Compactness vs Collusion Resistance in Functional Encryption*

Baiyu Li[†] Daniele Micciancio[‡]

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 to 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 [25], 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. [21] 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 [21], deniable encryption [30], and much more (e.g., see [16, 8].) However, the problem of building an obfuscator with a solid proof of security is still far from being solved. The multilinear-map problems [20, 18, 22, 19] underlying most known candidate $i\mathcal{O}$ constructions [21, 11, 6, 5, 29, 23] have recently been subject to attacks [15, 17], 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 towards the construction of $i\mathcal{O}$ from standard assumptions has recently been opened by Bitansky and Vaikuntanathan [9] 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 [24, 14, 32, 2, 12]. We recall that a (public key) functional encryption scheme [10, 28, 31, 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 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)

*This work was sponsored in part by the Defense Advanced Research Projects Agency (DARPA) and the U.S. Army Research Office under contract number W911NF-15-C-0226.

[†]University of California, San Diego, USA. E-mail: `bal007@cs.ucsd.edu`

[‡]University of California, San Diego, USA. E-mail: `daniele@cs.ucsd.edu`. Research supported in part by DARPA Safeware grant, and NSF grant CNS-1528068.

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 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 will focus on schemes for which f can be any polynomial sized circuit. Interestingly, [24] 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 keys (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 [9, 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 [9] 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 towards 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 [9, 3]) is summarized by the solid arrows in the following diagram:

$$\begin{tikzcd}
\text{Weak Collusion-Resistant} \arrow[d] & \text{Collusion-Resistant} \arrow[l] \arrow[d] \\
\text{Weakly Compact} \arrow[ur, dotted] & \text{Compact} \arrow[l]
\end{tikzcd}$$

where the horizontal implications are trivial (from stronger to weaker constraints on the t^{Enc}) and the vertical implications are from [9, 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 logarithmic $\log(Q)$, as opposed to linear $O(Q)$ as in the trivial construction. This mild dependence on Q results in much better security: while the security of the iterated product construction degrades linearly with Q, the security of the iterated squaring construction degrades only logarithmically in Q.

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 function. We consider the simplicity of the constructions in this work as a positive feature.

1.2 Other related work

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 [26], where sum and product constructions are used to build forward secure signature schemes supporting an arbitrary number of updates, starting from regular signatures (i.e., supporting no 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 [13] 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 [13] 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., key generation and encryption procedure of a cryptosystem) all use precisely κ bits of randomness, where κ is the 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.

We consider only public key functional encryption schemes in our work, 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* (PKey*,* Enc*,* Dec*,* Fun) *where*

- $\mathsf{PKey}(sk) = pk$ *is a public key generation algorithm that on input a random secret key $sk \in R$, produces a corresponding public key pk.*
- $\mathsf{Enc}(pk, m; r) = c$ *is an encryption algorithm that on input a public key pk, message $m \in M$ and randomness $r \in R$, produces a ciphertext c*
- $\mathsf{Fun}(sk, f, i) = fk$ *is a functional key derivation algorithm that on input a secret key sk, a function $f \in F$, and an index i, 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

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

Our syntax for functional encryption schemes slightly differs from the standard one is two respects. First, we identify the randomness used by the key generation procedure with the secret key of the scheme. This is without loss of generality, but provides a more convenient definition for our constructions. The other is that the functional key derivation algorithm Fun takes an index i as an additional parameter. The only requirement on this index is that different calls to $\mathsf{Fun}(sk, \cdot, i)$ use different values of i. The role of i is simply to put a bound on the number of calls to the key derivation algorithm. (In particular, the indexes $i \in I$ can be used in any order.) 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.

Security Since our work is primarily motivated by the application of FE to indistinguishability obfuscation [9, 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 [10], expressed in the functional/equational style of [27]. 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 $\mathcal{H}^{\mathsf{FE}}((m_0, m_1), \{f_i\}_{i \in I}) = (pk, c, \{fk^i\}_{i \in I})$ that receives as input a pair of message $(m_0, m_1) \in M^2$ and collection of function queries $f_i \in F$, and outputs a public key pk, ciphertext c and collection of functional keys fk^i for $i \in I$. The adversary is a function $\mathcal{A}(pk, c, \{fk^i\}_{i \in I}) = ((m_0, m_1), \{f_i\}_{i \in I}, b')$ that on input a public key pk, ciphertext c and functional keys $\{fk^i\}_{i \in I}$ outputs a pair of messages (m_0, m_1), function queries $\{f_i\}_{i \in I}$ and decision bit b'. 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 variable can take a special *undefined* value $\perp$, subject to the natural monotonicity requirements. See [27] for details.)

