Incompressible Functional Encryption

Rishab Goyal
University of Wisconsin-Madison, WI, USA

Venkata Koppula
IIT Delhi, India

Mahesh Sreekumar Rajasree[1]
IIT Delhi, India

Aman Verma
IIT Delhi, India

Abstract

Incompressible encryption (Dziembowski, Crypto'06; Guan, Wichs, Zhandry, Eurocrypt'22) protects from attackers that learn the entire decryption key, but cannot store the full ciphertext. In incompressible encryption, the attacker must try to compress a ciphertext within pre-specified memory bound S before receiving the secret key.

In this work, we generalize the notion of incompressibility to functional encryption. In incompressible functional encryption, the adversary can corrupt non-distinguishing keys at any point, but receives the distinguishing keys only after compressing the ciphertext to within S bits. An important efficiency measure for incompressible encryption is the ciphertext-rate (i.e., $\mathsf{rate} = |m|/|\mathsf{ct}|$). We give many new results for incompressible functional encryption for circuits, from minimal assumption of (non-incompressible) functional encryption, with

1. ct-rate-$\frac{1}{2}$ and *short* secret keys,
2. ct-rate-1 and *large* secret keys.

Along the way, we also give a new incompressible attribute-based encryption for circuits from standard assumptions, with ct-rate-$\frac{1}{2}$ and short secret keys. Our results achieve optimal efficiency, as incompressible attribute-based/functional encryption with ct-rate-1 as well as short secret keys has strong barriers for provable security from standard assumptions. Moreover, our assumptions are minimal as incompressible attribute-based/functional encryption are strictly stronger than their non-incompressible counterparts.

2012 ACM Subject Classification Security and privacy → Public key encryption

Keywords and phrases functional encryption, attribute-based encryption, incompressible encryption

Digital Object Identifier 10.4230/LIPIcs.ITCS.2025.56

Related Version *Full Version*: https://eprint.iacr.org/2024/798.pdf

Funding *Rishab Goyal*: Support for this research was provided by OVCRGE at UW–Madison with funding from the Wisconsin Alumni Research Foundation.

1 Introduction

A fundamental principle in cryptography is to leverage "secrets" for differentiating between an honest user legitimately accessing a system and an attacker trying to illegitimately access it. Consider data encryption as an example, where the goal is data secrecy. A central assumption underlying traditional encryption is that *an attacker cannot learn anything about a user's secret decryption key*. Moreover, this assumption seems inherent as if an attacker learns the secret key, then how can one distinguish between "capabilities" of an honest user and an attacker!?

[1] Corresponding author

In 2006, Dziembowzki [14] proposed an exquisite approach to *restrict the "capabilities" of an attacker*, even when the *entire* secret key gets corrupted. The proposal was to view *long-term memory storage* as a limited and expensive resource. In words, the intuition was that no real-world attacker can maintain an unbounded long-term memory, and it has to pick and choose what it wants to store in its memory. Dziembowzki [14] formalized this in the secret-key setting, and defined it as a *forward-secure storage* system. Recently, Guan, Wichs, and Zhandry (henceforth GWZ) [23] generalized this to the public-key setting. They labeled cryptography in this model as "incompressible" cryptography. The intuition being that an attacker cannot "compress" its unbounded short-term memory into its bounded long-term memory. GWZ provided multiple constructions for "incompressible" public-key encryption and signatures, from a variety of cryptographic assumptions with different efficiency tradeoffs.

Beyond Public-Key Encryption

Functional encryption (FE) [32, 2, 28, 7] is a powerful generalization of public-key encryption (PKE) [12]. FE generalizes PKE by providing fine-grained access to encrypted data. In FE, the master decryption key holder can create partial "functional" decryption keys sk_f for any supported function f of its choice. Such a key holder can recover $f(m)$ from any ciphertext ct, encrypting data m. Standard indistinguishability-based FE security states that no polytime attacker cannot distinguish between encryptions $\mathsf{ct}_0, \mathsf{ct}_1$ of messages m_0, m_1, unless it receives a secret key sk_{f^*} s.t. $f^*(m_0) \neq f^*(m_1)$. That is, unless an attacker receives a "distinguishing key" sk_{f^*}, it cannot distinguish between any two ciphertexts.[2] The ability to learn only function evaluation of encrypted data and nothing more has had fascinating consequences in cryptography, both theoretically and practically [33, 5, 11, 32, 22, 8, 2, 28, 7, 30, 17, 15, 20, 34, 10, 19, 29, 31, 21, 26, 27, 4, 13].

While functional keys offer fine-grained access over encrypted data, enabling many superior applications, they also enable far more attack opportunities compared to plain encryption. In PKE, there is just one secret key, and while that implies an all-or-nothing functionality, storing the key securely for the entire system lifetime is much easier. On the contrary, in FE, there are master secret keys as well as functional decryption keys. Each decryption key must be securely generated, distributed, and stored by the appropriate user. While it might be reasonable to expect the central trusted authority will (in most part) store master key msk securely; expecting this from every other user (holding a functional key) is unrealistic.

Our Contributions: Incompressible FE

In this work, we study incompressible functional encryption systems. We formalize the concept, and provide multiple constructions from a variety of cryptographic assumptions enabling different efficiency tradeoffs. As we elaborate in the full version [18], incompressibility in FE is extremely interesting, both theoretically and practically. In short, it pushes the traditional perception in FE that leaking an entire decryption key is equivalent to learning the function output on every previously encrypted data.

Formalizing incompressible FE. Interestingly, formally defining incompressibility for FE is more nuanced than for PKE. In public-key encryption, there is just one secret key. Thus, to define incompressibility for PKE, one considers a simple two-stage attacker: in stage 1,

[2] Throughout the sequel, by the phrase a "distinguishing key", we mean a secret key that can distinguish between two ciphertexts (by running decryption). Similarly, by the phrase "non-distinguishing keys", we mean secret keys that do not enable a trivial distinguishing attack between two ciphertexts.

the attacker receives a ciphertext that it must "compress" down to S bits[3]; and stage 2, where the attacker receives the secret key (along with the S-bit "compressed" ciphertext). As long as such an attacker cannot learn anything about the message, the PKE scheme is a secure incompressible scheme. In functional encryption setting, there are many more secret keys! Thus, depending upon the secret key(s) that a *stage 2* attacker learns, there are different formulations of incompressible FE security. Below, we briefly summarize three such formulations, and discuss them in further detail in the full version [18].[4]

- Standard security: the adversary receives *one* (or bounded number of) *distinguishing* secret key(s), after compressing challenge ciphertext (i.e., in stage 2). Moreover, it receives unbounded number of *non-distinguishing* secret keys before and after compressing the ciphertext.
- Semi-strong security: this is a strengthening of standard security, where the adversary can get an *unbounded* number of distinguishing keys in stage 2.
- Strong security: this further strengthens as now adversary gets the *master secret key* in stage 2.

Measuring incompressibility efficiency. Clearly, for the above to not be trivially impossible, the ciphertext size should be larger than S, the incompressibility limit. Further, the ciphertext should be larger than $|m|$ to uniquely encode m and guarantee correctness. However, it is not clear if there are any other necessary constraints on parameter sizes, beyond $|\mathsf{ct}| = \max(S, |m|) + \mathsf{poly}(\lambda)$. Here the $\max(S, |m|)$ term ensures both ciphertext constraints (i.e., $|\mathsf{ct}| > S$ and $|\mathsf{ct}| > |m|$).

Thus, an ideal goal is to design incompressible FE where $|\mathsf{mpk}|, |\mathsf{msk}|, |\mathsf{sk}_f|$ are all independent of S, and $|\mathsf{ct}| = \max(S, |m|) + \mathsf{poly}(\lambda)$. Such an incompressible FE scheme would have asymptotically optimal-sized parameters. As we discuss in the full version [18], achieving optimality in all parameter sizes is unachievable from standard falsifiable assumptions. Therefore, a standard approach in the literature [23, 9] is to design encryption with highest ciphertext-rate. Ciphertext-rate is defined as the ratio of message length and ciphertext length, i.e. $\frac{|m|}{|\mathsf{ct}|}$.

Prior works [23, 9] built incompressible PKE with optimal ct-rate of 1. Unfortunately, that comes at the cost of large secret keys, i.e. $|\mathsf{sk}| = \mathsf{poly}(S, \lambda)$. In this work, we design incompressible ABE/FE in two incomparable parameter regimes – ct-rate of 1 with *"large"* secret keys, and ct-rate of $\frac{1}{2}$ with *"short"* secret keys. We believe both parameter regimes to be equally interesting. In FE applications, where an honest user stores only secret keys and not ciphertexts in its persistent long-term storage, it might be beneficial to have short keys with constant ct-rate. While, in applications where an honest user stores only a few secret keys and a large number of ciphertexts, having a ct-rate of 1 might be more useful.

Our results

We summarize our main results below. All our schemes are in the standard model and under standard assumptions. We do not make any ideal-cipher, random oracle, or any other non-standard/non-falsifiable cryptographic assumption.

[3] S is typically considered to be the bound on the adversary's long-term storage.

[4] We can also consider joint compression of ciphertexts and secret keys in FE. The focus of this work is only on ciphertext incompressibility as discussed in the full version [18]. It is an interesting open question of whether new tradeoffs and applications can be enabled, if one considers joint incompressibility of ciphertexts and secret keys.

Incompressible FE. Our first set of results contains new constructions for incompressible FE for polynomial-sized circuits. We provide three separate and incomparable FE constructions as each provides a unique efficiency-security tradeoff. All three construction rely on a standard public-key FE scheme for polynomial-sized circuits.

