Compactness vs Collusion Resistance in Functional Encryption*

Baiyu Li† Daniele Micciancio‡

April 10, 2017

Abstract

We present two general constructions that can be used to combine any two functional encryption (FE) schemes (supporting a bounded number of key queries) into a new functional encryption scheme supporting a larger number of key queries. By using these constructions iteratively, we transform any primitive FE scheme supporting a single functional key query (from a sufficiently general class of functions) and has certain weak compactness properties into a collusion-resistant FE scheme with the same or slightly weaker compactness properties. Together with previously known reductions, this shows that the compact, weakly compact, collusion-resistant, and weakly collusion-resistant versions of FE are all equivalent under polynomial time reductions. These are all FE variants known to imply the existence of indistinguishability obfuscation, and were previously thought to offer slightly different avenues toward the realization of obfuscation from general assumptions. Our results show that they are indeed all equivalent, improving our understanding of the minimal assumptions on functional encryption required to instantiate indistinguishability obfuscation.

1 Introduction

Indistinguishability obfuscation ($i\mathcal{O}$), first formalized in [7] and further investigated in [27], is currently one of the most intriguing notions on the cryptographic landscape, and it has attracted a tremendous amount of attention in the last few years. Since Garg et al. [22] put forward a plausible candidate obfuscation algorithm, $i\mathcal{O}$ has been successfully used to solve a wide range of complex cryptographic problems, including functional encryption [22], deniable encryption [34], and much more (e.g., see [17, 8].) However, the problem of building an obfuscator with a solid proof of security is still far from being solved. The multilinear-map problems [21, 19, 24, 20] underlying most known candidate $i\mathcal{O}$ constructions [22, 12, 6, 5, 33, 25] have recently been subject to attacks [16, 18], and basing $i\mathcal{O}$ on a solid, well-understood standard complexity assumption, has rapidly emerged as perhaps the single most important open problem in the area of cryptographic obfuscation.

An alternative path toward the construction of $i\mathcal{O}$ from standard assumptions has recently been opened by Bitansky and Vaikuntanathan [10] and Ananth and Jain [3], who independently showed that $i\mathcal{O}$ can be built from any (subexponentially secure) public key functional encryption scheme satisfying certain compactness requirements. While general constructions of compact functional encryption (for arbitrary functions) are only known using $i\mathcal{O}$, functional encryption is typically considered a weaker primitive than general $i\mathcal{O}$, or, at very least, a more manageable one, closer to what cryptographers know how to build. In fact, several functional encryption schemes (for restricted, but still rather broad classes of functions) are known achieving various notions of security [26, 15, 36, 2, 13]. We recall that a (public key) functional encryption scheme [11, 32, 35, 1] is an encryption scheme with a special type of *functional* secret decryption keys sk_f (indexed by functions f) such that encrypting a message m (using the public key) and then decrypting the resulting

*Research supported in part by the Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Research Office (ARO) under contract numbers W911NF-15-C-0226 and W911NF-15-C-0236, and the National Science Foundation (NSF) under grant CNS-1528068.

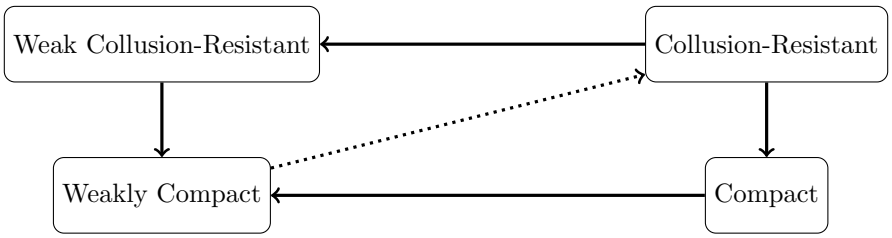
†University of California, San Diego, USA. E-mail: baiyu@cs.ucsd.edu

‡University of California, San Diego, USA. E-mail: daniele@cs.ucsd.edu.

ciphertext using sk_f produces the output of the function $f(m)$, without revealing any other information about the message. Parameters of interest in the study of functional encryption (in relation to obfuscation) are the time (or circuit) complexity of the encryption function t^{Enc} and the number of functional keys sk_f that can be released without compromising the security of the scheme. (See Section 2 for formal definitions and details about security.) Ideally, we would like the encryption time t^{Enc} to depend (polynomially) only on the message size $|m|$ (irrespective of the complexity of the functions f computed during decryption), and the scheme to support an arbitrary polynomial number q of functional decryption keys sk_f. Schemes satisfying these two properties are usually called *compact* (when t^{Enc} is independent of the size $|f|$ of the circuit computing the function), and *collusion-resistant* (when q can be an arbitrary polynomial).

The class of functions f supported by the scheme is also an important parameter, but for simplicity here we focus on schemes for which f can be any polynomial sized circuit. Interestingly, [26] gives a functional encryption scheme (based on standard lattice assumptions) which supports arbitrary functions f. However, the scheme allows to release only $q = 1$ decryption key (i.e., it is not collusion resistant) and the complexity of encryption depends polynomially on the output size and circuit depth of f (i.e., the scheme is not compact.) It is easy to see that any number q of functional decryption keys can always be supported simply by picking q independent public keys. But this makes the complexity of encryption grow linearly with q. So, technically, the constraint that a scheme is collusion-resistant can be reformulated by requiring that the complexity of encryption t^{Enc} is independent of q. One can also consider weaker versions of both compactness and collusion resistance where the complexity of encryption t^{Enc} is required to be just sublinear in $|f|$ or q.

Using this terminology, the main result of [10, 3] states that any (weakly) compact (but not necessarily collusion-resistant) functional encryption scheme can be used to build an $i\mathcal{O}$ obfuscator.[1] In an effort to further reduce (or better understand) the minimal assumptions on functional encryption required to imply obfuscation, the full version of [10] also gives a polynomial reduction from weakly compact functional encryption to (non-compact) weakly collusion-resistant functional encryption. A similar polynomial reduction from compact functional encryption to (non-compact) collusion-resistant functional encryption is also given in [4], where it is suggested that non-compact functional encryption may be easier to achieve, and the reduction is presented as a further step toward basing obfuscation on standard assumptions. In summary, the relation between these four variants of functional encryption (all known to imply $i\mathcal{O}$ by the results of [10, 3]) is summarized by the solid arrows in the following diagram,



where the horizontal implications are trivial (from stronger to weaker constraints on the t^{Enc}) and the vertical implications are from [10, 3].

1.1 Our results and techniques.

In this paper we further investigate the relation between these four variants of functional encryption, and prove (among other things) the following result:

[1] The reduction incurs a loss in security that is exponential in the input size, which can be accounted for by assuming the functional encryption scheme is exponentially hard to break.

Theorem 1. *(Informal) There is a polynomial time reduction from collusion-resistant functional encryption to weakly compact functional encryption.*

This adds precisely the (dotted) diagonal arrow to the previous diagram, showing (by transitivity) that all four variants are equivalent under polynomial time reductions. Technically, proving the above theorem requires showing that any single key ($q = 1$) functional encryption scheme satisfying some weak compactness requirement can be turned into a scheme supporting an arbitrary large polynomial number Q of functional key queries. We do so in a modular way, analyzing two general constructions that can be used to combine two arbitrary functional encryption schemes, which we call the SUM construction and the PRODUCT construction.

- The SUM construction takes two functional encryption schemes FE_1, FE_2 supporting q_1 and q_2 functional key queries, and combines them into a new scheme $FE_1 + FE_2$ supporting $q_1 + q_2$ key queries.
- The PRODUCT construction takes two functional encryption schemes FE_1, FE_2 supporting q_1 and q_2 functional key queries, and combines them into a new scheme $FE_1 \times FE_2$ supporting $q_1 \cdot q_2$ key queries.

The two constructions can be recursively combined in a number of different ways, exhibiting various efficiency/security tradeoffs. For example, Theorem 1 corresponds to starting from a scheme FE_1 supporting a single key ($q_1 = 1$), using the SUM construction $FE_2 = (FE_1 + FE_1)$ to support $q_2 = 2$ keys, and then iterating the PRODUCT construction $(FE_2 \times \cdots \times FE_2)$ precisely $\log(Q)$ times, where Q is the desired number of key queries in the final scheme. (Here for simplicity Q is chosen in advance, but our operations are flexible enough to design a scheme where Q is chosen dynamically by the adversary, and the public key does not depend on Q.)

Another possible instantiation is given by repeatedly squaring the scheme FE_2, i.e., defining $FE_4 = FE_2 \times FE_2$, $FE_{16} = FE_4 \times FE_4$, etc. The squaring operation is repeated $\log(\log((Q))$ times, to yield a scheme supporting Q queries. (Again, we are assuming Q is fixed in advance for simplicity, and our results are easily extended to dynamically chosen Q.) Interestingly (and perhaps surprisingly) this produces a different scheme than the iterated product described before, offering different trade-offs. Specifically, the iterated squaring scheme is no longer compact, and the complexity of encryption now depends on Q. However, the dependence is pretty mild, just double-logarithmic $\log(\log(Q))$, as opposed to linear $O(Q)$ as in the trivial construction. This mild dependence on Q results in different ciphertext lengths: while the iterated product construction produces a ciphertext of length twice as long as that of the underlying single-key FE scheme, the iterated squaring construction produces a ciphertext that is a single encryption of a slightly longer message using the underlying FE scheme.

The methods used by the SUM and PRODUCT constructions are relatively standard: the SUM construction is essentially a formalization and generalization of the trivial "repetition" construction to turn a single-key scheme into one supporting q key queries by picking q public keys. The PRODUCT construction is based on the same type of "chaining" techniques used in many bootstrapping theorems before this work. The main technical novelty of this work is the general modular framework to combine the operations, and the detailed analysis of the efficiency and security of the SUM and PRODUCT construction. We remark that, even for the trivial construction, a detailed analysis is needed in order to evaluate the parameters growth when the constructions are applied iteratively an arbitrary (non-constant) number of times. The details of our SUM and PRODUCT constructions are also particularly simple: both constructions combine the component FE schemes making a simple use of just a length doubling pseudorandom generator. Similar constructions in the literature typically make use of more complex building blocks, like puncturable pseudorandom functions. We consider the simplicity of the constructions in this work as a positive feature.

1.2 Other related work

In a concurrent and independent work, Garg and Srinivasan [23] present a generic transformation from a polynomially-hard single-key weakly-compact FE scheme to a collusion-resistant compact FE scheme.

While their work and ours have similar complexity theoretic implications about the existence of functional encryption schemes, they are quite different at the technical level, and produce very different constructions of (collusion resistant) functional encryption, which may be of independent interest. At a high level, the construction of [23] is based on the use of single-key FE schemes to simulate the role of the *obfuscation* in the $i\mathcal{O}$-to-FE transformations of [10, 3], together with a *prefix puncturable pseudorandom function*. Our constructions are much more direct, and make use of just a simple *pseudorandom generator*. An easy way to see how this leads to very different schemes is that our FE public keys are just the same as the public keys of the underlying (single-key) FE scheme, while the *public keys* of [23] consist of polynomially many *functinal decryption keys* from the underlying (single-key) FE scheme. So, one advantage of our work over [23] is simplicity and, potentially, efficiency.[2] On the other hand the construction of [23] produces a *compact* FE scheme even when starting from a *weakly compact* one, while our construction only preserves the compactness properties, from weakly compact to weakly compact or from fully compact to fully compact.

Our transformations can be adapted to the secret-key settings as well: we can obtain a selective-secure multi-key L-message secret-key FE scheme from a selective-secure single-key L-message secret-key FE scheme while maintaining compactness properties, where L is a-priori fixed. Our transformations also preserve function-privacy properties of the base scheme. However, we are not able to extend such transformations to obtain a multi-message secret-key FE schemes (e.g., L is an arbitrary polynomial in κ.) Details of the secret-key versions of our results are included in Appendix B.

Our definition of a SUM and PRODUCT construction, and their combined use to build different schemes exhibiting a variety of efficiency/security tradeoffs is somehow similar to the work [30], where sum and product constructions are used to build forward secure signature schemes supporting an arbitrary number of key updates, starting from regular (one-time) signatures (i.e., supporting no key updates) and hash functions. However, beside this high level similarity, we deal with completely different cryptographic primitives. The chaining technique used in our product construction has been used many times before in previous bootstrapping theorems for functional encryption, but it is most closely related to the work of [14] where chaining is used in a tree fashion to achieve a hierarchical functional encryption scheme. Our composition approach can be easily adapted to that setting to make the construction and analysis of [14] more modular.

2 Background

We first set up the notation and terminology used in our work.

2.1 Functional Encryption

For notational simplicity, we assume that all randomized algorithms (e.g., the key generation and encryption procedures of a cryptosystem) use precisely κ bits of randomness, where κ is a security parameter. This is without loss of generality, as κ bits of randomness can be used to generate polynomially many pseudorandom bits using a pseudorandom generator.

For simplicity, we consider only public key functional encryption schemes in the following sections, though our transformations and results can be adapted to the secret key setting as well. (See Appendix B.) So from now on we omit "public key" and just say functional encryption. We use the following syntax for functional encryption schemes, where $R = \{0,1\}^\kappa$.

Definition 1. *A Functional Encryption scheme is specified by four sets M, R, I, F (the* message*,* randomness*,* index *and* function *spaces) and four algorithms* (Setup*,* Enc*,* Dec*,* KeyGen) *where*

- $\mathsf{Setup}(sk) = pk$ *is a public key generation algorithm that on input a random secret key $sk \in R$, produces a corresponding public key pk.*

[2] Since the single-key FE schemes underlying [23] and our work lack a known instantiation from standard assumptions, it is hard to make a concrete efficiency comparison between the two schemes. However, based on the asymptotics of the two constructions, one can easily estimate the public keys of [23] to be larger than our public keys at least by a factor κ, proportional to the security parameter.

- $\mathsf{Enc}(pk, m; r) = c$ *is an encryption algorithm that on input a public key* pk*, message* $m \in M$ *and randomness* $r \in R$*, produces a ciphertext* c*.*
- $\mathsf{KeyGen}(sk, f, i) = fk$ *is a functional key derivation algorithm that on input a secret key* sk*, a function* $f \in F$*, and an index* $i \in I$*, produces a functional decryption key* fk *associated to* f*.*
- $\mathsf{Dec}(fk, c) = m'$ *is a decryption algorithm that on input a functional decryption key* fk *and ciphertext* c*, outputs a plaintext message* m'*.*

The scheme is correct if with overwhelming probability (over the choice of $sk, r \in R$*), for any message* $m \in M$*, function* $f \in F$ *and index* $i \in I$*, it holds that*

$$\mathsf{Dec}(\mathsf{KeyGen}(sk, f, i), \mathsf{Enc}(\mathsf{Setup}(sk), m; r)) = f(m).$$

Our syntax for functional encryption schemes slightly differs from the standard one in two respects. First, we identify the randomness used by the public key generation procedure Setup with the master secret key of the scheme. This is without loss of generality, but provides a more convenient syntax for our constructions. The other is that the functional key derivation algorithm KeyGen takes an index i as an additional parameter. The only requirement on this index is that different calls to $\mathsf{KeyGen}(sk, \cdot, i)$ use different values of i. The role of i is simply to put a bound on the number of calls to the functional key derivation algorithm. (In particular, the indices $i \in I$ can be used in any order, and the key derivation algorithm does not need to keep any state.) For example, a functional encryption scheme supporting the release of a single functional key fk will have an index space $I = \{1\}$ of size 1.

Remark 1. The standard definition of *bounded collusion* resistant functional encryption typically allows an arbitrary number of calls to KeyGen, and imposes a bound only on the number of functional decryption keys that are released to the adversary. This always requires the set I to have exponential (or, at least, superpolynomial) size in the security parameter. So, our definition of I-bounded FE is somehow more restrictive than $|I|$-bounded collusion resistance. When the set I has superpolynomial size (e.g., as obtained in the main results of this paper,) it is easy to make KeyGen completely stateless and match the standard FE security definition with only a negligible loss in security, e.g., by letting KeyGen pick $i \in I$ at random, or setting $i = H(f)$ for some collision resistant hash function $H \colon F \to I$.