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

$$\begin{aligned}
&\mathcal{H}_b^{\mathsf{FE}}((m_0, m_1), \{f_i\}_{i \in I}) = (pk, c, \{fk^i\}_{i \in I}) \\
&\quad \text{where } sk \leftarrow R,\ r \leftarrow R \\
&\qquad\quad pk = \mathsf{PKey}(sk) \\
&\qquad\quad c = \mathsf{Enc}(pk, m_b; r) \\
&\qquad\quad \text{For all } i \in I: \\
&\qquad\qquad \textbf{if } fk^i = (f_i(m_0) = f_i(m_1) \neq \perp) \textbf{ then } \mathsf{Fun}(sk, f_i, i) \textbf{ else } \perp
\end{aligned}$$

By the notation $x \leftarrow R$ we mean the operation of selecting an element uniformly at random from R. Note that, 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

- $pk =$ **if** $((m_0, m_1) \neq \perp)$ **then** $\mathsf{PKey}(sk)$ **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.
- $fk^i = \mathsf{Fun}(sk, f_i, i)$ and $c =$ **if** $(\forall i. f_i(m_0) = f_i(m_1))$ **then** $\mathsf{Enc}(pk, m_b; r)$ **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.

A FE game $\mathsf{Exp}_{FE}[\mathcal{H}_{(\cdot)}^{\mathsf{FE}}, \mathcal{A}]$ is defined by the following system of equations:

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

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 game outputs $\top$, 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':

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

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

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

The two formulations are easily seen to be perfectly equivalent.

Definition 2. *A functional encryption scheme* FE *is* (q, ϵ)*-non-adaptively (or selectively/adaptively) secure if* $|I| = q$ *and for any efficient adversary* $\mathcal{A}$ *there exists a negligible function* $\epsilon(\kappa)$ *such that the advantage of* $\mathcal{A}$ *in the non-adaptive (or selective/adaptive) FE game is bounded by* $\mathsf{Adv}_{\mathsf{FE}}[\mathcal{A}] \leq \epsilon(\kappa)$*.*

When $\epsilon(\kappa)$ is negligible, for simplicity we sometimes omit it and just say a FE scheme is q-secure, where $q = |I|$ as in the definition above.

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

- FE is *compact*[2] 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*[3] if t^{Enc} is sub-linear in $|I|$ 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 $|I|$ but is polynomials in n, κ, and d.

The notion of compact FE has been considered in [3, 4], and also in [9] under the name *fully circuit succinct*. Here we choose the name "compact" to distinguish other variants of succinctness notions. It was shown in [3, 9] that a 1-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 [24], where their definition was based on ciphertext length. 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 [24] FE scheme is still ciphertext-succinct with our definition.

Furthermore, 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 secure when $|I|$ is any polynomial in κ. When we also allow sub-linear dependence on $|I|$, the FE scheme is called *weakly collusion-resistant*.

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. In the following we give its definition and some conventions in using it.

[2] Also known as *fully (circuit) succinct* in [9].
[3] Also known as *weakly (circuit) succinct* in [9].

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 of stretch* $\ell(\kappa)$ *if for all* $x \in R$ *we have* $|\mathsf{G}(x)| = \ell(|x|)$*, where* $\ell(\kappa)$ *is a polynominal in* κ*, and 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 the advantage of* $\mathcal{A}$ *in breaking the PRG* G*.*

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. The assumption is that $\mathsf{G}(r)$ is computationally indistinguishable from a random string of length $|\mathsf{G}(r)|$, i.e., G is μ-secure for some negligible function $\mu(\kappa)$.

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 $\mathsf{FE}_b = (\mathsf{PKey}_b, \mathsf{Enc}_b, \mathsf{Dec}_b, \mathsf{Fun}_b)$ for $b \in \{0, 1\}$. Then, $\mathsf{FE} = (\mathsf{PKey}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Fun})$ is defined as

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

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 of the scheme is easily verified by substitution. 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, which can be easily adapted to the other models.