1. Our first incompressible FE scheme is proven secure in the *semi-strong* incompressibility model. If the underlying FE scheme has ct-rate of $r \in [0, 1]$, then our incompressible FE scheme has an $\frac{r}{2}$ ct-rate.
2. Our second scheme is also proven secure in the *semi-strong* incompressibility model, except we only prove selective security.[5] Moreover, this construction is a ct-rate preserving construction. That is, we obtain ct-rate r incompressible FE scheme from any ct-rate r standard FE scheme. However, the secret keys are large to avoid known barriers.
3. Our third scheme is an extension of our above scheme. It is proven secure in the *selective standard* incompressibility model, but it provides a rather non-trivial efficiency guarantee. We prove that if the underlying FE scheme has short keys, then so does our incompressible FE scheme (without lowering the ct-rate). As we elaborate later in the overview, this does not contradict known implausibility results (see discussion in the full version [18]), but rather tightly matches it.

By plugging in known optimal-ciphertext-rate FE schemes [23, 25], we obtain incompressible FE schemes under the assumption of poly-secure FE with:

1. a ct-rate-$\frac{1}{2}$, short keys, and selective semi-strong security,
2. a ct-rate-$\frac{1}{4}$, short keys, and *adaptive* semi-strong security,
3. a ct-rate-1, large keys, and selective semi-strong security,
4. a ct-rate-1, *short* keys, and selective *standard* security.

Incompressible ABE. Our second result is an incompressible attribute-based encryption (ABE) scheme for polynomial-sized circuits achieves (standard) incompressible ABE security, with ciphertext size $|\mathsf{ct}| = S + |m| + \mathsf{poly}(\lambda)$ and all other parameter sizes, $|\mathsf{mpk}|, |\mathsf{msk}|, |\mathsf{sk}_f|$, are independent of S. That is, this scheme has ct-rate of $\frac{1}{2}$, and short keys. The construction and security proof are available in the full version [18].

2 Preliminaries and Notations

Throughout this paper, we will use λ to denote the security parameter and $\mathrm{negl}(\cdot)$ to denote a negligible function in the input. We will use the short-hand notation PPT for "probabilistic polynomial time". For any finite set X, $x \leftarrow X$ denotes the process of picking an element x from X uniformly at random. Similarly, for any distribution $\mathcal{D}$, $x \leftarrow \mathcal{D}$ denotes an element x drawn from the distribution $\mathcal{D}$. For any natural number $n \in \mathbb{N}$, $[n]$ denotes the set $\{1, 2, \ldots, n\}$. For any two binary string x and y, $x||y$ denotes the concatenation of x and y. We use the following two notations to denote a family of circuits - $\{\mathcal{C}_n\}_n$ is a set of families of circuits indexed by some parameter n and $\{\mathcal{C}_{d,\ell}\}_{d,\ell}$ is a set of families of circuits indexed by the depth of the circuits d and the number of inputs to the circuits ℓ.

[5] Here selective means that the adversary must submit challenge messages before receiving secret keys in stage 1.

2.1 Randomness Extractors

▶ **Definition 1** (Strong Average Min-Entropy Extractor). *A (k, ϵ)-strong average min-entropy extractor is an efficient function* $\mathsf{Ext} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^m$ *such that for all jointly distributed random variable X, Y where X takes values $\{0,1\}^d$ and $H_\infty(X|Y) \geq k$, we have $(U_d, \mathsf{Ext}(X, U_d), Y) \approx_\epsilon (U_d, U_m, Y)$ where U_d, U_m are uniformly random strings of length d, m respectively. Here $H_\infty(X|Y) = -\log \mathop{\mathbb{E}}_{y \leftarrow Y}(\max_x \mathsf{Pr}(X = x|Y = y))$ is the average min-entropy of X conditioned on Y.*

▶ **Theorem 2.** *There exists an explicit efficient $(k, 2^{-\lambda})$-strong average min-entropy extractor* $\mathsf{Ext} : \{0,1\}^d \times \{0,1\}^n \to \{0,1\}^\lambda$ *such that $k = \mathsf{poly}(\lambda)$, $d = \mathsf{poly}(\lambda)$, $n = S + k$ and the depth of the extractor circuit is* $\mathsf{poly}(\lambda, \log(n))$.

2.2 Pseudorandom Functions

A family of pseudorandom functions $\mathsf{PRF} = (\mathsf{KeyGen}, \mathsf{Eval})$ with key space $\{\mathcal{K}_\lambda\}_\lambda$, input space $\{\mathcal{X}_\lambda\}_\lambda$ and output space $\{\mathcal{Y}_\lambda\}_\lambda$ consists of the following algorithms.

- $\mathsf{KeyGen}(1^\lambda)$: The key generation algorithm is a randomized algorithm that takes as input the security parameter 1^λ and outputs a key $k \in \mathcal{K}_\lambda$.
- $\mathsf{Eval}(k, x)$: The evaluation algorithm is a deterministic algorithm that takes as input a key $k \in \mathcal{K}_\lambda$ and $x \in \mathcal{X}_\lambda$ and outputs $y \in \mathcal{Y}_\lambda$.

▶ **Definition 3.** *A PRF scheme* PRF *is secure if for all PPT adversary $\mathcal{A}$, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$,*

$$|\Pr[\mathcal{A}^{\mathsf{Eval}(k,\cdot)}(1^\lambda) = 1 \ : \ k \leftarrow \mathsf{KeyGen}(1^\lambda)] - \Pr[\mathcal{A}^{R(\cdot)}(1^\lambda) = 1 \ : \ R \leftarrow \mathcal{U}_\lambda]| \leq \mathrm{negl}(\lambda)$$

where $\mathcal{U}_\lambda$ is the set of all functions from $\mathcal{X}_\lambda$ to $\mathcal{Y}_\lambda$.

2.3 Garbling Scheme

A garbling scheme $\mathsf{GC} = (\mathsf{Garble}, \mathsf{Eval})$ for a class of circuits $\{\mathcal{C}_\lambda\}_\lambda$ consists of the following algorithms.

- $\mathsf{Garble}(1^\lambda, C)$: The garbling algorithm is a randomized algorithm that takes as input the security parameter 1^λ and a circuit $C \in \mathcal{C}_\lambda$ such that $C : \{0,1\}^n \to \{0,1\}$ and outputs a garbled circuit $\hat{C}$ and a set of labels $\{\mathsf{lab}_{i,b}\}_{i \in [n], b \in \{0,1\}}$.
- $\mathsf{Eval}(\hat{C}, \{\mathsf{lab}_i\}_{i \in [n]})$: The evaluation algorithm takes as input a garbled circuit $\hat{C}$ and a set of n labels $\{\mathsf{lab}_i\}_{i \in [n]}$ and outputs $y \in \{0,1\}$.

Correctness

For correctness of a garbling scheme GC for a class of circuits $\{\mathcal{C}_\lambda\}_\lambda$, we require that for all $\lambda \in \mathbb{N}, C \in \mathcal{C}_\lambda$, $x \in \{0,1\}^n$,

$$\mathsf{Eval}(\hat{C}, \{\mathsf{lab}_{i,x_i}\}_{i \in [n]}) = C(x)$$

where $(\hat{C}, \{\mathsf{lab}_{i,b}\}_{i \in [n], b \in \{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C)$.

Security

For security, we define simulation based security [35, 1].

▶ **Definition 4.** *A garbling scheme* $\mathsf{GC} = (\mathsf{Garble}, \mathsf{Eval})$ *for a class of circuits* $\{\mathcal{C}_\lambda\}_\lambda$ *is said to be secure if there exists a PPT algorithm* Sim *such that for all PPT adversaries* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*, the following holds.*

$$\left| \Pr\left[\mathcal{A}_2(\hat{C}, \{\mathsf{lab}_{i,x_i}\}_{i\in[n]}, \mathsf{aux}) = 1 : \begin{array}{c}(C, x, \mathsf{aux}) \leftarrow \mathcal{A}_1(1^\lambda),\\ (\hat{C}, \{\mathsf{lab}_{i,x_i}\}_{i\in[n]}) \leftarrow \mathsf{Sim}(1^\lambda, 1^n, 1^{|C|}, C(x))\end{array}\right] \right.$$

$$\left. - \Pr\left[\mathcal{A}_2(\hat{C}, \{\mathsf{lab}_{i,x_i}\}_{i\in[n]}, \mathsf{aux}) = 1 : \begin{array}{c}(C, x, \mathsf{aux}) \leftarrow \mathcal{A}_1(1^\lambda),\\ (\hat{C}, \{\mathsf{lab}_{i,b}\}_{i\in[n],b\in\{0,1\}}) \leftarrow \mathsf{Garble}(1^\lambda, C))\end{array}\right] \right| \leq \frac{1}{2} + \mathrm{negl}(\lambda).$$

2.4 Incompressible Secret Key Encryption

An incompressible secret key encryption scheme $\mathsf{IncSKE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec})$ with message space $\{\mathcal{M}_\lambda\}_\lambda$ consists of the following PPT algorithms.

- $\mathsf{Setup}(1^\lambda, 1^S, 1^n)$: The setup algorithm is a randomized algorithm that takes as input the security parameter λ, a parameter 1^S, the length of the message 1^n and outputs a secret key sk.
- $\mathsf{Enc}(\mathsf{sk}, m)$: The encryption algorithm is a randomized algorithm that takes as input a secret key sk and a message $m \in \mathcal{M}_\lambda$ and outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: The decryption algorithm takes as input a secret key sk and a ciphertext ct and outputs either a message $m \in \mathcal{M}_\lambda$ or $\perp$.

Correctness

For correctness, we require that for all $\lambda \in \mathbb{N}, S \in \mathbb{N}, n \in \mathbb{N}, m \in \mathcal{M}_\lambda$ and $\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda, 1^S, 1^n)$,

$$\Pr[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{sk}, m)) = m] = 1$$

where the probability is over the random bits used in the encryption algorithm.

