On the Optimal Succinctness and Efficiency of Functional Encryption and Attribute-Based Encryption

Aayush Jain[1], Huijia Lin[2], and Ji Luo[2][0000−0003−1225−5310]

[1] Carnegie Mellon University, USA
aayushja@andrew.cmu.edu
[2] Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, USA
{rachel,luoji}@cs.washington.edu

Abstract. We investigate the best-possible (asymptotic) efficiency of functional encryption (FE) and attribute-based encryption (ABE) by proving inherent space-time trade-offs and constructing nearly optimal schemes. We consider the general notion of partially hiding functional encryption (PHFE), capturing both FE and ABE, and the most efficient computation model of random-access machine (RAM). In PHFE, a secret key sk_f is associated with a function f, whereas a ciphertext $\mathsf{ct}_x(y)$ is tied to a public input x and encrypts a private input y. Decryption reveals $f(x, y)$ and nothing else about y.

We present the first PHFE for RAM solely based on the necessary assumption of FE for circuits. Significantly improving upon the efficiency of prior schemes, our construction achieves nearly optimal succinctness and computation time:

- Its secret key sk_f is of *constant size* (optimal), independent of the function description length $|f|$, i.e., $|\mathsf{sk}_f| = \mathrm{poly}(\lambda)$.
- Its ciphertext $\mathsf{ct}_x(y)$ is *rate-2* in the private input length $|y|$ (nearly optimal) and *independent* of the public input length $|x|$ (optimal), i.e., $|\mathsf{ct}_x(y)| = 2|y| + \mathrm{poly}(\lambda)$.
- Decryption time is *linear* in the *instance* running time T of the RAM computation, plus the function and public/private input lengths, i.e., $T_{\mathsf{Dec}} = (T + |f| + |x| + |y|)\,\mathrm{poly}(\lambda)$.

As a corollary, we obtain the first ABE with both keys and ciphertexts being constant-size, while enjoying the best-possible decryption time matching the lower bound by Luo [ePrint '22]. We also separately achieve several other optimal ABE subject to the known lower bound.

We study the barriers to further efficiency improvements. We prove the first unconditional space-time trade-offs for (PH-)FE:

- *No* secure (PH-)FE can have $|\mathsf{sk}_f|$ and T_{Dec} *both* sublinear in $|f|$.
- *No* secure PHFE can have $|\mathsf{ct}_x(y)|$ and T_{Dec} *both* sublinear in $|x|$.

Our lower bounds apply even to the weakest secret-key 1-key 1-ciphertext selective schemes. Furthermore, we show a conditional barrier towards the optimal decryption time $T_{\mathsf{Dec}} = T\,\mathrm{poly}(\lambda)$ while keeping linear size dependency — any such (PH-)FE scheme implies doubly efficient private information retrieval (DE-PIR) with linear-size preprocessed database, for which so far there is no candidate.

1 Introduction

Functional encryption (FE) [15,50] and attribute-based encryption (ABE) [34, 52] are powerful enhancement of public-key encryption with many fascinating applications. In this work, we investigate the best-possible efficiency of these primitives, proving inherent space-time trade-offs for FE and presenting nearly optimal constructions of FE and ABE.

To this end, we consider the general notion of partially hiding functional encryption (PHFE) [5,33,38] capturing both FE and ABE. In PHFE, a secret key sk_f is associated with a function f, whereas a ciphertext $\mathsf{ct}_x(y)$ is tied to a public input x and encrypts a private input y. Their decryption recovers the computation output $f(x, y)$. Collusion-resistant (indistinguishability-based) security ensures that given unboundedly (polynomially) many secret keys $\{\mathsf{sk}_{f_q}\}_q$ for different functions $\{f_q\}_q$, ciphertexts $\mathsf{ct}_x(y_0)$ and $\mathsf{ct}_x(y_1)$ tied to the same public input x and encrypting different private inputs y_0, y_1 remain indistinguishable so long as none of the keys separate them by functionality, i.e., $f_q(x, y_0) = f_q(x, y_1)$. Put simply, the only information revealed about the private input y is the outputs $\{f_q(x, y)\}_q$.

Over the past decade, significant progress has been made in establishing the feasibility of (PH-)FE, for various classes of computation, with different levels of efficiency and security, and from different assumptions. However, we are yet to understand the asymptotic optimality and theoretical limits of its efficiency. We ask:

What is the best-possible asymptotic efficiency of PHFE?
Are there trade-offs among different aspects of efficiency?
Can we construct optimally efficient PHFE?

We make progress towards answering the above questions.

For the lower bounds, we prove inherent trade-offs between the size of keys or ciphertexts and the decryption time, and show barriers towards achieving the optimal decryption time.

On the constructive front, we present the first collusion-resistant PHFE for RAM solely based on the necessary assumption of (polynomially secure) collusion-resistant FE for circuits, which in turn can be based on well-studied assumptions [39,40]. Our scheme has nearly minimally sized keys and ciphertexts, and nearly optimal decryption time matching our lower bounds. As a corollary, we obtain the first ABE with both constant-size keys and constant-size ciphertexts, and the best-possible decryption time matching the recently discovered lower bound [48]. By slighting tweaking the construction, we also obtain ABE with linear-size keys and/or ciphertexts and the optimal decryption time subject to the known lower bound.

Dream Efficiency. Before describing our results, we first picture the dream efficiency with respect to three important dimensions. Each dimension has been a consistent research theme across many primitives in cryptography.

Efficient Computation Model. Functions should be represented by random-access machines (RAM), the most efficient computation model subsuming both circuits and Turing machines. RAM is also closer to real-world computers.

We consider a RAM U (fixed[3] at set-up time) with random access to three tapes, a function tape containing f, an input tape containing $x\|y$, and a working tape. It may produce *arbitrarily long* output, e.g., one bit at *every* step. This flexible model captures many natural scenarios, e.g., binary search where the database could be part of f, x, y. It can emulate the evaluation of a circuit C on input (x, y) by putting the circuit description on the function tape. In ciphertext-policy ABE, each ciphertext is tied to a predicate P, which can be captured by setting $x = P$. These examples tell us that any or even all of f, x, y could be long, and we want to optimize the efficiency dependency on their lengths.

Succinctness. Enjoying low communication and storage overhead means having short master public key mpk, secret keys sk_f, and ciphertexts $\mathsf{ct}_x(y)$. At the most basic level, the size of each component should be *polynomial* in the length of the information it is associated with — $|\mathsf{mpk}| = \mathrm{O}(1)$,[4] $|\mathsf{sk}_f| = \mathrm{poly}(|f|)$, and $|\mathsf{ct}_x(y)| = \mathrm{poly}(|x|, |y|)$, where $|f|, |x|, |y|$ are the description lengths of f, x, y, respectively — referred to as *polynomial efficiency*.[5] However, there is much to be desired beyond this basic level of efficiency. For instance, linear efficiency means $|\mathsf{sk}_f| = \mathrm{O}(|f|)$ and $|\mathsf{ct}| = \mathrm{O}(|x| + |y|)$, and rate-1 efficiency means $|\mathsf{sk}_f| = |f| + \mathrm{O}(1)$ and $|\mathsf{ct}| = |x| + |y| + \mathrm{O}(1)$.

In fact, even smaller parameters are possible. Since (PH-)FE does not aim to hide the function f, it is allowed to put the description of f in the clear in the secret key, and the *non-trivial* part of the secret key may be shorter than f. In this case, the right measure of efficiency should be the size of the non-trivial part (i.e., the overhead), which we now view as *the* secret key. We can aim for secret keys of size *independent* of that of the function — i.e., $|\mathsf{sk}_f| = \mathrm{O}(1)$ — referred to as *constant-size* keys. The same observation applies to the public input x tied to the ciphertext and we can hope for ciphertexts of size *independent* of $|x|$ while having optimal, rate-1 dependency on the private input length $|y|$ — i.e., $|\mathsf{ct}_x(y)| = |y| + \mathrm{O}(1)$. In summary:

$$\underline{\textit{Ideal Succinctness:}} \quad |\mathsf{mpk}| = \mathrm{O}(1),\ |\mathsf{sk}_f| = \mathrm{O}(1),\ |\mathsf{ct}_x(y)| = |y| + \mathrm{O}(1).$$

Note that the ideal component sizes are completely independent of the running time or the output length of the computation.

[3] We can think of U as a universal RAM.

[4] In this introduction, $\mathrm{O}(\cdot)$ hides a multiplicative factor of $\mathrm{poly}(\lambda)$.

[5] It may appear that polynomial efficiency is the bare minimum. However, it is possible to consider components whose sizes depend on an upper bound of the length of some information *not* tied to them. Many early schemes are as such, e.g., the FE scheme of [26] has $|\mathsf{mpk}| = \mathrm{O}(\mathrm{poly}(\max |y|))$, and the celebrated ABE scheme by [14] has $|\mathsf{mpk}|, |\mathsf{ct}|$ growing polynomially with the maximum depth of the computation. When a scheme requires fixing an upper bound on parameter Z (e.g., input length, depth, or size), it is said to be Z-bounded.

Decryption Time. Decryption is also a RAM computation, $\mathsf{Dec}^{f,x,\mathsf{sk}_f,\mathsf{ct}_x(y)}(\mathsf{mpk})$, which on input mpk and with random access to $f, x, \mathsf{sk}_f, \mathsf{ct}_x(y)$, computes the output $U^{f,x\|y}()$. We want decryption to be efficient, ideally taking time linear in the *instance* running time T of the RAM computation in the clear:

$$\underline{\textit{Ideal Decryption Time}}: \quad T_{\mathsf{Dec}} = \mathrm{O}(T).$$

Organization. In Sect. 1.1, we describe our results. In Sect. 1.2, we present an overview of our techniques. In Sect. 1.3, we discuss the related works. In Sect. 2, we lay out our formulations of succinct garbled RAM and PHFE. In Sect. 3, we formally prove our unconditional lower bounds. Due to the space constraint, we refer the readers to the full version [37] for the complete details on definitions, constructions, applications, and proofs.

1.1 Our Results

The question is whether the dream efficiency is attainable simultaneously in all above three dimensions. Towards understanding this, we present both new constructions and lower bounds.

New PHFE for RAM with Nearly Optimal Succinctness. Starting from selectively and polynomially secure bounded FE for circuits, i.e., all of $|\mathsf{mpk}|$, $|\mathsf{sk}_f|, |\mathsf{ct}(y)|$ are just $\mathrm{poly}(|f|, |y|)$, which in turn can be constructed from three well-studied assumptions [39,40], we construct an adaptively secure (unbounded) PHFE for RAM with nearly optimal succinctness.

Theorem 1 (informal)**.** *Assuming polynomially secure FE for circuits, there exists an adaptively secure PHFE for RAM:*

$$\underline{\textit{PHFE Efficiency}}: \quad |\mathsf{mpk}| = \mathrm{O}(1),\ |\mathsf{sk}_f| = \mathrm{O}(1),\ |\mathsf{ct}_x(y)| = 2|y| + \mathrm{O}(1),$$
$$T_{\mathsf{Dec}} = \mathrm{O}(T + |f| + |x| + |y|).$$

Our construction gives the first collusion-resistant (PH-)FE for RAM, and also the first (PH-)FE for any model of computation (e.g., circuit or TM) with nearly optimal succinctness. The only gap to optimality is that the ciphertext is rate-2 in $|y|$ instead of rate-1. Prior constructions work with either circuits [26,39,40] or Turing machines [1,8,42], except for the recent concurrent and independent work of [3], which also constructs FE for RAM. All of them only achieve polynomial efficiency as summarized in Table 1. We further discuss related works in Sect. 1.3.

As a corollary, we obtain the first ABE for RAM from falsifiable assumptions, and the first for any model of computation with both constant-size keys and constant-size ciphertexts. The only prior construction of ABE for RAM by [31] relies on non-falsifiable assumptions like SNARK and differing-input obfuscation. In terms of succinctness, existing schemes achieve *either* constant-size keys *or* constant-size ciphertexts [9–11,45,46,53–55]. Achieving constant-size keys and ciphertexts *simultaneously* has been an important theoretical open question (see discussion in [45]). The state-of-the-art is summarized in Table 2.