Theorem 2 (SUM construction)**.** *If* FE_i *for* $i \in \{0, 1\}$ *is a succinct* (q_i, ϵ_i)*-non-adaptively secure FE scheme for functions in the class* F*, with public key size* ℓ_i^k *and ciphertext length* ℓ_i^c*, and if* G *is a* μ*-secure pseudorandom generator, then* $\mathsf{FE} = \mathsf{FE}_0 + \mathsf{FE}_1$ *is a succinct* $(q_0 + q_1, \epsilon_0 + \epsilon_1 + 4\mu)$*-non-adaptively secure FE scheme for* F *with public-key size* $\ell_0^k + \ell_1^k$ *and ciphertext length* $\ell_0^c + \ell_1^c$*.*

Moreover, if the algorithms $\mathsf{PKey}_i, \mathsf{Dec}_i, \mathsf{Fun}_i$ *and* Enc_i *of* FE_i *run in time* t_i^{PKey}*,*t_i^{Dec}*,*t_i^{Fun} *and* $t_i^{\mathsf{Enc}}(n, \kappa, d_i)$*, respectively, where* d_i *is the maximum depth of functions in* F*, and if* G *runs in time* t^{G}*, then the running times of the algorithms in* $\mathsf{FE} = \mathsf{FE}_0 + \mathsf{FE}_1$ *are:*

- PKey*:* $t_0^{\mathsf{PKey}} + t_1^{\mathsf{PKey}} + t^{\mathsf{G}}$
- Enc *:* $t_0^{\mathsf{Enc}} + t_1^{\mathsf{Enc}} + t^{\mathsf{G}}$
- Dec *:* $\max\{t_0^{\mathsf{Dec}}, t_1^{\mathsf{Dec}}\}$

- Fun : $\max\{t_0^{\mathsf{Fun}}, t_1^{\mathsf{Fun}}\} + t^{\mathsf{G}}$

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 $\mathcal{H}_b^{(j)}((m_0, m_1), \{f_{(h,i)}\}_{(h,i)\in I}) = (pk, c, \{fk^{(h,i)}\}_{(h,i)\in I})$, where $I = I_0 + I_1$.

$\mathcal{H}_b^{(0)}$: 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 $\mathcal{H}_b^{(0)}$:

$$
\begin{aligned}
&\mathcal{H}_b^{(0)}((m_0, m_1), \{f_{(h,i)}\}) = (pk, c, \{fk^{(h,i)}\})\\
&\quad \textbf{where } sk \leftarrow R,\ r \leftarrow R\\
&\qquad\qquad sk_0 = \mathsf{G}_0(sk),\ sk_1 = \mathsf{G}_1(sk)\\
&\qquad\qquad pk_0 = \mathsf{PKey}_0(sk_0),\ pk_1 = \mathsf{PKey}_1(sk_1),\ pk = (pk_0, pk_1)\\
&\qquad\qquad c_0 = \mathsf{Enc}_0(pk_0, m_b; \mathsf{G}_0(r)),\ c_1 = \mathsf{Enc}_1(pk_1, m_b; \mathsf{G}_1(r)),\ c = (c_0, c_1)\\
&\qquad\qquad \text{For all } (h,i) \in I_0 + I_1:\\
&\qquad\qquad\quad fk^{(h,i)} = \textbf{if } (f_{(h,i)}(m_0) = f_{(h,i)}(m_1) \neq \perp) \textbf{ then } (h, \mathsf{Fun}_h(sk_h, f; i))
\end{aligned}
$$

$\mathcal{H}_b^{(1)}$: 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.

$$
\begin{aligned}
&\mathcal{H}_b^{(1)}((m_0, m_1), \{f_{(h,i)}\}) = (pk, c, \{fk^{(h,i)}\})\\
&\quad \textbf{where } sk_0 \leftarrow R,\ sk_1 \leftarrow R,\ r_0, r_1 \leftarrow R\\
&\qquad\qquad pk_0 = \mathsf{PKey}_0(sk_0),\ pk_1 = \mathsf{PKey}_1(sk_1),\ pk = (pk_0, pk_1)\\
&\qquad\qquad c_0 = \mathsf{Enc}_0(pk_0, m_b; r_0),\ c_1 = \mathsf{Enc}_1(pk_1, m_b; r_1),\ c = (c_0, c_1)\\
&\qquad\qquad \text{For all } (h,i) \in I_0 + I_1:\\
&\qquad\qquad\quad fk^{(h,i)} = \textbf{if } (f_{(h,i)}(m_0) = f_{(h,i)}(m_1) \neq \perp) \textbf{ then } (h, \mathsf{Fun}_h(sk_h, f; i))
\end{aligned}
$$