Security Since our work is primarily motivated by the application of FE to indistinguishability obfuscation [10, 3], we will use an indistinguishability security definition for FE, which is the most relevant one in this context. We follow the indistinguishability security notions as defined in [11], expressed in the functional/equational style of [31, 28]. Security for functional encryption is defined by a game between a challenger and an adversary. Both the challenger and the adversary are reactive programs, modeled by monotone functions: the challenger is a function $\mathcal{H}^{\mathsf{FE}}((m_0, m_1), \{f_i\}_{i \in Q}) = (pk, c, \{fk^i\}_{i \in Q})$ that receives as input a pair of message $(m_0, m_1) \in M^2$ and a collection of function queries $f_i \in F$, and outputs a public key pk, a ciphertext c and a collection of functional keys fk^i, one for each f_i. The adversary is a function $\mathcal{A}(pk, c, \{fk^i\}_{i \in Q}) = ((m_0, m_1), \{f_i\}_{i \in Q}, b')$ that on input a public key pk, ciphertext c and functional keys $\{fk^i\}_{i \in Q}$ outputs a pair of messages (m_0, m_1), function queries $\{f_i\}_{i \in Q}$ and a decision bit b'. Note that $Q \subseteq I$ can be chosen adaptively by the adversary. We recall that, as reactive programs, $\mathcal{H}$ and $\mathcal{A}$ can produce some outputs before receiving all the inputs. (Formally, each of the input or output variables can take a special *undefined* value $\perp$, subject to the natural monotonicity requirements. See [31] for details.)

Security is defined by the following challenger program $\mathcal{H}^{\mathsf{FE}}_b$, parameterized by a bit $b \in \{0, 1\}$:

$$\begin{array}{l}
\mathcal{H}^{\mathsf{FE}}_b((m_0, m_1), \{f_i\}_{i \in Q}) = (pk, c, \{fk^i\}_{i \in Q}) \\
\quad \text{where } sk \leftarrow R,\ r \leftarrow R \\
\qquad\quad pk = \mathsf{Setup}(sk) \\
\qquad\quad c = \mathsf{Enc}(pk, m_b; r) \\
\qquad\quad \text{For all } i \in Q\text{:} \\
\qquad\qquad fk^i = \textbf{if } ((m_0, m_1) = \perp) \vee (f = \perp) \textbf{ then } \perp \\
\qquad\qquad\qquad\quad \textbf{else if } (f_i(m_0) = f_i(m_1)) \textbf{ then } \mathsf{KeyGen}(sk, f_i, i) \textbf{ else } \top
\end{array}$$

By the notation $x \leftarrow R$ we mean the operation of selecting an element uniformly at random from R. Besides, we use the symbol "$\top$" to denote the error condition that the adversary's query is invalid.[3] Security is defined by running the challenger $\mathcal{H}_b^{\mathsf{FE}}$ together with an adversary $\mathcal{A}$ attacking the scheme and trying to guess the value of a randomly chosen bit b. Formally, the FE game $\mathsf{Exp}_{FE}[\mathcal{H}_{(\cdot)}^{\mathsf{FE}}, \mathcal{A}]$ is defined by the following system of equations:

$$\begin{aligned}\mathsf{Exp}_{FE}[\mathcal{H}_{(\cdot)}^{\mathsf{FE}}, \mathcal{A}] &= (b \stackrel{?}{=} b') \\ \text{where } b &\leftarrow \{0,1\} \\ (pk, c, \{fk^i\}_{i\in Q}) &= \mathcal{H}_b^{\mathsf{FE}}((m_0, m_1), \{f_i\}_{i\in Q}) \\ ((m_0, m_1), \{f_i\}_{i\in Q}, b') &= \mathcal{A}(pk, c, \{fk^i\}_{i\in Q})\end{aligned}$$

The output of the game can be obtained by finding the least fixed point of $[\mathcal{H}_b^{\mathsf{FE}}, \mathcal{A}]$, which describes the output when the computation stabilizes. We say that the adversary $\mathcal{A}$ wins the game $\mathsf{Exp}_{FE}[\mathcal{H}_{(\cdot)}^{\mathsf{FE}}, \mathcal{A}]$ if the condition $b = b'$ is satisfied (denoted by the output $\top$), and we define the advantage of $\mathcal{A}$ in breaking the FE scheme FE as

$$\mathsf{Adv}_{\mathsf{FE}}[\mathcal{A}] = \left| 2\Pr\{\mathsf{Exp}_{FE}[\mathcal{H}_{(\cdot)}^{\mathsf{FE}}, \mathcal{A}] = \top\} - 1 \right|.$$

Alternatively, we can let the FE game be parameterized by b and output a bit b':

$$\begin{aligned}[\mathcal{H}_{(b)}^{\mathsf{FE}}, \mathcal{A}] &= b' \\ \text{where } (pk, c, \{fk^i\}_{i\in Q}) &= \mathcal{H}_b^{\mathsf{FE}}((m_0, m_1), \{f_i\}_{i\in Q}, b') \\ ((m_0, m_1), \{f_i\}_{i\in Q}, b') &= \mathcal{A}(pk, c, \{fk^i\}_{i\in Q})\end{aligned}$$

Then the advantage of $\mathcal{A}$ in breaking the FE scheme FE can be defined as

$$\mathsf{Adv}_{\mathsf{FE}}[\mathcal{A}] = \left| \Pr\{[\mathcal{H}_0^{\mathsf{FE}}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_1^{\mathsf{FE}}, \mathcal{A}] = 1\} \right|.$$

The two formulations are easily seen to be perfectly equivalent.

Note that, in the definition of $\mathcal{H}_b^{\mathsf{FE}}$, if $f_i = \perp$ or $m_j = \perp$, then $f_i(m_j) = \perp$. So, this challenger corresponds to a *non-adaptive* security definition where the adversary cannot get any functional key before choosing the challenge messages (m_0, m_1). On the other hand, the public key pk is computed (and given to the adversary) right away. So the (distribution of the) messages (m_0, m_1) may depend on the value of the public key. Alternative definitions can be obtained by setting

- $pk = \textbf{if } ((m_0, m_1) \neq \perp) \textbf{ then } \mathsf{Setup}(sk) \textbf{ else } \perp$, which corresponds to the *selective* (i.e., fully non-adaptive) attack where the adversary has to choose the messages before seeing the public key.
- $fk^i = \mathsf{KeyGen}(sk, f_i, i)$ and $c = \textbf{if } (\forall i. f_i(m_0) = f_i(m_1)) \textbf{ then } \mathsf{Enc}(pk, m_b; r) \textbf{ else } \perp$, which corresponds to allowing function queries (only) before choosing the messages (m_0, m_1).

All our results and constructions are easily adapted to all these different definitional variants, as well as *fully adaptive* settings where message and function queries can be specified in any order, subject to the natural non-triviality requirements.

Definition 2. *A functional encryption scheme FE is $\epsilon(\kappa, q)$-non-adaptively (or selectively/adaptively) secure if for any efficient adversary $\mathcal{A}$ who makes at most q functional key queries, the advantage of $\mathcal{A}$ in the non-adaptive (or selective/adaptive) FE game is bounded by $\mathsf{Adv}_{\mathsf{FE}}[\mathcal{A}] \leq \epsilon(\kappa, q)^{\Omega(1)}$.*

When $\epsilon(\kappa, q)$ is independent of q, we simply say the FE scheme is $\epsilon(\kappa)$-secure. Furthermore, if $\epsilon(\kappa)$ is a negligible function in κ, we sometimes omit it and just say that the FE scheme is secure. We say that a FE scheme is *single-key* secure if $|I| = 1$, and we say that it is *q-key* secure if it is $\epsilon(\kappa, q)$-secure where $\epsilon(\kappa, q)$ is negligible. Finally, one may naturally require a FE scheme to be secure even when a large number of functional keys are released. We say a FE scheme is *collusion-resistant* if it is q-key secure for any polynomial $q(\kappa)$.

[3] Formally we can extend the flat CPO of all functional decryption keys with an "top" element that is greater than all keys, such that the challenger remains a monotone function. Note that, if $m_0 = \perp$ or $m_1 = \perp$ then $(m_0, m_1) = \perp$.

Efficiency For a FE scheme to be useful in applications or when building other cryptographic constructs, we need to measure its efficiency. Several efficiency notions have been considered in the literature, and here we mention those that are used in our work. Let FE be a q-key secure FE scheme with security parameter κ, and let n be the length of messages to be encrypted. Then we say that

- FE is *compact*[4] if the running time t^{Enc} of the encryption procedure Enc is polynomial in n and κ, and it is independent of other parameters.
- FE is *weakly compact*[5] if t^{Enc} is sub-linear in q and the maximal circuit size s of functions in F, and it is polynomial in n and κ.
- FE is *ciphertext-succinct* or simply *succinct* if t^{Enc} is polynomials in n, κ, and the maximal circuit depth d of functions in F.
- FE is *weakly ciphertext-succinct* or simply *weakly succinct* if t^{Enc} is sub-linear in q and polynomial in n, κ, and d, but otherwise independent of the circuit size.
- FE is *weakly collusion-resistant* if t^{Enc} is sub-linear in q.

The notion of compact FE has been considered in [3, 4], and also in [10] under the name of *fully circuit succinctness*. Here we chose the name "compact" to distinguish it from other variants of succinctness notions. It was shown in [3, 10] that a secure compact FE with sub-exponential security for all circuits implies an indistinguishability obfuscation for all circuits.

Succinct FE scheme, a weaker notion, was considered in [26], where their definition was based on ciphertext length, rather than encryption time. They constructed a succinct FE scheme based on standard sub-exponential lattice assumptions. We note that, although our definition is stronger due to using the complexity of encryption, the FE scheme of [26] is still ciphertext-succinct according to our definition.

2.2 Pseudorandom Generators

Our construction assumes the existence of pseudorandom generators that can stretch a short random seed to a polynomially long pseudorandom bit-string.

Definition 3. *Let $\mathsf{G} : R \to S$ be a deterministic function that can be computed in polynomial time. We say that G is a $\mu(\kappa)$-secure pseudorandom generator if for any efficient adversary $\mathcal{A}$ we have*

$$\mathsf{Adv}_{\mathsf{G}}[\mathcal{A}] = \left| \Pr_{s \leftarrow S}\{\mathcal{A}(s) = 1\} - \Pr_{r \leftarrow R}\{\mathcal{A}(\mathsf{G}(r)) = 1\} \right| \leq \mu(\kappa).$$

The quantity $\mathsf{Adv}_{\mathsf{G}}[\mathcal{A}]$ is called the advantage of $\mathcal{A}$ in breaking the PRG G, and it is usually assumed to be less than $\mu(\kappa)$ for some negligible function μ. Also, it is usually assumed that the generator output strings of length $|\mathsf{G}(x)| = \ell(|x|)$ for all $x \in R$, where $\ell(\kappa)$ is some fixed polynominal function of κ. We write $\mathsf{G}(r)$ to denote the output of a pseudorandom generator on input a (randomly chosen) seed r, with the domain and range of G usually defined implicitly by the context. We write $\mathsf{G}_i(r)$ to denote a specific part of the output, i.e., $\mathsf{G}(r) = \mathsf{G}_0(r)\mathsf{G}_1(r)\ldots\mathsf{G}_k(r)$, where the blocks $\mathsf{G}_i(r)$ usually have all the same length.

3 The SUM construction

We describe a simple method to combine two functional encryption schemes FE_0 and FE_1 with index spaces I_0 and I_1, into a new scheme $\mathsf{FE} = \mathsf{FE}_0 + \mathsf{FE}_1$ with index space $I = I_0 + I_1 = \{(b, i) \mid b \in \{0, 1\}, i \in I_b\}$ given by the disjoint union of I_0 and I_1. Let $\mathsf{FE}_b = (\mathsf{Setup}_b, \mathsf{Enc}_b, \mathsf{Dec}_b, \mathsf{KeyGen}_b)$ for $b \in \{0, 1\}$. Then, $\mathsf{FE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{KeyGen})$ is defined as

[4] Also known as *fully (circuit) succinct* in [10].
[5] Also known as *weakly (circuit) succinct* in [10].

- $\mathsf{Setup}(sk) = (\mathsf{Setup}_0(\mathsf{G}_0(sk)), \mathsf{Setup}_1(\mathsf{G}_1(sk)))$
- $\mathsf{Enc}((pk_0, pk_1), m; r) = (\mathsf{Enc}_0(pk_0, m; \mathsf{G}_0(r)), \mathsf{Enc}_1(pk_1, m; \mathsf{G}_1(r)))$
- $\mathsf{Dec}((b, fk), (c_0, c_1)) = \mathsf{Dec}_b(fk, c_b)$
- $\mathsf{KeyGen}(sk, f, (b, i)) = (b, \mathsf{KeyGen}_b(\mathsf{G}_b(sk), f, i))$

for all $sk, r \in R$, $m \in M$, $b \in \{0,1\}$ and $i \in I_b$. Informally, the *SUM* scheme works by generating two public keys (one for each component scheme FE_b), and encrypting each message under both public keys. When applied to two copies of the same scheme $\mathsf{FE}_0 = \mathsf{FE}_1$, this doubles the size of the index space $|I| = 2|I_b|$ (allowing twice as many functional decryption keys,) but at the cost of doubling also the public key and ciphertext size. The complexity of decryption and functional key generation stays essentially the same as that of the component schemes (no doubling, only a small additive increase for multiplexing), as only one of the two ciphertexts gets decrypted.

The correctness and efficiency properties of the SUM construction are summarized in the following theorem, which is easily verified by inspection/substitution.

Theorem 2 (SUM construction)**.** *If (for $i \in \{0,1\}$) FE_i is a correct FE scheme for function class F with index space of size $|I_i|$, public key size ℓ_i^k, ciphertext length ℓ_i^c, and algorithms $\mathsf{Setup}_i, \mathsf{Dec}_i, \mathsf{Enc}_i, \mathsf{KeyGen}_i$ running in time $t_i^{\mathsf{Setup}}, t_i^{\mathsf{Dec}}, t_i^{\mathsf{Enc}}, t_i^{\mathsf{KeyGen}}$ respectively, then $\mathsf{FE} = \mathsf{FE}_0 + \mathsf{FE}_1$ is a correct FE scheme for the same function class F with index space of size $|I| = |I_0| + |I_1|$, public-key size $\ell^k = \ell_0^k + \ell_1^k$, ciphertext length $\ell^c = \ell_0^c + \ell_1^c$, and algorithms $\mathsf{Setup}_i, \mathsf{Dec}_i, \mathsf{Enc}_i, \mathsf{KeyGen}_i$ running in time*

$$\begin{aligned} t^{\mathsf{Setup}} &= t_0^{\mathsf{Setup}} + t_1^{\mathsf{Setup}} + t^{\mathsf{G}} \\ t^{\mathsf{Enc}} &= t_0^{\mathsf{Enc}} + t_1^{\mathsf{Enc}} + t^{\mathsf{G}} \\ t^{\mathsf{Dec}} &= \max\{t_0^{\mathsf{Dec}}, t_1^{\mathsf{Dec}}\} \\ t^{\mathsf{KeyGen}} &= \max\{t_0^{\mathsf{KeyGen}}, t_1^{\mathsf{KeyGen}}\} + t^{\mathsf{G}} \end{aligned}$$

where t^{G} is the running time of the PRG G. In particular, if FE_0 and FE_1 are both succint, then $\mathsf{FE} = \mathsf{FE}_0 + \mathsf{FE}_1$ is also succint.

Security (proved in the next theorem) is not entirely trivial, as it requires a careful use of the pseudorandom generator, but it still follows by a fairly standard hybrid argument. The construction preserves the non-adaptive/selective/adaptive security properties. We prove the non-adaptive version. Proofs for other definitional variants are similar.

Theorem 3 (Security of SUM construction)**.** *If (for $i \in \{0,1\}$) FE_i is non-adaptively $\epsilon_i(\kappa)$-secure, and G is a μ-secure pseudorandom generator, then $\mathsf{FE} = \mathsf{FE}_0 + \mathsf{FE}_1$ is also non-adaptively $\epsilon(\kappa)$-secure for $\epsilon = 4\mu + \epsilon_0 + \epsilon_1$.*

Proof. We build 6 hybrids to reduce the security of the SUM construction $\mathsf{FE}_0 + \mathsf{FE}_1$ to the security of the PRG G and the security of the FE schemes FE_0 and FE_1. We denote a hybrid by $\mathcal{H}_b^{(j)}$ for $b \in \{0,1\}$ and an index j. Like the challenger in a FE game, a hybrid is a monotone function $\mathcal{H}_b^{(j)}((m_0, m_1), \{f_{(h,i)}\}_{(h,i)\in Q}) = (pk, c, \{fk^{(h,i)}\}_{(h,i)\in Q})$, where $Q \subseteq I$ is (adaptively) chosen by the adversary. Let $Q_0 = \{i \mid (0,i) \in Q\} \subseteq I_0$ and $Q_1 = \{i \mid (1,i) \in Q\} \subseteq I_1$. Proofs of lemmas can be found in Appendix A.