Incompressible SKE Security

Consider the following experiment with an adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$.

- **Initialization Phase:** $\mathcal{A}_1$ on input 1^λ, outputs an upper bound on the state size 1^S and the length of the message 1^n. The challenger runs $\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda, 1^S, 1^n)$.
- **Challenge Phase:** $\mathcal{A}_1$ outputs a message m, along with an auxiliary information aux. The challenger randomly chooses $b \in \{0, 1\}$. If $b = 0$, it samples a truly random string ct^*. Else, it computes a ciphertext $\mathsf{ct}^* = \mathsf{Enc}(\mathsf{sk}, m)$ and sends it to $\mathcal{A}_1$.[6]
- **First Response Phase:** $\mathcal{A}_1$ computes a state st such that $|\mathsf{st}| \leq S$.
- **Second Response Phase:** $\mathcal{A}_2$ receives $(\mathsf{sk}, \mathsf{aux}, \mathsf{st})$ and outputs b'. $\mathcal{A}$ wins the experiment if $b = b'$.

▶ **Definition 5.** *An SKE scheme is said to be incompressible secure if for all PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$\Pr[\mathcal{A} \textit{ wins in the above experiment}] \leq \frac{1}{2} + \mathrm{negl}(\lambda)$$

[6] In the incompressible security definition presented in prior works [9, 24], the adversary sends two messages m_0, m_1 and receives an encryption of one of these message. Note that this is a weaker notion, which is implied by the incompressible security notion defined in this paper.

CPA-SKE Security

Consider the following experiment with an adversary $\mathcal{A}$ where Setup algorithm takes only 1^λ and 1^n as input.

- **Initialization Phase:** The challenger runs $\mathsf{sk} \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$.
- **Pre-Challenge Query Phase:** $\mathcal{A}$ is allowed to make polynomially many queries. For each query m, the challenger computes $\mathsf{ct} \leftarrow \mathsf{SKE.Enc}(\mathsf{sk}, m)$ and returns ct to Adv.
- **Challenge Phase:** $\mathcal{A}$ outputs a message m^*. The challenger randomly chooses $b \in \{0,1\}$. If $b = 0$, it samples a truly random string ct^*. Else, it computes a ciphertext $\mathsf{ct}^* = \mathsf{Enc}(\mathsf{sk}, m^*)$ and sends it to $\mathcal{A}$.[7]
- **Post-Challenge Query Phase:** $\mathcal{A}$ is allowed to make polynomially many queries. For each query m, the challenger computes $\mathsf{ct} \leftarrow \mathsf{SKE.Enc}(\mathsf{sk}, m)$ and returns ct to Adv.
- **Response Phase:** $\mathcal{A}$ outputs $b' \in \{0,1\}$ and wins the experiment if $b = b'$.

▶ **Definition 6.** *An SKE scheme is said to be secure if for all PPT adversaries $\mathcal{A}$, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}, n \in \mathbb{N}$,*

$$\Pr[\mathcal{A} \text{ wins in the above experiment}] \leq \frac{1}{2} + \mathrm{negl}(\lambda)$$

▶ **Theorem 7** ([14])**.** *Assuming the existence of one-way functions, there exists an incompressible SKE scheme with* $\mathsf{poly}(\lambda)$ *secret-key size and $S + n + \mathsf{poly}(\lambda)$ ciphertext size where n is the size of the message and S is the compressibility parameter. The depth of the decryption circuit is* $\mathsf{poly}(\lambda, \log(S, n))$.

2.5 Functional Encryption

A functional encryption scheme $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for the function space $\{\mathcal{F}_n : \mathcal{X}_n \to \mathcal{Y}_n\}_n$ consists of the following PPT algorithms.

- $\mathsf{Setup}(1^\lambda, 1^n)$: The setup algorithm is a randomized algorithm that takes as input the security parameter 1^λ and an index 1^n and outputs a master public key mpk and a secret key msk.
- $\mathsf{KeyGen}(\mathsf{msk}, f)$: The key generation algorithm is a randomized algorithm that takes as input the master secret msk and a function $f \in \mathcal{F}_n$ and outputs a secret sk_f.
- $\mathsf{Enc}(\mathsf{mpk}, m)$: The encryption algorithm is a randomized algorithm that takes as input a public key mpk and a message $m \in \mathcal{X}_n$ and outputs a ciphertext ct.
- $\mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct})$: The decryption algorithm takes as input a secret key sk_f and a ciphertext ct and outputs either a $y \in \mathcal{Y}_n$ or $\perp$.

Correctness

For correctness, we require that for all $\lambda \in \mathbb{N}, n \in \mathbb{N}, m \in \mathcal{X}_n, f \in \mathcal{F}_n$ and $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$,

$$\Pr[\mathsf{Dec}(\mathsf{KeyGen}(\mathsf{msk}, f), \mathsf{Enc}(\mathsf{mpk}, m)) = f(m)] = 1$$

where the probability is over the random bits used in the encryption and key generation algorithm.

[7] Note that this security notion implies the standard indistinguishability security notion where the adversary sends two messages m_0, m_1 and receives an encryption of one of the messages.

Adaptive IND-based Security

Consider the following experiment with an adversary $\mathcal{A}$.

- **Initialization Phase:** The challenger runs $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$ and sends mpk to $\mathcal{A}$.
- **Pre-Challenge Query Phase:** $\mathcal{A}$ is allowed to make polynomially many queries. For each query f, the challenger computes $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$ and returns sk_f to $\mathcal{A}$.
- **Challenge Phase:** $\mathcal{A}$ outputs two message m_0, m_1. If there exists a function f queried by $\mathcal{A}$ such that $f(m_0) \neq f(m_1)$, the challenger aborts the game. Else, it randomly chooses $b \in \{0, 1\}$ and computes a ciphertext $\mathsf{ct}^* = \mathsf{Enc}(\mathsf{mpk}, m_b)$ and sends it to $\mathcal{A}$.
- **Post-Challenge Query Phase:** $\mathcal{A}$ is allowed to make polynomially many queries. For each query f, if $f(m_0) \neq f(m_1)$, the challenger sends $\perp$. Else, computes $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$ and returns sk_f to $\mathcal{A}$.
- **Response Phase:** $\mathcal{A}$ outputs b'. $\mathcal{A}$ wins the experiment if $b = b'$.

▶ **Definition 8.** *An FE scheme satisfies adaptive indistinguishability-based security if for all PPT adversaries $\mathcal{A}$, there exists a negligible function* $\mathrm{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}$,*

$$\Pr[\mathcal{A} \textit{ wins in the above experiment}] \leq \frac{1}{2} + \mathrm{negl}(\lambda)$$

If the FE is secure against an unbounded number of queries from the adversary, then we say that the scheme is collision-resistant.

A relaxed notion of the FE indistinguishability-based security is the well-known selective security model. An FE scheme is said to be *selectively* secure if it is secure against PPT adversaries $\mathcal{A}$ that do not make any key queries during the pre-challenge query phase, and have to submit the challenge message before obtaining the public parameters.

The following two theorems addresses the optimal ciphertext-rate and secret-key size FE schemes for both adaptive and selective settings. It is important to note that in these FE scheme, the decryption algorithm requires the complete description of the function f in addition to the secret key sk_f to perform decryption.

▶ **Theorem 9** ([25])**.** *Assuming selectively secure FE for circuits, there exists an adaptively secure FE such that*

$$|\mathsf{mpk}| = O_\lambda(1), \quad |\mathsf{sk}_f| = O_\lambda(1)^8, \quad |\mathsf{ct}| = 2|x| + O_\lambda(1)$$

where $O_\lambda(\cdot)$ hides factors of $\mathsf{poly}(\lambda)$*, λ is the security parameter,* ct *is a ciphertext and* sk_f *is a secret key for the function f, x is any element from the input space of f and* mpk *is the master public key generated by the scheme.*

▶ **Theorem 10** ([25])**.** *Assuming selectively secure FE for circuits, there exists an adaptively secure FE such that*

$$|\mathsf{mpk}| = O_\lambda(1), \quad |\mathsf{sk}_f| = O_\lambda(1)^9, \quad |\mathsf{ct}| = |x| + O_\lambda(1)$$

where $O_\lambda(\cdot)$ hides factors of $\mathsf{poly}(\lambda)$*, λ is the security parameter,* ct *is a ciphertext and* sk_f *is a secret key for the function f that outputs a single bit, x is any element from the input space of f and* mpk *is the master public key generated by the scheme.*

Simulation security

Consider the following real and simulated experiment with an adversary $\mathcal{A}$ where Sim is a PPT simulator.

Real Experiment

- **Initialization Phase:** The challenger runs $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$ and sends mpk to $\mathcal{A}$.
- **Challenge Phase:** $\mathcal{A}$ outputs a message m and a function f. The challenger computes a ciphertext $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, m)$ and $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$ and sends $(\mathsf{ct}, \mathsf{sk}_f)$ to $\mathcal{A}$.
- **Response Phase:** $\mathcal{A}$ outputs b.

Simulated Experiment

- **Initialization Phase:** The simulator runs $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$ and sends mpk to $\mathcal{A}$.
- **Challenge Phase:** $\mathcal{A}$ outputs a message m and a function f. The simulator computes $(\mathsf{ct}, \mathsf{sk}_f) \leftarrow \mathsf{Sim}(\mathsf{mpk}, f, f(m), 1^{|m|})$ and sends $(\mathsf{ct}, \mathsf{sk}_f)$ to $\mathcal{A}$.
- **Response Phase:** $\mathcal{A}$ outputs b.