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:

$$
\begin{aligned}
&\mathcal{H}_b^{(2)}((m_0, m_1), \{f_{(h,i)}\}) = (pk, c, \{fk^{(h,i)}\})\\
&\quad \textbf{where } sk_0 \leftarrow R,\ sk_1 \leftarrow R,\ r_0, r_1 \leftarrow R,\\
&\qquad\qquad pk_0 = \mathsf{PKey}_0(sk_0),\ pk_1 = \mathsf{PKey}_1(sk_1),\ pk = (pk_0, pk_1)\\
&\qquad\qquad c_0 = \mathsf{Enc}_0(pk_0, m_0; r_0),\ c_1 = \mathsf{Enc}_1(pk_1, m_1; r_1),\ c = (c_0, c_1)\\
&\qquad\qquad \text{For all } (h,i) \in I_0 + I_1:\\
&\qquad\qquad\quad fk^{(h,i)} = \textbf{if } (f_{(h,i)}(m_0) = f_{(h,i)}(m_1) \neq \perp) \textbf{ then } (h, \mathsf{Fun}_h(sk_h, f; i))
\end{aligned}
$$

Lemma 2. *If FE_1 is a (q_1, ϵ_1)-non-adaptively secure FE scheme, 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. *f FE_0 is a (q_0, ϵ_0)-non-adaptively secure FE scheme, 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 $\mathsf{Adv}_{\mathsf{FE}}[\mathcal{A}] = |\Pr\{[\mathcal{H}_0^{(0)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_1^{(0)}, \mathcal{A}] = 1\}| \leq 2\mu + \epsilon_1 + 0 + \epsilon_0 + 2\mu = 4\mu + \epsilon_0 + \epsilon_1$. $\square$ $\square$

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 $\mathsf{FE}_b = (\mathsf{PKey}_b, \mathsf{Enc}_b, \mathsf{Dec}_b, \mathsf{Fun}_b)$ 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 $pk \in K$:

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

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

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

$$\begin{aligned} sk_0 &= \mathsf{G}_0(sk) \\ sk_1^i &= \mathsf{G}_i(\mathsf{G}_1(sk)) \\ pk_1^i &= \mathsf{PKey}_1(sk_1^i) \\ c_i &= \mathsf{G}_i(\mathsf{G}_2(sk)) \\ fk_0^i &= \mathsf{Fun}_0(sk_0, e_i[c_i, pk_1^i], i) \\ fk_1^{i,j} &= \mathsf{Fun}_1(sk_1^i, f, j) \end{aligned}$$

The re-encryption function can work in 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 input; in the special mode where m is not given (denoted by the special symbol $\perp$), it pads a hard-wired ciphertext c with pseudo-randomness derived from the random seed from input. Note that the special mode is never invoked in a real world execution of the scheme, but it is only used in security proofs.

Let $\mathbb{RE}_{\mathsf{FE}}$ be the class of functions that include $e_i[c_i, pk_1^i](\cdot, \cdot)$ defined using Enc of the FE scheme FE. Then we state the security of our PRODUCT construction as follows. Again, the analysis can be easily adapted to other (e.g., selective/adaptive) models.

Theorem 3 (PRODUCT construction)**.** *Assume FE_0 and FE_1 are succinct public-key FE which are (q_0, ϵ_0)- and (q_1, ϵ_1)-non-adaptively secure for functions in the classes $\mathbb{RE}_{\mathsf{FE}_0}$ and F respectively, whose key sizes are ℓ_0^k and ℓ_1^k, ciphertext lengths $\ell_0^c(n, \kappa, d_0)$ 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. Also assume G is a μ-secure pseudorandom generator. Then $\mathsf{FE}_0 \times \mathsf{FE}_1$ is a $(q_0 q_1, \epsilon_1 + 2\epsilon_0 + 12\mu)$-non-adaptively secure succinct public-key FE scheme for F with public-key sizes $\ell_0^k + \ell_1^k$ and ciphertext length $\ell_0^c(n + \kappa, \kappa, d_0)$.*