$\mathcal{H}_b^{(0)}$: This hybrid is the same as the original challenger $\mathcal{H}_b^{\mathsf{FE}}$ in the FE game for the FE scheme $\mathsf{FE}_0 + \mathsf{FE}_1$. For a fixed $b \in \{0,1\}$, by expanding the SUM construction, we get the following definition of $\mathcal{H}_b^{(0)}$:

$$\mathcal{H}_b^{(0)}((m_0,m_1),\{f_{(h,i)}\}_{(h,i)\in Q}) = (pk, c, \{fk^{(h,i)}\}_{(h,i)\in Q})$$
where
$sk \leftarrow R$, $r \leftarrow R$
$sk_0 = \mathsf{G}_0(sk)$, $sk_1 = \mathsf{G}_1(sk)$
$pk_0 = \mathsf{Setup}_0(sk_0)$, $pk_1 = \mathsf{Setup}_1(sk_1)$, $pk = (pk_0, pk_1)$
$c_0 = \mathsf{Enc}_0(pk_0, m_b; \mathsf{G}_0(r))$, $c_1 = \mathsf{Enc}_1(pk_1, m_b; \mathsf{G}_1(r))$, $c = (c_0, c_1)$
For all $(h,i) \in Q$:
$fk^{(h,i)} =$ **if** $((m_0,m_1) = \bot) \vee (f_{(h,i)} = \bot)$ **then** $\bot$
else if $(f_{(h,i)}(m_0) = f_{(h,i)}(m_1))$ **then** $(h, \mathsf{KeyGen}_h(sk_h, f_{(h,i)}; i))$ **else** $\top$

$\mathcal{H}_b^{(1)}$**:** In this hybrid we replace the PRG outputs by truly random strings. So sk and r are no longer needed and hence we remove them from the hybrid.

$$\mathcal{H}_b^{(1)}((m_0,m_1),\{f_{(h,i)}\}_{(h,i)\in Q}) = (pk, c, \{fk^{(h,i)}\}_{(h,i)\in Q})$$
where
$sk_0 \leftarrow R$, $sk_1 \leftarrow R$, $r_0 \leftarrow R$, $r_1 \leftarrow R$
$pk_0 = \mathsf{Setup}_0(sk_0)$, $pk_1 = \mathsf{Setup}_1(sk_1)$, $pk = (pk_0, pk_1)$
$c_0 = \mathsf{Enc}_0(pk_0, m_b; r_0)$, $c_1 = \mathsf{Enc}_1(pk_1, m_b; r_1)$, $c = (c_0, c_1)$
For all $(h,i) \in Q$:
$fk^{(h,i)} =$ **if** $((m_0,m_1) = \bot) \vee (f_{(h,i)} = \bot)$ **then** $\bot$
else if $(f_{(h,i)}(m_0) = f_{(h,i)}(m_1))$ **then** $(h, \mathsf{KeyGen}_h(sk_h, f_{(h,i)}; i))$ **else** $\top$

Lemma 1. If G is a μ-secure pseudorandom generator, then for any $b \in \{0,1\}$ and adversary $\mathcal{A}$ we have $|\Pr\{[\mathcal{H}_b^{(0)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_b^{(1)}, \mathcal{A}] = 1\}| \leq 2\mu(\kappa)$.

$\mathcal{H}_b^{(2)}$**:** In this hybrid the ciphertext c encrypts both m_0 and m_1:

$$\mathcal{H}_b^{(2)}((m_0,m_1),\{f_{(h,i)}\}_{(h,i)\in Q}) = (pk, c, \{fk^{(h,i)}\}_{(h,i)\in Q})$$
where
$sk_0 \leftarrow R$, $sk_1 \leftarrow R$, $r_0 \leftarrow R$, $r_1 \leftarrow R$
$pk_0 = \mathsf{Setup}_0(sk_0)$, $pk_1 = \mathsf{Setup}_1(sk_1)$, $pk = (pk_0, pk_1)$
$c_0 = \mathsf{Enc}_0(pk_0, m_0; r_0)$, $c_1 = \mathsf{Enc}_1(pk_1, m_1; r_1)$, $c = (c_0, c_1)$
For all $(h,i) \in Q$:
$fk^{(h,i)} =$ **if** $((m_0,m_1) = \bot) \vee (f_{(h,i)} = \bot)$ **then** $\bot$
else if $(f_{(h,i)}(m_0) = f_{(h,i)}(m_1))$ **then** $(h, \mathsf{KeyGen}_h(sk_h, f_{(h,i)}; i))$ **else** $\top$

Lemma 2. If FE_1 is a non-adaptively $\epsilon_1(\kappa)$-secure FE scheme with an index space I_1, then for any adversary $\mathcal{A}$ we have $|\Pr\{[\mathcal{H}_0^{(1)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_0^{(2)}, \mathcal{A}] = 1\}| \leq \epsilon_1(\kappa)$.

By symmetric argument, we can also obtain the following lemma.

Lemma 3. If FE_0 is a non-adaptively $\epsilon_0(\kappa)$-secure FE scheme with an index space I_0, then for any adversary $\mathcal{A}$ we have $|\Pr\{[\mathcal{H}_1^{(1)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_1^{(2)}, \mathcal{A}] = 1\}| \leq \epsilon_0(\kappa)$.

Finally, we observe that the last hybrid $\mathcal{H}_b^{(2)}$ does not depend on the bit b, and therefore $\Pr\{[\mathcal{H}_0^{(2)}, \mathcal{A}] = 1\} = \Pr\{[\mathcal{H}_1^{(2)}, \mathcal{A}] = 1\}$. It follows by triangle inequality that the advantage of adversary $\mathcal{A}$ in breaking the SUM FE scheme is at most $\mathsf{Adv}_{\mathsf{FE}}[\mathcal{A}] = |\Pr\{[\mathcal{H}_0^{(0)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_1^{(0)}, \mathcal{A}] = 1\}| \leq 2\mu + \epsilon_1 + 0 + \epsilon_0 + 2\mu = 4\mu + \epsilon_0 + \epsilon_1$. □

4 The PRODUCT construction

We now define a different method to combine FE_0 and FE_1 into a new scheme $\mathsf{FE} = \mathsf{FE}_0 \times \mathsf{FE}_1$ with index space $I_0 \times I_1$ equal to the cartesian product of the index spaces I_0, I_1 of FE_0 and FE_1. Let $\mathsf{FE}_b = (\mathsf{Setup}_b, \mathsf{Enc}_b, \mathsf{Dec}_b, \mathsf{KeyGen}_b)$ for $b \in \{0,1\}$. First, for each $i \in I_0$, we define a "re-encryption" function $e_i[c, pk] : M \times R \to M$, parameterized by $c \in M$ and $pk \in R$:

$$e_i[c, pk](m, \tilde{r}) = \begin{cases} \mathsf{G}_i(\tilde{r}) \oplus c & \text{if } m = \perp \\ \mathsf{Enc}_1(pk, m; \mathsf{G}_i(\tilde{r})) & \text{otherwise} \end{cases}$$

Then, $\mathsf{FE} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{KeyGen})$ is defined as follows:

- $\mathsf{Setup}(sk) = \mathsf{Setup}_0(\mathsf{G}_0(sk))$
- $\mathsf{Enc}(pk, m; r) = \mathsf{Enc}_0(pk, (m, \mathsf{G}_0(r)); \mathsf{G}_1(r))$
- $\mathsf{Dec}((fk_0, fk_1), c) = \mathsf{Dec}_1(fk_1, \mathsf{Dec}_0(fk_0, c))$
- $\mathsf{KeyGen}(sk, f, (i, j)) = (fk_0^i, fk_1^{(i,j)})$ where

$$\begin{array}{l} sk_0 = \mathsf{G}_0(sk) \\ sk_1^i = \mathsf{G}_i(\mathsf{G}_1(sk)) \\ pk_1^i = \mathsf{Setup}_1(sk_1^i) \\ c_i = \mathsf{G}_i(\mathsf{G}_2(sk)) \\ fk_0^i = \mathsf{KeyGen}_0(sk_0, e_i[c_i, pk_1^i], i) \\ fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f, j) \end{array}$$

The re-encryption function has two modes: in the regular mode where a message m is given, it computes the FE_1 ciphertext of m under a hard-wired public key pk with pseudo-randomness supplied by a random seed from the input; in the special mode where m is not given (denoted by the special symbol $\perp$), it masks a hard-wired ciphertext c with pseudo-randomness derived from the random seed from the input. Note that the special mode is never invoked in a real world execution of the scheme, but it is used in security proofs.

For any set $Q \subseteq I_0 \times I_1$ of functional key query indices, let $\mathrm{proj}_0(Q) = \{i \mid (i,j) \in Q \text{ for some } j \in I_1\}$, and for $i \in I_0$ let $\mathrm{proj}_1^i(Q) = \{j \mid (i,j) \in Q\}$. In addition, let $\mathbb{RE}_{\mathsf{FE}}$ be the class of functions that includes $e_i[c_i, pk_1^i](\cdot,\cdot)$ defined using Enc of a FE scheme FE.

Theorem 4 (PRODUCT construction)**.** *If FE_0 and FE_1 are succinct FE schemes for functions in the classes $\mathbb{RE}_{\mathsf{FE}_0}$ and F respectively, whose index spaces are I_0 and I_1, key sizes are ℓ_0^k and ℓ_1^k, ciphertext lengths are $\ell_0^c(n, \kappa, d_0)$ and $\ell_1^c(n, \kappa, d_1)$, where n is the message length and d_0, d_1 are the maximum depth of functions in $\mathbb{RE}_{\mathsf{FE}_0}, F$, respectively. Then $\mathsf{FE} = \mathsf{FE}_0 \times \mathsf{FE}_1$ is a succinct FE scheme for F with an index space $I_0 \times I_1$, public-key size ℓ_0^k, and ciphertext length $\ell_0^c(n + \kappa, \kappa, d_0)$.*

Moreover, for $i \in \{0,1\}$, let $t_i^{\mathsf{Setup}}, t_i^{\mathsf{Enc}}, t_i^{\mathsf{Dec}}, t_i^{\mathsf{KeyGen}}$ be the running times of $\mathsf{Setup}_i, \mathsf{Enc}_i, \mathsf{Dec}_i, \mathsf{KeyGen}_i$ of FE_i, where $t_i^{\mathsf{Enc}} = t_i^{\mathsf{Enc}}(n, \kappa, d_i)$, and let t^{G} be the running time of the PRG G. Then FE runs in time:

$$\begin{aligned} t^{\mathsf{Setup}} &= t_0^{\mathsf{Setup}} + t^{\mathsf{G}} \\ t^{\mathsf{Enc}} &= t_1^{\mathsf{Enc}}(n + \kappa, \kappa, d_0) + t^{\mathsf{G}} \\ t^{\mathsf{Dec}} &= t_0^{\mathsf{Dec}} + t_1^{\mathsf{Dec}} \\ t^{\mathsf{KeyGen}} &= t_1^{\mathsf{Setup}} + t_0^{\mathsf{KeyGen}} + t_1^{\mathsf{KeyGen}} + 3t^{\mathsf{G}} \end{aligned}$$

Next we state the security of our PRODUCT construction. Again, the analysis can be easily adapted to other (e.g., selective/adaptive) models.

Theorem 5 (Security of PRODUCT construction)**.** *Assume* FE_0 *and* FE_1 *are succinct FE schemes which are non-adaptively* $\epsilon_0(\kappa, q')$*- and* $\epsilon_1(\kappa, q')$*-secure with index spaces* I_0 *and* I_1*. Also assume* G *is a* μ*-secure pseudorandom generator. Then for any efficient adversary* $\mathcal{A}$ *who makes at most* $q \leq |I_0 \times I_1|$ *functional key queries,* $\mathsf{FE} = \mathsf{FE}_0 \times \mathsf{FE}_1$ *is a non-adaptively* $\epsilon(\kappa, q)$*-secure succinct FE scheme for* F*, where*

$$\epsilon(\kappa, q) \leq \max_{q_1+\cdots+q_p=q, q_i \leq |I_1|} \sum_i \epsilon_1(\kappa, q_i) + 2\epsilon_0(\kappa, \min\{q, |I_0|\}) + 12\mu(\kappa).$$

Proof. We build a series of hybrids to reduce the security of $\mathsf{FE}_0 \times \mathsf{FE}_1$ to the security of the PRG and the security of FE schemes FE_0 and FE_1. We denote our hybrids by $\mathcal{H}_b^{(h)}$ for $b \in \{0,1\}$ and h an index, and a hybrid is a monotone function $\mathcal{H}_b^{(h)}((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}) = (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})$. An adversary $\mathcal{A}$ wins the game against $\mathcal{H}_b^{(h)}$ if $[\mathcal{H}_b^{(h)}, \mathcal{A}] = 1$, and its advantage over $\mathcal{H}_b^{(h)}$ is $\mathsf{Adv}[\mathcal{A}]_b^{(h)} = \Pr\{[\mathcal{H}_b^{(h)}, \mathcal{A}] = 1\}$. Proofs of lemmas can be found in Appendix A.

$\mathcal{H}_b^{(0)}$: This is the same as the original challenger $\mathcal{H}_b^{\mathsf{FE}_0 \times \mathsf{FE}_1}$ in the FE game for the scheme $\mathsf{FE}_0 \times \mathsf{FE}_1$. By expanding the PRODUCT construction, we get the following definition of $\mathcal{H}_b^{(0)}$:

$\mathcal{H}_b^{(0)}((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}) = (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})$
where
$sk \leftarrow R$, $r \leftarrow R$
$sk_0 = \mathsf{G}_0(sk)$, $pk = \mathsf{Setup}_0(sk_0)$
$c = \mathsf{Enc}_0(pk, (m_b, \mathsf{G}_0(r)); \mathsf{G}_1(r))$
For all $(i,j) \in Q$:
$fk^{(i,j)} =$ **if** $((m_0, m_1) = \bot) \vee (f_{(i,j)} = \bot))$ **then** $\bot$
else if $(f_{(i,j)}(m_0) = f_{(i,j)}(m_1))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$
where $sk_1^i = \mathsf{G}_i(\mathsf{G}_1(sk))$, $pk_1^i = \mathsf{Setup}_1(sk_1^i)$, $c_i = \mathsf{G}_i(\mathsf{G}_2(sk))$
$fk_0^i = \mathsf{KeyGen}_0(sk_0, e_i[c_i, pk_1^i], i)$
$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_{(i,j)}, j)$

$\mathcal{H}_b^{(1)}$: In this hybrid some uses of the PRG G are replaced by truly random strings. In addition, sk is no longer needed so we remove it from the hybrid.

$\mathcal{H}_b^{(1)}((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}) = (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})$
where $sk_0 \leftarrow R$, $pk = \mathsf{Setup}_0(sk_0)$
$r' \leftarrow R$, $r'' \leftarrow R$, $c = \mathsf{Enc}_0(pk, (m_b, r'); r'')$
For all $i \in \mathrm{proj}_0(Q)$:
$sk_1^i \leftarrow R$, $pk_1^i = \mathsf{Setup}_1(sk_1^i)$, $c_i \leftarrow R$
$fk_0^i = \mathsf{KeyGen}_0(sk_0, e_i[c_i, pk_1^i], i)$
For all $j \in \mathrm{proj}_1^i(Q)$:
$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_{(i,j)}, j)$
$fk^{(i,j)} =$ **if** $((m_0, m_1) = \bot) \vee (f_{(i,j)} = \bot))$ **then** $\bot$
else if $(f_{(i,j)}(m_0) = f_{(i,j)}(m_1))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$

Lemma 4. If G is a μ-secure pseudorandom generator, then for any $b \in \{0,1\}$ and any efficient adversary $\mathcal{A}$, we have $|\mathsf{Adv}[\mathcal{A}]_b^{(0)} - \mathsf{Adv}[\mathcal{A}]_b^{(1)}| \leq 4\mu(\kappa)$.

$\mathcal{H}_b^{(2)}$: In this hybrid we slightly modify how the c_i's are generated without changing their distribution.

$\mathcal{H}_b^{(1)}((m_0,m_1),\{f_{(i,j)}\}_{(i,j)\in Q}) = (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})$
where $sk_0 \leftarrow R$, $pk = \mathsf{Setup}_0(sk_0)$
$r' \leftarrow R$, $r'' \leftarrow R$, $c = \mathsf{Enc}_0(pk, (m_b, r'); r'')$
For all $i \in \mathrm{proj}_0(Q)$:
$sk_1^i \leftarrow R$, $pk_1^i = \mathsf{Setup}_1(sk_1^i)$
$s_i \leftarrow R$, $\tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r'))$, $c_i = s_i \oplus \tilde{c}_1^i$
$fk_0^i = \mathsf{KeyGen}_0(sk_0, e_i[c_i, pk_1^i], i)$
For all $j \in \mathrm{proj}_1^i(Q)$:
$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_{(i,j)}, j)$
$fk^{(i,j)} =$ **if** $((m_0,m_1) = \bot) \vee (f_{(i,j)} = \bot))$ **then** $\bot$
else if $(f_{(i,j)}(m_0) = f_{(i,j)}(m_1))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$

Lemma 5. For any $b \in \{0,1\}$ and adversary $\mathcal{A}$, we have $\mathsf{Adv}[\mathcal{A}]_b^{(1)} = \mathsf{Adv}[\mathcal{A}]_b^{(2)}$.