▶ **Definition 11.** *An FE scheme is said to be* ***single-key*** *simulation secure if for all PPT adversaries $\mathcal{A}$, there exists a PPT simulator* Sim *and a negligible function* $\mathrm{negl}(\cdot)$ *such that for all $\lambda \in \mathbb{N}, n \in \mathbb{N}$,*

$$\Big|\Pr[\mathcal{A} \text{ outputs 0 in the real experiment}] - \Pr[\mathcal{A} \text{ outputs 0 in the simulated experiment}]\Big| \leq \mathrm{negl}(\lambda)$$

▶ **Theorem 12** ([16])**.** *Assuming the hardness of* LWE*, there exists a single-key simulation secure* FE *for a class of circuits $\{\mathcal{C}_{n,d}\}_{n,d}$ that takes input of size n, outputs a single bit and has depth d such that*

$$|\mathsf{ct}| = O_\lambda(n \cdot d^2), \quad |\mathsf{sk}_f| = O_\lambda(d^2), \quad |mpk| = O_\lambda(n \cdot d^2)$$

where $O_\lambda(\cdot)$ hides factors of $\mathsf{poly}(\lambda)$, λ is the security parameter, ct *is a ciphertext and* sk_f *is a secret key for the circuit $f \in \mathcal{C}_{n,d}$ and* mpk *is the master public key generated by the scheme.*

▶ **Corollary 13.** *Assuming the hardness of* LWE*, there exists a single-key simulation secure* FE *for a class of circuits $\{\mathcal{C}_{n,d}\}_{n,d}$ that takes input of size n, outputs m-bits and has depth d such that*

$$|\mathsf{ct}| = O_\lambda(m \cdot n \cdot d^2), \quad |\mathsf{sk}_f| = O_\lambda(m \cdot d^2), \quad |mpk| = O_\lambda(m \cdot n \cdot d^2)$$

where $O_\lambda(\cdot)$ hides factors of $\mathsf{poly}(\lambda)$, λ is the security parameter, ct *is a ciphertext and* sk_f *is a secret key for the circuit $f \in \mathcal{C}_{n,d}$ and* mpk *is the master public key generated by the scheme.*

Specializing FE

An attribute based encryption scheme (ABE) is a function encryption scheme where the message space is $\{\mathcal{M}_n \times \mathcal{X}_n\}_n$, i.e., it is a tuple of a message and an attribute and the function class is a set of function $\{\mathcal{F}_n\}_n$ such that any function $f_C \in \mathcal{F}_n$ is associated with a predicate function C with the following definition.

$$\mathcal{F}_C(m, x) = \begin{cases} m & if \;\; C(x) = 1 \\ \perp & if \;\; C(x) = 0 \end{cases}$$

In the security experiment, the adversary has to output m_0, m_1, x^* in the challenge phase with the restriction that it has never queried a key for a function f_C such that $C(x^*) = 1$.

▶ **Theorem 14** ([6]). *Assuming the hardness of* LWE*, there exists an selectively secure ABE scheme for any family of circuits* $\{\mathcal{C}_{d,\ell}\}_{d,\ell}$ *that has* ℓ*-length input and* d*-depth satisfying*

$$|\mathsf{ct}| = O_\lambda(\ell \cdot d^2), \qquad |\mathsf{sk}_f| = O_\lambda(d^2), \qquad |mpk| = O_\lambda(\ell \cdot d^2)$$

where $O_\lambda(\cdot)$ *hides factors of* $\mathsf{poly}(\lambda)$*,* λ *is the security parameter,* ct *is a ciphertext and* sk_f *is a secret key for the predicate* $f \in \mathcal{C}_{d,\ell}$ *and* mpk *is the master public key generated by the scheme.*

An identity based encryption scheme (IBE) is a attribute based encryption scheme where the message space is $\{\mathcal{M}_\lambda \times \mathcal{ID}_\lambda\}_\lambda$, i.e., it is a tuple of a message and an identity and the function class is a set of function $\{\mathcal{F}_\lambda\}_\lambda$ such that any function $f_{\mathsf{id}} \in \mathcal{F}_\lambda$ is associated with an identity id with the following definition.

$$\mathcal{F}_{\mathsf{id}}(m, \mathsf{id}') = \begin{cases} m & if \ \ \mathsf{id} = \mathsf{id}' \\ \perp & if \ \ \mathsf{id} \neq \mathsf{id}' \end{cases}$$

In the security experiment, the adversary has to output m_0, m_1, id^* in the challenge phase with the restriction that it has never queried a key for id^*.

3 Incompressible Functional Encryption: Definitions

In this section, we will proceed to define the incompressible version of the security game for FE scheme where Setup takes an additional input 1^S. Similar to the incompressible SKE schemes, we will consider two adversaries $\mathcal{A}_1, \mathcal{A}_2$. The first adversary $\mathcal{A}_1$ will be provided with the complete challenge ciphertext and produce a compressed version of it. The second adversary $\mathcal{A}_2$ is provided with the master public key, compressed challenge ciphertext which was created by $\mathcal{A}_1$ and certain secret keys.

▶ **Definition 15** (Incompressible FE Security). *Let* $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ *be an FE scheme in which the setup algorithm takes an additional parameter* 1^S *as input. Consider the following experiment with an adversary* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*.*

- ***Initialization Phase:*** *$\mathcal{A}_1$ on input 1^λ, outputs an upper bound on the state size 1^S and 1^n. The challenger runs $(\mathsf{msk}, \mathsf{mpk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^S, 1^n)$ and sends mpk to $\mathcal{A}_1$.*
- ***Pre-Challenge Query Phase:*** *In this phase, $\mathcal{A}_1$ is allowed to make polynomially many key queries. For each query f sent to the challenger, the challenger computes $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$ and returns sk_f to $\mathcal{A}_1$.*
- ***Challenge Phase:*** *$\mathcal{A}_1$ outputs two messages m_0, m_1, along with an auxiliary information aux. If there exists a function f queried by $\mathcal{A}_1$ such that $f(m_0) \neq f(m_1)$, the challenger aborts the game. Else, it randomly chooses $b \in \{0, 1\}$ and computes a ciphertext $\mathsf{ct}^* = \mathsf{Enc}(\mathsf{mpk}, m_b)$ and sends it to $\mathcal{A}_1$.*
- ***Post-Challenge Query Phase:*** *This is similar to the pre-challenge query phase. The adversary $\mathcal{A}_1$ is allowed to send polynomially many key queries. For each query f, if $f(m_0) \neq f(m_1)$, the challenger sends $\perp$. Else, computes $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$ and returns sk_f to $\mathcal{A}_1$.*
- ***First Response Phase:*** *$\mathcal{A}_1$ computes a state st such that $|st| \leq S$.*
- ***Second Response Phase:*** *$\mathcal{A}_2$ receives (mpk, aux, st) and can make the following query.*
  - ***Key Query Phase:*** *In this phase, $\mathcal{A}_2$ is allowed to make **a single distinguishing key** query but polynomially many queries for non-distinguishing keys. The challenger computes $\mathsf{sk}_f \leftarrow \mathsf{KeyGen}(\mathsf{msk}, f)$ and returns sk_f to $\mathcal{A}_2$.*

  Finally, $\mathcal{A}_2$ outputs b'. $\mathcal{A}$ wins the experiment if $b = b'$.

An FE scheme is said to be incompressible secure if for all PPT adversaries $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, there exists a negligible function* $\text{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$\Pr[\mathcal{A} \text{ wins in the above experiment}] \leq \frac{1}{2} + \text{negl}(\lambda)$$

We consider two stronger variants of the above security - strongly incompressible and semi-strongly incompressible. In the strongly incompressible version, $\mathcal{A}_2$ is provided with the master secret msk whereas in the semi-strongly incompressible version, it is allowed to make polynomially many distinguishing key queries.

▶ **Definition 16** (Strongly Incompressible FE Security)**.** *An FE scheme is said to be strongly incompressible secure if for all PPT adversaries* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, there exists a negligible function* $\text{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$\Pr[\mathcal{A} \text{ wins in the Incompressible } \mathsf{FE} \text{ experiment}] \leq \frac{1}{2} + \text{negl}(\lambda)$$

provided the second adversary $\mathcal{A}_2$ *is also given the* ***master secret key*** msk *at the beginning of Second Response Phase.*

▶ **Definition 17** (Semi-Strongly Incompressible FE Security)**.** *An FE scheme is said to be semi-strongly incompressible secure if for all PPT adversaries* $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$*, there exists a negligible function* $\text{negl}(\cdot)$ *such that for all* $\lambda \in \mathbb{N}$*,*

$$\Pr[\mathcal{A} \text{ wins in the Incompressible } \mathsf{FE} \text{ experiment}] \leq \frac{1}{2} + \text{negl}(\lambda)$$

provided the second adversary $\mathcal{A}_2$ *is allowed to make* ***polynomially many distinguishing key queries*** *in the Key Query Phase.*

We can consider similar incompressible security notions for IBE and ABE schemes.

4 Rate-$\frac{1}{2}$ Incompressible FE with Short Keys and Semi-Strong Security

In this section, we present a *semi-strong incompressible* FE scheme, using a regular FE scheme along with an incompressible SKE and regular SKE scheme. If the underlying FE has ct-rate of $r \in [0, 1]$, then our incompressible FE scheme has an $\frac{r}{2}$ ct-rate.

Our Construction

Our FE scheme supports functions from the class $\{\mathcal{F}_n : \mathcal{N}_n \to \{0,1\}\}_n$ such that any function in this class has a $\mathsf{polylog}(|\mathcal{N}_n|)$ depth circuit using the following primitives. Throughout S denotes the (in)compressibility parameter.