Table 1. Comparison among some (PH-)FE schemes. All rows except this work are FE, and this work is PHFE. C is the circuit. T is the instance running time of TM/RAM. All $\mathrm{poly}(\cdot)$ and $\mathrm{O}(\cdot)$ implicitly contains λ. For assumptions, FE is always for circuits, "sls" means sublinearly succinct, "subexp" means subexponentially secure, and "PK-DE-PIR" means public-key doubly efficient private information retrieval.

reference	functionality	$\lvert\mathsf{sk}\rvert$	$\lvert\mathsf{ct}\rvert$	T_{Dec}	adaptive	assumption
GGHRSW [26]	circuit	$\mathrm{poly}(\lvert C\rvert)$	$\mathrm{poly}(\lvert y\rvert)$	$\mathrm{poly}(\lvert C\rvert)$		$i\mathcal{O}$
KNTY [42]	circuit	$\mathrm{poly}(\lvert C\rvert)$	$\mathrm{poly}(\lvert y\rvert)$	$\mathrm{poly}(\lvert C\rvert)$	✓	1-key sls FE
GWZ [35]	circuit	$\mathrm{poly}(\lvert C\rvert)$	$\lvert y\rvert + \mathrm{O}(1)$	$\mathrm{poly}(\lvert C\rvert)$		$i\mathcal{O}$
AS [8]	TM	$\mathrm{poly}(\lvert f\rvert)$	$\mathrm{poly}(\lvert y\rvert)$	$\mathrm{poly}(\lvert f\rvert, \lvert y\rvert)T$	✓	$i\mathcal{O}$
AJS [6]	TM	$c\lvert f\rvert + \mathrm{O}(1)$	$c\lvert y\rvert + \mathrm{O}(1)$	$\mathrm{poly}(\lvert f\rvert, \lvert y\rvert)T$	✓	subexp $i\mathcal{O}$
AM [1]	TM	$\mathrm{poly}(\lvert f\rvert)$	$\mathrm{O}(\lvert y\rvert)$	$\mathrm{poly}(\lvert f\rvert)T$	✓	FE
KNTY [42]	TM	$\mathrm{poly}(\lvert f\rvert)$	$\mathrm{poly}(\lvert y\rvert)$	$\mathrm{poly}(\lvert f\rvert, \lvert y\rvert)T$		1-key sls FE
ACFQ [3]	RAM	$\mathrm{poly}(\lvert f\rvert)$	$\mathrm{poly}(\lvert y\rvert)$	$\mathrm{O}(T)$		PK-DE-PIR & FE
this work	RAM	$\mathrm{O}(1)$	$2\lvert y\rvert + \mathrm{O}(1)$	$\mathrm{O}(T + \lvert f\rvert + \lvert x\rvert + \lvert y\rvert)$	✓	FE

Table 2. Comparison among some KP-ABE schemes. Notations shared with Table 1. ABP means arithmetic branching programs (also using C). For assumptions, "k-Lin" means k-Linear in pairing groups, "GGM" means generic pairing group model, and "$di\mathcal{O}$" means differing-input obfuscation.

reference	functionality	$\lvert\mathsf{sk}\rvert$	$\lvert\mathsf{ct}\rvert$	T_{Dec}	adaptive	assumption
BGGHNSVV [14]	circuit	$\mathrm{poly}(d)$	$\lvert x\rvert\,\mathrm{poly}(d)$	$\lvert C\rvert\,\mathrm{poly}(d)$		LWE
LL [46]	ABP	$\mathrm{O}(\lvert C\rvert \cdot \lvert x\rvert)$	$\mathrm{O}(1)$	$\mathrm{O}(\lvert C\rvert \cdot \lvert x\rvert)$	✓	k-Lin
LLL [45]	circuit	$\mathrm{O}(1)$	$\mathrm{poly}(d)$	$\lvert C\rvert\,\mathrm{poly}(d)$	✓	GGM & LWE
GKPVZ [31]	RAM	$\mathrm{O}(1)$	$\mathrm{poly}(\lvert x\rvert)$	$\mathrm{O}(T + \lvert f\rvert + \lvert x\rvert)$		SNARK & diO
this work	RAM	$\mathrm{O}(1)$	$\mathrm{O}(1)$	$\mathrm{O}(T + \lvert f\rvert + \lvert x\rvert)$	✓	FE

Corollary 2 (informal)**.** *Assuming polynomially secure FE for circuits, there exists an adaptively secure key-policy ABE (KP-ABE) for RAM as well as an adaptively secure ciphertext-policy ABE (CP-ABE) for RAM:*

$$\begin{aligned}
&\underline{\textit{KP-ABE Efficiency}}\textit{:} \quad \lvert\mathsf{mpk}\rvert = \mathrm{O}(1), \quad \lvert\mathsf{sk}_f\rvert = \mathrm{O}(1), \quad \lvert\mathsf{ct}_x\rvert = \mathrm{O}(1),\\
&\qquad\qquad\qquad\qquad T_{\mathsf{Dec}} = \mathrm{O}(T + \lvert f\rvert + \lvert x\rvert);\\
&\underline{\textit{CP-ABE Efficiency}}\textit{:} \quad \lvert\mathsf{mpk}\rvert = \mathrm{O}(1), \quad \lvert\mathsf{sk}_x\rvert = \mathrm{O}(1), \quad \lvert\mathsf{ct}_f\rvert = \mathrm{O}(1),\\
&\qquad\qquad\qquad\qquad T_{\mathsf{Dec}} = \mathrm{O}(T + \lvert f\rvert + \lvert x\rvert).
\end{aligned}$$

The decryption time of our PHFE and ABE *appears* suboptimal. In addition to the necessary linear dependency on T, it also grows linearly with $|f|, |x|, |y|$. It turns out that *ideal succinctness and ideal decryption time are at conflict*. We prove that under *sublinear* succinctness, the linear dependency of T_{Dec} on $|f|, |x|$ is *inherent*. We also show barriers towards removing the dependency of T_{Dec} on $|y|$ or $|f|, |x|$ while maintaining linear succinctness.

Our PHFE scheme matches the lower bounds and barriers — it is Pareto-optimal with respect to the dependency on $|f|, |x|$. For ABE, our lower bounds and barriers do not apply. Nevertheless, our ABE scheme matches an existing lower bound by [48], which states that any moderately expressive ABE must

satisfy $|\mathsf{ct}_x| \cdot T_{\mathsf{Dec}} = \Omega(|x|)$ and $|\mathsf{sk}_f| \cdot T_{\mathsf{Dec}} = \Omega(|f|)$.[6] Given that our scheme has constant-size keys and ciphertexts, its decryption time matches the lower bound of [48], hence it is thus Pareto-optimal. By tweaking the construction, we obtain several other Pareto-optimal ABE schemes:

Theorem 3 (informal)**.** *Assuming polynomially secure FE for circuits, there exist adaptively secure KP-/CP-ABE schemes for RAM:*

$$\begin{aligned} \underline{\textit{KP-ABE Efficiency:}} \quad & |\mathsf{mpk}| = \mathrm{O}(1),\ |\mathsf{sk}_f| = |f|^\alpha + \mathrm{O}(1),\ |\mathsf{ct}_x| = |x|^\beta + \mathrm{O}(1), \\ & T_{\mathsf{Dec}} = \mathrm{O}(T + |f|^{1-\alpha} + |x|^{1-\beta}). \\ \underline{\textit{CP-ABE Efficiency:}} \quad & |\mathsf{mpk}| = \mathrm{O}(1),\ |\mathsf{sk}_x| = |x|^\beta + \mathrm{O}(1),\ |\mathsf{ct}_f| = |f|^\alpha + \mathrm{O}(1), \\ & T_{\mathsf{Dec}} = \mathrm{O}(T + |f|^{1-\alpha} + |x|^{1-\beta}). \end{aligned}$$

All four combinations of $\alpha, \beta \in \{0, 1\}$ are possible for both KP- and CP-ABE.

Contention Between Succinct Components and Fast Decryption. We now describe our lower bounds in more detail. Consider the efficiency dependency on the lengths of public information f and x. We show that unconditionally, it is impossible to simultaneously achieve key size sublinear in $|f|$ and decryption time sublinear in $|f|$. Similarly, it is impossible to have both ciphertext size and decryption time sublinear in $|x|$. In fact, these trade-offs apply to the weakest secret-key 1-key 1-ciphertext selectively secure PHFE, and the first trade-off with respect to $|f|$ also applies to plain FE. More precisely:

Theorem 5 (informal)**.** *For a secret-key 1-key 1-ciphertext selectively secure moderately expressive PHFE with*

$$|\mathsf{sk}| = \mathrm{O}(|f|^\alpha) \quad \text{and} \quad T_{\mathsf{Dec}} = (T + |f|^\beta + |y|)\,\mathrm{poly}(|x|),$$

it must hold that $\alpha \geq 1$ or $\beta \geq 1$. The same (without x) is true for FE.

Theorem 6 (informal)**.** *For a secret-key 1-key 1-ciphertext selectively secure moderately expressive PHFE with*

$$|\mathsf{ct}| = |x|^\alpha\,\mathrm{poly}(|y|) \quad \text{and} \quad T_{\mathsf{Dec}} = (T + |f| + |x|^\beta)\,\mathrm{poly}(|y|),$$

it must hold that $\alpha \geq 1$ or $\beta \geq 1$.

Our PHFE scheme achieves one profile of Pareto-optimality, $\alpha = 0$ and $\beta = 1$.

A natural question that our work leaves open is whether the other Pareto-optimal profile, $\alpha = 1$ and $\beta = 0$ (or even just $\beta < 1$), is attainable. Another question is whether the decryption time must grow with the length of the private input y.

[6] The lower bounds apply as long as the ABE scheme supports broadcast encryption. Theorem 14 in [48] is the first trade-off between $|\mathsf{ct}_x|$ and T_{Dec}. Essentially the same proof yields the second trade-off between $|\mathsf{sk}_f|$ and T_{Dec}.

Barriers to Ideal Decryption Time. We illustrate barriers to positive answers to the above two questions. We show that PHFE with decryption time independent of $|y|$, $|x|$, or $|f|$ implies secret-key doubly efficient private information retrieval (SK-DE-PIR) with small preprocessed database.

Theorem 4 (informal)**.** *Suppose a moderately expressive secret-key PHFE with selective security has*

$$|\mathsf{sk}_f| = \ell_{\mathsf{sk}}(\lambda, |f|), \qquad |\mathsf{ct}_x(y)| = \ell_{\mathsf{ct}}(\lambda, |x|, |y|),$$
$$T_{\mathsf{Dec}} = (|f|^{e_f} + |x|^{e_x} + |y|^{e_y})\,\mathrm{poly}(T).$$

Then the following hold:

- *If $e_x = 0$, there exists an SK-DE-PIR scheme with preprocessed database size $\ell_{\mathsf{ct}}(N, \mathrm{poly}(\lambda), \lambda)$, where N is the length of the original database. The PHFE only has to be 1-ciphertext secure.*
- *If $e_y = 0$, the SK-DE-PIR preprocessed database will have size $\ell_{\mathsf{ct}}(0, N, \lambda)$. The PHFE only has to be 1-ciphertext secure.*
- *If $e_f = 0$, the SK-DE-PIR preprocessed database will have size $\ell_{\mathsf{sk}}(N, \lambda)$. The PHFE only has to be 1-key secure.*

SK-DE-PIR, introduced by [17,22], allows a client to privately encode a database D into $\widetilde{D}$ while keeping a secret key k. Later, client can outsource the encoded database $\widetilde{D}$ to a remote storage server, and *obliviously* query the database using k hiding the logical access pattern. Different from ORAM, the server never updates the encoded database nor keeps any additional state. Different from PIR, SK-DE-PIR allows the database to be privately encoded in exchange for *double efficiency* — for each query, the complexities of both the client and the server are, ideally, independent of the database size $|D|$, whereas PIR necessarily has server complexity $\Omega(|D|)$. The double efficiency of SK-DE-PIR makes it highly desirable. The initial works [17,22] presented candidate constructions based on a new conjecture that permuted local-decoding queries (for a Reed–Muller code with suitable parameters) are computationally indistinguishable from uniformly random sets of points. More recently, a simple "toy conjecture" inspired by (though formally unrelated to) those SK-DE-PIR schemes has been broken [16]. Very recently, in a concurrent and independent work, Lin, Mook, and Wichs [47] constructed DE-PIR with public preprocessing from the ring-LWE assumption. The most important efficiency metrics of DE-PIR are the preprocessed database size and the complexity per query. None of the existing schemes simultaneously achieve constant complexity per query and linear-size preprocessed database.

Our theorem shows that constructing PHFE with short decryption time entails constructing SK-DE-PIR with preprocessed database size inherited from ciphertext/key size. In particular, a PHFE scheme with decryption time independent of $|y|$ and ciphertext size linear in $|y|$ implies an SK-DE-PIR with preprocessed database of length $\mathrm{O}(N)$ and constant complexity per query. Since no such SK-DE-PIR is known even under non-standard assumptions, this represents a barrier towards improving the decryption time dependency on $|y|$ in our PHFE construction.

Succinct Garbled RAM and Constant-Overhead $i\mathcal{O}$. The main tool in our construction of PHFE for RAM is succinct garbled RAM (GRAM). Initiated by [13,21,43], a sequence of works have constructed succinct garbled RAM [2, 19,20,23] based on subexponentially secure FE for circuits and succinct garbled Turing machines [6,7,30,43] based on polynomially secure FE for circuits.

In this work, we formulate a new notion of succinct GRAM (informally described in the technical overview and formally defined in Sect. 2.1) geared for building highly efficient PHFE for RAM, and construct it based on polynomially secure FE for circuits. Our construction has two consequences: *i)* we obtain the first succinct GRAM (our or the standard notion) based on polynomial hardness, as opposed to subexponential hardness as in prior constructions, and *2)* using $i\mathcal{O}$ for circuits, we obtain $i\mathcal{O}$ for RAM with constant overhead — the size of the obfuscated program is $2|M| + \mathrm{poly}(\lambda, \ell)$, where M is the original RAM and ℓ is the input length. Previously, constant-overhead $i\mathcal{O}$ was only known for Turing machines [6].

1.2 Technical Overview

We start with an overview of our negative results.

Unconditional Lower Bounds. As introduced earlier, we show that it is impossible for a secure PHFE to enjoy sublinear dependency on $|f|$ (resp. $|x|$) simultaneously for $|\mathsf{sk}_f|$ (resp. $|\mathsf{ct}_x(y)|$) and T_{Dec} when T_{Dec} is linear in $T, |x|, |y|$ (resp. $T, |f|, |y|$). We illustrate our ideas of proving the contention between

$$|\mathsf{sk}_f| = \mathrm{O}(|f|^\alpha) \quad \text{and} \quad T_{\mathsf{Dec}} = \mathrm{O}(T + |f|^\beta + |x| + |y|) \qquad \text{for } \alpha < 1 \text{ and } \beta < 1$$

by exhibiting an efficient adversary breaking the security of PHFE for RAM (polynomial factors in the security parameter are ignored).

Adversarial Function and Strategy. The adversary will selectively request one secret key and one ciphertext. Let $n < N$ be determined later.

- The function f is described by a string $R \in \{0,1\}^N$.
- There is no public input, so $x = \perp$.
- The private input y is either $(I \subseteq [N], w \in \{0,1\}^n)$ or $z \in \{0,1\}^n$, where I is a set containing n indices and w is a one-time pad.

The functionality is simply *reading and XORing* or *outputting as-is*, i.e.,

$$f(x, y) = \begin{cases} R[I] \oplus w, & \text{if } y = (I, w); \\ z, & \text{if } y = z; \end{cases}$$

where $R[I]$ means the n bits of R at the indices in I. Clearly,

$$|f| = \mathrm{O}(N), \quad |x| = \mathrm{O}(1), \quad |y| = \mathrm{O}(n), \quad T = \mathrm{O}(n),$$
$$|\mathsf{sk}| = \mathrm{O}(N^\alpha), \qquad T_{\mathsf{Dec}} = \mathrm{O}(n + N^\beta).$$

More precisely, $|y| = \mathrm{O}(n \log n)$, but the $\log n$ factor is absorbed by the $\mathrm{poly}(\lambda)$ factor hidden in $\mathrm{O}(\cdot)$.