$\mathcal{H}_b^{(3)}$: In this hybrid we replace the truly random s_i with a pseudorandom string.

$\mathcal{H}_b^{(3)}((m_0,m_1),\{f_{(i,j)}\}_{(i,j)\in Q}) = (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})$
where $sk_0 \leftarrow R$, $pk = \mathsf{Setup}_0(sk_0)$
$r' \leftarrow R$, $r'' \leftarrow R$, $c = \mathsf{Enc}_0(pk, (m_b, r'); r'')$, $s \leftarrow R$
For all $i \in \mathrm{proj}_0(Q)$:
$sk_1^i \leftarrow R$, $pk_1^i = \mathsf{Setup}_1(sk_1^i)$
$s_i = \mathsf{G}_i(s)$, $\tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r'))$, $c_i = s_i \oplus \tilde{c}_1^i$
$fk_0^i = \mathsf{KeyGen}_0(sk_0, e_i[c_i, pk_1^i], i)$
For all $j \in \mathrm{proj}_1^i(Q)$:
$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_{(i,j)}, j)$
$fk^{(i,j)} =$ **if** $((m_0,m_1) = \bot) \vee (f_{(i,j)} = \bot))$ **then** $\bot$
else if $(f_{(i,j)}(m_0) = f_{(i,j)}(m_1))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$

Lemma 6. If G is a μ-secure pseudorandom generator, then for any $b \in \{0,1\}$ and adversary $\mathcal{A}$, we have $|\mathsf{Adv}[\mathcal{A}]_b^{(2)} - \mathsf{Adv}[\mathcal{A}]_b^{(3)}| \leq \mu(\kappa)$.

$\mathcal{H}_b^{(4)}$: In this hybrid we modify c to encrypt $(\bot, s)$ instead of (m_b, r).

$\mathcal{H}_b^{(4)}((m_0,m_1),\{f_{(i,j)}\}_{(i,j)\in Q}) = (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})$
where $sk_0 \leftarrow R$, $pk = \mathsf{Setup}_0(sk_0)$
$r' \leftarrow R$, $r'' \leftarrow R$, $c = \mathsf{Enc}_0(pk, (\bot, s); r'')$, $s \leftarrow R$
For all $i \in \mathrm{proj}_0(Q)$:
$sk_1^i \leftarrow R$, $pk_1^i = \mathsf{Setup}_1(sk_1^i)$
$s_i = \mathsf{G}_i(s)$, $\tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r'))$, $c_i = s_i \oplus \tilde{c}_1^i$
$fk_0^i = \mathsf{KeyGen}_0(sk_0, e_i[c_i, pk_1^i], i)$
For all $j \in \mathrm{proj}_1^i(Q)$:
$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_{(i,j)}, j)$
$fk^{(i,j)} =$ **if** $((m_0,m_1) = \bot) \vee (f_{(i,j)} = \bot))$ **then** $\bot$
else if $(f_{(i,j)}(m_0) = f_{(i,j)}(m_1))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$

Lemma 7. If FE_0 is a non-adaptive $\epsilon_0(\kappa, q')$-secure FE scheme for functions in $\mathbb{RE}_{\mathsf{FE}_0}$ with an index space I_0, then for any $b \in \{0,1\}$ and any efficient adversary $\mathcal{A}$ who makes at most q functional key queries, we have $|\mathsf{Adv}[\mathcal{A}]_b^{(3)} - \mathsf{Adv}[\mathcal{A}]_b^{(4)}| \leq \epsilon_0(\kappa, \min\{q, |I_0|\})$.

$\mathcal{H}_b^{(5)}$: Now we use fresh randomness to generate $\tilde{c}_i$ instead of sharing a pseudorandom string.

$$
\begin{aligned}
&\mathcal{H}_b^{(5)}((m_0,m_1),\{f_{(i,j)}\}_{(i,j)\in Q}) = (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})\\
&\quad \textbf{where } sk_0 \leftarrow R,\ pk = \mathsf{Setup}_0(sk_0)\\
&\qquad\qquad r'' \leftarrow R,\ c = \mathsf{Enc}_0(pk,(\bot,s);r''),\ s \leftarrow R\\
&\qquad\qquad \text{For all } i \in \mathrm{proj}_0(Q):\\
&\qquad\qquad\quad sk_1^i \leftarrow R,\ pk_1^i = \mathsf{Setup}_1(sk_1^i)\\
&\qquad\qquad\quad s_i = \mathsf{G}_i(s),\ r_i \leftarrow R,\ \tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; r_i),\ c_i = s_i \oplus \tilde{c}_1^i\\
&\qquad\qquad\quad fk_0^i = \mathsf{KeyGen}_0(sk_0, e_i[c_i, pk_1^i], i)\\
&\qquad\qquad\quad \text{For all } j \in \mathrm{proj}_1^i(Q):\\
&\qquad\qquad\qquad fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_{(i,j)}, j)\\
&\qquad\qquad\qquad fk^{(i,j)} = \textbf{if } ((m_0,m_1)=\bot) \vee (f_{(i,j)}=\bot)) \textbf{ then } \bot\\
&\qquad\qquad\qquad\qquad\qquad \textbf{else if } (f_{(i,j)}(m_0) = f_{(i,j)}(m_1)) \textbf{ then } (fk_0^i, fk_1^{(i,j)}) \textbf{ else } \top
\end{aligned}
$$

Lemma 8. If G is a μ-secure pseudorandom generator, then for any $b \in \{0,1\}$ and any adversary $\mathcal{A}$ we have $|\mathsf{Adv}[\mathcal{A}]_b^{(4)} - \mathsf{Adv}[\mathcal{A}]_b^{(5)}| \le \mu(\kappa)$.

Lemma 9. If FE_1 is a non-adaptive $\epsilon_1(\kappa, q')$-secure FE scheme with an index space I_1, then for any efficient adversary $\mathcal{A}$ who makes at most q functional key queries, we have $|\mathsf{Adv}[\mathcal{A}]_0^{(5)} - \mathsf{Adv}[\mathcal{A}]_1^{(5)}| \le \max_{q_1+\cdots+q_p=q, q_i \le |I_1|} \sum_i \epsilon_1(\kappa, q_i)$.

Finally, by applying previous lemmas, we see that the advantage of any adversary $\mathcal{A}$ to the PRODUCT scheme FE can be bounded by

$$
\begin{aligned}
\mathsf{Adv}_{\mathsf{FE}}[\mathcal{A}] &= |\Pr\{[\mathcal{H}_0^{(0)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_1^{(0)}, \mathcal{A}] = 1\}|\\
&\le 2(4\mu(\kappa) + 0 + \mu(\kappa) + \epsilon_0(\kappa, \min\{q, |I_0|\}) + \mu(\kappa)) + \max_{q_1+\cdots+q_p=q, q_i \le |I_1|} \sum_i \epsilon_1(\kappa, q_i)\\
&= \max_{q_1+\cdots+q_p=q, q_i \le |I_1|} \sum_i \epsilon_1(\kappa, q_i) + 2\epsilon_0(\kappa, \min\{q, |I_0|\}) + 12\mu(\kappa).
\end{aligned}
$$

□

5 Compositions using SUM and PRODUCT Constructions

SUM and PRODUCT constructions provide ways to build new FE schemes with larger function spaces. They also have nice efficiency preserving properties. Using them as building blocks, we propose two composition methods to define transformations from a FE scheme supporting only one functional key query to a new FE scheme that supports any polynomially many functional key queries without losing much security and efficiency guarantees.

Throughout this section, we assume FE_0 is a single-key $\epsilon_0(\kappa)$-secure FE scheme, where $\epsilon_0(\kappa)$ is negligible, for functions in a class F with some minimal efficiency guarantees, for example, succinct. FE_0 can be either selective-, non-adaptive-, or adaptive-secure, and our transformations preserve these security notions. We also assume G is a μ-secure PRG, for negligible $\mu(\kappa)$. Let $t_0^{\mathsf{Setup}}, t_0^{\mathsf{Enc}}, t_0^{\mathsf{Dec}}, t_0^{\mathsf{KeyGen}}$ be the running times of the four algorithms in FE_0, and let ℓ_0^k, ℓ_0^c, ℓ_0^{fk} be the lengths of public key, ciphertext, and functional keys of FE_0. Since FE_0 is succinct, $t_0^{\mathsf{Enc}} = t_0^{\mathsf{Enc}}(n,\kappa,d)$ and $\ell_0^c = \ell_0^c(n,\kappa,d)$ are both polynomials in the message length n, security parameter κ, and the maximal depth d of functions in F. Let t^{G} be the running time of the PRG G. Our main results are two reductions from collusion-resistant compact FE schemes for F to FE_0 assuming F meets some requirements (more details later.)

5.1 Iterated Squaring Composition

Our first transformation can be obtained by repeatedly squaring the previously composed FE scheme. At the beginning, we use the SUM construction to obtain FE schemes supporting 2 functional key queries. Then PRODUCT construction is used on the FE schemes of the previous iteration.

Formally, we can define the *iterated squaring composition* method by:

$$\mathsf{FE}_1 = \mathsf{FE}_0 + \mathsf{FE}_0, \text{and for } h \geq 1, \mathsf{FE}_{h+1} = \mathsf{FE}_h \times \mathsf{FE}_h. \tag{1}$$

So FE_1 is secure for at most 2 functional queries. For $h \geq 1$, the index space I_{h+1} of FE_{h+1} has size 2^{2^h}. Let $p = \omega(\log\log\kappa)$ and $p = O(\log\kappa)$; then FE_{p+1} has a superpolynomial-sized index space. In the following we discuss in detail the security and performance of FE'_{p+1}.

Security: For any polynomial $q(\kappa)$, the advantage of any efficient adversary $\mathcal{A}$ who makes at most q functional key queries over FE'_{p+1} is bounded by

$$\mathsf{Adv}_{\mathsf{FE}'_{p+1}}[\mathcal{A}] \leq q\kappa \cdot (\epsilon_0(\kappa) + \mu(\kappa)). \tag{2}$$

This upper bound can be obtained as follows. Let $\epsilon_h(\kappa, q')$ be the security upper bound of FE_h against any adversary $\mathcal{A}$ making at most $q' < |I_h|$ functional key queries. We show by induction on h that $\epsilon_h(\kappa, q') \leq 3^{h-1}q' \cdot (\epsilon_1 + \mu)$. For $h = 1$, by Theorem 3 we have $\epsilon_1(\kappa) \leq 2\epsilon_0(\kappa) + 4\mu(\kappa)$, and in particular, ϵ_1 is independent of q'. So for any $1 \leq q' \leq |I_1| = 2$, $\epsilon_1(\kappa, q') = \epsilon_1(\kappa, 1) \leq q'\epsilon_1(\kappa)$. Next, suppose $\epsilon_h(\kappa, q') \leq 3^{h-1}q' \cdot (\epsilon_1 + \mu)$ for $h \geq 1$ and $1 \leq q' \leq |I_h|$. By Theorem 5, for any $1 \leq q' \leq |I_{h+1}|$ we have

$$\begin{aligned}
\epsilon_{h+1}(\kappa, q') &\leq \max_{q_1+\cdots+q_k=q', q_i \leq |I_h|} \sum_i \epsilon_h(\kappa, q_i) + 2\epsilon_h(\kappa, \min\{q', |I_h|\}) + 12\mu \\
&\leq \max_{q_1+\cdots+q_k=q', q_i \leq |I_h|} \sum_i (3^{h-1} q_i \cdot (\epsilon_1 + \mu)) + 2 \cdot 3^{h-1} \min\{q', |I_h|\} \cdot (\epsilon_1 + \mu) + 12\mu \\
&\leq 3^h q' \cdot (\epsilon_1 + \mu).
\end{aligned}$$

Now, since $p = O(\log\kappa)$, the upper bound in Equation 2 follows.

Running times and output lengths:

- Setup: $2t_0^{\mathsf{Setup}} + (p+1)t^{\mathsf{G}} \leq 2t_0^{\mathsf{Setup}} + \log\kappa \cdot t^{\mathsf{G}}$
- Enc : $2t_0^{\mathsf{Enc}}(n + p\kappa, \kappa, d) + (p+1)t^{\mathsf{G}} \leq 2t_0^{\mathsf{Enc}}(n + \kappa\log\kappa, \kappa, d) + \log\kappa \cdot t^{\mathsf{G}}$
- Dec : $2^p t_0^{\mathsf{Dec}} \leq \kappa \cdot t_0^{\mathsf{Dec}}$
- KeyGen : $2(2^p - 1)t_0^{\mathsf{Setup}} + 2^p t_0^{\mathsf{KeyGen}} + (\sum_{i=0}^{p}(p+2-i)2^i + 2^{p+1} - 1)t^{\mathsf{G}} \leq \kappa \cdot t_0^{\mathsf{Setup}} + \kappa \cdot t_0^{\mathsf{KeyGen}} + \kappa \cdot t^{\mathsf{G}}$
- $\ell_{p+1}^k = 2\ell_0^k$
- $\ell_{p+1}^c = \ell_0^c(n + p\kappa, \kappa, d) \leq \ell_0^c(n + \kappa\log\kappa, \kappa, d)$
- $\ell_{p+1}^{fk} = 2^p \ell_0^{fk} \leq \kappa \cdot \ell_0^{fk}$

Clearly FE_{p+1} is a q-key secure FE scheme for any polynomial $q(\kappa)$, and the transformation incurs only linear (in terms of q) security loss. Since FE_0 is succinct, t_0^{Enc} is a polynomial in n, κ, and d. So t_{p+1}^{Enc} can be bounded by $\mathrm{poly}(n, \kappa, d)$ for some fixed polynomial poly, and hence FE_{p+1} is succinct.

Besides, for the iterated squaring composition to be viable, we must be careful about the function classes supported in each iteration of the composition. Let F_h be the class of functions supported by KeyGen_h of the FE scheme FE_h, for $h \geq 0$. First we have $F_1 = F_0$. In the steps using PRODUCT construction on FE_h

to derive FE_{h+1}, a functional key $\mathit{fk} = (\mathit{fk}_0, \mathit{fk}_1)$ for any function f consists of two keys under FE_h: fk_0 is for a "re-encryption" function $e_i^{(h)}[c, pk](\cdot,\cdot)$, and fk_1 is for f. Hence for the composition to go through, FE_h must be capable of generating functional keys for these two classes of functions, namely

$$F_{h+1} \cup \{e_i^{(h)}[c, pk] \mid c \in M, pk \in R\} \subseteq F_h.$$

Recall from Section 4 that $\mathbb{RE}_{\mathsf{FE}_h}$ is the class containing $e_i^{(h)}[c, pk]$ for all $c \in M, pk \in R$. Let $\mathbb{RE}^p_{\mathsf{FE}_0} = \cup_{h=1}^p \mathbb{RE}_{\mathsf{FE}_h}$. By expanding the above recursion, we see that to support function class F_{p+1} the FE scheme FE_0 must be capable of functional keys for the functions in $F_{p+1} \cup \mathbb{RE}^p_{\mathsf{FE}_0}$ and the PRG G.

Theorem 6. *Let p be such that $p = \omega(\log\log\kappa)$ and $p = O(\log\kappa)$. Assume G is a secure PRG, and assume FE_0 is a succinct 1-key non-adaptive (or selective/adaptive) secure FE scheme for a class F of functions such that $\mathbb{RE}^p_{\mathsf{FE}_0} \subseteq F$ and $\mathsf{G} \in F$. Then FE_{p+1} defined by Equation 1 is a succinct collusion-resistant non-adaptive (or selective/adaptive, respectively) secure FE scheme for F.*

5.2 Iterated Linear Composition

A drawback of the iterated squaring composition is that the base scheme FE_0 must be capable of generating functional keys for the re-encryption functions of all iteration steps. It is usually hard to check if this condition holds for a concrete FE scheme. We now present another composition method which requires only the base scheme to be capable of functional keys for its own encryption function.

The *iterated linear composition* is defined recursively by

$$\mathsf{FE}_1 = \mathsf{FE}_0 + \mathsf{FE}_0, \text{and for } h \geq 1, \mathsf{FE}_{h+1} = \mathsf{FE}_1 \times \mathsf{FE}_h. \tag{3}$$

Under this composition, FE_1 supports 2 functional keys, and for $h \geq 1$, FE_{h+1} has an index space I_{h+1} of size 2^{h+1}. For FE_h to be secure with q functional keys for any polynomial $q(\kappa)$, we can set $h = p$ for $p = \omega(\log\kappa)$ and $p = O(\kappa)$. We analyze the security and performance of FE_p in the following.