- $\mathsf{IncSKE} = (\mathsf{IncSKE.Setup}, \mathsf{IncSKE.Enc}, \mathsf{IncSKE.Dec})$ be an incompressible SKE. We define $\alpha = \alpha(\lambda, S)$ as the length of the secret key of this scheme and $\beta = \beta(\lambda, S)$ as the length of the ciphertext for a message of length λ encrypted using this scheme.
- $\mathsf{SKE} = (\mathsf{SKE.Setup}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ be an SKE scheme with secret keys of length λ.
- $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.Keygen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ be an FE scheme with message spaces $\{\tilde{\mathcal{N}}_n\}_n$ and function space $\{\tilde{\mathcal{F}}_n\}_n$ that contains the decryption circuits of IncSKE and SKE.

Let $\tilde{n} = \tilde{n}(\lambda, n, S)$ be the lexicographically smallest functionality index such that $\tilde{\mathcal{F}}_{\tilde{n}}$ contains $\mathcal{F}_n$, the decryption circuit of SKE and IncSKE. We describe the algorithms for our incompressible FE scheme below.

- $\mathsf{Setup}(1^\lambda, 1^S, 1^n)$: The setup algorithm samples a master public key and a master secret key of the FE scheme by computing $(\mathsf{fe.mpk}, \mathsf{fe.msk}) \leftarrow \mathsf{FE.Setup}(1^\lambda, 1^{\tilde{n}})$. It generates a secret key of the incompressible SKE scheme by computing $\mathsf{inc.sk} \leftarrow \mathsf{IncSKE.Setup}(1^\lambda, 1^S, 1^\lambda)$ and two secret keys of the SKE scheme by computing $\mathsf{ske.sk} \leftarrow \mathsf{SKE.Setup}(1^\lambda)$ and $\mathsf{ske.sk}' \leftarrow \mathsf{SKE.Setup}(1^\lambda)$. It sets $\mathsf{mpk} = \mathsf{fe.mpk}$ and $\mathsf{msk} = (\mathsf{fe.msk}, \mathsf{inc.sk}, \mathsf{ske.sk}, \mathsf{ske.sk}')$.
- $\mathsf{KeyGen}(\mathsf{msk}, f)$: Let $\mathsf{msk} = (\mathsf{fe.msk}, \mathsf{inc.sk}, \mathsf{ske.sk}, \mathsf{ske.sk}')$. The key generation algorithm first computes ciphertext $\mathsf{ske.ct} = \mathsf{SKE.Enc}(\mathsf{ske.sk}, (0, 0^\alpha))$ and $\mathsf{ske.ct}' = \mathsf{SKE.Enc}(\mathsf{ske.sk}', 0)$. Then, it generates a key of the FE scheme by computing $\mathsf{fe.sk}_f \leftarrow \mathsf{FE.Keygen}(\mathsf{fe.msk}, C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'})$ where the function $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$ is described in Figure 1.

Input: Messages m, SKE key $\mathsf{ske.sk}$, incompressible ciphertext $\mathsf{inc.ct}$ and a flag b
Hardwired: Function f, two SKE ciphertexts $\mathsf{ske.ct}, \mathsf{ske.ct}'$
Output: y

1. Check if $b = 0$:
   – If yes, output $y = f(m)$.
2. Otherwise, compute $(flag, \mathsf{inc.sk}) \leftarrow \mathsf{SKE.Dec}(\mathsf{ske.sk}, \mathsf{ske.ct})$.
3. Check if $flag = 0$:
   – If yes, output $y = f(m)$.
4. Otherwise, compute $\mathsf{ske.sk}' \leftarrow \mathsf{IncSKE.Dec}(\mathsf{inc.sk}, \mathsf{inc.ct})$.
5. Output $y = \mathsf{SKE.Dec}(\mathsf{ske.sk}', \mathsf{ske.ct}')$.

Figure 1 Description of $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$.

- $\mathsf{Enc}(\mathsf{mpk}, m)$: Let $\mathsf{mpk} = \mathsf{fe.mpk}$. The encryption algorithm randomly samples $\mathsf{inc.ct} \leftarrow \{0,1\}^\beta$ and generates $\mathsf{fe.ct} \leftarrow \mathsf{FE.Enc}(\mathsf{fe.mpk}, (m, \perp, \mathsf{inc.ct}, 0))$[10]. It returns $\mathsf{fe.ct}$.
- $\mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct})$: Let $\mathsf{sk}_f = \mathsf{fe.sk}_f$. The decryption algorithm decrypts the ciphertext by computing $m = \mathsf{FE.Dec}(\mathsf{fe.sk}_f, \mathsf{ct})$. It returns m.

Correctness

A ciphertext corresponding to a message m in the above scheme is an encryption of $(m, \perp, \mathsf{inc.ct}, 0)$ using the FE scheme, i.e., $\mathsf{ct} \leftarrow \mathsf{FE.Enc}(\mathsf{fe.mpk}, (m, \perp, \mathsf{inc.ct}, 0))$ where $\mathsf{inc.ct}$ is a truly random string.

Consider that a decryptor possesses the secret key sk_f for some function f. The secret key sk_f is a key $\mathsf{fe.sk}_f$ of the underlying FE scheme for the circuit $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$ which takes as input in $(m, \mathsf{ske.sk}, \mathsf{inc.ct}, b)$ and outputs $f(m)$ if $b = 0$. The decryption algorithm simply computes the decryption algorithm of the underlying FE scheme on the ciphertext ct using $\mathsf{fe.sk}_f$ and returns the output. Since, b is set to 0 in the ciphertext ct, decrypting ct using $\mathsf{fe.sk}_f$ gives $f(m)$ due to the correctness of the underlying FE scheme.

Size of the master public key

The master public key of our scheme is $\mathsf{fe.mpk}$. Therefore, the size is $|\mathsf{fe.mpk}|$.

[10] By $\perp$, we mean it sets a null secret key.

Size of the ciphertext

Recall that the ciphertext is an FE encryption of $(m, \perp, \mathsf{inc.ct}, 0)$ where the third component has size β. From the definition of β, it is the length of an incompressible ciphertext which is intended encrypt an SKE secret key, therefore, $\beta = S + \mathsf{poly}(\lambda)$ using Dziembowski's construction (see Theorem 7). So, the size of $(m, \perp, \mathsf{inc.ct}, 0)$ is $|m| + S + \mathsf{poly}(\lambda)$. Assuming that FE has ct-rate of r, the size of $\mathsf{fe.ct}$ is also $r^{-1} \cdot (|m| + S + \mathsf{poly}(\lambda))$. Therefore, the rate of the proposed scheme is

$$\frac{r \cdot |m|}{(|m| + S + \mathsf{poly}(\lambda))} = \frac{r}{\left(1 + \frac{S}{|m|} + \frac{\mathsf{poly}(\lambda)}{|m|}\right)}$$

Size of the secret keys

The size of the secret key associated with a function f is $|\mathsf{fe.sk}_f|$ where $\mathsf{fe.sk}_f$ is an FE secret key for the circuit $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$. To use this secret key, the decryption algorithm would require the entire description of $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$ (see Theorem 9 and Theorem 10).

In addition to the description of f and the decryption circuits of SKE and IncSKE, the circuit $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$ contains two SKE ciphertexts generated during the key generation process. Therefore, $\mathsf{fe.sk}_f$ should contain both an FE secret key for the circuit $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$ and the two ciphertexts $\mathsf{ske.ct}, \mathsf{ske.ct}'$. The first SKE ciphertext is an encryption of an incompressible SKE scheme's secret key $\mathsf{inc.sk}$ along with a bit $flag$, and the second is just a bit y. Using Dziembowski's construction (see Theorem 7), we have $|\mathsf{inc.sk}| = \mathsf{poly}(\lambda)$. Therefore, $|\mathsf{fe.sk}_f|$ is the size of an FE secret key associated with $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$, with an additional $\mathsf{poly}(\lambda)$.

▶ **Theorem 18.** *Assuming* $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.Keygen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ *is an* ct*-rate-*r *adaptively (selectively) secure FE scheme,* $\mathsf{SKE} = (\mathsf{SKE.Setup}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ *is a secure SKE scheme against unbounded number of ciphertexts and* $\mathsf{IncSKE} = (\mathsf{IncSKE.Setup}, \mathsf{IncSKE.Enc}, \mathsf{IncSKE.Dec})$ *be a secure incompressible SKE scheme, then the above construction is an adaptively (selectively) secure semi-strongly incompressible FE scheme. Also,*

$$|\mathsf{mpk}| = |\mathsf{fe.mpk}|, \quad |\mathsf{sk}| = |\mathsf{fe.sk}|, \quad |\mathsf{ct}| = (|m| + S + \mathsf{poly}(\lambda))/r$$

Proof-Sketch. The proof proceeds via a sequence of hybrid experiments.

- H_0: This corresponds to the adaptive semi-strong incompressibility security game. The adversary first receives the master public key mpk. It queries for polynomially many non-distinguishing keys, and receives the corresponding secret keys. Then, it sends its challenge messages m_0, m_1 and receives the challenge ciphertext which is encryption of m_b. Again, the adversary queries for polynomially many non-distinguishing keys, and receives the corresponding secret keys. It then compresses its state, and queries for both distinguishing and non-distinguishing functions. Upon receiving the secret keys, it must guess which message was encrypted.
- H_1 : In this hybrid, the challenger keeps the challenge ciphertext same as before, but modifies the *distinguishing* secret keys. For every *distinguishing* key query for some function f, it encrypts $f(m_b)$ instead of 0 to generate $\mathsf{ske.ct}'$.
  Note that H_0 and H_1 are indistinguishable due to SKE security. This is because the attacker does not get access to the SKE secret key, except in the form of ciphertexts as part of functional secret keys. We highlight we know the value $f(m_b)$ because the adversary is allowed to make distinguishing key queries in the second phase only.