The adversary chooses

$$\begin{array}{ll} \text{key query} & f \text{ with } R \xleftarrow{\$} \{0,1\}^N, \\ \text{challenge} & x \leftarrow \bot, \qquad y_0 \xleftarrow{\$} \text{random } (I, w), \quad y_1 \leftarrow z = R[I] \oplus w. \end{array}$$

By our choice, $f(x, y_0) = R[I] \oplus w = z = f(x, y_1)$, so the challenge is well-formed. Upon receiving sk and ct, the adversary simply runs the decryption algorithm on $(\mathsf{sk}, \mathsf{ct})$ with random access to the function, i.e., R, in the clear. Let $L \subseteq [N]$ be the set of indices in R that is read during decryption.

$$R[I] \oplus w \leftarrow \mathsf{Dec}^{R, x=\bot}(\mathsf{sk}, \mathsf{ct}), \text{ where } \mathsf{Dec} \text{ reads } R[L]$$

The adversary determines that

$$\mathsf{ct} \text{ is an encryption of } \begin{cases} y_0 = (I, w), & \text{if } |L \cap I| \text{ is large;} \\ y_1 = z, & \text{if } |L \cap I| \text{ is small;} \end{cases}$$

where the threshold for *large* and *small* is described below.

Intuition and Toy Proof. The intuition behind the adversary's strategy is simple. Let L_b be the index set L that decryption accessed when decrypting ct encrypting y_b.

- When ct encrypts y_0, decryption must correctly recover $R[I]$ (as the adversary knows w). The decryption algorithm can only obtain information of $R[I]$ either from sk or via accesses to the tape R. Since $R[I]$ contains n bits of information, by setting $|\mathsf{sk}| = \mathrm{O}(N^\alpha) \ll n$, decryption must read a *large* portion of information of $R[I]$ from the tape R, implying $|L \cap I|$ is *large*, namely, $\Omega(n)$.
- In contrast, when ct encrypts y_1, observe that the joint distribution of $(R, \mathsf{ct}, \mathsf{sk})$ is independent of I, as w is a one-time pad and completely hides I in $y_1 = R[I] \oplus w$. Therefore, the behavior of Dec is independent of I. Since Dec runs in a short time $\mathrm{O}(n + N^\beta) \ll N$, without knowing I, where it reads in R cannot overlap with I for a large portion. Therefore, $|L \cap I|$ is likely to be *small*.

It remains to analyze how large and small $|L \cap I|$ is in the above two cases. Let us first consider a toy proof, under the hypothesis that sk contains no information about $R[I]$ at all. We will remove this hypothesis below. By this hypothesis, when ct encrypts y_0, the decryption algorithm must read the entire $R[I]$ from the tape R and hence $|L \cap I| = n$. When ct encrypts y_1, since the indices L_1 that the decryption algorithm reads from R is independent of I, the intersection size $|L_1 \cap I|$ follows a hypergeometric distribution, and hence

$$\mathbb{E}\big[\,|L_1 \cap I|\,\big] \le \frac{T_{\mathsf{Dec}} \cdot n}{N} \ll n \tag{1}$$

This means the adversary can distinguish when ct encrypts y_0 or y_1 with good probability, and contradicts the security of PHFE.

Removing the Hypothesis. The hypothesis that sk contains no information about $R[I]$ at all may well be false. When it is removed, we can no longer argue that $I \subseteq L_0$, as the adversary may obtain some information of $R[I]$ from sk. Our intuition is $|L_0 \cap I| \geq |I| - |\mathsf{sk}|$, but proving this formally is not trivial as sk may contain arbitrary information of $R[I]$.

We employ a compression argument. The basic idea behind a compression argument is that there is no pair of encoding and decoding algorithms (Encode, Decode), with arbitrarily long shared randomness s, is able to transmit an n-bit random string u (independent of s) from one end to the other, via an encoding v containing less than n bits. Informally,

$$\text{if } \Pr\begin{bmatrix} s \xleftarrow{\$} \mathcal{D}_s \\ u \xleftarrow{\$} \{0,1\}^n & : \mathsf{Decode}(s,v) = u \\ v \leftarrow \mathsf{Encode}(s,u) \end{bmatrix} = 1, \quad \text{then } |v| \geq |u|.$$

(Lemma 9 is the formal statement by [24].) We show that if $|L_0 \cap I| < |I| - |\mathsf{sk}|$, then we can design a pair of $(\mathsf{Encode}, \mathsf{Decode})$ violating the above information-theoretic bound.

- The shared randomness s is the PHFE randomness and $I, w, R[[N] \setminus I]$.
- To encode $u \in \{0,1\}^n$, the procedure Encode first sets $R[I] = u$. Using s, it then generates a PHFE key sk for R and a ciphertext ct encrypting $y_0 = (I, w)$, runs Dec to obtain the locations L_0 in R that decryption reads. Lastly, it sets the codeword to be $v = (\mathsf{sk}, R[L_0 \cap I])$.
- To decode, Decode regenerates ct using s, and runs Dec to obtain the output $z = R[I] \oplus w$ and recover $u = z \oplus w$. During decryption, Dec queries for locations L_0 in R. Every query is in either $R[[N] \setminus I]$ or $R[L_0 \cap I]$; the former can be answered by finding the right element in s and the latter in v.

Suppose $|L_0 \cap I| < |I| - |\mathsf{sk}|$, then $|v|$ is less than $|I| = |u|$, which contradicts the incompressibility of u. (In the formal proof, we make v fixed-length and suffer from incorrect decoding, hence the statements are probabilistic. See Sect. 3.1 for more details.)

Stepping back, the compression argument shows that $|L_0 \cap I| \geq |I| - |\mathsf{sk}|$. In contrast, by Eq. (1), $|L_1 \cap I| \leq n/2$ with high probability. To show that the adversary can distinguish ct encrypting y_0 or y_1, we can set, e.g., $n = N^{(\alpha+1)/2}$, so that $|\mathsf{sk}| = \mathrm{O}(N^\alpha) \ll n$, and $|L_0 \cap I| \geq |I| - |\mathsf{sk}| \geq n/2$. (In the formal proof, N itself is a large $\mathrm{poly}(\lambda)$ to overwhelm $\mathrm{poly}(\lambda)$ factors, which is ignored in this overview.) In summary, any PHFE with both $|\mathsf{sk}|$ sublinear in $|f|$ and T_{Dec} sublinear in $|f|$ (and linear in $T, |x|, |y|$) is insecure.

Technical Barrier Towards Fast Decryption. As described earlier, we show barriers in current techniques against constructing a PHFE scheme with fast decryption. Consider a PHFE scheme whose decryption time is

$$\mathrm{O}(T^{\beta_T}_{\varphi(f,x,y)} + |f|^{\beta_f} + |x|^{\beta_x} + |y|^{\beta_y})$$

for constants $\beta_T, \beta_f, \beta_x\beta_y$. We show that even if just one of β_x, β_y, and β_f is zero, then the scheme implies SK-DE-PIR (an informal description of SK-DE-PIR is in the introduction and formal definition in the full version [37]).

To illustrate our main idea, we start by describing this transformation for the case when the decryption is efficient in the length of the public input x, namely when $\beta_x = 0$. The ideas naturally extend to the other cases.

Since the decryption is efficient in $|x|$, as a first attempt, we set $\mathsf{DB} \in \{0,1\}^n$ as x, and y as empty. The client processes the database DB by first sampling $(\mathsf{mpk}, \mathsf{msk})$ for the PHFE scheme and then encodes the database into

$$\widetilde{\mathsf{DB}} = (\mathsf{DB}, \mathsf{ct}_{\mathsf{PHFE}} = \mathsf{PHFE.Enc}(\mathsf{mpk}, (x = \mathsf{DB}, y = \bot)))$$

and sends it over to the server. To look up a location DB_{i_j}, the client can compute a PHFE function key sk_{f_j} for the program f_j that looks up and outputs DB_{i_j} and sends the key as the query to the server.

The server responds to the query by decrypting $\mathsf{ct}_{\mathsf{PHFE}}$ in $\widetilde{\mathsf{DB}}$ using the key sk_{f_j} and with random access to DB, and learns DB_{i_j}. Note that double efficiency requirement is already satisfied. Client only needs to compute a function key sk_{f_j} that can be computed in time polynomial in the description length of f_j, and hence polynomial in λ and $\log n$. On the other hand, due to the supposed efficiency of decryption, the decryption time is polynomial in $T_{f_j(x,y)}, |f_j|, |y|$, which are also polynomial in λ and $\log n$.

While this idea solves the core issue, we have completely missed one important aspect. The scheme reveals the indices $\{i_j\}_j$ to the server as the keys $\{\mathsf{sk}_{f_j}\}_j$ are not guaranteed to hide the function descriptions $\{f_j\}_j$. To resolve this issue, we observe that if we had a function-hiding PHFE scheme, we would have been done. To enable this, we will use similar techniques as used to convert any FE to a function-hiding FE [18]. Namely, we will compute a symmetric-key encryption SKE of the index i (denoted as ctk_1). We will hardwire ctk_1 in the function secret key instead of the index i. The corresponding secret key $\mathsf{SKE.sk}_1$ will be put in the private component y, which will be used to decrypt ctk_1 to learn index i. While this might seem to be enough, we face yet another issue. Learning DB_{i_j} in the clear upon decryption can reveal information about the index i_j to the server. To fix this, the decryption will output an encryption of DB_{i_j} under another secret key $\mathsf{SKE.sk}_2$ of the secret-key encryption scheme. We will put this key in the private input y along with a PRF key to derive randomness to compute the encryption.

$$\begin{aligned}\widetilde{\mathsf{DB}} &= (\mathsf{DB}, \mathsf{ct}_{\mathsf{PHFE}} = \mathsf{PHFE.Enc}(\mathsf{mpk}, (\mathsf{DB}, (\mathsf{SKE.sk}_1, \mathsf{SKE.sk}_2, \mathsf{PRF}.k)))),\\ \text{query} = \mathsf{sk}_{f_j} \text{ where } & f_j[\mathsf{ckt}_1(i_j), \$]^{\mathsf{DB}, (\mathsf{SKE.sk}_1, \mathsf{SKE.sk}_2, \mathsf{PRF}.k)}\\ &= \mathsf{SKE}(\mathsf{SKE.sk}_2, \mathsf{DB}_{i_j} \ ; \ \mathsf{PRF}(\mathsf{PRF}.k, \$)).\end{aligned}$$

Double efficiency is still preserved. We have increased the complexity of f_j multiplicatively by a polynomial amount (in λ and $\log n$), similarly the size of y is also polynomial in λ to store secret keys of SKE and a PRF key. There are a few more subtleties. To make the proof go through, we need to use the

Trojan method in the FE literature [25], which requires another encryption key $\mathsf{SKE.sk}_3$ and additional programming.

Overview of Our PHFE for RAM. At a very high level, we use a *succinct garbled RAM (GRAM) scheme* to lift a FE for circuits to a PHFE for RAM. This former can be viewed as a 1-key, 1-ciphertext, secret-key FE for RAM, where succinctness implies that the running time of key generation and encryption is independent of the running time of the RAM computation. The (collusion-resistant) FE for circuits is then used to lift the one-time security to many-time security. This high-level approach first appeared in [8] for building FE for TM. In this work, towards nearly optimally efficient FE for RAM, we first observe that existing definitions and constructions of succinct GRAM [2,13,19–21,23] are insufficient. Therefore, we first formulate a new variant of succinct GRAM, termed *laconic GRAM*, and then construct it using polynomially hard FE for circuit. Along the way, we weaken the assumption underlying succinct GRAM schemes from iO, which requires subexponentially hard FE for circuit, to polynomially-hard FE for circuits.

Let us first review the syntax and security of standard GRAM schemes. They consist of the following algorithms. The encoding algorithm encodes a database D into $\hat{D}$ and outputs a private state τ. The garbling algorithms uses τ to garble a RAM M into $\hat{M}$ and outputs a collection of input labels $\{L_{i,b}\}_{i,b}$. The evaluation algorithm given the garbled RAM $\hat{M}$, a subset of labels L_k corresponding to an input k, and random access to $\hat{D}$, returns the output $M^D(k)$ of the RAM computation. Simulation based security ensures that $\hat{D}, \hat{M}, L_k = \{L_{i,k_i}\}_i$ can be simulated using only the output $M^D(k)$. The efficiency of different algorithms is is described below.

$$(\hat{D}, \tau) \leftarrow \mathsf{Encode}(D)\ , \quad \hat{M}, \{L_{i,b}\}_{i,b} \leftarrow \mathsf{Garble}(M, \tau)\ , \quad M^D(k) = \mathsf{Eval}^{\hat{D}}(\hat{M}, L_k)$$
$$|\hat{D}| = |D|\,\mathrm{poly}(\lambda)\ , \qquad |\hat{M}| = \mathrm{poly}(\lambda), \qquad T_{\mathsf{Eval}} = T\,\mathrm{poly}(|M|, \lambda)$$

We now describe why the standard notion falls short for our purpose of building very efficient FE for RAM and how to address these issues. An informal definition of our succinct GRAM is provided in Fig. 1.

- *many-tape v.s. single-tape*: To start with, we consider RAM computation with multiple tapes $U^{x,y,f}(1)$ instead of a single tape $M^D(k)$.
- *public-tape v.s. private-tape*: Some of the tapes we consider, such as x and f, are public. But standard GRAM only provides a mechanism for encoding private tape, and the encoding is necessarily at least as long as the tape itself. However, optimal succinctness requires the FE ciphertext- and key-size to be independent of $|x|$ and $|f|$. Hence we cannot afford to encode x, f as in standard GRAM. Instead, our new notion of succinct GRAM has a $\mathsf{Compress}$ algorithm that compresses the public tapes into hashes/digests h_x and h_f; the Garble algorithm "ties" these hashes to the garbled program $\hat{U}$; and finally Eval makes random access to x and f in the clear directly (just as the decryption algorithm of RAM-FE does).