Security: For any polynomial $q(\kappa)$, the advantage of any efficient adversary $\mathcal{A}$ who makes at most q functional key queries over FE'_p is bounded by

$$\mathsf{Adv}_{\mathsf{FE}'_p}[\mathcal{A}] \leq q\kappa \cdot (\epsilon_0(\kappa) + \mu(\kappa)). \tag{4}$$

To obtain this upper bound, let $\epsilon_h(\kappa, q')$ be the security bound of FE_h against any adversary who makes at most $q' \leq |I_h|$ functional key queries. We use induction on h to show that $\epsilon_h(\kappa, q') \leq q'h \cdot (\epsilon_1 + \mu)$. First note that $\epsilon_1(\kappa) = 2\epsilon_0(\kappa) + 4\mu(\kappa)$ is independent of q', so for any $q' \leq |I_1| = 2$, $\epsilon_1(\kappa, q') \leq q'\epsilon_1(\kappa)$. Next, suppose our bound holds for $h \geq 1$ and $2 \leq q' \leq |I_h|$. Then for $h+1$, it follows from Theorem 5 that

$$\begin{aligned}
\epsilon_{h+1}(\kappa, q') &\leq \max_{q_1+q_2=q', q_i \leq |I_h|} \sum_i \epsilon_h(\kappa, q_i) + 2\epsilon_1(\kappa) + 12\mu \\
&\leq \max_{q_1+q_2=q', q_i \leq |I_h|} \sum_i (q_i h \cdot (\epsilon_1 + \mu)) + 2\epsilon_1(\kappa) + 12\mu \\
&\leq q'(h+1) \cdot (\epsilon_1 + \mu).
\end{aligned}$$

Since $p = O(\kappa)$, the upper bound in Equation 4 holds.

Running times and output lengths :

- Setup: $2t_0^{\mathsf{Setup}} + 2t^{\mathsf{G}}$
- Enc : $2t_0^{\mathsf{Enc}}(n + \kappa, \kappa, d) + 2t^{\mathsf{G}}$
- Dec : $pt_0^{\mathsf{Dec}} \leq \kappa \cdot t_0^{\mathsf{Dec}}$

- KeyGen : $pt_0^{\mathsf{KeyGen}} + 2(p-1)t_0^{\mathsf{Setup}} + (6p-5)t^{\mathsf{G}} \leq \kappa \cdot t_0^{\mathsf{Setup}} + \kappa \cdot t_0^{\mathsf{KeyGen}} + \kappa \cdot t^{\mathsf{G}}$
- $\ell_p^k = 2\ell_0^k$
- $\ell_p^c = 2\ell_0^c(n+\kappa, \kappa, d)$
- $\ell_p^{fk} = p\ell_0^{fk} \leq \kappa \cdot \ell_0^{fk}$

The FE scheme FE_p is also q-key secure for any polynomial $q(\kappa)$, and this transformation too incurs linear (in terms of q) security loss. This time, the running time of the encryption procedure no longer depends on q, so FE_p is fully succinct.

Again, for this composition method to be viable, we should consider the functions can be handled at each iteration. Let F_h denote the function class supported by FE_h, for $h \geq 0$. As in the squaring composition, we have $F_1 = F_0$. For $h \geq 1$, to derive a functional key for any function f in FE_{h+1}, the scheme FE_1 must generate functional keys for the re-encryption function $e_i[pk, c]$, and FE_h must be capable of generating functional keys of f. This implies that

$$F_p \cup \{e_i[pk, c] \mid pk \in R, c \in M\} \subseteq F_0.$$

Since $e_i[pk, c](\cdot, \cdot)$ can be easily built using basic operations on $\mathsf{Enc}_1(pk, \cdot; \cdot)$ and $\mathsf{G}(\cdot)$, it is sufficient to require that FE_0 can generate functional keys for these two classes of functions.

Theorem 7. *Let p be such that $p = \omega(\log \kappa)$ and $p = O(\kappa)$. Assume G is a secure PRG, and assume FE_0 is a succinct 1-key non-adaptive (or selective/adaptive) secure FE scheme for the class F of functions such that $\mathsf{Enc}_0(pk, \cdot; \cdot), \mathsf{G}(\cdot) \in F$ for any $pk \in R$. Then, the FE scheme FE_p defined in Equation 3 is a succinct collusion-resistant non-adaptive (or selective/adaptive, respectively) secure FE scheme for F.*

Comparing with the iterated squaring composition to support q functional key queries, one can see that the security loss, the running times, and key lengths of Setup and KeyGen are about the same, and the iterated linear composition gives better encryption performance: Enc runs slightly faster. The trade-off is in the ciphertext length: our linear composition simply doubles the ciphertext length of the underlying single-key FE scheme, while the iterated squaring composition produces a ciphertext that encrypts a slightly longer message in the single-key FE scheme.

5.3 On the implications of our reductions

So far we have obtained two transformations from a single-key succinct FE scheme to a succinct FE scheme that supports polynomially many functional key queries. In this subsection we explore the implications of our reductions.

A q-key secure FE scheme for F is called *weakly collusion-succinct* if t^{Enc} grows sub-linearly in q but polynomially in n, κ, and the maximum circuit size of functions in F. If the sub-linear dependence on q is removed, then the FE scheme is called *collusion-succinct*. For succinct FE_0, let us consider the following two cases about the encryption time t_{p+1}^{Enc} of FE_{p+1} obtained by our transformations on FE_0:

1. If FE_{p+1} is as in the iterated squaring composition, then for $p = O(\log \kappa)$ we have $t_{p+1}^{\mathsf{Enc}} \leq t_0^{\mathsf{Enc}}(n + \kappa \cdot \log \kappa, \kappa, d) + \log \kappa \cdot t^{\mathsf{G}}(\kappa)$. Since t_{p+1}^{Enc} is independent of q and t_0^{Enc} is polynomial in n,κ, and d, FE_{p+1} is collusion-succinct.

2. If FE_{p+1} is as in the iterated linear composition, then we have $t_{p+1}^{\mathsf{Enc}} = 2t_0^{\mathsf{Enc}}(n+\kappa, \kappa, d) + 2t^{\mathsf{G}}(\kappa)$, which is independent of q. So FE_{p+1} is also collusion-succinct.

As we have mentioned in Remark 1, since the index space of FE_{p+1} as in our transformations has superpolynomial size, we can eliminate i from the interface of KeyGen to make it completely stateless. Such conversions incur only a negligible security loss, and they do not affect the security properties of FE_{p+1} in either transformation.

Bitansky and Vaikuntanathan [10] described a reduction from any (weakly) compact q-key secure FE scheme to a (weakly) collusion-succinct q-key secure FE scheme for the same class of functions. We note that, although in [10] the notion of collusion-succinct was defined in terms of ciphertext length, their reduction still holds with our encryption time based definition. By applying their reduction together with our transformations, we get the following new reductions:

Theorem 8. *1. If there exists a succinct single-key secure FE scheme FE_0 for a class F of functions such that $\mathbb{RE}^p_{\mathsf{FE}_0} \subseteq F$ for $p = \omega(\log\log\kappa)$ and that $\mathsf{G} \in F$, then there exists a compact and collusion-resistant secure FE scheme for F;*

2. If there exists a succinct single-key secure FE scheme FE_0 for a class F of functions such that its encryption function Enc_0 satisfies $\mathsf{Enc}_0(pk, \cdot; \cdot) \in F$ for any $pk \in R$ and that $\mathsf{G} \in F$, then there exists a compact and collusion-resistant secure FE scheme for F.

Notice that a (weakly) compact FE scheme is necessarily (weakly) succinct. Our results show that weakly compact (non-collusion-resistant) FE schemes (supporting a sufficiently general class of functions,) imply collusion-resistant FE schemes. As shown in [10, 3], (non-compact) collusion-resistant FE schemes imply compact FE schemes. So now we can see these variants as equivalent notions under polynomial time reductions.

One may attempt to instantiate a compact collusion-resistant FE scheme using our transformations on a succinct single-key secure FE scheme. Based on sub-exponential lattice assumption, Goldwasser et al. [26] showed that, for any polynomial $d(n)$, there exists a succinct single-key secure FE scheme for the class of functions with 1-bit output and depth d circuits. However, it is not clear how to efficiently "upgrade" this FE scheme to be capable of generating a functional key of its own encryption function so that the assumptions of our transformations can be met. This is not surprising because any instantiation would immediately give an indistinguishability obfuscator. We find it very interesting to answer such question and we leave it for future work.

Acknowledgement

We would like to thank Fuyuki Kitagawa and Ilan Komargodski for pointing out mistakes in earlier versions of this paper, and we thank anonymous TCC reviewers for useful comments.

References

[1] Agrawal, S., Gorbunov, S., Vaikuntanathan, V., Wee, H.: Functional encryption: New perspectives and lower bounds. In: CRYPTO. pp. 500–518 (2013)

[2] Ananth, P., Brakerski, Z., Segev, G., Vaikuntanathan, V.: From selective to adaptive security in functional encryption. In: CRYPTO. Lecture Notes in Computer Science, vol. 9216, pp. 657–677. Springer (2015)

[3] Ananth, P., Jain, A.: Indistinguishability obfuscation from compact functional encryption. In: CRYPTO. Lecture Notes in Computer Science, vol. 9215, pp. 308–326. Springer (2015)

[4] Ananth, P., Jain, A., Sahai, A.: Indistinguishability obfuscation from functional encryption for simple functions. IACR Cryptology ePrint Archive 2015, 730 (2015), `http://eprint.iacr.org/2015/730`

[5] Applebaum, B., Brakerski, Z.: Obfuscating circuits via composite-order graded encoding. In: Theory of Cryptography Conference, TCC. pp. 528–556 (2015)

[6] Barak, B., Garg, S., Kalai, Y.T., Paneth, O., Sahai, A.: Protecting obfuscation against algebraic attacks. In: EUROCRYPT. pp. 221–238 (2014)

[7] Barak, B., Goldreich, O., Impagliazzo, R., Rudich, S., Sahai, A., Vadhan, S.P., Yang, K.: On the (im)possibility of obfuscating programs. J. ACM 59(2), 6 (2012), prelim. version in CRYPTO 2001

[8] Bitansky, N., Goldwasser, S., Jain, A., Paneth, O., Vaikuntanathan, V., Waters, B.: Time-lock puzzles from randomized encodings. In: Innovations in Theoretical Computer Science. pp. 345–356 (2016)

[9] Bitansky, N., Nishimaki, R., Passelègue, A., Wichs, D.: From cryptomania to obfustopia through secret-key functional encryption. IACR Cryptology ePrint Archive 2016, 558 (2016), http://eprint.iacr.org/2016/558

[10] Bitansky, N., Vaikuntanathan, V.: Indistinguishability obfuscation from functional encryption. In: Foundations of Computer Science, FOCS. pp. 171–190 (2015)

[11] Boneh, D., Sahai, A., Waters, B.: Functional encryption: a new vision for public-key cryptography. Commun. ACM 55(11), 56–64 (2012)

[12] Brakerski, Z., Rothblum, G.N.: Virtual black-box obfuscation for all circuits via generic graded encoding. In: Theory of Cryptography Conference, TCC. pp. 1–25 (2014)

[13] Brakerski, Z., Segev, G.: Function-private functional encryption in the private-key setting. In: Theory of Cryptography Conference, TCC. Lecture Notes in Computer Science, vol. 9015, pp. 306–324. Springer (2015)

[14] Brakerski, Z., Segev, G.: Hierarchical functional encryption. IACR Cryptology ePrint Archive 2015, 1011 (2015), http://eprint.iacr.org/2015/1011

[15] Chandran, N., Chase, M., Vaikuntanathan, V.: Functional re-encryption and collusion-resistant obfuscation. In: Theory of Cryptography Conference, TCC. Lecture Notes in Computer Science, vol. 7194, pp. 404–421. Springer (2012)

[16] Cheon, J.H., Han, K., Lee, C., Ryu, H., Stehlé, D.: Cryptanalysis of the multilinear map over the integers. In: EUROCRYPT. pp. 3–12 (2015)

[17] Chung, K., Lin, H., Pass, R.: Constant-round concurrent zero-knowledge from indistinguishability obfuscation. In: CRYPTO. pp. 287–307 (2015)

[18] Coron, J., Gentry, C., Halevi, S., Lepoint, T., Maji, H.K., Miles, E., Raykova, M., Sahai, A., Tibouchi, M.: Zeroizing without low-level zeroes: New MMAP attacks and their limitations. In: CRYPTO. pp. 247–266 (2015)

[19] Coron, J., Lepoint, T., Tibouchi, M.: Practical multilinear maps over the integers. In: CRYPTO. pp. 476–493 (2013)

[20] Coron, J., Lepoint, T., Tibouchi, M.: New multilinear maps over the integers. In: CRYPTO. pp. 267–286 (2015)

[21] Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: EUROCRYPT. pp. 1–17 (2013)

[22] Garg, S., Gentry, C., Halevi, S., Raykova, M., Sahai, A., Waters, B.: Candidate indistinguishability obfuscation and functional encryption for all circuits. In: Foundations of Computer Science, FOCS. pp. 40–49 (2013)

[23] Garg, S., Srinivasan, A.: Single-key to multi-key functional encryption with polynomial loss. To appear in TCC-2016B. Available at IACR ePrint http://eprint.iacr.org/2016/524

[24] Gentry, C., Gorbunov, S., Halevi, S.: Graph-induced multilinear maps from lattices. In: Theory of Cryptography, TCC. pp. 498–527 (2015)

[25] Gentry, C., Lewko, A.B., Sahai, A., Waters, B.: Indistinguishability obfuscation from the multilinear subgroup elimination assumption. In: Foundations of Computer Science, FOCS. pp. 151–170 (2015)

[26] Goldwasser, S., Kalai, Y.T., Popa, R.A., Vaikuntanathan, V., Zeldovich, N.: Reusable garbled circuits and succinct functional encryption. In: Symposium on Theory of Computing Conference, STOC. pp. 555–564 (2013)

[27] Goldwasser, S., Rothblum, G.N.: On best-possible obfuscation. J. Cryptology 27(3), 480–505 (2014), prelim. version in TCC 2007

[28] Li, B., Micciancio, D.: Equational security proofs of oblivious transfer protocols. IACR Cryptology ePrint Archive 2016, 624 (2016), `http://eprint.iacr.org/2016/624`

[29] Lin, H., Pass, R., Seth, K., Telang, S.: Indistinguishability obfuscation with non-trivial efficiency. In: Public-Key Cryptography - PKC 2016 - 19th IACR International Conference on Practice and Theory in Public-Key Cryptography, Taipei, Taiwan, March 6-9, 2016, Proceedings, Part II. pp. 447–462 (2016)

[30] Malkin, T., Micciancio, D., Miner, S.K.: Efficient generic forward-secure signatures with an unbounded number of time periods. In: EUROCRYPT. pp. 400–417 (2002)

[31] Micciancio, D., Tessaro, S.: An equational approach to secure multi-party computation. In: Innovations in Theoretical Computer Science. pp. 355–372. ITCS '13, ACM, New York, NY, USA (2013)

[32] O'Neill, A.: Definitional issues in functional encryption. IACR Cryptology ePrint Archive 2010, 556 (2010), `http://eprint.iacr.org/2010/556`

[33] Pass, R., Seth, K., Telang, S.: Indistinguishability obfuscation from semantically-secure multilinear encodings. In: CRYPTO. pp. 500–517 (2014)

[34] Sahai, A., Waters, B.: How to use indistinguishability obfuscation: deniable encryption, and more. In: Symposium on Theory of Computing, STOC. pp. 475–484 (2014)

[35] Waters, B.: Functional encryption: Origins and recent developments. In: Public-Key Cryptography - PKC. pp. 51–54 (2013)

[36] Waters, B.: A punctured programming approach to adaptively secure functional encryption. In: CRYPTO. Lecture Notes in Computer Science, vol. 9216, pp. 678–697. Springer (2015)

A Proofs of Lemmas

Let $\mathsf{G} : R \to S$ be a $\mu(\kappa)$-secure PRG. Recall the following well-known facts:

- The function $\mathsf{G}'(r_1 \cdots r_m) = \mathsf{G}(r_1) \cdots \mathsf{G}(r_m)$ defined by concatenating m pseudorandom strings generated by G on $r_1, \ldots, r_m \in R$ is a $m\mu(\kappa)$-secure pseudorandom generator.
- The function $\mathsf{G}''(r) = \mathsf{G}(\mathsf{G}_i(r))$, where $|\mathsf{G}_i(r)| = |r| = n$, is a $2\mu(\kappa)$-secure pseudorandom generator.