- H_2 : In this hybrid, the challenger generates $\mathsf{inc.ct}^*$ by encrypting $\mathsf{ske.sk}'$ instead of sampling a truly random string. The rest of the game proceeds as before.
  Indistinguishability of hybrids H_1 and H_2 follows from standard indistinguishability-based security of the incompressible SKE scheme. This is because the secret key $\mathsf{inc.sk}$ is used only during the generation of $\mathsf{inc.ct}^*$.
- H_3 : In this hybrid experiment, the challenger keeps the challenge ciphertext same as before, but modifies the *distinguishing* secret keys. For every *distinguishing* key query for some function f, it encrypts $(1, \mathsf{inc.sk})$ instead of $(0, 0^\alpha)$ to generate $\mathsf{ske.ct}$.
  Note that H_2 and H_3 are indistinguishable due to SKE security. This is because the attacker does not get access to the SKE secret key, except in the form of ciphertexts as part of functional secret keys.
- H_4 : In this hybrid, the challenger modifies the challenge ciphertext ct^* by encrypting $(m_0, \mathsf{ske.sk}, \mathsf{inc.ct}^*, 1)$ instead of $(m_b, \bot, \mathsf{inc.ct}^*, 0)$. The rest of the game proceeds as before.
  Indistinguishability of hybrids H_3 and H_4 follows from adaptive indistinguishability-based security of FE, as the output of the function stays the same for all key queries. This is because in H_4, the non-distinguishing keys would decrypt the challenge ciphertext to $f(m_0)$ (since b is set to 1 and $flag = 0$), which is equal to $f(m_b)$. Whereas, the distinguishing keys would decrypt to $f(m_b)$ (since b is set to 1 and $flag = 1$, see Figure 1).
- H_5 : In this hybrid, the challenger reverts to generating $\mathsf{inc.ct}^*$ by sampling a truly random string. The rest of the game proceeds as before.
  Indistinguishability of hybrids H_4 and H_5 follows from the *incompressible* indistinguishability-based security of the incompressible SKE scheme. This is because the information $\mathsf{inc.sk}$ is needed to generate distinguishing keys in the second phase.
  Finally, note that in H_5, the bit b is used only in the second phase while generating distinguishing keys. That is, $\mathsf{ske.ct}'$ is generated by encrypting $f(m_b)$. Therefore, the winning probability for any adversary in G_5 depends on the standard indistinguishability-based security of the regular SKE scheme. ◀

The proof for the above theorem is provided in the full version [18]. Using Theorem 9, we have the following corollary.

▶ **Corollary 19.** *Assuming the existence of selectively secure FE for circuits, there exists a* ct*-rate-*$\frac{1}{4}$ *adaptively secure semi-strongly incompressible FE scheme for circuits. Also,*

$$|\mathsf{mpk}| = \mathsf{poly}(\lambda), \quad |\mathsf{sk}_f| = \mathsf{poly}(\lambda), \quad |\mathsf{ct}| = 2(|m| + S) + \mathsf{poly}(\lambda)$$

Using Theorem 10, we have the following corollary.

▶ **Corollary 20.** *Assuming the existence of selectively secure FE for circuits, there exists a* ct*-rate-*$\frac{1}{2}$ *selectively secure semi-strongly incompressible FE scheme for circuits. Also,*

$$|\mathsf{mpk}| = \mathsf{poly}(\lambda), \quad |\mathsf{sk}_f| = \mathsf{poly}(\lambda), \quad |\mathsf{ct}| = |m| + S + \mathsf{poly}(\lambda)$$

5 Rate-1 Incompressible FE with Large Keys and Semi-Strong Security

In this section, we present a *selective semi-strong incompressible* FE scheme, using a (regular) public key FE, together with other standard cryptographic primitives. If the underlying FE scheme has ct-rate-1, then our incompressible FE scheme also has ct-rate-1. Although the functional key sizes are larger.

Our construction

For simplicity, we consider boolean-valued functions. To design an incompressible FE supporting function class $\{\mathcal{F}_n : \{0,1\}^n \to \{0,1\}\}_{n\in\mathbb{N}}$, we use the following primitives. Throughout S denotes the (in)compressibility parameter.

- $\mathsf{Ext} : \{0,1\}^d \times \{0,1\}^\ell \to \{0,1\}^\lambda$ be a strong average-min entropy randomness extractor, where $d = d(\lambda)$ and $\ell = \ell(\lambda, S)$.
- $\mathsf{SKE} = (\mathsf{SKE.Setup}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ be an SKE scheme that can encrypt messages of length $n + d + 1$. (Here n is the FE message length and d is the seed length. W.l.o.g., we assume the secret key length to be λ.)
- $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.Keygen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ be an FE scheme with message spaces $\{\{0,1\}^n\}_n$ and function space $\left\{\tilde{\mathcal{F}}_n\right\}_n$.
  Let $\tilde{n} = \max(n, S) + \lambda + 1$. We require the function class $\tilde{\mathcal{F}}_{\tilde{n}}$ to be such that it contains $\mathcal{F}_n$, and the decryption circuit of SKE and extractor Ext.

We describe the algorithms for our incompressible FE scheme below.

- $\mathsf{Setup}(1^\lambda, 1^S, 1^n)$: Let $\tilde{n} = \max(n, S) + \lambda + 1$. The setup algorithm samples a master public and secret key of the FE scheme by sampling $(\mathsf{fe.msk}, \mathsf{fe.mpk}) \leftarrow \mathsf{FE.Setup}(1^\lambda, 1^{\tilde{n}})$. It also samples an SKE secret key $\mathsf{ske.sk} \leftarrow \mathsf{SKE.Setup}(1^\lambda)$. It sets $\mathsf{msk} = (\mathsf{ske.sk}, \mathsf{fe.msk})$ and $\mathsf{mpk} = \mathsf{fe.mpk}$.
- $\mathsf{KeyGen}(\mathsf{msk}, f)$: Parse master key as $\mathsf{msk} = (\mathsf{ske.sk}, \mathsf{fe.msk})$. The key generation algorithm first computes ciphertext $\mathsf{ske.ct} = \mathsf{SKE.Enc}(\mathsf{ske.sk}, \mathbf{0})$. That is, it encrypts $n+d+1$ zeros. Then, it generates an FE secret key by computing $\mathsf{fe.sk}_f \leftarrow \mathsf{FE.Keygen}(\mathsf{fe.msk}, C_{f,\mathsf{ske.ct}})$, where C is described in Figure 2.

Input: $x \in \{0,1\}^{\max(n,S)}$, an SKE secret key $\mathsf{ske.sk} \in \{0,1\}^\lambda$, bit $\mathsf{flag} \in \{0,1\}$.
Hardwired: Function f, an SKE ciphertext $\mathsf{ske.ct}$.
Output: $y \in \{0,1\}$.

1. Check if $\mathsf{flag} = 0$:
   – If yes, parse x as an n-bit message m, and output $f(m)$.
   That is, if $S > n$, then $m = x[1:n]$, else $m = x$.
2. Otherwise, compute $(b, s, v) = \mathsf{SKE.Dec}(\mathsf{ske.sk}, \mathsf{ske.ct})$.
   Here $b \in \{0,1\}$, $s \in \{0,1\}^d, v \in \{0,1\}^n$.
3. Check if $b = 0$:
   – If yes, then output first bit of s, i.e. $s[1]$.
4. Otherwise, parse x as an S-bit source R, and output $f(v \oplus \mathsf{Ext}_s(R))$.
   That is, if $S < n$, then set $R = x[1:S]$, else set $R = x$.
   Then compute message $m = v \oplus \mathsf{Ext}_s(R)$ and output $f(m)$.

Figure 2 Description of $C_{f,\mathsf{ske.ct}}$.

- $\mathsf{Enc}(\mathsf{msk}, m)$: The encryption algorithm generates the ciphertext by computing $\mathsf{fe.ct} \leftarrow \mathsf{FE.Enc}(\mathsf{msk}, (m, \perp, 0))$. It outputs $\mathsf{fe.ct}$. (We point out that the message m is padded to be of length S in case $S > n$. Also, by $\perp$, we mean it sets a null secret key.)
- $\mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct})$: The decryption algorithm decrypts the ciphertext by computing $m = \mathsf{FE.Dec}(\mathsf{sk}_f, \mathsf{ct})$. It returns m.

Correctness

Follows immediately from the construction.

Size of the master public key

The master public key of our scheme is $\mathsf{fe.mpk}$. Therefore, the size is $|\mathsf{fe.mpk}|$.

Size of the ciphertext

The size of the message encrypted by $\mathsf{FE.Enc}$ is $\tilde{n} = \max(n, S) + \lambda + 1$. Now if the base FE scheme has ct-rate-1, then ciphertext size in our scheme is also $\max(S, n) + \mathsf{poly}(\lambda)$.

Size of the secret key

The size of the secret key associated with a function f is $|\mathsf{fe.sk}_f|$ where $\mathsf{fe.sk}_f$ is an FE secret key for the circuit $C_{f,\mathsf{ske.ct}}$. To use this secret key, the decryption algorithm would require the entire description of $C_{f,\mathsf{ske.ct}}$ (see Theorem 10).

In addition to the description of f and the decryption circuit of SKE, the circuit $C_{f,\mathsf{ske.ct}}$ contains an SKE ciphertext generated during the key generation process. So, $\mathsf{fe.sk}_f$ should contain both an FE secret key for the circuit $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$ and the ciphertext $\mathsf{ske.ct}$. The SKE ciphertext is an encryption of a message of length $n + d + 1$ where d is the seed length of the extractor[11] and n is the length of the message. Therefore, $|\mathsf{fe.sk}_f|$ is equal to $n + d + 1$ plus the size of an FE secret key associated with $C_{f,\mathsf{ske.ct}}$.