Moreover, for $i \in \{0, 1\}$, let t_i^{PKey},t_i^{Enc},t_i^{Dec},t_i^{Fun} be the running times of algorithms $\mathsf{PKey}_i, \mathsf{Enc}_i, \mathsf{Dec}_i, \mathsf{Fun}_i$ 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 G. Then the running times of FE are:

- PKey*:* $t_0^{\mathsf{PKey}} + t^{\mathsf{G}}$

- Enc : $t_1^{\mathsf{Enc}}(n+\kappa,\kappa,d_0)+t^{\mathsf{G}}$
- Dec : $t_0^{\mathsf{Dec}}+t_1^{\mathsf{Dec}}$
- Fun : $t_1^{\mathsf{PKey}}+t_0^{\mathsf{Fun}}+t_1^{\mathsf{Fun}}+3t^{\mathsf{G}}$

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. Let $I = I_0 \times I_1$. A hybrid is a monotone function $\mathcal{H}_b^{(h)}((m_0,m_1),\{f_i\}_{i\in I}) = (pk, c, \{fk^i\}_{i\in I})$. An adversary $\mathcal{A}$ wins the game against $\mathcal{H}_b^{(h)}$ if $b' = [\mathcal{H}_b^{(h)},\mathcal{A}] = 1$, and its advantage over $\mathcal{H}_b^{(h)}$ is $\mathsf{Adv}[\mathcal{A}]_b^{(h)} = \Pr\{[\mathcal{H}_b^{(h)},\mathcal{A}]=1\}$.

$\mathcal{H}_b^{(0)}$: 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 $\mathcal{H}_b^{(0)}$:

$$
\begin{array}{l}
\mathcal{H}_b^{(0)}((m_0,m_1),\{f_{(i,j)}\}_{(i,j)\in I}) = (pk, c, \{fk^{(i,j)}\}_{(i,j)\in I})\\
\quad \textbf{where } sk \leftarrow K,\ r \leftarrow R\\
\qquad\quad sk_0 = \mathsf{G}_0(sk),\ pk = \mathsf{PKey}_0(sk_0)\\
\qquad\quad c = \mathsf{Enc}_0(pk,(m_b,\mathsf{G}_0(r));\mathsf{G}_1(r))\\
\qquad\quad \text{For all } i \in I_0, j \in I_1:\\
\qquad\qquad fk^{i,j} = \textbf{if } (f_{i,j}(m_0) = f_{i,j}(m_1) \neq \bot) \textbf{ then } (fk_0^i, fk_1^{i,j})\\
\qquad\qquad\qquad \textbf{where } sk_1^i = \mathsf{G}_i(\mathsf{G}_1(sk)),\ pk_1^i = \mathsf{PKey}_1(sk_1^i),\ c_i = \mathsf{G}_i(\mathsf{G}_2(sk))\\
\qquad\qquad\qquad\qquad fk_0^i = \mathsf{Fun}_0(sk_0, e_i[c_i, pk_1^i], i)\\
\qquad\qquad\qquad\qquad fk_1^{i,j} = \mathsf{Fun}_1(sk_1^i, f_{i,j}, j)
\end{array}
$$

$\mathcal{H}_b^{(1)}$: 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.

$$
\begin{array}{l}
\mathcal{H}_b^{(1)}((m_0,m_1),\{f_{i,j}\}_{(i,j)\in I}) = (pk, c, \{fk^{i,j}\}_{(i,j)\in I})\\
\quad \textbf{where } sk_0 \leftarrow K,\ r \leftarrow R\\
\qquad\quad pk = \mathsf{PKey}_0(sk_0)\\
\qquad\quad r' \leftarrow K,\ r'' \leftarrow K,\ c = \mathsf{Enc}_0(pk,(m_b,r');r'')\\
\qquad\quad \text{For all } i \in I_0, j \in I_1:\\
\qquad\qquad fk^{i,j} = \textbf{if } (f_{i,j}(m_0) = f_{i,j}(m_1) \neq \bot) \textbf{ then } (fk_0^i, fk_1^{i,j})\\
\qquad\qquad\qquad \textbf{where } sk_1^i \leftarrow K,\ pk_1^i = \mathsf{PKey}_1(sk_1^i),\ c_i \leftarrow K\\
\qquad\qquad\qquad\qquad fk_0^i = \mathsf{Fun}_0(sk_0, e_i[c_i, pk_1^i], i)\\
\qquad\qquad\qquad\qquad fk_1^{i,j} = \mathsf{Fun}_1(sk_1^i, f_{i,j}, j)
\end{array}
$$