– *rate-2 encoding v.s. rate-*$\mathrm{poly}(\lambda)$ *encoding of private tape:* Our setting also has private tape, namely y. But optimal succinctness requires concretely efficient, rate-1 or rate-2, encoding of y, whereas standard GRAM allows much worse rate $\mathrm{poly}(\lambda)$. To achieve concretely efficient encoding, we can only encrypt y using a rate-1 encryption scheme. To bind the encryption $\hat{y}$ with a garbled program, we simply treat $\hat{y}$ as yet another public tape just like x, f. In other words, we consider the modified RAM computation $\bar{U}^{x,\hat{y},f}(k) = U^{x,y,f}(1)$, where k is the secret key of the rate-1 encryption. As such, our succinct GRAM only need to handle public tapes.

  In our construction of FE, additional care needs to be taken to ensure that our GRAM security implies that y is hidden. To achieve this, We rely on existing techniques [49], which requires two (rate-1) encryption of y with independent keys so that different security hybrids can invoke the semantic security of different encryption. This is is why our FE has rate-2 dependency on $|y|$, instead of rate-1. We omit details in this overview.

– *reusable digests v.s. one-time encoding:* In standard GRAM, the database encoding $\hat{D}$ can only be used, *once*, by a single garbled program $\hat{M}$ generated using the right state τ. The technical cause of the one-time security of $\hat{D}$ is due to the use of ORAM in order to hide the access pattern of M to D; the same ORAM storage $\hat{D}$ cannot be used by multiple garbled programs.[7] The one-time security means that when using succinct GRAM to construct RAM-FE, the decryption of every ciphertext and key pair must generate fresh encoding $\hat{D}$ (and $\hat{M}$). Such fresh encoding can only be generated using the underlying FE for circuit, by encoding D in its key or ciphertext, which would lead to large polynomial dependency on $|D|$.

  Our notion of succinct GRAM compresses the public tapes into hashes $h_x, h_f, h_{\hat{y}}$. For the same reasons, we cannot afford to generate fresh hashes at decryption of every pair of ciphertext and key. Instead, our hashes are *reusable* – they can be tied to multiple garbled programs; this is ensured by the fact that our Garble algorithm does not take any private state from the $\mathsf{Compress}$ algorithm. A technical issue we must resolve is how to hide access pattern to the public tapes $x, \hat{y}, f$ since they are not encoded using ORAM, which we discuss later.

– *Difference in decryption time:* The reusability comes at a cost. In our new notion, evaluation time is $(T + |x| + |y| + |f|)\,\mathrm{poly}(\lambda)$ whereas standard GRAM has evaluation time $T\,\mathrm{poly}(|M|, \lambda)$ independent of tape size $|D|$. Nevertheless, our lower bound for RAM-FE implies that the dependency on $|x|, |y|, |f|$ is hard to get around (as our succinct GRAM implies RAM-FE with the same decryption time).

[7] This should be distinguished from the scenario of GRAM with persistent database where a sequence of garbled program $(\hat{M}_1, \hat{M}_2, \cdots)^{\hat{D}}$ are executed sequentially with $\hat{D}$. The difference lies in that in sequential execution, each garbled RAM can modify $\hat{D}$ and the changes are kept persistently to the next computation. Here, we are considering the scenario where the unmodified $\hat{D}$ is used by multiple garbled program, which breaks ORAM security.

– *RAM with long outputs v.s. single-bit output:* Standard succinct GRAM handles RAM computation with a single-bit output. To handle RAM with m-bit output, it reduces to creating m instances of garbled RAM, one for each output bit. Under simulation security, the size of the garbled RAM necessarily grows linearly with the output length m.
In our notion, we require garbling RAM with arbitrarily long outputs, without efficiency degradation in the output length. To do so, we switch to indistinguishability based security instead of simulation security.

Putting the above pieces together, we formulate succinct GRAM as in Fig. 1.

Our Notion of Succinct Garbled RAM

– $\mathsf{Compress}(\tau, D_\tau)$ compresses the τ'th public tape D_τ into a short hash digest_τ of length $\mathrm{poly}(\lambda)$. It runs in time $\mathrm{O}(|D_\tau|)$.
– $\mathsf{Garble}(U, \{\mathsf{digest}_\tau\}_{\tau\in[\mathcal{T}]})$ outputs a garbled program $\widehat{U}$ tied with hashes of the public tapes, and pairs of labels $\{L_{i,b}\}_{i,b}$. It runs in time $\mathrm{poly}(\lambda)$ (U has constant-size).
– $\mathsf{Eval}^{D_1,\ldots,D_\mathcal{T}}(U, \{\mathsf{digest}_\tau\}_{\tau\in[\mathcal{T}]}, \widehat{U}, L_k)$ returns the (long) output of RAM computation $U^{D_1,\ldots,D_\mathcal{T}}(k)$ if $L_k = \{L_{i,k_i}\}_i$. It runs in time $(T + \sum_\tau |D_\tau|)\,\mathrm{poly}(\lambda)$.
Security guarantees that for two computations $U^{D_1,\ldots,D_\mathcal{T}}(k_0)$ and $U^{D_1,\ldots,D_\mathcal{T}}(k_1)$ with different inputs but identical outputs and running time, the distributions of $(\widehat{U}, \{\mathsf{digest}_\tau\}_{\tau\in[\mathcal{T}]}, L_{k_0})$ and $(\widehat{U}, \{\mathsf{digest}_\tau\}_{\tau\in[\mathcal{T}]}, L_{k_1})$ are indistinguishable. This holds when the tapes $\{D_i\}$ are chosen adaptively dependent on the hashes of previously chosen tapes, before the program U and inputs k_0, k_1 are chosen.

Fig. 1. Our notion of succinct GRAM.

Our Construction of Succinct GRAM. One approach towards constructing succinct GRAM for TM or RAM is starting from a non-succinct GRAM for TM or RAM where the size of the garble program scales with the worst-case time complexity of the TM/RAM U, into one that is succinct. First introduced in [13], this approach uses iO to obfuscate a circuit that on input an index t, outputs the t'th component in the non-succinct GRAM. If every component of the non-succinct GRAM can be locally generated using a small circuit of size $\mathrm{poly}(|U|, \lambda)$, then the obfuscated circuit also has size $\mathrm{poly}(|U|, \lambda)$ and can be viewed as the succinct garbled program. To prove security, [13] identified that the non-succinct garbling scheme must satisfy another property, articulated later by [7], called *local simulation*. Informally, it requires that the non-succinct garbled scheme is proven secure via a sequence of hybrids, where components of every hybrid garbled program can be locally generated using a small circuit, and in neighboring hybrids, only a few components changes. Beyond succinct garbling, local simulation has also found application in achieving adaptive security [12] of garbling schemes. A sequence of works developed local simulation strategies

for garbled circuit [29,36], TM [7,30], and RAM [27]. Most notably, the work by Garg and Srinivasan [29] introduced a beautiful pebbling technique for realizing local simulation.

Our construction of succinct GRAM proceeds in steps. First, we use the Garg–Srinivasan [29] pebbling technique to obtain a non-succinct GRAM that has a local simulation proof for, however, weak indistinguishability security called fixed memory security. Indistinguishability only holds when the two RAM computations have not only identical outputs and running time, but also identical memory access pattern and content. Then by the same approach of [7, 13,30], we turn it into a succinct GRAM, still with only *fixed memory security*, relying on iO for polynomial-size domain which is implied by polynomially-hard FE for circuits. Many details need to be ironed out in order to realize our new notion of succinct GRAM. For example, prior works [7,29,30] deal with single-bit output RAM and can build intermediate security hybrid where the suffix (i.e., the last certain number of steps) of a computation is simulated by hardwiring the single-bit output. In contrast, we directly garble RAM with arbitrarily long outputs. Hardwiring the long output would compromise the local simulation property (since the hybrid garbled program can no longer be locally generated by a small circuit). To avoid this, we build a hybrid GRAM that runs with one input k_0 in the prefix of the computation and with another input k_1 in the suffix (recall that these two inputs produce identical memory). This ensures that the output is always correctly computed, while keeping local simulation. Similar hybrids appeared in [27] for different reasons.

Finally, we lift *fixed memory security* to full security using punctured PRF and ORAM to protect the memory content and access pattern. One issue is that in our succinct GRAM, the public input tapes $D_1, \cdots, D_{\mathcal{T}}$ are not encoded using ORAM, and evaluation is given random access to them in the clear. Yet, to ensure security, evaluation must access these tapes in an oblivious way, independent of the input k_0 or k_1. To solve this issue, we consider a modified RAM program U', which has random access to $D_1, \cdots, D_{\mathcal{T}}$ and additionally a work-tape that contains an ORAM storage that initially contains no elements. The program U' starts with linearly scanning every tape D_τ and inserting every element into the ORAM storage. Only after that, it emulates the execution of the original RAM program U; every time U read from/write to a location in tape D_τ, U' accesses the corresponding location on its work-tape through ORAM, which hides the access pattern of U. The intuition is that since the access pattern of U' is independent of the input, it suffices to garble it using GRAM with fixed memory security. Clearly, the running time of U' scales linearly with the total length of all tapes $\sum_\tau |D_\tau|$. This is why the evaluation time of our succinct GRAM is linear in $\sum_\tau |D_\tau|$. Nevertheless, our lower bound shows that this dependency is hard to circumvent. Lastly, we mention that to prove security, one must ensure that the use of ORAM does not hurt local simulation. Fortunately, the techniques by [20] provide a solution.

1.3 Related Works

Our new construction significantly improve upon the efficiency of prior FE and ABE schemes. The state-of-the-art is summarized in Tables 1 and 2. Below, we compare with prior works in more detail.

FE for Circuits. The first construction of collusion-resistant FE for polynomial-size circuits is by [26] and based on $i\mathcal{O}$, which in turn relies on subexponential hardness. Later works [28,41,42,44] improved the assumption from $i\mathcal{O}$ to 1-key FE with sublinearly compact ciphertext, $|\mathsf{ct}(y)| = \mathrm{poly}(|y|)|T_f|^{1-\varepsilon}$, where ε is a positive constant and T_f is the circuit complexity of f. The latter has been recently constructed by [39,40] from the polynomial hardness of three well-studied assumptions. However, these circuit-FE schemes have polynomial efficiency. The only exception is the recent construction by [35], which has rate-1 ciphertext, i.e., $|\mathsf{ct}(y)| = |y| + \mathrm{O}(1)$, but still large secret keys $|\mathsf{sk}_f| = \mathrm{poly}(T_f)$.

FE for Turing Machines. Several works constructed FE for Turning machines with arbitrary-length inputs, first from the assumption of iO [8], then from FE for circuits [1], and more recently from 1-key sublinearly compact FE [42] (which implies collusion-resistant FE for circuits). The construction of [8] relies on a 1-key 1-ciphertext secret-key FE for TM, which is essentially a succinct garbling scheme for TM with indistinguishability security. They constructed it by modifying the succinct garbling for TM of [43]. Later, the works of [7,30] improved and simplified the construction of succinct garbling for TM. The work of [42] improved the assumption to the existence of 1-key sublinearly succinct FE, and showed that their garbling scheme can be combined with [8] to obtain FE for TM. On the other hand, the work of [1] presented an alternative direct approach to FE for TM from FE for circuits without going through succinct garbling for TM. Prior constructions of FE for TM [1,8,42] focus more on weakening the underlying assumptions, and only show polynomial efficiency. Examining their schemes, we conclude that they achieve efficiency listed in Table 1.

FE for Bounded-Input RAM. A line of research obtained *bounded-input* $i\mathcal{O}$ for Turning machines [6,30,43] and RAM [2,13,19–21,23]. Plugging these $i\mathcal{O}$ schemes into the construction of [26] yields *bounded-input* FE for TM and RAM. Unfortunately, these schemes are not full-fledged FE for TM or RAM for the following reason: Existing $i\mathcal{O}$ only handles bounded-input TM and RAM in the sense that the obfuscator needs to know the maximum input length $\max |y|$ to the TM/RAM f being obfuscated. (Constructing $i\mathcal{O}$ for unbounded input TM/RAM remains a major open question.) Plugging them into [26] yields schemes where the key generation algorithm needs to know the maximum input length $\max |y|$, despite that the TM/RAM f could process arbitrarily long inputs. Such FE is said to have *bounded input*. In terms of efficiency, the secret key contains an obfuscated program of size $\mathrm{poly}(|f|, \max |y|)$ when using the RAM-$i\mathcal{O}$ of [20,21], and $\mathrm{poly}(|f|, \max |y|, S)$, where S is the space complexity of f when using the RAM-$i\mathcal{O}$ of [13].

In summary, our construction gives the first full-fledged (PH-)FE scheme for RAM computation with arbitrarily long inputs and outputs, significantly improves the efficiency of prior FE schemes, and matches newly proven lower bounds.

ABE for Circuits and Turing Machines. Since FE implies ABE, the aforementioned FE schemes immediately imply ABE with the same level of efficiency. The literature on ABE focuses on constructing ABE from weaker assumptions, and achieving better efficiency, among others. The celebrated works of [14,32] showed that ABE for *bounded-depth* circuits can be constructed from the learning With errors (LWE) assumption. Parameters of these schemes however depend polynomially on the maximum depth d of the circuits supported, namely, $|\mathsf{mpk}| = \mathrm{poly}(d)$, $|\mathsf{sk}_f| = \mathrm{poly}(d)$, $|\mathsf{ct}(x, m)| = \mathrm{poly}(d)|x|$, and the decryption time is $T_{\mathsf{Dec}} = \mathrm{poly}(d)T$. A recent work [45] improved it to obtain constant-size keys while keeping the sizes of master public key and ciphertext intact, but at the cost of additionally relying on the generic (pairing) group model (GGM). ABE for low-depth computation such as NC^1 or (arithmetic) branching programs can be constructed using pairing groups, where several schemes have either constant-size keys *or* constant-size ciphertexts, but never both [9–11,46,53–55].