▶ **Theorem 21.** *Assuming* $\mathsf{SKE} = (\mathsf{SKE.Setup}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ *is IND-CPA secure,* $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.Keygen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ *is a selectively secure FE with* ct*-rate-*r*, then the above FE construction is a selective semi-strong incompressible FE scheme. Also,*

$$|\mathsf{mpk}| = |\mathsf{fe.mpk}|, \quad |\mathsf{sk}| = |\mathsf{fe.sk}| + |\mathsf{ske.ct}|, \quad |\mathsf{ct}| = (\max(n, S) + \mathsf{poly}(\lambda))/r$$

Proof-Sketch. The proof proceeds via a sequence of hybrid experiments.

- H_0: This corresponds to the selective semi-strong incompressibility security game. The adversary first sends its challenge messages m_0, m_1 and receives the challenge ciphertext which is encryption of m_b. Next, the adversary queries for polynomially many non-distinguishing keys, and receives the corresponding secret keys. It then compresses its state, and queries for both distinguishing and non-distinguishing functions. Upon receiving the secret keys, it must guess which message was encrypted.
- H_1 : In this hybrid, the challenger keeps the challenge ciphertext same as before, but modifies the secret keys. During setup, the challenger samples a *single* random string $R \leftarrow \{0,1\}^S$ as the extractor source, and a *single* random string $s \leftarrow \{0,1\}^d$ as the extractor seed. Now, instead of using an SKE encryption of $\mathbf{0}$ within each secret key, the challenger does the following:
  - For every *non-distinguishing* key query for some function f, it encrypts $(0, f(m_0), \mathbf{0})$. That is, the second message bit is $f(m_0)$ and rest are still all zeros.
  - For every *distinguishing* key query for some function f, it sets $v = m_b \oplus \mathsf{Ext}_s(R)$, and computes the SKE ciphertext as to be an encryption of $(1, s, v)$.

[11] d is equal to $\mathsf{poly}(\lambda)$, see Theorem 2

Note that H_0 and H_1 are indistinguishable due to SKE security. This is because the attacker does not get access to the SKE secret key, except in the form of ciphertexts as part of functional secret keys. We highlight that since we need to know $f(m_0)$ to answer even a non-distinguishing key query, thus we can only prove selective security of our incompressible FE scheme.

- H_2 : In this hybrid, the challenger encrypts $(R, \mathsf{ske.sk}, 1)$ instead of $(m_b, \perp, 0)$. (Note that here we are overloading notation and consider that there is a fixed deterministic way, e.g. by padding, to write R and m as $\max(n, S)$-bit strings.) The rest of the game proceeds as before.
  Indistinguishability of hybrids H_1 and H_2 follows standard selective indistinguishability-based security of FE, as the output of the function stays the same for all key queries. This is because for non-distinguishing keys we know that $f(m_0) = f(m_b)$, while for non-distinguishing keys, the function output is still $f(m_b)$.
- H_3 : In this hybrid experiment, v is chosen uniformly at random (instead of setting it as $m_b \oplus \mathsf{Ext}_s(R)$. Using the strong extractability guarantee of the extractor, we can argue that H_2 and H_3 are negligibly close.
  Finally, note that the bit b is not used in H_3, and therefore the adversary has no advantage in this experiment. ◀

The detailed proof for the above theorem is provided in the full version [18]. Using Theorem 10, we have the following corollary.

▶ **Corollary 22.** *Assuming the existence of* ct*-rate-*$\frac{1}{2}$ *selectively secure FE for circuits, there exists a* ct*-rate-*$\frac{1}{2}$ *selectively secure semi-strongly incompressible FE scheme for circuits. Also,*

$$|\mathsf{mpk}| = \mathsf{poly}(\lambda), \quad |\mathsf{sk}_f| = |m| + \mathsf{poly}(\lambda), \quad |\mathsf{ct}| = |m| + S + \mathsf{poly}(\lambda)$$

6 Rate-1 Incompressible FE with Short Keys and Standard Security

In this section, we present a *selective (standard) incompressible* FE scheme, using a (regular) public key FE, together with other standard cryptographic primitives. If the underlying FE scheme has ct-rate-1, then our incompressible FE scheme also has ct-rate-1. Interestingly, we show that if the underlying FE scheme has short keys, then our resulting incompressible FE scheme has short keys as well. While it might seem to contradict the lower bound [3], we observe that since we only prove standard incompressibility security (where only one non-distinguishing key gets corrupted) and the functions supported by our FE scheme has single-bit output, thus this does not contradict the result. This is because the lower bound actually states that if an incompressible FE scheme has ct-rate-1, then the size of the functional secret key must grow with the output length of the function, and not necessarily the input/message length.

Our construction

As discussed in the overview, this construction relies on the same ingredients as our FE construction from Section 5. The core difference is that now the functions that we create secret keys for change slightly. Below we describe the algorithms where we make modifications, and underline all the changes.

- $\mathsf{Setup}(1^\lambda, 1^S, 1^n)$: This is the same as in Section 5.

- $\mathsf{KeyGen}(\mathsf{msk}, f)$: Parse master key as $\mathsf{msk} = (\mathsf{ske.sk}, \mathsf{fe.msk})$. The key generation algorithm first computes ciphertext $\mathsf{ske.ct} = \mathsf{SKE.Enc}(\mathsf{ske.sk}, \underline{\mathbf{0}})$. That is, it encrypts $d+2$ zeros. Then, it generates an FE secret key by computing $\mathsf{fe.sk}_f \leftarrow \mathsf{FE.Keygen}(\mathsf{fe.msk}, C'_{f,\mathsf{ske.ct}})$, where C is described in Figure 3.

Input: $x \in \{0,1\}^{\max(n,S)}$, an SKE secret key $\mathsf{ske.sk} \in \{0,1\}^{\lambda}$, bit $\mathsf{flag} \in \{0,1\}$.
Hardwired: Function f, an SKE ciphertext $\mathsf{ske.ct}$.
Output: $y \in \{0,1\}$.

1. Check if $\mathsf{flag} = 0$:
   – If yes, parse x as an n-bit message m, and output $f(m)$.
2. Otherwise, compute $(b, s, v) = \mathsf{SKE.Dec}(\mathsf{ske.sk}, \mathsf{ske.ct})$.
   Here $b \in \{0,1\}, s \in \{0,1\}^d, \underline{v \in \{0,1\}}$.
3. Check if $b = 0$:
   – If yes, then output first bit of s, i.e. $s[1]$.
4. Otherwise, parse x as an S-bit source R, and output $\underline{v \oplus \mathsf{Ext}_s(R)}$.
   That is, it no longer computes f in this case.

Figure 3 Description of $C'_{f,\mathsf{ske.ct}}$.

- $\mathsf{Enc}(\mathsf{msk}, m), \mathsf{Dec}(\mathsf{sk}_f, \mathsf{ct})$: The encryption and decryption algorithms are the same as in Section 5.

Correctness

Follows immediately from the construction.

Size of the master public key and ciphertext

This is the same as in Section 5.

Size of the secret key

The size of the secret key associated with a function f is $|\mathsf{fe.sk}_f|$ where $\mathsf{fe.sk}_f$ is an FE secret key for the circuit $C'_{f,\mathsf{ske.ct}}$. To use this secret key, the decryption algorithm would require the entire description of $C'_{f,\mathsf{ske.ct}}$ (see Theorem 10).

In addition to the description of f and the decryption circuit of SKE, the circuit $C_{f,\mathsf{ske.ct}}$ contains an SKE ciphertext generated during the key generation process. So, $\mathsf{fe.sk}_f$ should contain both an FE secret key for the circuit $C_{f,\mathsf{ske.ct},\mathsf{ske.ct}'}$ and the ciphertext $\mathsf{ske.ct}$. The SKE ciphertext is an encryption of a message of length $d+2$ where d is the seed length of the extractor[12]. Therefore, $|\mathsf{fe.sk}_f|$ is the size of an FE secret key associated with $C_{f,\mathsf{ske.ct}}$, with an addition $d+1$.

▶ **Theorem 23.** *Assuming* $\mathsf{SKE} = (\mathsf{SKE.Setup}, \mathsf{SKE.Enc}, \mathsf{SKE.Dec})$ *is IND-CPA secure,* $\mathsf{FE} = (\mathsf{FE.Setup}, \mathsf{FE.Keygen}, \mathsf{FE.Enc}, \mathsf{FE.Dec})$ *is a selectively secure FE with* ct*-rate-*r*, then the above FE construction is a selective (standard) incompressible FE scheme. Also,*

$$|\mathsf{mpk}| = |\mathsf{fe.mpk}|, \quad |\mathsf{sk}| = |\mathsf{fe.sk}|, \quad |\mathsf{ct}| = (\max(n, S) + \mathsf{poly}(\lambda))/r$$

[12] d is equal to $\mathsf{poly}(\lambda)$, see Theorem 2

Proof-Sketch. The proof is very similar to the security proof of Theorem 21, except with one change. Since we only prove standard incompressibility of our FE scheme, then we only need to answer a single distinguishing key query. Therefore, rather than programming v as $m_b \oplus \mathsf{Ext}_s(R)$ as in the previous proof, we simply program $v = f(m_b) \oplus \mathsf{Ext}_s(R)$. That is, we only hardwire one function value inside v rather than the full message m_b. Because of this change, we can only prove standard incompressibility security (because we cannot answer more than one non-distinguishing key), but the circuit $C'_{f,\mathsf{ske.ct}}$ (that we have to create a functional key for) now has a succinct description as it only needs f and $\mathsf{ske.ct}$ (which is short anyways). For completeness, we sketch the hybrids below.