We will use them to shorten our security proofs.

First we prove lemmas in Sections 3 that are used to establish security of the SUM constructions.

Lemma 1. If G is a μ-secure pseudorandom generator, then for any $b \in \{0,1\}$ and adversary $\mathcal{A}$ we have $|\Pr\{[\mathcal{H}_b^{(0)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_b^{(1)}, \mathcal{A}] = 1\}| \leq 2\mu(\kappa)$.

Proof. We define the following adversary $\mathcal{B}$ using $\mathcal{A}$ as an oracle to attack the PRG G, where $\mathcal{H}_b^{(1)}[sk_0, sk_1, r_0, r_1]$ is the hybrid obtained by replacing sk_0, sk_1, r_0, r_1 of $\mathcal{H}_b^{(1)}$ by the given values. By the notation $sk_0 \| sk_1 \| r_0 \| r_1 = x$ we mean to parse x as a concatenation of four bit-strings sk_0, sk_1, r_0, r_1 of appropriate lengths.

$\mathcal{B}(x) = b'$
where $sk_0 \| sk_1 \| r_0 \| r_1 = x$
$(pk, c, \{fk^{(h,i)}\}_{(h,i)\in Q}) = \mathcal{H}_b^{(1)}[sk_0, sk_1, r_0, r_1]((m_0, m_1), \{f_{(h,i)}\}_{(h,i)\in Q})$
$((m_0, m_1), \{f_{(h,i)}\}_{(h,i)\in Q}, b') = \mathcal{A}(pk, c, \{fk^{(h,i)}\}_{(h,i)\in Q})$

Notice that if x is generated by the PRG G then $\mathcal{B}$ is running the system $[\mathcal{H}_b^{(0)}, \mathcal{A}]$, and if x is uniformly at random then $\mathcal{B}$ is running $[\mathcal{H}_b^{(1)}, \mathcal{A}]$. Since in $\mathcal{H}_b^{(1)}$ we replaced two calls to G with truly random seeds, we have $|\mathsf{Adv}[\mathcal{A}]_b^{(0)} - \mathsf{Adv}[\mathcal{A}]_b^{(1)}| = \mathsf{Adv}_{\mathsf{G}}[\mathcal{B}] \le 2\mu(\kappa)$. □

Lemma 2. If FE_1 is a non-adaptively $\epsilon_1(\kappa)$-secure FE scheme with an index space I_1, then for any adversary $\mathcal{A}$ we have $|\Pr\{[\mathcal{H}_0^{(1)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_0^{(2)}, \mathcal{A}] = 1\}| \ \le \ \epsilon_1(\kappa)$.

Proof. We define the following adversary $\mathcal{B}$ using $\mathcal{A}$ as an oracle to attack the FE scheme FE_1.

$\mathcal{B}(pk_1, c_1, \{fk_1^{(1,i)}\}_{i\in Q_1}) = ((m_0, m_1), \{f_{(1,i)}\}_{i\in Q_1})$
where
$(pk, c, \{fk^{(h,i)}\}_{(h,i)\in Q}) = \mathcal{H}_0^{(2)}[pk_1, c_1, \{fk_1^{(1,i)}\}_{i\in Q_1}]((m_0, m_1), \{f_{(h,i)}\}_{(h,i)\in Q})$
$((m_0, m_1), \{f_{(h,i)}\}_{(h,i)\in Q}, b') = \mathcal{A}(pk, c, \{fk^{(h,i)}\}_{(h,i)\in Q})$

Since $\mathcal{A}$ is a valid adversary to $\mathsf{FE}_0 + \mathsf{FE}_1$, we must have $f_{(1,i)}(m_0) = f_{(1,i)}(m_1)$ for all $i \in Q_1$; and hence $\mathcal{B}$ is valid for FE_1. Notice that if the input c_1 to $\mathcal{B}$ is an encryption of m_0, i.e., $c_1 = \mathsf{Enc}_1(pk_1, m_0; r_1)$ for some random string $r_1 \in R$, then $\mathcal{B}$ is running $[\mathcal{H}_0^{(1)}, \mathcal{A}]$; if $c_1 = \mathsf{Enc}_1(pk_1, m_1; r_1)$ for some $r_1 \in R$, then $\mathcal{B}$ is running $[\mathcal{H}_0^{(2)}, \mathcal{A}]$. Hence the advantage of $\mathcal{B}$ in winning the FE game for the scheme FE_1 is $\mathsf{Adv}_{\mathsf{FE}_1}[\mathcal{B}] = |\mathsf{Adv}[\mathcal{A}]_0^{(1)} - \mathsf{Adv}[\mathcal{A}]_0^{(2)}| \le \epsilon_1(\kappa)$. □

Next we prove lemmas that are used to establish security of the PRODUCT constructions. From now on, hybrids refer to those defined in Section 4. Recall that $\mathcal{A}$ is an efficient adversary attacking $\mathsf{FE} = \mathsf{FE}_0 \times \mathsf{FE}_1$ and it makes q functional key queries. Let Q denote the index set of functional key queries made by $\mathcal{A}$, and suppose $Q_0 = \mathrm{proj}_0(Q) = \{i_1, \ldots, i_p\}$.

Lemma 4. If G is a μ-secure pseudorandom generator, then for any $b \in \{0, 1\}$ and any efficient adversary $\mathcal{A}$, we have $|\mathsf{Adv}[\mathcal{A}]_b^{(0)} - \mathsf{Adv}[\mathcal{A}]_b^{(1)}| \le 4\mu(\kappa)$.

Proof. We build an adversary $\mathcal{B}$ using $\mathcal{A}$ as an oracle to attack the PRG G. As in previous proofs, by $\mathcal{H}_b^{(1)}[sk_0, r', r'', \{sk_1^i\}_{i\in Q_0}, \{c_i\}_{i\in Q_0}]$ we mean the hybrid obtained by substituting $sk_0, r', r'', \{sk_1^i\}_{i\in Q_0}$, $\{c_i\}_{i\in Q_0}$ with the given values. The adversary $\mathcal{B}$ is defined as:

$\mathcal{B}(x) = b'$
where
$sk_0 \| r' \| r'' \| sk_1^{i_1} \| \cdots \| sk_1^{i_p} \| c_{i_1} \| \cdots \| c_{i_p} = x$
$(pk, c, \{fk^{(i,j)}\}_Q) = \mathcal{H}_b^{(1)}[sk_0, r', r'', \{sk_1^i\}_{i\in Q_0}, \{c_i\}_{i\in Q_0}]((m_0, m_1), \{f_{(i,j)}\}_Q)$
$((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}, b') = \mathcal{A}(pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})$

Notice that if x is generated by four calls to G then $\mathcal{B}$ is running $[\mathcal{H}_b^{(0)}, \mathcal{A}]$, and if x is truly random then $\mathcal{B}$ is running $[\mathcal{H}_b^{(1)}, \mathcal{A}]$. Since G is a μ-secure pseudorandom generator, we have $|\mathsf{Adv}[\mathcal{A}]_b^{(0)} - \mathsf{Adv}[\mathcal{A}]_b^{(1)}| \le 4\mu(\kappa)$. □

Lemma 5. For any $b \in \{0, 1\}$ and adversary $\mathcal{A}$, we have $\mathsf{Adv}[\mathcal{A}]_b^{(1)} = \mathsf{Adv}[\mathcal{A}]_b^{(2)}$.

Proof. Since c_i is sampled uniformly at random from K in $\mathcal{H}_b^{(1)}$ and $c_i = s_i \oplus c_i'$ in $\mathcal{H}_b^{(2)}$ where s_i is sampled uniformly at random from K, the distributions of c_i in the two experiments are identical. Therefore $\mathsf{Adv}[\mathcal{A}]_b^{(1)} = \mathsf{Adv}[\mathcal{A}]_b^{(2)}$. □

Lemma 6. If G is a μ-secure pseudorandom generator, then for any $b \in \{0,1\}$ and adversary $\mathcal{A}$, we have $|\mathsf{Adv}[\mathcal{A}]_b^{(2)} - \mathsf{Adv}[\mathcal{A}]_b^{(3)}| \leq \mu(\kappa)$.

Proof. We build an adversary $\mathcal{B}$ using $\mathcal{A}$ as an oracle to attack G:

$$\begin{aligned}
&\mathcal{B}(x) = b' \\
&\quad \textbf{where } s_{i_1} \| \cdots \| s_{i_p} = x \\
&\qquad (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q}) = \mathcal{H}_b^{(1)}[\{s_i\}_{i\in Q_0}]((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}) \\
&\qquad ((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}, b') = \mathcal{A}(pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})
\end{aligned}$$

Notice that if x is chosen uniformly at random then $\mathcal{B}$ is running $[\mathcal{H}_b^{(2)}, \mathcal{A}]$, and if x is generated by G then $\mathcal{B}$ is running $[\mathcal{H}_b^{(3)}, \mathcal{A}]$. Thus we have $|\mathsf{Adv}[\mathcal{A}]_b^{(2)} - \mathsf{Adv}[\mathcal{A}]_b^{(3)}| \leq \mu(\kappa)$. □

Lemma 7. If FE_0 is a non-adaptive $\epsilon_0(\kappa, q')$-secure FE scheme for functions in $\mathbb{RE}_{\mathsf{FE}_0}$ with an index space I_0, then for any $b \in \{0,1\}$ and any efficient adversary $\mathcal{A}$ who makes at most q functional key queries, we have $|\mathsf{Adv}[\mathcal{A}]_b^{(3)} - \mathsf{Adv}[\mathcal{A}]_b^{(4)}| \leq \epsilon_0(\kappa, \min\{q, |I_0|\})$.

Proof. We build an adversary $\mathcal{B}$ using $\mathcal{A}$ as an oracle to attack FE_0. For $b \in \{0,1\}$, we define $\mathcal{B}$ as follows:

$$\begin{aligned}
&\mathcal{B}(pk_0, c_0, \{fk_0^i\}_{i\in Q_0}) = ((x_0, x_1), \{e_i[c_i, pk_1^i]\}_{i\in Q_0}, b') \\
&\quad \textbf{where } pk = pk_0,\ c = c_0 \\
&\qquad r \leftarrow R,\ r' \leftarrow R,\ s \leftarrow R \\
&\qquad x_0 = (m_b, r'),\ x_1 = (\bot, s) \\
&\qquad \text{For all } (i,j) \in Q\text{:} \\
&\qquad\quad fk^{(i,j)} = \textbf{if } ((m_0, m_1) = \bot) \vee (f_{(i,j)} = \bot)) \textbf{ then } \bot \\
&\qquad\qquad\quad \textbf{else if } (f_{(i,j)}(m_0) = f_{(i,j)}(m_1)) \textbf{ then } (fk_0^i, fk_1^{(i,j)}) \textbf{ else } \top \\
&\qquad\qquad\quad \textbf{where } sk_1^i \leftarrow R \\
&\qquad\qquad\qquad pk_1^i = \mathsf{Setup}_1(sk_1^i) \\
&\qquad\qquad\qquad s_i = \mathsf{G}_i(s),\ \tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r')),\ c_i = s_i \oplus \tilde{c}_1^i \\
&\qquad\qquad\qquad fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_{(i,j)}, j) \\
&\qquad ((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}, b') = \mathcal{A}(pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})
\end{aligned}$$

We show that $\mathcal{B}$ is a valid adversary for the FE game, that is, the functions $e_i[c_i, pk_1^i]$ that appear in $\mathcal{B}$'s queries satisfy $e_i[c_i, pk_1^i](x_0) = e_i[c_i, pk_1^i](x_1)$ for all $i \in Q_0$. Since $x_0 = (m_b, r')$ and $x_1 = (\bot, s)$, by definition of $e_i[c_i, pk_1^i]$ we have

$$\begin{aligned}
e_i[c_i, pk_1^i](x_0) &= e_i[c_i, pk_1^i](m_b, r') = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r')), \\
e_i[c_i, pk_1^i](x_1) &= e_i[c_i, pk_1^i](\bot, s) = \mathsf{G}_i(s) \oplus c_i = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r')).
\end{aligned}$$

So indeed $e_i[c_i, pk_1^i](x_0) = e_i[c_i, pk_1^i](x_1)$.

Notice that if the input ciphertext c_0 is an encryption of m_0, i.e., $c_0 = \mathsf{Enc}_0(pk_0, (m_b, r'); r'')$ for some random string r'', then $\mathcal{B}$ is running $[\mathcal{H}_b^{(3)}, \mathcal{A}]$, and if $c_0 = \mathsf{Enc}_0(pk_0, (\bot, s); r'')$ then $\mathcal{B}$ is running $[\mathcal{H}_b^{(4)}, \mathcal{A}]$. Thus the advantage of $\mathcal{B}$ in the FE game is

$$\begin{aligned}
|2\Pr\{b_0' = b_0\} - 1| &= |\Pr\{b_0' = 0 \mid b_0 = 0\} + \Pr\{b_0' = 1 \mid b_0 = 1\} - 1| \\
&= |\Pr\{b' = 1 \mid b_0 = 0\} - \Pr\{b' = 1 \mid b_0 = 1\}|,
\end{aligned}$$

where $\Pr\{b' = 1 \mid b_0 = 0\} = \mathsf{Adv}[\mathcal{A}]_b^{(3)}$ and $\Pr\{b' = 1 \mid b_0 = 1\} = \mathsf{Adv}[\mathcal{A}]_b^{(4)}$. Since FE_0 is non-adaptively $\epsilon_0(\kappa, q')$-secure, and the number of functional key queries made by $\mathcal{B}$ is $|\mathrm{proj}_0(Q)| \leq \min\{q, |I_0|\}$, we have that $|\mathsf{Adv}[\mathcal{A}]_b^{(3)} - \mathsf{Adv}[\mathcal{A}]_b^{(4)}| \leq \epsilon_0(\kappa, \min\{q, |I_0|\})$. □

Lemma 8. If G is a μ-secure pseudorandom generator, then for any $b \in \{0,1\}$ and any adversary $\mathcal{A}$ we have $|\mathsf{Adv}[\mathcal{A}]_b^{(4)} - \mathsf{Adv}[\mathcal{A}]_b^{(5)}| \leq \mu(\kappa)$.

Proof. We build an adversary $\mathcal{B}$ to attack G using $\mathcal{A}$ as an oracle.

$$\begin{aligned}
&\mathcal{B}(x) = b' \\
&\quad \textbf{where } r_{i_1} \| \cdots \| r_{i_p} = x \\
&\qquad (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q}) = \mathcal{H}_b^{(1)}[\{r_i\}_{i\in Q_0}]((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}) \\
&\qquad ((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}, b') = \mathcal{A}(pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})
\end{aligned}$$

Notice that if x is truly random then $\mathcal{B}$ is running $[\mathcal{H}_b^{(4)}, \mathcal{A}]$, and if x is generated by G then $\mathcal{B}$ is running $[\mathcal{H}_b^{(5)}, \mathcal{A}]$. So we have

$$\Pr\{\mathcal{B}(x) = 1 \mid x \leftarrow R\} - \Pr\{\mathcal{B}(x) = 1 \mid \exists y. x = \mathsf{G}(y)\} = |\mathsf{Adv}[\mathcal{A}]_b^{(4)} - \mathsf{Adv}[\mathcal{A}]_b^{(5)}|.$$

Since G is μ-secure, $|\mathsf{Adv}[\mathcal{A}]_b^{(4)} - \mathsf{Adv}[\mathcal{A}]_b^{(5)}| \le \mu(\kappa)$. □

Lemma 9. If FE_1 is a non-adaptive $\epsilon_1(\kappa, q')$-secure FE scheme with an index space I_1, then for any efficient adversary $\mathcal{A}$ who makes at most q functional key queries, we have $|\mathsf{Adv}[\mathcal{A}]_0^{(5)} - \mathsf{Adv}[\mathcal{A}]_1^{(5)}| \le \max_{q_1+\cdots+q_p=q, q_i\le|I_1|} \sum_i \epsilon_1(\kappa, q_i)$.