The work of [31] constructed ABE for Turing machines and RAM with constant-size secret keys $|\mathsf{sk}_f| = \mathrm{O}(1)$, but still large ciphertexts $|\mathsf{ct}_x| = \mathrm{poly}(|x|)$. Their scheme uses SNARK and differing-input $i\mathcal{O}$, which cannot be based on falsifiable assumptions. Another work [4] tries to construct ABE for RAM from LWE, at the cost of making the master public key, secret keys, and ciphertexts all grow polynomially with the maximum running time of the RAM supported, i.e., it is an ABE for bounded-time RAM.

In summary, we give the first ABE for RAM from falsifiable assumptions, simultaneously having constant-size secret keys, constant-size ciphertexts, and the best-possible decryption time matching the known lower bound [48] under the constraint of having constant-size keys and ciphertexts.

Concurrent and Independent Work on FE for RAM. Concurrently and independently of our work, the recent work by Ananth, Chung, Fan, and Qian (ACFQ) [3] also considers the question of FE for RAM. Despite an apparent overlap between both these works, there are many differences. The two works start with different motivations. Our goal is to understand the optimal succinctness and efficiency of PHFE, both constructively and from a lower-bound perspective, whereas ACFQ aims to construct FE for RAM with optimal decryption time $T_{\mathsf{Dec}} = \mathrm{O}(T)$. Consequently, the two works obtain mostly complementary results.

First, we prove unconditional trade-offs between the sizes of secret keys/ciphertexts and decryption time; it shows that no PHFE can have both optimal succinctness and optimal decryption time. We then construct PHFE for RAM with (nearly) optimal succinctness, while minimizing the decryption time to the best-possible matching our lower bounds. ACFQ, on the other hand,

constructs FE for RAM with optimal decryption time. (They did not attempt to simultaneously minimizing the sizes of secret keys and ciphertexts.)

On the common front, both works show that any (PH-)FE scheme for RAM with optimal decryption time implies SK-DE-PIR. We regard this as a barrier to optimal efficiency due to lack of DE-PIR schemes from well-studied assumptions, whereas in ACFQ, public key version of DE-PIR (PK-DE-PIR) is used as a building block to realizing such PHFE. As a result their scheme relies on ideal obfuscation and a new assumption of permuted puzzles inherited from current candidate PK-DE-PIR, whereas our storage optimal PHFE scheme is based on circuit-FE, which is necessary and can in turn be based on well-studied assumptions.

There are two other major differences in the schemes: Our scheme handles arbitrarily long output, where as ACFQ consider single-bit output. To handle long output, they proposes to generate a separate key for computing each output bit, meaning that the key-size grows linearly with the output length, which could be as long as the running time in many scenarios. Moreover, our scheme achieves adaptive security, whereas that ACFQ scheme is only selectively secure.

In terms of techniques, both works demonstrate that the main bottleneck towards (PH-)FE for RAM is that existing notions of succinct GRAM are insufficient — it needs GRAM with reusable tape encoding. The two works develop different techniques to achieve this: Our construction lets an GRAM instance build fresh ORAM storage at the beginning of every evaluation and hence ORAM is never reused, whereas ACFQ uses PK-DE-PIR which is essentially a reusable ORAM.

2 Preliminaries

We present our formulations of laconic garbled RAM and partially hiding functional encryption, essential for considering the optimal succinctness and efficiency and the lower bounds. The details can be found in the full version [37].

Multi-Tape RAM. We consider $\mathcal{T}$-tape RAM for natural number $\mathcal{T}$. Such a machine has $\mathcal{T}$ read-only input tapes and one read/write working tape. Each input tape consists of multiple ℓ_{cell}-bit *cells* indexed by ℓ_{addr}-bit *addresses*. For the working tape, the lengths are ℓ_{CELL} and ℓ_{ADDR}. The machine is also given an ℓ_{in}-bit (short) input that remains constant during one execution, and it maintains an ℓ_{st}-bit internal state. At each step, the machine could produce an optional output bit. We denote an execution of M with input tape contents $D_1, \ldots, D_{\mathcal{T}}$ and short input w by $M^{D_1,\ldots,D_{\mathcal{T}}}(w)$, and write $\mathsf{time}(M, D_1, \ldots, D_{\mathcal{T}}, w)$ and $\mathsf{outS}(M, D_1, \ldots, D_{\mathcal{T}}, w)$ for its running time and its output sequence (a sequence of elements in $\{\bot, 0, 1\}$).

2.1 Laconic Garbled RAM

Our notion of garbling RAM laconically involves two steps. First, a reusable short digest is created for each input tape. The digest has length independent of

that of the tape itself and must be computable in linear time. Second, the RAM and the short digests are put together to produce a garbled program and the labels. This procedure runs in time poly-logarithmic in the RAM running time. Given a garbled program and one set of labels (selected by the bits of the short input), the evaluation procedure computes the output sequence in time linear in the RAM running time.

We consider indistinguishability-based security for the short input. The input tape contents can be chosen adaptively, but the short input cannot depend on the garbled program (i.e., selectiveness).

Definition 1 (LGRAM)**.** *Let $\mathcal{T}$ be a natural number. A* laconic garbling scheme *for $\mathcal{T}$-tape RAM consists of three algorithms:*

- $\mathsf{Compress}(1^\lambda, 1^{\ell_{\mathrm{cell}}}, 1^{\ell_{\mathrm{addr}}}, \tau, D_\tau)$ *takes as input a cell length ℓ_{cell}, an address length ℓ_{addr}, an input tape index $\tau \in [\mathcal{T}]$, and the content of that input tape, $D_\tau \in (\{0,1\}^{\ell_{\mathrm{cell}}})^{\leq 2^{\ell_{\mathrm{addr}}}}$. It outputs a short digest digest_τ. The algorithm runs in time $|D_\tau|\,\mathrm{poly}(\lambda, \ell_{\mathrm{cell}}, \ell_{\mathrm{addr}})$ and its output length is $\mathrm{poly}(\lambda, \ell_{\mathrm{cell}}, \ell_{\mathrm{addr}})$.*
- $\mathsf{Garble}(1^\lambda, T_{\max}, M, \{\mathsf{digest}_\tau\}_{\tau\in[\mathcal{T}]})$ *takes as input a time bound $T_{\max} \in \mathbb{N}_+$, a $\mathcal{T}$-tape RAM M, and $\mathcal{T}$ input tape digests. It outputs a garbled program $\widehat{M}$ and ℓ_{in} pairs of labels $\{L_{i,b}\}_{i\in[\ell_{\mathrm{in}}],b\in\{0,1\}}$ in polynomial time.*
- $\mathsf{Eval}^{D_1,\ldots,D_\mathcal{T}}(1^\lambda, T_{\max}, M, \{\mathsf{digest}_\tau\}_{\tau\in[\mathcal{T}]}, \widehat{M}, \{L_i\}_{i\in[\ell_{\mathrm{in}}]})$ *takes as input $T_{\max}$, M, the input tape digests, $\widehat{M}$, and one set of labels. Given random access to the input tapes, it is supposed to compute the output sequence. The algorithm runs in time*

$$\left(\min\left\{T_{\max}, \mathsf{time}(M, D_1, \ldots, D_\mathcal{T}, w)\right\} + \sum_{i=1}^{\mathcal{T}} |D_\tau|\right) \mathrm{poly}(\lambda, |M|, \log T_{\max}),$$

where w is the short input corresponding to the labels.

The scheme must be correct*, i.e., for all $\lambda, \ell_{\mathrm{in}} \in \mathbb{N}$, $\ell_{\mathrm{cell}}, \ell_{\mathrm{addr}}, T_{\max} \in \mathbb{N}_+$, input tape contents $D_1, \ldots, D_\mathcal{T} \in (\{0,1\}^{\ell_{\mathrm{cell}}})^{\leq 2^{\ell_{\mathrm{addr}}}}$, short input $w \in \{0,1\}^{\ell_{\mathrm{in}}}$, $\mathcal{T}$-tape RAM M such that $M^{D_1,\ldots,D_\mathcal{T}}(w)$ halts in time at most $T_{\max}$,*

$$\Pr\left[\begin{array}{r} \mathsf{digest}_\tau \xleftarrow{\$} \mathsf{Compress}(1^\lambda, 1^{\ell_{\mathrm{cell}}}, 1^{\ell_{\mathrm{addr}}}, \tau, D_\tau)\ \forall \tau \in [\mathcal{T}] \\ (\widehat{M}, \{L_{i,b}\}_{i\in[\ell_{\mathrm{in}}],b\in\{0,1\}}) \xleftarrow{\$} \mathsf{Garble}(1^\lambda, T_{\max}, M, \{\mathsf{digest}_\tau\}_{\tau\in[\mathcal{T}]}) \\ \mathsf{Eval}^{D_1,\ldots,D_\mathcal{T}}(1^\lambda, T_{\max}, M, \{\mathsf{digest}_\tau\}_{\tau\in[\mathcal{T}]}, \widehat{M}, \{L_{i,w[i]}\}_{i\in[\ell_{\mathrm{in}}]}) \\ : \qquad = \mathsf{outS}(M, D_1, \ldots, D_\mathcal{T}, w) \end{array}\right] = 1.$$

Remark 1 (unboundedness)*.* Our notion of LGRAM is *unbounded*, i.e., it is not necessary to know a polynomial upper bound of the instance running time upon garbling. By choosing an exponentially large $T_{\max}$, one garbling works for all polynomial-time computation. In contrast is a *bounded* scheme for all polynomial-time computation, where $T_{\max}$ can be *any* polynomial, but it *must* be a polynomial, hence every garbling is restricted to *some* polynomial time bound upon creation. Unboundedness is reflected in both efficiency and security (below), where $T_{\max}$ is written in binary.

Definition 2 (LGRAM security)**.** *An LGRAM scheme (Definition 1) is* (tape-adaptively, indistinguishability-based) secure *if* $\mathsf{Exp}^0_{\mathrm{LGRAM}} \approx \mathsf{Exp}^1_{\mathrm{LGRAM}}$*, where* $\mathsf{Exp}^\beta_{\mathrm{LGRAM}}(1^\lambda, \mathcal{A})$ *proceeds as follows:*

- **Setup.** Launch $\mathcal{A}(1^\lambda)$ and receive $(1^{\ell_{\mathrm{cell}}}, 1^{\ell_{\mathrm{addr}}})$ from it.
- **Tape Choices.** Repeat this for $\mathcal{T}$ rounds. In each round, $\mathcal{A}$ chooses $\tau \in [\mathcal{T}]$ and $D_\tau \in (\{0,1\}^{\ell_{\mathrm{cell}}})^{\leq 2^{\ell_{\mathrm{addr}}}}$. Upon receiving such choice, run
$$\mathsf{digest}_\tau \xleftarrow{\$} \mathsf{Compress}(1^\lambda, 1^{\ell_{\mathrm{cell}}}, 1^{\ell_{\mathrm{addr}}}, \tau, D_\tau)$$
and send digest_τ to $\mathcal{A}$.
- **Challenge.** $\mathcal{A}$ chooses an instance running time bound $1^{\overline{T}}$ (in unary), a time bound $T_{\max}$ (in binary), a $\mathcal{T}$-tape RAM M, and two inputs (w_0, w_1). Run
$$(\widehat{M}, \{L_{i,b}\}_{i\in[\ell_{\mathrm{in}}],b\in\{0,1\}}) \xleftarrow{\$} \mathsf{Garble}(1^\lambda, T_{\max}, M, \{\mathsf{digest}_\tau\}_{\tau\in[\mathcal{T}]})$$
and send $(\widehat{M}, \{L_{i,w_\beta[i]}\}_{i\in[\ell_{\mathrm{in}}]})$ to $\mathcal{A}$.
- **Guess.** $\mathcal{A}$ outputs a bit β'. The output of the experiment is β' if all of the following conditions hold:
  - The τ's in all rounds of **Tape Choices** are distinct.
  - Both $M^{D_1,\ldots,D_\mathcal{T}}(w_0)$ and $M^{D_1,\ldots,D_\mathcal{T}}(w_1)$ halt in time $T \leq \overline{T} \leq T_{\max}$ with identical *output sequences* $\mathsf{outS}(\cdots)$.

  Otherwise, the output is set to 0.

Remark 2 (polynomial security)*.* Although $T_{\max}$ can be exponentially large, we only require security for polynomially large *instance* running time, which is captured by the requirement that the adversary must produce $1^{\overline{T}}$, an upper bound of the instance running time in unary.

2.2 Partially Hiding Functional Encryption and FE for Circuits

We define partially hiding functional encryption with respect to functionality

$$\varphi : F \times X \times Y \to \{\bot\} \cup (\mathbb{N}_+ \times Z),$$

where F, X, Y, Z are the sets of function description, public input, private input, and output, respectively. Each key is associated with some $f \in F$, and each ciphertext encrypts some private input $y \in Y$ and is tied to some public input $x \in X$. The decryptor should be able to recover z if $\varphi(f, x, y) = (T, z)$, in which case T is the time to compute z from f, x, y in the clear and serves as a baseline for decryption efficiency. For security, we only consider f, x, y for which T is polynomially bounded. On the other hand, if $\varphi(f, x, y) = \bot$, we require neither correctness nor security. This can be used to exclude non-halting computation.