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 c_i is generated without changing its distribution.

$$
\begin{aligned}
&\mathcal{H}_b^{(1)}((m_0,m_1),\{f_{i,j}\}_{(i,j)\in I}) = (pk, c, \{fk^{i,j}\}_{(i,j)\in I})\\
&\quad \textbf{where } sk_0 \leftarrow K,\ r \leftarrow R\\
&\qquad\qquad pk = \mathsf{PKey}_0(sk_0)\\
&\qquad\qquad r' \leftarrow K,\ r'' \leftarrow K,\ c = \mathsf{Enc}_0(pk,(m_b,r');r'')\\
&\qquad\qquad \text{For all } i\in I_0, j\in I_1:\\
&\qquad\qquad\quad fk^{i,j} = \textbf{if } (f_{i,j}(m_0) = f_{i,j}(m_1) \neq \bot) \textbf{ then } (fk_0^i, fk_1^{i,j})\\
&\qquad\qquad\qquad\qquad \textbf{where } sk_1^i \leftarrow K,\ pk_1^i = \mathsf{PKey}_1(sk_1^i)\\
&\qquad\qquad\qquad\qquad\qquad s_i \leftarrow K,\ \tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r')),\ c_i = s_i \oplus \tilde{c}_1^i\\
&\qquad\qquad\qquad\qquad\qquad fk_0^i = \mathsf{Fun}_0(sk_0, e_i[c_i, pk_1^i], i)\\
&\qquad\qquad\qquad\qquad\qquad fk_1^{i,j} = \mathsf{Fun}_1(sk_1^i, f_{i,j}, j)
\end{aligned}
$$

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.

$$
\begin{aligned}
&\mathcal{H}_b^{(3)}((m_0,m_1),\{f_{i,j}\}_{(i,j)\in I}) = (pk, c, \{fk^{i,j}\}_{(i,j)\in I})\\
&\quad \textbf{where } sk_0 \leftarrow K,\ r \leftarrow R,\ s \leftarrow K\\
&\qquad\qquad pk = \mathsf{PKey}_0(sk_0)\\
&\qquad\qquad r' \leftarrow K,\ r'' \leftarrow K,\ c = \mathsf{Enc}(pk, m_b; r) = \mathsf{Enc}_0(pk,(m_b,r');r'')\\
&\qquad\qquad \text{For all } i\in I_0, j\in I_1:\\
&\qquad\qquad\quad fk^{i,j} = \textbf{if } (f_{i,j}(m_0) = f_{i,j}(m_1) \neq \bot) \textbf{ then } (fk_0^i, fk_1^{i,j})\\
&\qquad\qquad\qquad\qquad \textbf{where } sk_1^i \leftarrow K,\ pk_1^i = \mathsf{PKey}_1(sk_1^i)\\
&\qquad\qquad\qquad\qquad\qquad s_i = \mathsf{G}_i(s),\ \tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r')),\ c_i = s_i \oplus \tilde{c}_1^i\\
&\qquad\qquad\qquad\qquad\qquad fk_0^i = \mathsf{Fun}_0(sk_0, e_i[c_i, pk_1^i], i)\\
&\qquad\qquad\qquad\qquad\qquad fk_1^{i,j} = \mathsf{Fun}_1(sk_1^i, f_{i,j}, j)
\end{aligned}
$$

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).

$$
\begin{aligned}
&\mathcal{H}_b^{(4)}((m_0,m_1),\{f_{i,j}\}_{(i,j)\in I}) = (pk, c, \{fk^{i,j}\}_{(i,j)\in I})\\
&\quad \textbf{where } sk_0 \leftarrow K,\ r \leftarrow R,\ s \leftarrow K\\
&\qquad\qquad pk = \mathsf{PKey}_0(sk_0)\\
&\qquad\qquad r' \leftarrow K,\ r'' \leftarrow K,\ c = \mathsf{Enc}_0(pk,(\bot,s);r'')\\
&\qquad\qquad \text{For all } i\in I_0, j\in I_1:\\
&\qquad\qquad\quad fk^{i,j} = \textbf{if } (f_{i,j}(m_0) = f_{i,j}(m_1) \neq \bot) \textbf{ then } (fk_0^i, fk_1^{i,j})\\
&\qquad\qquad\qquad\qquad \textbf{where } sk_1^i \leftarrow K,\ pk_1^i = \mathsf{PKey}_1(sk_1^i)\\
&\qquad\qquad\qquad\qquad\qquad s_i = \mathsf{G}_i(s),\ \tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r')),\ c_i = s_i \oplus \tilde{c}_1^i\\
&\qquad\qquad\qquad\qquad\qquad fk_0^i = \mathsf{Fun}_0(sk_0, e_i[c_i, pk_1^i], i)\\
&\qquad\qquad\qquad\qquad\qquad fk_1^{i,j} = \mathsf{Fun}_1(sk_1^i, f_{i,j}, j)
\end{aligned}
$$