- H_0: This corresponds to the selective standard incompressibility security game. The adversary first sends its challenge messages m_0, m_1 and receives the challenge ciphertext which is encryption of m_b. Next, the adversary queries for polynomially many non-distinguishing keys, and receives the corresponding secret keys. It then compresses its state, and queries for one distinguishing function $f\mathsf{st}$ and many non-distinguishing functions. Upon receiving the secret keys, it must guess which message was encrypted.
- H_1 : In this hybrid, the challenger keeps the challenge ciphertext same as before, but modifies the secret keys. During setup, the challenger samples a single random string $R \leftarrow \{0,1\}^S$ as the extractor source, and a random string $s \leftarrow \{0,1\}^d$ as the extractor seed. Now, instead of using an SKE encryption of $\mathbf{0}$ within each secret key, the challenger does the following:
  - For every *non-distinguishing* key query for some function f, it encrypts $(0, f(m_0), \mathbf{0})$. That is, the second message bit is $f(m_0)$ and rest are still all zeros.
  - For *distinguishing* key query f, it sets $v = f(m_b) \oplus \mathsf{Ext}_s(R)$, and computes the SKE ciphertext as to be an encryption of $(1, s, v)$.
- H_2 : In this hybrid, the challenger encrypts $(R, \mathsf{ske.sk}, 1)$ instead of $(m_b, \perp, 0)$. The rest of the game proceeds as before.
- H_3 : In this hybrid experiment, v is chosen uniformly at random (instead of setting it as $f(m_b) \oplus \mathsf{Ext}_s(R)$.

  Finally, note that the bit b is not used in H_3, and therefore the adversary has no advantage in this experiment. ◀

The detailed proof for the above theorem is provided in the full version [18]. Using Theorem 10, we have the following corollary.

▶ **Corollary 24.** *Assuming the existence of* ct*-rate-*1 *selectively secure FE for circuits, there exists a* ct*-rate-*1 *selectively secure regular incompressible FE scheme for circuits. Also,*

$$|\mathsf{mpk}| = \mathsf{poly}(\lambda), \quad |\mathsf{sk}_f| = \mathsf{poly}(\lambda), \quad |\mathsf{ct}| = \max(S, n) + \mathsf{poly}(\lambda)$$

▶ Remark 25 (Handling functions with longer outputs, or a fixed number of distinguishing keys). The above FE construction can be naturally extended to support functions with longer outputs, or a fixed number of distinguishing keys. The idea is quite simply to hardwire either the multi-bit function output (in the first case) or the function output for every distinguishing function (in the second case). Note that if the secret keys for the underlying FE scheme are short (independent of the function description), then the above scheme achieves incompressible FE with optimal ciphertexts as well as secret keys.

References

1 Mihir Bellare, Viet Tung Hoang, and Phillip Rogaway. Foundations of garbled circuits. In *CCS '12*, 2012. doi:10.1145/2382196.2382279.

2 John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334, 2007. doi:10.1109/SP.2007.11.

3 Kaartik Bhushan, Rishab Goyal, Venkata Koppula, Varun Narayanan, Manoj Prabhakaran, and Mahesh Sreekumar Rajasree. Leakage-resilient incompressible cryptography: Constructions and barriers. In *Advances in Cryptology–ASIACRYPT*, 2024.

4 Nir Bitansky and Tomer Solomon. Bootstrapping homomorphic encryption via functional encryption. In Yael Tauman Kalai, editor, *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPIcs*, pages 17:1–17:23. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. doi:10.4230/LIPIcs.ITCS.2023.17.

5 Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, 2001.

6 Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit abe and compact garbled circuits. In *Advances in Cryptology–EUROCRYPT 2014: 33rd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Copenhagen, Denmark, May 11-15, 2014. Proceedings 33*, pages 533–556. Springer, 2014. doi:10.1007/978-3-642-55220-5_30.

7 Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: definitions and challenges. In *TCC*, 2011.

8 Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, 2007.

9 Pedro Branco, Nico Döttling, and Jesko Dujmović. Rate-1 Incompressible Encryption from Standard Assumptions. In Eike Kiltz and Vinod Vaikuntanathan, editors, *Theory of Cryptography*, Lecture Notes in Computer Science, pages 33–69, Cham, 2022. Springer Nature Switzerland. doi:10.1007/978-3-031-22365-5_2.

10 Chongwon Cho, Nico Döttling, Sanjam Garg, Divya Gupta, Peihan Miao, and Antigoni Polychroniadou. Laconic oblivious transfer and its applications. In Jonathan Katz and Hovav Shacham, editors, *Advances in Cryptology – CRYPTO 2017 – 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part II*, volume 10402 of *Lecture Notes in Computer Science*, pages 33–65. Springer, 2017. doi:10.1007/978-3-319-63715-0_2.

11 Clifford Cocks. An identity based encryption scheme based on Quadratic Residues. In *Cryptography and Coding, IMA International Conference*, volume 2260 of LNCS, pages 360–363, 2001. doi:10.1007/3-540-45325-3_32.

12 Whitfield Diffie and Martin E. Hellman. New directions in cryptography, 1976.

13 Nico Döttling, Phillip Gajland, and Giulio Malavolta. Laconic function evaluation for turing machines. In Alexandra Boldyreva and Vladimir Kolesnikov, editors, *Public-Key Cryptography – PKC 2023 – 26th IACR International Conference on Practice and Theory of Public-Key Cryptography, Atlanta, GA, USA, May 7-10, 2023, Proceedings, Part II*, volume 13941 of *Lecture Notes in Computer Science*, pages 606–634. Springer, 2023. doi:10.1007/978-3-031-31371-4_21.

14 Stefan Dziembowski. On Forward-Secure Storage. In Cynthia Dwork, editor, *Advances in Cryptology – CRYPTO 2006*, Lecture Notes in Computer Science, pages 251–270, Berlin, Heidelberg, 2006. Springer. doi:10.1007/11818175_15.

15 Sanjam Garg, Omkant Pandey, and Akshayaram Srinivasan. Revisiting the cryptographic hardness of finding a nash equilibrium. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology – CRYPTO 2016 – 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 579–604. Springer, 2016. doi:10.1007/978-3-662-53008-5_20.

16 Shafi Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 555–564, 2013.

17 Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology – CRYPTO 2015 – 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 503–523. Springer, 2015. doi:10.1007/978-3-662-48000-7_25.

18 Rishab Goyal, Venkata Koppula, Mahesh Sreekumar Rajasree, and Aman Verma. Incompressible functional encryption. Cryptology ePrint Archive, Paper 2024/798, 2024. URL: https://eprint.iacr.org/2024/798.

19 Rishab Goyal, Venkata Koppula, Andrew Russell, and Brent Waters. Risky traitor tracing and new differential privacy negative results. In Hovav Shacham and Alexandra Boldyreva, editors, *Advances in Cryptology – CRYPTO 2018 – 38th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2018, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 467–497. Springer, 2018. doi:10.1007/978-3-319-96884-1_16.

20 Rishab Goyal, Venkata Koppula, and Brent Waters. Lockable obfuscation. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*, pages 612–621, 2017. doi:10.1109/FOCS.2017.62.

21 Rishab Goyal, Venkata Koppula, and Brent Waters. Collusion resistant traitor tracing from learning with errors. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 660–670. ACM, 2018. doi:10.1145/3188745.3188844.

22 Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, Ioctober 30 – November 3, 2006*, pages 89–98. ACM, 2006. doi:10.1145/1180405.1180418.

23 Jiaxin Guan, Daniel Wichs, and Mark Zhandry. Incompressible Cryptography. In Orr Dunkelman and Stefan Dziembowski, editors, *Advances in Cryptology – EUROCRYPT 2022*, Lecture Notes in Computer Science, pages 700–730, Cham, 2022. Springer International Publishing. doi:10.1007/978-3-031-06944-4_24.

24 Jiaxin Guan, Daniel Wichs, and Mark Zhandry. Multi-instance randomness extraction and security against bounded-storage mass surveillance. In *Theory of Cryptography Conference*, pages 93–122. Springer, 2023. doi:10.1007/978-3-031-48621-0_4.

25 Aayush Jain, Huijia Lin, and Ji Luo. On the optimal succinctness and efficiency of functional encryption and attribute-based encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 479–510. Springer, 2023. doi:10.1007/978-3-031-30620-4_16.

26 Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from well-founded assumptions. In *STOC*, 2021.

27 Aayush Jain, Huijia Lin, and Amit Sahai. Indistinguishability obfuscation from lpn over f p, dlin, and prgs in nc 0. In *Advances in Cryptology–EUROCRYPT 2022: 41st Annual International Conference on the Theory and Applications of Cryptographic Techniques, Trondheim, Norway, May 30–June 3, 2022, Proceedings, Part I*, pages 670–699. Springer, 2022.

28 Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT*, 2008.

29 Lucas Kowalczyk, Tal Malkin, Jonathan Ullman, and Daniel Wichs. Hardness of non-interactive differential privacy from one-way functions, 2018.

30 Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010.

31 Willy Quach, Hoeteck Wee, and Daniel Wichs. Laconic function evaluation and applications. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 859–870. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00086.

32 Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005. doi:10.1007/11426639_27.

33 Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, 1984.

34 Daniel Wichs and Giorgos Zirdelis. Obfuscating compute-and-compare programs under LWE. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017*, pages 600–611, 2017. doi:10.1109/FOCS.2017.61.

35 Andrew Yao. How to generate and exchange secrets. In *FOCS*, 1986.