Definition 3 (PHFE)**.** *Let* $\Phi = \{\Phi_\lambda\}_{\lambda\in\mathbb{N}}$ *be a sequence of functionality families*

$$\text{with} \quad \varphi : F_\varphi \times X_\varphi \times Y_\varphi \to \{\bot\} \cup (\mathbb{N}_+ \times Z_\varphi) \quad \text{for each } \varphi \in \Phi_\lambda.$$

A partially hiding functional encryption scheme *for* Φ *consists of four algorithms, with efficiency defined in Definition 4:*

– $\mathsf{Setup}(1^\lambda, \varphi)$ *takes a functionality* $\varphi \in \Phi_\lambda$ *as input, and outputs a pair of master public/secret keys* $(\mathsf{mpk}, \mathsf{msk})$.
– $\mathsf{KeyGen}(1^\lambda, \mathsf{msk}, f)$ *takes as input* msk *and a function description* $f \in F_\varphi$. *It outputs a secret key* sk_f *for* f.
– $\mathsf{Enc}(1^\lambda, \mathsf{mpk}, x, y)$ *takes as input* mpk, *a public input* $x \in X_\varphi$, *and a private input* $y \in Y_\varphi$. *It outputs a ciphertext* ct_x *of* y *tied to* x.
– $\mathsf{Dec}^{f,x,\mathsf{sk}_f,\mathsf{ct}_x}(1^\lambda, \mathsf{mpk})$ *takes* mpk *as input and is given random access to* $f, x, \mathsf{sk}_f, \mathsf{ct}_x$. *It is supposed to compute* z *in* $\varphi(f, x, y) = (T, z)$ *efficiently.*

The scheme must be correct, *i.e., for all* $\lambda \in \mathbb{N}$, $\varphi \in \Phi_\lambda$, $f \in F_\varphi$, $x \in X_\varphi$, $y \in Y_\varphi$ *such that* $\varphi(f, x, y) = (T, z) \neq \bot$, *it holds that*

$$\Pr\left[\begin{array}{lll}(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \varphi) & & \\ \mathsf{sk}_f \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda, \mathsf{msk}, f) & : & \mathsf{Dec}^{f,x,\mathsf{sk}_f,\mathsf{ct}_x}(1^\lambda, \mathsf{mpk}) = z \\ \mathsf{ct}_x \xleftarrow{\$} \mathsf{Enc}(1^\lambda, \mathsf{mpk}, x, y) & & \end{array}\right] = 1.$$

Definition 4 (PHFE efficiency)**.** *The basic efficiency requirements for a PHFE scheme (Definition 3) are as follows:*

– $\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}$ *are polynomial-time.*
– Dec *runs in time* $\mathrm{poly}(\lambda, |\varphi|, |f|, |x|, |y|, T)$ *if* $\varphi(f, x, y) = (T, z) \neq \bot$.

The following time-efficiency properties are considered:

– *It has* linear-time KeyGen *[resp.* Enc*] if* KeyGen *[resp.* Enc*] runs in time* $|f|\,\mathrm{poly}(\lambda, |\varphi|)$ *[resp.* $(|x| + |y|)\,\mathrm{poly}(\lambda, |\varphi|)$*];*
– *It has* $(T^{e_T} + |f|^{e_f} + |x|^{e_x} + |y|^{e_y})$-time Dec *(for constants* e_T, e_f, e_x, e_y*) if* Dec *runs in time*

$$(T^{e_T} + |f|^{e_f} + |x|^{e_x} + |y|^{e_y})\,\mathrm{poly}(\lambda, |\varphi|),$$

where $\varphi(M, f, x, y) = (T, z) \neq \bot$. *Furthermore, the scheme has* f-fast *[resp.* x-fast, y-fast*]* Dec *if it has* $(T^{e_T} + |f|^{e_f} + |x|^{e_x} + |y|^{e_y})$*-time* Dec *with* $e_f = 0$ *[resp.* $e_x = 0$, $e_y = 0$*].*

The following size-efficiency properties are considered:

– *It is* f-succinct *if* $|\mathsf{sk}_f| = \mathrm{poly}(\lambda, |\varphi|)$, *independent of* $|f|$.
– *It is* x-succinct *if* $|\mathsf{ct}_x| = \mathrm{poly}(\lambda, |\varphi|, |y|)$, *independent of* $|x|$.
– *It has* rate-c ciphertext *for some constant* c *if* $|\mathsf{ct}_x| = c|y| + \mathrm{poly}(\lambda, |\varphi|)$.

Security. We consider adaptive IND-CPA for polynomially bounded T:

Definition 5 (PHFE security)**.** *A PHFE scheme (Definition 3) is* (adaptively IND-CPA) secure *if* $\mathsf{Exp}^0_{\mathrm{PHFE}} \approx \mathsf{Exp}^1_{\mathrm{PHFE}}$, *where* $\mathsf{Exp}^\beta_{\mathrm{PHFE}}(1^\lambda, \mathcal{A})$ *proceeds as follows:*

– **Setup.** Launch $\mathcal{A}(1^\lambda)$ and receive from it some $\varphi \in \Phi_\lambda$ and $1^{\overline{T}}$. Run

$$(\mathsf{mpk}, \mathsf{msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda, \varphi)$$

and send mpk to $\mathcal{A}$.

– **Query I.** Repeat the following for arbitrarily many rounds determined by $\mathcal{A}$. In each round, $\mathcal{A}$ submits some $f_q \in F_\varphi$. Upon receiving such query, run

$$\mathsf{sk}_q \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda, \mathsf{msk}, f_q)$$

and send sk_q to $\mathcal{A}$.
– **Challenge.** $\mathcal{A}$ submits $x \in X_\varphi$ and $y_0, y_1 \in Y_\varphi$. Upon the challenge, run

$$\mathsf{ct} \xleftarrow{\$} \mathsf{Enc}(1^\lambda, \mathsf{mpk}, x, y_\beta)$$

and send ct to $\mathcal{A}$.
– **Query II.** Same as **Query I**.
– **Guess.** $\mathcal{A}$ outputs a bit β'. The outcome of the experiment is β' if

$$\begin{aligned} &|y_0| = |y_1|, \\ &\text{and } \varphi(f_q, x, y_0) = \varphi(f_q, x, y_1) = (T_q, z_q) \neq \bot && \text{for all } q, \\ &\text{and } T_q \leq \overline{T} && \text{for all } q. \end{aligned}$$

Otherwise, the outcome is set to 0.

FE for Circuits. As an example, we show how to instantaite Definition 3 into FE for circuits, a building block of our construction (see the full version [37]).

Definition 6 (FE for circuits)**.** *A* functional encryption scheme for circuits *is a PHFE scheme (Definition 3) for*

$$\begin{aligned} \Phi &= \{\Phi_\lambda\}_{\lambda\in\mathbb{N}}, \qquad \Phi_\lambda = \{\varphi_{\ell,s}\}_{\ell,s\in\mathbb{N}_+}, \\ \varphi_{\ell,s} &: F_{\ell,s} \times X \times Y_\ell \to \{\bot\} \cup (\mathbb{N}_+ \times Z), \\ F_{\ell,s} &= \{\text{ circuits of input length } \ell \text{ and size at most } s\}, \\ X &= \{\bot\}, \qquad Y_\ell = \{0,1\}^\ell, \qquad Z = \{0,1\}^*, \\ &\varphi_{\ell,s}(f, \bot, y) = \big(1, f(y)\big), \end{aligned}$$

where the functionality $\varphi_{\ell,s}$ is represented by $(1^\ell, 1^s)$.

Remark 3. The first output of $\varphi_{\ell,s}$ is just a placeholder value and all efficiency parameters (Definition 4) are always allowed arbitrary polynomial dependency on λ, ℓ, s by our choice of representing $\varphi_{\ell,s}$ by $(1^\ell, 1^s)$. This is intended as we use FE for circuits as a building block and do not wish to start with too strong a scheme.

2.3 Universal RAM and PHFE for RAM

In this section, we define PHFE for RAM after explaining some rationales behind certain subtleties in our formulation.

To obtain PHFE for RAM, we will employ the standard transformation [51] of using FE for circuits to compute LGRAM. However, in LGRAM (Definition 1),

the running time of Garble depends on the machine size. This dependency is inherited by every efficiency parameter of the resultant PHFE for RAM if we associate each key with a RAM. To get rid of this dependency, we fix some universal RAM U of size $\mathrm{poly}(\lambda)$[8] upon setting up the scheme, and associate with each key a piece of code interpreted by U.

The other issue is that LGRAM puts an upper bound on the running time and its incorrectness in case of exceeding the time limit is propagated to the PHFE scheme. We avoid it[9] by defining $\varphi = \bot$ if the running time exceeds some super-polynomial value prescribed upon set-up.

The above explains the intended usage of PHFE for RAM, yet we define it for general machines. Moreover, as an intermediate primitive, we will first consider PHFE for RAM with *bounded private input*, where the private input is simply the short input to the machine:

Definition 7 (PHFE for RAM with bounded private input)**.** *A* PHFE scheme for RAM with bounded private input *is a PHFE scheme (Definition 3) for*

$$\Phi = \{\Phi_\lambda\}_{\lambda\in\mathbb{N}}, \qquad \Phi_\lambda = \{\varphi_{M,T_{\max}}\}_{M \text{ is a 2-tape RAM and } T_{\max}\in\mathbb{N}_+},$$
$$\varphi_{M,T_{\max}} : F_M \times X_M \times Y_M \to \{\bot\} \cup (\mathbb{N}_+ \times Z),$$
$$F_M = X_M = (\{0,1\}^{\ell_{\mathrm{cell}}})^{\le 2^{\ell_{\mathrm{addr}}}}, \qquad Y_M = \{0,1\}^{\ell_{\mathrm{in}}}, \qquad Z = \{\bot,0,1\}^*,$$
$$\varphi_{M,T_{\max}}(f,x,y) = \begin{cases} \big(T, \mathsf{outS}(M,f,x,y)\big), & \text{if } \mathsf{time}(M,f,x,y) = T \le T_{\max}; \\ \bot, & \text{otherwise;} \end{cases}$$

where $\varphi_{M,T_{\max}}$ *is represented by* $(M, T_{\max})$.

In a full-fledged PHFE for RAM, the machine has no short input, and the private input is encoded on a tape:

Definition 8 (full-fledged PHFE for RAM)**.** *A* full-fledged PHFE scheme for RAM *is a PHFE scheme (Definition 3) for*

$$\Phi = \{\Phi_\lambda\}_{\lambda\in\mathbb{N}}, \qquad \Phi_\lambda = \{\varphi_{M,T_{\max}}\}_{M \text{ is a 2-tape RAM with } \ell_{\mathrm{in}}=0, \text{ and } T_{\max}\in\mathbb{N}_+},$$
$$\varphi_{M,T_{\max}} : F_M \times X_M \times Y_M \to \{\bot\} \cup (\mathbb{N}_+ \times Z),$$
$$F_M = X_M = Y_M = (\{0,1\}^{\ell_{\mathrm{cell}}})^{\le 2^{\ell_{\mathrm{addr}}}}, \qquad Z = \{\bot,0,1\}^*,$$
$$\varphi_{M,T_{\max}}(f,x,y) = \begin{cases} \big(T, \mathsf{outS}(M,f,x\|y,\varepsilon)\big), \\ \quad \text{if } |x|+|y| \le 2^{\ell_{\mathrm{addr}}} \text{ and } \mathsf{time}(M,f,x\|y,\varepsilon) = T \le T_{\max}; \\ \bot, \qquad \text{otherwise;} \end{cases}$$

where ε *is the empty string and* $\varphi_{M,T_{\max}}$ *is represented by* $(M, T_{\max})$.

[8] U is not the same RAM across different values of λ — its input address length should be $\omega(\log\lambda)$ to accommodate all polynomially long input.

[9] An alternative solution is to blatantly reveal everything if the running time is too large so that correctness in that case can be implemented by executing the machine in the clear. Security is not affected because the adversary is not allowed to choose keys and ciphertexts with super-polynomial instance running time. However, non-halting computation still needs to be handled separately.

Remark 4 (unbounded scheme and polynomial security). When Definitions 3 and 5 are instantiated into PHFE for RAM (Definitions 7 and 8), the scheme is *unbounded*, meaning that $T_{\max}$ can be exponentially large, yet security only holds for polynomially bounded instance running time. This is similar to the case in Sect. 2.1.

3 Efficiency Trade-Offs of PHFE for RAM

We present the unconditional lower bounds. Additional contents about technique barriers can be found in the full version [37].

3.1 Contention Between Storage Overhead and Decryption Time

In this section, we show that it is impossible to achieve

$$|\mathsf{sk}| = \mathrm{O}(|f|^{\alpha}) \quad \text{and} \quad T_{\mathsf{Dec}} = \mathrm{O}(T + |f|^{\beta} + |x| + |y|)$$

simultaneously for a secure PHFE for RAM when $\alpha, \beta < 1$, where polynomial factors in the security parameter are ignored. This leaves us with two candidate optima:

- $\alpha = 0$ and $\beta = 1$ for succinct keys; or
- $\alpha = 1$ and $\beta = 0$ for f-fast decryption.

Similarly, it is impossible to achieve

$$|\mathsf{ct}| = \mathrm{O}(|x|^{\alpha})\,\mathrm{poly}(|y|) \quad \text{and} \quad T_{\mathsf{Dec}} = \mathrm{O}(T + |f| + |x|^{\beta} + |y|)$$

simultaneously if $\alpha, \beta < 1$, which implies a contention between succinct ciphertexts and x-fast decryption.

Formally, our theorems are slightly stronger than the discussion above:

Theorem 5 (contention of $|f|$-dependency between $|\mathsf{sk}|$ and T_{Dec}; ¶)**.** *For a secure full-fledged PHFE for RAM (Definitions 3, 5, and 8), if*

$$|\mathsf{sk}| \le |f|^{\alpha}(\lambda + |\varphi|)^{C} \quad \text{and} \quad T_{\mathsf{Dec}} \le (T + |f|^{\beta} + |y|)(\lambda + |\varphi| + |x|)^{C}$$

for infinitely many λ, where α, β, C are constants, then $\alpha \ge 1$ or $\beta \ge 1$.

Theorem 6 (contention of $|x|$-dependency between $|\mathsf{ct}|$ and T_{Dec})**.** *For a secure full-fledged PHFE for RAM (Definitions 3, 5, and 8), if*

$$|\mathsf{ct}| \le |x|^{\alpha}(\lambda + |\varphi| + |y|)^{C} \quad \text{and} \quad T_{\mathsf{Dec}} \le (T + |f| + |x|^{\beta})(\lambda + |\varphi| + |y|)^{C}$$

for infinitely many λ, where α, β, C are constants, then $\alpha \ge 1$ or $\beta \ge 1$.

We will only prove Theorem 5. The proof of Theorem 6 is similar.