Proof. To close the gap between $\mathcal{H}_0^{(5)}$ and $\mathcal{H}_1^{(5)}$, we build a sequence of hybrids. First we fix Q and suppose $p = |\mathrm{proj}_0(Q)|$. Without loss of generality, suppose $Q_0 = \mathrm{proj}_0(Q)$ contains indices $1, 2, \ldots, p$, and let $q_i = |\mathrm{proj}_1^i(Q)|$ for all $i \in Q_0$; then $q_1 + \cdots + q_p = q$. Let $\mathcal{H}_0^{(5.0)} = \mathcal{H}_0^{(5)}$, and for each $h \in Q_0$, we obtain $\mathcal{H}_0^{(5.h)}$ from $\mathcal{H}_0^{(5.(h-1))}$ by changing $\tilde{c}_1^h$ from encrypting m_0 to encrypting m_1. Notice that $\mathcal{H}_0^{(5,q_0)}$ is same as $\mathcal{H}_1^{(5)}$. Formally, for all $h \in Q_0$, let $\mathcal{H}_0^{(5.h)}$ be:

$$\begin{aligned}
&\mathcal{H}_0^{(5.h)}((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}) = (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q}) \\
&\quad \textbf{where} \\
&\qquad sk_0 \leftarrow R,\ r \leftarrow R,\ s \leftarrow R \\
&\qquad pk = \mathsf{Setup}_0(sk_0) \\
&\qquad r'' \leftarrow R,\ c = \mathsf{Enc}_0(pk, (\bot, s); r'') \\
&\qquad \text{For all } (i,j) \in Q\text{:} \\
&\qquad\quad fk^{(i,j)} = \textbf{if } ((m_0, m_1) = \bot) \vee (f_{(i,j)} = \bot)) \textbf{ then } \bot \\
&\qquad\qquad\qquad \textbf{else if } (f_{(i,j)}(m_0) = f_{(i,j)}(m_1)) \textbf{ then } (fk_0^i, fk_1^{(i,j)}) \textbf{ else } \top \\
&\qquad\qquad\qquad \textbf{where } sk_1^i \leftarrow R \\
&\qquad\qquad\qquad\qquad pk_1^i = \mathsf{Setup}_1(sk_1^i) \\
&\qquad\qquad\qquad\qquad s_i = \mathsf{G}_i(s),\ r_i \leftarrow R \\
&\qquad\qquad\qquad\qquad \tilde{c}_1^i = \textbf{if } i \le h \textbf{ then } \mathsf{Enc}_1(pk_1^i, m_1; r_i) \\
&\qquad\qquad\qquad\qquad\qquad\quad \textbf{else } \mathsf{Enc}_1(pk_1^i, m_0; r_i) \\
&\qquad\qquad\qquad\qquad c_i = s_i \oplus \tilde{c}_1^i \\
&\qquad\qquad\qquad\qquad fk_0^i = \mathsf{KeyGen}_0(sk_0, e_i[c_i, pk_1^i], i) \\
&\qquad\qquad\qquad\qquad fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_{(i,j)}, j)
\end{aligned}$$

We can build an adversary $\mathcal{B}$ to attack the FE scheme FE_1 using $\mathcal{A}$ as an oracle. The adversary $\mathcal{B}$ makes q_h functional key queries, using indices from $\mathrm{proj}_1^h(Q)$:

$$\begin{aligned}
&\mathcal{B}(pk_1, c_1, \{fk_1^{(h,j)}\}_{j\in\mathrm{proj}_1^h(Q)}) = ((m_0, m_1), \{f_{(h,j)}\}_{j\in\mathrm{proj}_1^h(Q)}, b') \\
&\quad \textbf{where} \\
&\qquad pk_1^h = pk_1,\ \tilde{c}_1^h = c_1 \\
&\qquad (pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q}) = \mathcal{H}_b^{(5.h)}[pk_1^h, \tilde{c}_1^h, \{fk_1^{(h,j)}\}_{j\in\mathrm{proj}_1^h(Q)}]((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}) \\
&\qquad ((m_0, m_1), \{f_{(i,j)}\}_{(i,j)\in Q}, b') = \mathcal{A}(pk, c, \{fk^{(i,j)}\}_{(i,j)\in Q})
\end{aligned}$$

If $c_1 = \mathsf{Enc}_1(pk_1, m_0; \tilde{r})$ for some randomness $\tilde{r}$ then $\mathcal{B}$ is running $[\mathcal{H}_0^{(5.(h-1))}, \mathcal{A}]$, and if $c_1 = \mathsf{Enc}_1(pk_1, m_1; \tilde{r})$ then $\mathcal{B}$ is running $[\mathcal{H}_0^{(5.h)}, \mathcal{A}]$. So the advantage of $\mathcal{B}$ in winning the FE game for FE_1 is $|\mathsf{Adv}[\mathcal{A}]_0^{(5.(h-1))} - \mathsf{Adv}[\mathcal{A}]_0^{(5.h)}| \leq \epsilon_1(\kappa, q_h)$. Since $\mathcal{H}_0^{(5.0)}$ is same as $\mathcal{H}_0^{(5)}$ and $\mathcal{H}_0^{(5.q_0)}$ is same as $\mathcal{H}_1^{(5)}$, it follows by triangle inequality that

$$|\Pr\{[\mathcal{H}_0^{(5.0)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_0^{(5,q_0)}, \mathcal{A}] = 1\}| \leq \sum_i \epsilon_1(\kappa, q_i).$$

Therefore for any adversary $\mathcal{A}$ whose index set Q satisfies $|Q| = q$, it holds that

$$|\mathsf{Adv}[\mathcal{A}]_0^{(5)} - \mathsf{Adv}[\mathcal{A}]_1^{(5)}| \leq \max_{q_1+\cdots+q_p=q, q_i \leq |I_1|} \sum_i \epsilon_1(\kappa, q_i).$$

□

B Secret-Key Functional Encryption

In this section we present the secret-key version of our SUM and PRODUCT constructions and transformations composed using them as building blocks. First we formalize our syntax of secret-key FE. A *secret-key functional encryption scheme* is parameterized by four sets M, R, I, F just as in the public-key settings. For simplicity, we assume that the secret key sk is sampled by an external procedure. Thus a SKFE scheme consists of three algorithms:

- $\mathsf{Enc}(sk, m; r) = c$ is an encryption algorithm that on input a secret-key sk and a message $m \in M$, using the randomness $r \in R$, produces a ciphertext $c \in M$.
- $\mathsf{KeyGen}(sk, f, i) = \mathit{fk}$ is a functional key derivation algorithm that on input a secret-key sk, a function $f \in F$, and an index $i \in I$, produces a functional decryption key fk associated to f.
- $\mathsf{Dec}(\mathit{fk}, c) = m'$ is a decryption algorithm that on input a functional decryption key fk and a ciphertext c, produces a plaintext message m'.

The scheme is correct if with overwhelming probability (over the choice of $s, r \in R$), for any message $m \in M$, function $f \in F$, and index $i \in I$, it holds that

$$\mathsf{Dec}(\mathsf{KeyGen}(\mathsf{Setup}(s), f, i), \mathsf{Enc}(\mathsf{Setup}(s), m; r)) = m.$$

The security notions of secret-key FE have more variations than public-key FE. In general they can be classified into two categories: message-private security and function-private security. When we adapt the public-key FE security of Definition 2 to the secret-key settings, we obtain message-private security. On the other hand, function-private security is stronger than message-private security as it requires that an adversary can distinguish neither functional decryption keys of equivalent functions nor ciphertexts encrypting equivalent messages. Exactly speaking, in function-private models, an adversary can submit functional queries of the form (f_0, f_1) and message queries of the form (m_0, m_1), and he will receive the functional decryption key fk_b for f_b and the ciphertext c that encrypts m_b for a uniform random bit $b \in \{0, 1\}$. Such an adversary is valid if for any functional queries (f_0, f_1) and any message queries (m_0, m_1) it holds that $f_0(m_0) = f_1(m_1)$. Function-privacy is an important notion in the secret-key settings, but it is in general not achievable in the public-key settings. Brakerski and Segev [13] showed that any message-private SKFE scheme can be transformed generically into a function-private SKFE scheme. Such transformation preserves the maximal numbers of both ciphertexts and functional decryption keys that can be released to an adversary.

Unlike the public-key settings, a SKFE scheme that is secure with respect to L challenge ciphertexts may not be secure with respect to $L + 1$ challenge ciphertexts. So we consider L as another parameter in the security definitions.

Our transformations apply in all three variations of function-private security. We now state its definition as follows.

Fully function-private SKFE Let $\mathsf{Enc}_b(sk,\cdot,\cdot)$ and $\mathsf{KeyGen}_b(sk,\cdot,\cdot)$ be left-right encryption and functional key oracle with the following definitions:

$$\mathsf{Enc}_b(sk,m_0,m_1)=\mathsf{Enc}(sk,m_b),\quad \mathsf{KeyGen}_b(sk,f_0,f_1)=\mathsf{KeyGen}(sk,f_b,i) \text{ where } i\leftarrow I.$$

A SKFE scheme is *q-key L-message* $\epsilon(\kappa,q,L)$*-fully function-private secure* if for any efficient adversary $\mathcal{A}$ who makes at most q queries to the oracle $\mathsf{KeyGen}_b(sk,\cdot,\cdot)$ and at most L queries to the oracle $\mathsf{Enc}_b(sk,\cdot,\cdot)$ for $b\in\{0,1\}$, the function $\epsilon(\kappa,q,L)$ is negligible, and it holds that

$$\mathsf{Adv}^{\mathsf{FP}}_{\mathsf{FE}}[\mathcal{A}]=\left|\Pr\{\mathcal{A}^{\mathsf{KeyGen}_0(sk,\cdot,\cdot),\mathsf{Enc}_0(sk,\cdot,\cdot)}(1^\kappa)=1\}-\Pr\{\mathcal{A}^{\mathsf{KeyGen}_1(sk,\cdot,\cdot),\mathsf{Enc}_1(sk,\cdot,\cdot)}(1^\kappa)=1\}\right|\leq\epsilon(\kappa,q,L)^{\Omega(1)}.$$

When q can be an arbitrary polynomial in κ, we say that FE is *many-key secure* or *collusion-resistant*; when L can be an arbitrary polynomial in κ, we say that FE is *many-message secure*.

We can expand the definitions of both oracles in a challenger $\mathcal{H}^{\mathsf{FP}}_{(\cdot)}$ to let it answer oracle queries. Note that the oracle queries can still be adaptive with this notation. As in Remark 1, we can change the indices i from being uniformly random to being distinct elements of the index space I provided that $|I|$ is superpolynomial in κ.

$$\begin{aligned}
&\mathcal{H}^{\mathsf{FP}}_b(\{(m^l_0,m^l_1)\}_{l\in[L]},\{(f^i_0,f^i_1)\}_{i\in Q})=(\{c_l\}_{l\in[L]},\{\mathit{fk}^i\}_{i\in Q})\\
&\quad\text{where } sk\leftarrow R\\
&\qquad\text{For all } l\in[L]:\\
&\qquad\quad r_l\leftarrow R\\
&\qquad\quad c_l=\mathsf{Enc}(sk,m^l_b;r_l)\\
&\qquad\text{For all } i\in Q:\\
&\qquad\quad \mathit{fk}^i=\textbf{if } (f^i_0,f^i_1)=\perp \textbf{ then } \perp\\
&\qquad\qquad\qquad \textbf{else if } \forall l\in[L].(f^i_0(m^l_0)=f^i_1(m^l_1)) \textbf{ then } \mathsf{KeyGen}(sk,f^i_b,i)\\
&\qquad\qquad\qquad \textbf{else } \top
\end{aligned}$$

The symbol $\top$ denotes an error that the adversary has submitted an invalid query.[6] Then the security is defined via the following experiment:

$$\begin{aligned}
&\mathsf{Exp}_{SKFE}[\mathcal{H}^{\mathsf{FP}}_{(\cdot)},\mathcal{A}]=(b\stackrel{?}{=}b')\\
&\quad\text{where } b\leftarrow\{0,1\}\\
&\qquad(\{c_l\}_{i\in[L]},\{\mathit{fk}^i\}_{i\in Q})=\mathcal{H}^{\mathsf{SKFE}}_b(\{(m^l_0,m^l_1)\}_{l\in[L]},\{(f^i_0,f^i_1)\}_{i\in Q})\\
&\qquad(\{(m^l_0,m^l_1)\}_{l\in[L]},\{(f^i_0,f^i_1)\}_{i\in Q},b')=\mathcal{A}(\{c_l\}_{l\in[L]},\{\mathit{fk}^i\}_{i\in Q})
\end{aligned}$$

The advantage of any efficient adversary $\mathcal{A}$ must be bounded by $\epsilon(\kappa,q,L)$:

$$\mathsf{Adv}^{\mathsf{FP}}_{\mathsf{FE}}[\mathcal{A}]=\left|\Pr\{[\mathcal{H}^{\mathsf{FP}}_0,\mathcal{A}]=1\}-\Pr\{[\mathcal{H}^{\mathsf{FP}}_1,\mathcal{A}]=1\}\right|\leq\epsilon(\kappa,q,L)^{\Omega(1)}.$$

Other variations of function-privacy First, the notion of selective-message function-privacy requires that an adversary must submit message queries before he can access the functional key oracle. We can define this security by modifying the condition for releasing functional decryption keys:

$$\begin{aligned}
&\mathit{fk}^i=\textbf{if } f^i_0=\perp\vee f^i_1=\perp\vee(\exists l\in[L].(m^l_0=\perp\vee m^l_1=\perp)) \textbf{ then } \perp\\
&\qquad\textbf{else if } \forall l\in[L].(f^i_0(m^l_0)=f^i_1(m^l_1)) \textbf{ then } \mathsf{KeyGen}(sk,f^i_b,i)\\
&\textbf{else } \top
\end{aligned}$$

Next, the notion of selective-function function-privacy requires that an adversary must submit functional key queries before accessing the encryption oracle. We modify the condition for releasing the ciphertext and the functional key to model this security:

[6] Formally we can extend the flat CPO of all functional decryption keys with an "top" element that is greater than all keys, such that the challenger remains a monotone function.

$$c_l = \textbf{if } (\forall i.(f_0^i, f_1^i) \neq \perp) \textbf{ then } \mathsf{Enc}(sk, m_b^l; r_l) \textbf{ else } \perp$$
$$fk^i = \textbf{if } \exists l \in Q.(f_i = \perp) \textbf{ then } \perp$$
$$\textbf{else if } \forall l \in [L].(f_i(m_0^l) = f_i(m_1^l)) \textbf{ then } \mathsf{KeyGen}(sk, f_i, i) \textbf{ else } \top$$

In the following, we focus on the function-private security of our transformation, and we resort to the [13] transformation to achieve function-privacy from message-privacy. Thus, when we say a SKFE scheme is secure, we always mean it is function-private secure.

SUM construction for SKFE Let FE_0 and FE_1 be SKFE schemes for a function class F with index spaces I_0 and I_1. We can modify our SUM construction $\mathsf{FE} = \mathsf{FE}_0 + \mathsf{FE}_1$ in the secret-key settings as follows:

- $\mathsf{Enc}(sk, m; r) = (\mathsf{Enc}_0(\mathsf{G}_0(sk), m; \mathsf{G}_0(r)), \mathsf{Enc}_1(\mathsf{G}_1(sk), m; \mathsf{G}_1(r)))$
- $\mathsf{Dec}((b, fk), (c_0, c_1)) = \mathsf{Dec}_b(fk, c_b)$
- $\mathsf{KeyGen}(sk, f, (b, i)) = (b, \mathsf{KeyGen}_b(\mathsf{G}_b(sk), f, i))$

Theorem 9. *Assume for $i \in \{0, 1\}$, FE_i is L-message ϵ_i-fully (or selective-message, selective-function, respectively) secure with index spaces I_0 and I_1. Then $\mathsf{FE}_0 + \mathsf{FE}_1$ is L-message $(\epsilon_0 + \epsilon_1)$-fully (or selective-message, selective-function, respectively) secure with an index space $I_0 + I_1$.*

PRODUCT construction for SKFE In the secret-key settings, the PRODUCT construction can be modified as follows. Let I_i be the index space of FE_i for $i = 0, 1$. Suppose $L \geq 1$ is an fixed polynomial in κ such that both FE_0 and FE_1 are L-message secure. For $i \in I_0$, $c_l \in M$ for $l \in [L]$ and $sk \in R$, the "re-encryption" function $e_i[c_1, \ldots, c_L, sk] : M \times R \times [L] \to M$ is defined by:

$$e_i[c_1, \ldots, c_L, sk](m, \tilde{r}, l) = \begin{cases} \mathsf{G}_l(\mathsf{G}_i(\tilde{r})) \oplus c_l & \text{if } m = \perp \\ \mathsf{Enc}_1(sk, m; \mathsf{G}_i(\tilde{r})) & \text{otherwise} \end{cases}$$

The PRODUCT SKFE scheme $\mathsf{FE}_0 \times \mathsf{FE}_1$ can then be defined as:

- $\mathsf{Enc}(sk, m; r) = \mathsf{Enc}_0(sk, (m, \mathsf{G}_0(r), \perp); \mathsf{G}_1(r))$
- $\mathsf{Dec}((fk_0, fk_1), c) = \mathsf{Dec}_1(fk_1, \mathsf{Dec}_0(fk_0, c))$
- $\mathsf{KeyGen}(sk, f, (i, j)) = (fk_0^i, fk_1^{(i,j)})$ where

$$sk_0 = sk$$
$$sk_1^i = \mathsf{G}_i(\mathsf{G}_1(sk))$$
$$c_l^i = \mathsf{G}_l(\mathsf{G}_i(\mathsf{G}_2(sk))) \text{ for all } l \in [L]$$
$$fk_0^i = \mathsf{KeyGen}_0(sk_0, e_i[c_1^i, \ldots, c_L^i, sk_1^i], i)$$
$$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f, j)$$

Theorem 10. *Assume FE_h is L-message $\epsilon_h(\kappa, q, L)$-fully (selective-message, selective-function) secure with an index space I_h for $h = 0, 1$. Then $\mathsf{FE}_0 \times \mathsf{FE}_1$ is L-message $\epsilon(\kappa, q, L)$-fully (selective-message, selective-function, respectively) secure with an index space $I_0 \times I_1$ such that*

$$\epsilon(\kappa, q, L) \leq \max_{q_1 + \cdots + q_p = q, q_i \leq |I_1|} \sum_i \epsilon_1(\kappa, q_i, L) + 2\epsilon_0(\kappa, \min\{q, |I_0|\}, L) + 18\mu(\kappa).$$

The proof of security of the PRODUCT construction is similar to the public-key setting. In the following we show the full function-private version. We build a series of hybrids $\mathcal{H}_b^{(h)}$ to reduce the security of $\mathsf{FE}_0 \times \mathsf{FE}_1$ to the securities of SKFE schemes FE_0 and FE_1 and that of the PRG G.

$\mathcal{H}_b^{(0)}$: This is the same as the original challenger $\mathcal{H}_b^{\mathsf{FE}_0 \times \mathsf{FE}_1}$ in the SKFE game for the scheme $\mathsf{FE}_0 \times \mathsf{FE}_1$.

$\mathcal{H}_b^{(0)}(\{(m_0^l, m_1^l)\}_{l\in[L]}, \{(f_0^{(i,j)}, f_1^{(i,j)}\}_{(i,j)\in Q}) = (\{c_l\}_{l\in[L]}, \{fk^{(i,j)}\}_{(i,j)\in Q})$

where

$sk \leftarrow R$

For all $l \in [L]$:

$r_l \leftarrow R$

$c_l = \mathsf{Enc}_0(sk, (m_b^l, \mathsf{G}_0(r_l), \bot); \mathsf{G}_1(r_l))$

For all $(i,j) \in Q$:

$fk^{(i,j)} =$ **if** $(f_0^{(i,j)}, f_1^{(i,j)}) = \bot$ **then** $\bot$

else if $\forall l \in [L].(f_0^{(i,j)}(m_0^l) = f_1^{(i,j)}(m_1^l))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$

where $sk_1^i = \mathsf{G}_i(\mathsf{G}_1(sk))$

$\forall l \in [L].c_l^i = \mathsf{G}_l(\mathsf{G}_i(\mathsf{G}_2(sk)))$

$fk_0^i = \mathsf{KeyGen}_0(sk, e_i[c_1^i, \ldots, c_L^i, sk_1^i], i)$

$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_b^{(i,j)}, j)$

$\mathcal{H}_b^{(1)}$: In this hybrid we replace some uses of G with truly random bitstrings.

$\mathcal{H}_b^{(1)}(\{(m_0^l, m_1^l)\}_{l\in[L]}, \{(f_0^{(i,j)}, f_1^{(i,j)})\}_{(i,j)\in Q}) = (\{c_l\}_{l\in[L]}, \{fk^{(i,j)}\}_{(i,j)\in Q})$

where

$sk \leftarrow R$

For all $l \in [L]$:

$r'_l, r''_l \leftarrow R$

$c_l = \mathsf{Enc}_0(sk, (m_b^l, r'_l, \bot); r''_l)$

For all $i \in \mathrm{proj}_0(Q)$:

$sk_1^i \leftarrow R$

$\forall l \in [L].c_l^i \leftarrow R$

$fk_0^i = \mathsf{KeyGen}_0(sk, e_i[c_1^i, \ldots, c_L^i, sk_1^i], i)$

For all $j \in \mathrm{proj}_1^i(Q)$:

$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_b^{(i,j)}, j)$

$fk^{(i,j)} =$ **if** $(f_0^{(i,j)}, f_1^{(i,j)}) = \bot$ **then** $\bot$

else if $\forall l \in [L].(f_0^{(i,j)}(m_0^l) = f_1^{(i,j)}(m_1^l))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$

Since G is μ-secure, we have that $|\mathsf{Adv}[\mathcal{A}]_b^{(0)} - \mathsf{Adv}[\mathcal{A}]_b^{(1)}| \leq 6\mu(\kappa)$.

$\mathcal{H}_b^{(2)}$: In this hybrid we modify c_l^i without changing their distributions:

$\mathcal{H}_b^{(2)}(\{(m_0^l, m_1^l)\}_{l\in[L]}, \{(f_0^{(i,j)}, f_1^{(i,j)})\}_{(i,j)\in Q}) = (\{c_l\}_{l\in[L]}, \{fk^{(i,j)}\}_{(i,j)\in Q})$

where

$sk \leftarrow R$

For all $l \in [L]$:

$r'_l, r''_l \leftarrow R$

$c_l = \mathsf{Enc}_0(sk, (m_b^l, r'_l, \bot); r''_l)$

For all $i \in \mathrm{proj}_0(Q)$:

$sk_1^i \leftarrow R$

$\forall l \in [L].s_l^i \leftarrow R, c_l^i = s_l^i \oplus \mathsf{Enc}_1(sk_1^i, m_b^l; \mathsf{G}_i(r'_l))$

$fk_0^i = \mathsf{KeyGen}_0(sk, e_i[c_1^i, \ldots, c_L^i, sk_1^i], i)$

For all $j \in \mathrm{proj}_1^i(Q)$:

$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_b^{(i,j)}, j)$

$fk^{(i,j)} =$ **if** $(f_0^{(i,j)}, f_1^{(i,j)}) = \bot$ **then** $\bot$

else if $\forall l \in [L].(f_0^{(i,j)}(m_0^l) = f_1^{(i,j)}(m_1^l))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$

$\mathcal{H}_b^{(3)}$**:** Next we replace the truly random s_l^i with pseudorandom bitstrings:

$$\mathcal{H}_b^{(3)}(\{(m_0^l, m_1^l)\}_{l\in[L]}, \{(f_0^{(i,j)}, f_1^{(i,j)})\}_{(i,j)\in Q}) = (\{c_l\}_{l\in[L]}, \{fk^{(i,j)}\}_{(i,j)\in Q})$$
where
$sk \leftarrow R$, $s \leftarrow R$
For all $l \in [L]$:
$r'_l, r''_l \leftarrow R$
$c_l = \mathsf{Enc}_0(sk, (m_b^l, r'_l, \bot); r''_l)$
For all $i \in \mathrm{proj}_0(Q)$:
$sk_1^i \leftarrow R$
$\forall l \in [L].s_l^i = \mathsf{G}_l(\mathsf{G}_i(s)), c_l^i = s_l^i \oplus \mathsf{Enc}_1(sk_1^i, m_b^l; \mathsf{G}_i(r'_l))$
$fk_0^i = \mathsf{KeyGen}_0(sk, e_i[c_1^i, \ldots, c_L^i, sk_1^i], i)$
For all $j \in \mathrm{proj}_1^i(Q)$:
$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_b^{(i,j)}, j)$
$fk^{(i,j)} =$ **if** $(f_0^{(i,j)}, f_1^{(i,j)}) = \bot$ **then** $\bot$
else if $\forall l \in [L].(f_0^{(i,j)}(m_0^l) = f_1^{(i,j)}(m_1^l))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$

Again, since G is μ-secure, we have that $|\mathsf{Adv}[\mathcal{A}]_b^{(2)} - \mathsf{Adv}[\mathcal{A}]_b^{(3)}| \leq 2\mu(\kappa)$.

$\mathcal{H}_b^{(4)}$**:** Now we encrypt $(\bot, s, l)$ instead of $(m_b^l, r'_l, \bot)$ to get ciphertext c_l, and we generate functional keys for $e_i[c_1^i, \ldots, c_L^i, \bot]$ instead of $e_i[c_1^i, \ldots, c_L^i, sk_1^i]$ to get fk_0^i:

$$\mathcal{H}_b^{(4)}(\{(m_0^l, m_1^l)\}_{l\in[L]}, \{(f_0^{(i,j)}, f_1^{(i,j)})\}_{(i,j)\in Q}) = (\{c_l\}_{l\in[L]}, \{fk^{(i,j)}\}_{(i,j)\in Q})$$
where
$sk \leftarrow R$, $s \leftarrow R$
For all $l \in [L]$:
$r'_l, r''_l \leftarrow R$
$c_l = \mathsf{Enc}_0(sk, (\bot, s, l); r''_l)$
For all $i \in \mathrm{proj}_0(Q)$:
$sk_1^i \leftarrow R$
$\forall l \in [L].s_l^i = \mathsf{G}_l(\mathsf{G}_i(s)), c_l^i = s_l^i \oplus \mathsf{Enc}_1(sk_1^i, m_b^l; \mathsf{G}_i(r'_l))$
$fk_0^i = \mathsf{KeyGen}_0(sk, e_i[c_1^i, \ldots, c_L^i, \bot], i)$
For all $j \in \mathrm{proj}_1^i(Q)$:
$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_b^{(i,j)}, j)$
$fk^{(i,j)} =$ **if** $(f_0^{(i,j)}, f_1^{(i,j)}) = \bot$ **then** $\bot$ **else if** $\forall l \in [L].(f_0^{(i,j)}(m_0^l) = f_1^{(i,j)}(m_1^l))$ **else** $\top$

Notice that, c_l is an encryption of $(m_b^l, r'_l, \bot)$ in $\mathcal{H}_b^{(3)}$, and it is an encryption of $(\bot, s, l)$ in $\mathcal{H}_b^{(4)}$. Since for all $i \in \mathrm{proj}_0(Q)$ and $l \in [L]$ we have

$$e_i[c_1^i, \ldots, c_L^i, sk_1^i](m_b^l, r'_l, \bot) = \mathsf{Enc}_1(sk_1^i, m_b^l; \mathsf{G}_i(r'_l)),$$
$$e_i[c_1^i, \ldots, c_L^i, \bot](\bot, s, l) = \mathsf{G}_l(\mathsf{G}_i(s)) \oplus c_l = \mathsf{Enc}_1(sk_1^i, m_b^l, \mathsf{G}_i(r'_l)),$$

and since FE_0 is function-private, we can build an adversary $\mathcal{B}$ to attack FE_0 using an adversary $\mathcal{A}$ that attacks $\mathsf{FE}_0 \times \mathsf{FE}_1$ as follows: $\mathcal{B}$ submits function queries $(e_i[c_1^i, \ldots, c_L^i, sk_1^i], e_i[c_1^i, \ldots, c_L^i, \bot])$ for $i \in \mathrm{proj}_0(Q)$ and message queries $((m_b^l, r'_l, \bot), (\bot, s, l))$ for $l \in [L]$. With probability $1/2$, it is given either the ciphertext c_l that encrypts $(m_b^l, r'_l, \bot)$ and the functional keys for $e_i[c_1^i, \ldots, c_L^i, sk_1^i]$ as in $\mathcal{H}_b^{(3)}$, or the ciphertexts c_l that encrypts $(\bot, s, l)$ and the functional keys for $e_i[c_1^i, \ldots, c_L^i, \bot]$ as in $\mathcal{H}_b^{(4)}$. Since FE_0 is L-message $\epsilon_0(\kappa, q, L)$-secure, we have that $|\mathsf{Adv}[\mathcal{A}]_b^{(3)} - \mathsf{Adv}[\mathcal{A}]_b^{(4)}| \leq \epsilon_0(\kappa, \min\{q, |I_0|, L\})$.

$\mathcal{H}_b^{(5)}$**:** Finally we use fresh randomness to encrypt m_b^l under now independent secret keys sk_1^i:

$$\mathcal{H}_b^{(4)}(\{(m_0^l, m_1^l)\}_{l\in[L]}, \{(f_0^{(i,j)}, f_1^{(i,j)})\}_{(i,j)\in Q}) = (\{c_l\}_{l\in[L]}, \{fk^{(i,j)}\}_{(i,j)\in Q})$$

where

$sk \leftarrow R$, $s \leftarrow R$

For all $l \in [L]$:

$r'_l, r''_l \leftarrow R$

$c_l = \mathsf{Enc}_0(sk, (\bot, s, l); r''_l)$

For all $i \in \mathrm{proj}_0(Q)$:

$sk_1^i \leftarrow R$

$\forall l \in [L].s_l^i = \mathsf{G}_l(\mathsf{G}_i(s)), r_l^i \leftarrow R, c_l^i = s_l^i \oplus \mathsf{Enc}_1(sk_1^i, m_b^l; r_l^i)$

$fk_0^i = \mathsf{KeyGen}_0(sk, e_i[c_1^i, \ldots, c_L^i, \bot], i)$

For all $j \in \mathrm{proj}_1^i(Q)$:

$fk_1^{(i,j)} = \mathsf{KeyGen}_1(sk_1^i, f_b^{(i,j)}, j)$

$fk^{(i,j)} =$ **if** $(f_0^{(i,j)}, f_1^{(i,j)}) = \bot$ **then** $\bot$

else if $\forall l \in [L].(f_0^{(i,j)}(m_0^l) = f_1^{(i,j)}(m_1^l))$ **then** $(fk_0^i, fk_1^{(i,j)})$ **else** $\top$

Using the PRG security of G, we can bound the difference between $\mathcal{H}_b^{(4)}$ and $\mathcal{H}_b^{(5)}$:

$$|\mathsf{Adv}[\mathcal{A}]_b^{(4)} - \mathsf{Adv}[\mathcal{A}]_b^{(5)}| \leq \mu(\kappa).$$

Then notice that the hybrid $\mathcal{H}_b^{(5)}$ runs $|\mathrm{proj}_0(Q)|$ independent copies of FE_1, and also notice that the re-encryption function e_i no longer contains the secret key sk_1^i of FE_1, so

$$|\mathsf{Adv}[\mathcal{A}]_0^{(5)} - \mathsf{Adv}[\mathcal{A}]_1^{(5)}| \leq \max_{q_1+\cdots+q_p=q, q_i\leq|I_1|} \sum_i \epsilon_1(\kappa, q_i, L).$$

The upper bound in Theorem 10 follows by combining the above hybrids for $b = 0, 1$.

Achieving collusion-resistant SKFE from compact SKFE We can directly adapt the transformations of Section 5 to the secret-key settings, and as a result, we obtain generic transformations that can turn a single-key L-message function-private SKFE scheme into a collusion-resistant L-message function-private SKFE scheme for a fixed L.

Formally, assume FE_0 is a succinct L-message single-key function-private SKFE scheme for a function class F. Then we define two transformations inductively using the SUM and PRODUCT constructions:

1. $\mathsf{FE}_1 = \mathsf{FE}_0 + \mathsf{FE}_0$, $\mathsf{FE}_{h+1} = \mathsf{FE}_h \times \mathsf{FE}_h$ for $h > 0$.
2. $\mathsf{FE}_1 = \mathsf{FE}_0 + \mathsf{FE}_0$, $\mathsf{FE}_{h+1} = \mathsf{FE}_1 \times \mathsf{FE}_h$ for $h > 0$.

Let $\mathbb{RE}_{\mathsf{FE}_h}$ be the class of functions containing the "re-encryption" function $e_i^{(h)}[c_1, \ldots, c_L, sk]$ of FE_h in both transformations, where $L = 2^h$. Furthermore, assume G is a μ-secure pseudorandom generator. For the first transformation, let $p = \omega(\log\log\kappa)$ and $p = O(\log\kappa)$, and assume $\cup_{i=0}^p \mathbb{RE}_{\mathsf{FE}_i} \subseteq F$ and $\mathsf{G} \in F$, then FE_p is a L-message collusion-resistant function-private SKFE scheme for F. For the second transformation, let $p = \omega(\log\kappa)$ and $p = O(\kappa)$, and assume $\mathbb{RE}_{\mathsf{FE}_1} \subseteq F$ and $\mathsf{G} \in F$, then FE_p is a L-message collusion-resistant function-private SKFE scheme for F. For both transformations, the security and efficiency measures are the same as their public-key counterparts.

We remark that, our transformations work only for a fixed and bounded L: when L is unbounded, we would have to embed a superpolynomial number of ciphertexts c_l in the re-encryption function e_i and hence the complexity of the key derivation algorithm blows up. It is an interesting open question whether one can build a transformation from our SUM and PRODUCT constructions to produce SKFE schemes that are secure when an adversary can get an unbounded number of ciphertexts and an unbounded number of functional decryption keys.