Proof (Theorem 5). Let $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be a secure PHFE for RAM. Suppose for contradiction that $\alpha, \beta < 1 - 5\varepsilon$ for some $0 < \varepsilon \le \frac{1}{5}$. By enlarging C as needed, we could assume $|\varphi| \le \lambda^{C} - \lambda - 1$ for all sufficiently large λ, where

$$\varphi = (M_{\lambda}, 2^{\lambda}), \quad f = R \in \{0,1\}^{\le 2^{\lambda}}, \quad x = \bot,$$

$$y = \begin{cases} (I, w) = (i_1, w[1], \ldots, i_n, w[n]) \in ([2^\lambda] \times \{0,1\})^{\leq 2^\lambda}; \\ z = (\bot, z[1], \ldots, \bot, z[n]) \in (\{\bot\} \times \{0,1\})^{\leq 2^\lambda}; \end{cases}$$

$$M^{f,x\|y}() = \begin{cases} (R[i_1] \oplus w[1], \ldots, R[i_n] \oplus w[n]), & \text{if } y = (I, w); \\ (\quad z[1] \quad, \ldots, \quad z[n] \quad), & \text{if } y = z. \end{cases}$$

Under appropriate encoding and step circuit design, y has exactly n cells and M halts in exactly $(2n+1)$ steps.

We focus on the values of λ (hereafter, "λ with efficiency") such that

$$|\mathsf{sk}| \leq |f|^\alpha (\lambda + |\varphi|)^C \quad \text{and} \quad T_{\mathsf{Dec}} \leq (T + |f|^\beta + |y|)(\lambda + |\varphi| + |x|)^C$$

By setting

$$|R| = N = \lceil \lambda^{(C^2+1)/\varepsilon} \rceil, \qquad n = \lfloor N^{1-3\varepsilon} \rfloor,$$

we would have $n < N < 2^\lambda$ for sufficiently large λ. Consider the following adversary $\mathcal{A}$ (Definition 5):

- Upon launching, it computes φ, N, n defined above, sets up the PHFE scheme for φ, and submits 1^{2n+1} as the time bound.
- It samples $R \xleftarrow{\$} \{0,1\}^N$ and requests a key sk for $f = R$.
- It samples $w \xleftarrow{\$} \{0,1\}^n$ and a list I of n distinct random elements from $[N]$, sets

  $$z = (R[i_1] \oplus w[1], \ldots, R[i_n] \oplus w[n]).$$

  It challenges with

  $$x = \bot, \qquad y_0 = (I, w), \quad y_1 = z,$$

  and obtains a ciphertext ct encrypting either y_0 or y_1.
- It runs $\mathsf{Dec}^{f,x,\mathsf{sk},\mathsf{ct}}(\mathsf{mpk})$ and notes down the list L of indices into $R = f$ where it is read during decryption. $\mathcal{A}$ outputs 1 if and only if

  $$|L \cap I| > N^{1-4\varepsilon},$$

  where L and I are regarded as sets (unordered and deduplicated) for the intersection operation.

Clearly, $\mathcal{A}$ would be efficient and its challenge would satisfy the constraints of PHFE security for sufficiently large λ. We claim:

Claim 7 (¶). For sufficiently large λ with efficiency,

$$\Pr\left[\, |L \cap I| > N^{1-4\varepsilon} \text{ in } \mathsf{Exp}^0_{\mathrm{PHFE}} \,\right] \geq \frac{3}{4}.$$

Claim 8 (¶). For sufficiently large λ with efficiency,

$$\Pr\left[\, |L \cap I| > N^{1-4\varepsilon} \text{ in } \mathsf{Exp}^1_{\mathrm{PHFE}} \,\right] \leq \frac{1}{4}.$$

The two claims together would contradict the security of PHFE, as the advantage of $\mathcal{A}$ would be at least $\frac{1}{2}$ for infinitely many λ. Therefore, $\alpha \geq 1$ or $\beta \geq 1$. $\square$

To prove Claim 7, we need the following lemma about incompressibility of information:

Lemma 9 ([24]). *Suppose $E : S \times U \to V$ and $D : S \times V \to U$ are functions and $\mathcal{S}$ is a distribution over S, then*

$$|V| \geq |U| \cdot \Pr_{\substack{s \xleftarrow{\$} \mathcal{S} \\ u \xleftarrow{\$} U}} [D(s, E(s, u)) = u].$$

Proof (Claim 7). We use the PHFE scheme to compress a string u of length n. To encode, we embed u into a string R of length N at random locations (i.e., I) and generate a PHFE key for R. The encoding is the key plus some bits in R used during decryption. To decode, run the decryption algorithm. Lemma 9 will generate the following inequality equivalent to the desired one:

$$\Pr\left[\, |L \cap I| \leq \lfloor N^{1-4\varepsilon} \rfloor \text{ in } \mathsf{Exp}^0_{\mathrm{PHFE}} \,\right] \leq \frac{1}{4}.$$

Formally, let

$$\mathcal{S} = \left\{ \begin{pmatrix} \mathsf{mpk}, \mathsf{msk}, I, w, R', \\ r_{\mathsf{KeyGen}}, r_{\mathsf{Enc}}, r_{\mathsf{Dec}} \end{pmatrix} : \begin{aligned} (\mathsf{mpk}, \mathsf{msk}) &\xleftarrow{\$} \mathsf{Setup}(\varphi) \\ (I, w) &\text{ as how } \mathcal{A} \text{ samples it} \\ R'[i] &\xleftarrow{\$} \{0,1\} \text{ for } i \in [N] \setminus I \\ r_{\mathsf{KeyGen}}, r_{\mathsf{Enc}}, r_{\mathsf{Dec}} &\xleftarrow{\$} \text{algorithm randomness} \end{aligned} \right\},$$

$$U = \{0,1\}^n, \qquad V = \{0,1\}^{\lfloor N^{1-4\varepsilon} \rfloor} \times \{0,1\}^{\lfloor N^{1-4\varepsilon} \rfloor}.$$

The encoding procedure $E(s, u)$ works as follows:

- Parse $I = (i_1, \ldots, i_n)$ and set

$$R[i] = \begin{cases} R'[i], & \text{if } i \in [N] \setminus I; \\ u[j], & \text{if } i = i_j. \end{cases}$$

- Run

$$\begin{aligned} \mathsf{sk} &\leftarrow \mathsf{KeyGen}(\mathsf{msk}, R; r_{\mathsf{KeyGen}}), \\ \mathsf{ct} &\leftarrow \mathsf{Enc}(\mathsf{mpk}, \bot, (I, w); r_{\mathsf{Enc}}), \\ u \oplus w &\leftarrow \mathsf{Dec}^{R, \bot, \mathsf{sk}, \mathsf{ct}}(\mathsf{mpk}; r_{\mathsf{Dec}}), \end{aligned}$$

  and note down the list $L = (\ell_1, \ldots)$ of indices into R read by Dec.
- Output $v = (v_1, v_2)$ with $v_1, v_2 \in \{0,1\}^{\lfloor N^{1-4\varepsilon} \rfloor}$ and

$$v_1 = 0^{\lfloor N^{1-4\varepsilon} \rfloor - |\mathsf{sk}| - 1} 1 \| \mathsf{sk},$$

$$v_2[i] = \begin{cases} R[\ell_j], & \text{if } |\{\ell_1, \ldots, \ell_{j-1}\} \cap I| = i - 1 \text{ and } |\{\ell_1, \ldots, \ell_{j-1}, \ell_j\} \cap I| = i; \\ 0, & \text{if no such } j \text{ exists.} \end{cases}$$

Here, v_1 is a fixed-length encoding of sk and is indeed well-defined since

$$|\mathsf{sk}| \le |f|^{\alpha}(\lambda + |\varphi|)^C \le N^{1-5\varepsilon}\big(\lambda + (\lambda^C - \lambda - 1)\big)^C \le N^{1-5\varepsilon}\lambda^{C^2} < \lfloor N^{1-4\varepsilon} \rfloor - 1$$

for sufficiently large λ with efficiency. The string v_2 records, *sequentially*, the bits in R at each *distinct* index read by Dec that are part of u and not known from R', for at most $\lfloor N^{1-4\varepsilon} \rfloor$ bits.

The decoding procedure $D(s, v)$ works as follows:

- Run $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \bot, (I, w); r_{\mathsf{Enc}})$.
- Parse $v = (v_1, v_2)$ and recover sk from v_1 as specified in E.
- Initialize j, an index into v_2, by $j \leftarrow 0$, and initialize R by

$$R[i] = \begin{cases} R'[i], & \text{if } i \in [N] \setminus I; \\ \bot, & \text{if } i \in I. \end{cases}$$

  Run $z \leftarrow \mathsf{Dec}^{R,\bot,\mathsf{sk},\mathsf{ct}}(\mathsf{mpk}; r_{\mathsf{Dec}})$ with R filled on the fly. When Dec reads $R[i]$:
  - if $R[i] = \bot$ and $j < \lfloor N^{1-4\varepsilon} \rfloor$, then let $j \leftarrow j + 1$ and set $R[i] \leftarrow v_2[j]$;
  - if $R[i] = \bot$ and $j = \lfloor N^{1-4\varepsilon} \rfloor$, then abort by outputting 0^n;
  - otherwise, $R[i] \neq \bot$, then just proceed without aborting;

  and return $R[i]$ to Dec if not aborting.
- Output $z \oplus w$.

D will fill v_2 into the correct indices of R since the PHFE algorithms are derandomized with the same randomness as in E.

The sampling of s, u and the setting of R in $E(s, u)$ simulate $\mathcal{A}$ in $\mathsf{Exp}^0_{\mathrm{PHFE}}$. If s and u are such that $|L \cap I| \le \lfloor N^{1-4\varepsilon} \rfloor$ in $E(s, u)$, then D will successfully recover u. By Lemma 9,

$$\begin{aligned} & \Pr\left[\, |L \cap I| \le \lfloor N^{1-4\varepsilon} \rfloor \text{ in } \mathsf{Exp}^0_{\mathrm{PHFE}} \,\right] \\ = & \Pr_{\substack{s \xleftarrow{\$} \mathcal{S} \\ u \xleftarrow{\$} U}}\left[\, |L \cap I| \le \lfloor N^{1-4\varepsilon} \rfloor \text{ in } E(s, u) \,\right] \\ \le & \Pr_{\substack{s \xleftarrow{\$} \mathcal{S} \\ u \xleftarrow{\$} U}}[D(s, E(s, u)) = u] \le \frac{|V|}{|U|} = \frac{2^{2\lfloor N^{1-4\varepsilon} \rfloor}}{2^n} = 2^{2\lfloor N^{1-4\varepsilon} \rfloor - \lfloor N^{1-3\varepsilon} \rfloor} \le \frac{1}{4} \end{aligned}$$

for sufficiently large λ with efficiency. □

Proof (Claim 8). For sufficiently large λ with efficiency,

$$\begin{aligned} |L| \le T_{\mathsf{Dec}} &\le (T + |f|^{\beta} + |y|)(\lambda + |\varphi| + |x|)^C \\ &\le \big((2n+1) + N^{1-5\varepsilon} + n\big)\big(\lambda + (\lambda^C - \lambda - 1) + 1\big)^C \\ &\le \big(3N^{1-3\varepsilon} + N^{1-5\varepsilon} + 1\big)\lambda^{C^2} \le N^{1-2\varepsilon}. \end{aligned}$$

In $\mathsf{Exp}^1_{\mathrm{PHFE}}$, the input to Dec is independent of I, which only symbolically appears in ct as

$$y_1 = z = (R[i_1] \oplus w[1], \ldots, R[i_n] \oplus w[n])$$

and is fully hidden by the one-time pad w. Therefore, the list of indices into R read by Dec (i.e., L) is independent of I. Conditioned on L, the intersection size $|L \cap I|$ follows a hypergeometric distribution. By the law of total expectation,

$$\mathbb{E}\big[\,|L \cap I|\,\big] = \mathbb{E}\Big[\mathbb{E}\big[\,|L \cap I| \;\big|\; L\,\big]\Big] = \mathbb{E}\left[\frac{|I| \cdot |L|}{N}\right] \leq \frac{N^{1-3\varepsilon} \cdot N^{1-2\varepsilon}}{N} = N^{1-5\varepsilon}$$

for sufficiently large λ with efficiency, which implies, by Markov's inequality,

$$\Pr[\,|L \cap I| > N^{1-4\varepsilon} \text{ in } \mathsf{Exp}^1_{\mathrm{PHFE}}\,] \leq \frac{\mathbb{E}\big[\,|L \cap I|\,\big]}{N^{1-4\varepsilon}} \leq \frac{N^{1-5\varepsilon}}{N^{1-4\varepsilon}} = N^{-\varepsilon} \leq \frac{1}{4}. \qquad \square$$

Acknowledgement. Aayush Jain was supported by departmental funds from CMU Computer Science Department and a gift from CyLab. Huijia Lin and Ji Luo were supported by NSF grants CNS-1936825 (CAREER), CNS-2026774, a JP Morgan AI Research Award, a Cisco Research Award, and a Simons Collaboration on the Theory of Algorithmic Fairness. The views expressed are those of the authors and do not reflect the official policy or position of the funding agencies. The authors thank the anonymous reviewers of Eurocrypt 2023 for their valuable comments.

References

1. Agrawal, S., Maitra, M.: FE and iO for turing machines from minimal assumptions. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part II. LNCS, vol. 11240, pp. 473–512. Springer, Heidelberg (Nov 2018). https://doi.org/10.1007/978-3-030-03810-6_18
2. Ananth, P., Chen, Y.C., Chung, K.M., Lin, H., Lin, W.K.: Delegating RAM computations with adaptive soundness and privacy. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B, Part II. LNCS, vol. 9986, pp. 3–30. Springer, Heidelberg (Oct / Nov 2016). https://doi.org/10.1007/978-3-662-53644-5_1
3. Ananth, P., Chung, K.M., Fan, X., Qian, L.: Collusion-resistant functional encryption for RAMs. In: Agrawal, S., Lin, D. (eds.) ASIACRYPT 2022, Part I. LNCS, vol. 13791, pp. 160–194. Springer, Heidelberg (Dec 2022). https://doi.org/10.1007/978-3-031-22963-3_6
4. Ananth, P., Fan, X., Shi, E.: Towards attribute-based encryption for RAMs from LWE: Sub-linear decryption, and more. In: Galbraith, S.D., Moriai, S. (eds.) ASIACRYPT 2019, Part I. LNCS, vol. 11921, pp. 112–141. Springer, Heidelberg (Dec 2019). https://doi.org/10.1007/978-3-030-34578-5_5
5. Ananth, P., Jain, A., Lin, H., Matt, C., Sahai, A.: Indistinguishability obfuscation without multilinear maps: New paradigms via low degree weak pseudorandomness and security amplification. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 284–332. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_10
6. Ananth, P., Jain, A., Sahai, A.: Indistinguishability obfuscation for turing machines: Constant overhead and amortization. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part II. LNCS, vol. 10402, pp. 252–279. Springer, Heidelberg (Aug 2017). https://doi.org/10.1007/978-3-319-63715-0_9

7. Ananth, P., Lombardi, A.: Succinct garbling schemes from functional encryption through a local simulation paradigm. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part II. LNCS, vol. 11240, pp. 455–472. Springer, Heidelberg (Nov 2018). https://doi.org/10.1007/978-3-030-03810-6_17
8. Ananth, P.V., Sahai, A.: Functional encryption for turing machines. In: Kushilevitz, E., Malkin, T. (eds.) TCC 2016-A, Part I. LNCS, vol. 9562, pp. 125–153. Springer, Heidelberg (Jan 2016). https://doi.org/10.1007/978-3-662-49096-9_6
9. Attrapadung, N.: Dual system encryption framework in prime-order groups via computational pair encodings. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 591–623. Springer, Heidelberg (Dec 2016). https://doi.org/10.1007/978-3-662-53890-6_20
10. Attrapadung, N., Libert, B., de Panafieu, E.: Expressive key-policy attribute-based encryption with constant-size ciphertexts. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 90–108. Springer, Heidelberg (Mar 2011). https://doi.org/10.1007/978-3-642-19379-8_6
11. Attrapadung, N., Tomida, J.: Unbounded dynamic predicate compositions in ABE from standard assumptions. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part III. LNCS, vol. 12493, pp. 405–436. Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64840-4_14
12. Bellare, M., Hoang, V.T., Rogaway, P.: Foundations of garbled circuits. In: Yu, T., Danezis, G., Gligor, V.D. (eds.) ACM CCS 2012. pp. 784–796. ACM Press (Oct 2012). https://doi.org/10.1145/2382196.2382279
13. Bitansky, N., Garg, S., Lin, H., Pass, R., Telang, S.: Succinct randomized encodings and their applications. In: Servedio, R.A., Rubinfeld, R. (eds.) 47th ACM STOC. pp. 439–448. ACM Press (Jun 2015). https://doi.org/10.1145/2746539.2746574
14. 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: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (May 2014). https://doi.org/10.1007/978-3-642-55220-5_30
15. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Ishai, Y. (ed.) TCC 2011. LNCS, vol. 6597, pp. 253–273. Springer, Heidelberg (Mar 2011). https://doi.org/10.1007/978-3-642-19571-6_16
16. Boyle, E., Holmgren, J., Ma, F., Weiss, M.: On the security of doubly efficient PIR. Cryptology ePrint Archive, Report 2021/1113 (2021), https://eprint.iacr.org/2021/1113
17. Boyle, E., Ishai, Y., Pass, R., Wootters, M.: Can we access a database both locally and privately? In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part II. LNCS, vol. 10678, pp. 662–693. Springer, Heidelberg (Nov 2017). https://doi.org/10.1007/978-3-319-70503-3_22
18. Brakerski, Z., Segev, G.: Function-private functional encryption in the private-key setting. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part II. LNCS, vol. 9015, pp. 306–324. Springer, Heidelberg (Mar 2015). https://doi.org/10.1007/978-3-662-46497-7_12
19. Canetti, R., Chen, Y., Holmgren, J., Raykova, M.: Adaptive succinct garbled RAM or: How to delegate your database. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B, Part II. LNCS, vol. 9986, pp. 61–90. Springer, Heidelberg (Oct / Nov 2016). https://doi.org/10.1007/978-3-662-53644-5_3
20. Canetti, R., Holmgren, J.: Fully succinct garbled RAM. In: Sudan, M. (ed.) ITCS 2016. pp. 169–178. ACM (Jan 2016). https://doi.org/10.1145/2840728.2840765

21. Canetti, R., Holmgren, J., Jain, A., Vaikuntanathan, V.: Succinct garbling and indistinguishability obfuscation for RAM programs. In: Servedio, R.A., Rubinfeld, R. (eds.) 47th ACM STOC. pp. 429–437. ACM Press (Jun 2015). https://doi.org/10.1145/2746539.2746621
22. Canetti, R., Holmgren, J., Richelson, S.: Towards doubly efficient private information retrieval. In: Kalai, Y., Reyzin, L. (eds.) TCC 2017, Part II. LNCS, vol. 10678, pp. 694–726. Springer, Heidelberg (Nov 2017). https://doi.org/10.1007/978-3-319-70503-3_23
23. Chen, Y.C., Chow, S.S.M., Chung, K.M., Lai, R.W.F., Lin, W.K., Zhou, H.S.: Cryptography for parallel RAM from indistinguishability obfuscation. In: Sudan, M. (ed.) ITCS 2016. pp. 179–190. ACM (Jan 2016). https://doi.org/10.1145/2840728.2840769
24. De, A., Trevisan, L., Tulsiani, M.: Time space tradeoffs for attacks against one-way functions and PRGs. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 649–665. Springer, Heidelberg (Aug 2010). https://doi.org/10.1007/978-3-642-14623-7_35
25. De Caro, A., Iovino, V., Jain, A., O'Neill, A., Paneth, O., Persiano, G.: On the achievability of simulation-based security for functional encryption. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 519–535. Springer, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40084-1_29
26. Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: 54th FOCS. pp. 40–49. IEEE Computer Society Press (Oct 2013). https://doi.org/10.1109/FOCS.2013.13
27. Garg, S., Ostrovsky, R., Srinivasan, A.: Adaptive garbled RAM from laconic oblivious transfer. In: Shacham, H., Boldyreva, A. (eds.) CRYPTO 2018, Part III. LNCS, vol. 10993, pp. 515–544. Springer, Heidelberg (Aug 2018). https://doi.org/10.1007/978-3-319-96878-0_18
28. Garg, S., Srinivasan, A.: Single-key to multi-key functional encryption with polynomial loss. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B, Part II. LNCS, vol. 9986, pp. 419–442. Springer, Heidelberg (Oct / Nov 2016). https://doi.org/10.1007/978-3-662-53644-5_16
29. Garg, S., Srinivasan, A.: Adaptively secure garbling with near optimal online complexity. In: Nielsen, J.B., Rijmen, V. (eds.) EUROCRYPT 2018, Part II. LNCS, vol. 10821, pp. 535–565. Springer, Heidelberg (Apr / May 2018). https://doi.org/10.1007/978-3-319-78375-8_18
30. Garg, S., Srinivasan, A.: A simple construction of iO for turing machines. In: Beimel, A., Dziembowski, S. (eds.) TCC 2018, Part II. LNCS, vol. 11240, pp. 425–454. Springer, Heidelberg (Nov 2018). https://doi.org/10.1007/978-3-030-03810-6_16
31. Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: How to run turing machines on encrypted data. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part II. LNCS, vol. 8043, pp. 536–553. Springer, Heidelberg (Aug 2013). https://doi.org/10.1007/978-3-642-40084-1_30
32. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th ACM STOC. pp. 545–554. ACM Press (Jun 2013). https://doi.org/10.1145/2488608.2488677
33. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Predicate encryption for circuits from LWE. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 503–523. Springer, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_25

34. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006. pp. 89–98. ACM Press (Oct / Nov 2006). https://doi.org/10.1145/1180405.1180418, available as Cryptology ePrint Archive Report 2006/309
35. Guan, J., Wichs, D., Zhandry, M.: Incompressible cryptography. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part I. LNCS, vol. 13275, pp. 700–730. Springer, Heidelberg (May / Jun 2022). https://doi.org/10.1007/978-3-031-06944-4_24
36. Hemenway, B., Jafargholi, Z., Ostrovsky, R., Scafuro, A., Wichs, D.: Adaptively secure garbled circuits from one-way functions. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016, Part III. LNCS, vol. 9816, pp. 149–178. Springer, Heidelberg (Aug 2016). https://doi.org/10.1007/978-3-662-53015-3_6
37. Jain, A., Lin, H., Luo, J.: On the optimal succinctness and efficiency of functional encryption and attribute-based encryption. Cryptology ePrint Archive, Report 2022/1317 (2022), https://eprint.iacr.org/2022/1317
38. Jain, A., Lin, H., Matt, C., Sahai, A.: How to leverage hardness of constant-degree expanding polynomials overa $\mathbb{R}$ to build $i\mathcal{O}$. In: Ishai, Y., Rijmen, V. (eds.) EUROCRYPT 2019, Part I. LNCS, vol. 11476, pp. 251–281. Springer, Heidelberg (May 2019). https://doi.org/10.1007/978-3-030-17653-2_9
39. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from well-founded assumptions. In: Khuller, S., Williams, V.V. (eds.) 53rd ACM STOC. pp. 60–73. ACM Press (Jun 2021). https://doi.org/10.1145/3406325.3451093
40. Jain, A., Lin, H., Sahai, A.: Indistinguishability obfuscation from LPN over $\mathbb{F}_p$, DLIN, and PRGs in NC^0. In: Dunkelman, O., Dziembowski, S. (eds.) EUROCRYPT 2022, Part I. LNCS, vol. 13275, pp. 670–699. Springer, Heidelberg (May / Jun 2022). https://doi.org/10.1007/978-3-031-06944-4_23
41. Kitagawa, F., Nishimaki, R., Tanaka, K.: Simple and generic constructions of succinct functional encryption. In: Abdalla, M., Dahab, R. (eds.) PKC 2018, Part II. LNCS, vol. 10770, pp. 187–217. Springer, Heidelberg (Mar 2018). https://doi.org/10.1007/978-3-319-76581-5_7
42. Kitagawa, F., Nishimaki, R., Tanaka, K., Yamakawa, T.: Adaptively secure and succinct functional encryption: Improving security and efficiency, simultaneously. In: Boldyreva, A., Micciancio, D. (eds.) CRYPTO 2019, Part III. LNCS, vol. 11694, pp. 521–551. Springer, Heidelberg (Aug 2019). https://doi.org/10.1007/978-3-030-26954-8_17
43. Koppula, V., Lewko, A.B., Waters, B.: Indistinguishability obfuscation for turing machines with unbounded memory. In: Servedio, R.A., Rubinfeld, R. (eds.) 47th ACM STOC. pp. 419–428. ACM Press (Jun 2015). https://doi.org/10.1145/2746539.2746614
44. Li, B., Micciancio, D.: Compactness vs collusion resistance in functional encryption. In: Hirt, M., Smith, A.D. (eds.) TCC 2016-B, Part II. LNCS, vol. 9986, pp. 443–468. Springer, Heidelberg (Oct / Nov 2016). https://doi.org/10.1007/978-3-662-53644-5_17
45. Li, H., Lin, H., Luo, J.: ABE for circuits with constant-size secret keys and adaptive security. In: Kiltz, E., Vaikuntanathan, V. (eds.) TCC 2022, Part I. LNCS, vol. 13747, pp. 680–710. Springer, Heidelberg (Nov 2022). https://doi.org/10.1007/978-3-031-22318-1_24
46. Lin, H., Luo, J.: Succinct and adaptively secure ABE for ABP from k-lin. In: Moriai, S., Wang, H. (eds.) ASIACRYPT 2020, Part III. LNCS, vol. 12493, pp. 437–

466. Springer, Heidelberg (Dec 2020). https://doi.org/10.1007/978-3-030-64840-4_15
47. Lin, W.K., Mook, E., Wichs, D.: Doubly efficient private information retrieval and fully homomorphic RAM computation from ring LWE. Cryptology ePrint Archive, Report 2022/1703 (2022), https://eprint.iacr.org/2022/1703
48. Luo, J.: *Ad hoc* (decentralized) broadcast, trace, and revoke. Cryptology ePrint Archive, Report 2022/925 (2022), https://eprint.iacr.org/2022/925
49. Naor, M., Yogev, E.: Bloom filters in adversarial environments. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 565–584. Springer, Heidelberg (Aug 2015). https://doi.org/10.1007/978-3-662-48000-7_28
50. O'Neill, A.: Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556 (2010), https://eprint.iacr.org/2010/556
51. Quach, W., Wee, H., Wichs, D.: Laconic function evaluation and applications. In: Thorup, M. (ed.) 59th FOCS. pp. 859–870. IEEE Computer Society Press (Oct 2018). https://doi.org/10.1109/FOCS.2018.00086
52. Sahai, A., Waters, B.R.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (May 2005). https://doi.org/10.1007/11426639_27
53. Takashima, K.: Expressive attribute-based encryption with constant-size ciphertexts from the decisional linear assumption. In: Abdalla, M., Prisco, R.D. (eds.) SCN 14. LNCS, vol. 8642, pp. 298–317. Springer, Heidelberg (Sep 2014). https://doi.org/10.1007/978-3-319-10879-7_17
54. Yamada, S., Attrapadung, N., Hanaoka, G., Kunihiro, N.: A framework and compact constructions for non-monotonic attribute-based encryption. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 275–292. Springer, Heidelberg (Mar 2014). https://doi.org/10.1007/978-3-642-54631-0_16
55. Zhang, K., Gong, J., Tang, S., Chen, J., Li, X., Qian, H., Cao, Z.: Practical and efficient attribute-based encryption with constant-size ciphertexts in outsourced verifiable computation. In: Chen, X., Wang, X., Huang, X. (eds.) ASIACCS 16. pp. 269–279. ACM Press (May / Jun 2016)