Lemma 7. *If FE_0 is a (q_0, ϵ_0)-non-adaptive secure FE scheme for functions in the class $\mathbb{RE}_{\mathsf{FE}_0}$, then for any $b \in \{0,1\}$ and any efficient adversary $\mathcal{A}$, we have $|\mathsf{Adv}[\mathcal{A}]_b^{(3)} - \mathsf{Adv}[\mathcal{A}]_b^{(4)}| \leq \epsilon_0(\kappa)$.*

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

$$
\begin{aligned}
&\mathcal{H}_b^{(5)}((m_0, m_1), \{f_{i,j}\}_{(i,j)\in I}) = (pk, c, \{fk^{i,j}\}_{(i,j)\in I}) \\
&\quad \textbf{where } sk_0 \leftarrow K,\ r \leftarrow R,\ s \leftarrow K \\
&\qquad\qquad pk = \mathsf{PKey}_0(sk_0) \\
&\qquad\qquad r'' \leftarrow K,\ c = \mathsf{Enc}_0(pk, (\bot, s); r'') \\
&\qquad\qquad \text{For all } f_{i,j} \textbf{ where } i \in I_0, j \in I_1\text{:} \\
&\qquad\qquad\quad fk^{i,j} = \textbf{if } (f_{i,j}(m_0) = f_{i,j}(m_1) \neq \bot) \textbf{ then } (fk_0^i, fk_1^{i,j}) \\
&\qquad\qquad\qquad\qquad \textbf{where } sk_1^i \leftarrow K \\
&\qquad\qquad\qquad\qquad\qquad pk_1^i = \mathsf{PKey}_1(sk_1^i) \\
&\qquad\qquad\qquad\qquad\qquad s_i = \mathsf{G}_i(s),\ r_i \leftarrow K,\ \tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; r_i),\ c_i = s_i \oplus \tilde{c}_1^i \\
&\qquad\qquad\qquad\qquad\qquad fk_0^i = \mathsf{Fun}_0(sk_0, e_i[c_i, pk_1^i], i) \\
&\qquad\qquad\qquad\qquad\qquad fk_1^{i,j} = \mathsf{Fun}_1(sk_1^i, f_{i,j}, j)
\end{aligned}
$$

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

Lemma 9. *If FE_1 is a (q_1, ϵ_1)-non-adaptive secure FE scheme, then for any efficient adversary $\mathcal{A}$ we have* $|\mathsf{Adv}[\mathcal{A}]_0^{(5)} - \mathsf{Adv}[\mathcal{A}]_1^{(5)}| \leq \epsilon_1(\kappa)$.

Finally, by applying previous lemmas, we see that the advantage of any adversary $\mathcal{A}$ to the PRODUCT scheme FE can be bounded by $\mathsf{Adv}_{\mathsf{FE}}[\mathcal{A}] = |\Pr\{[\mathcal{H}_0^{(0)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_1^{(0)}, \mathcal{A}] = 1\}| \leq 2(4\mu + 0 + \mu + \epsilon_0 + \mu) + \epsilon_1 = \epsilon_1 + 2\epsilon_0 + 12\mu$. □ □

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 $(1, \epsilon_0)$-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^{PKey},t_0^{Enc},t_0^{Dec},t_0^{Fun} 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 (weakly) 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:

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

So FE_1 supports 2 functional queries, and for $p \geq 1$, the FE scheme FE_{p+1} supports 2^{2^p} functional queries. For any polynomial $Q(\kappa)$, when $p \geq \log\log Q$, the FE scheme FE_{p+1} supports $Q(\kappa)$ functional queries, and its security and performance can be characterized as follows.

Security : The advantage of FE_{p+1} over any efficient adversary $\mathcal{A}$ is

$$\mathsf{Adv}_{\mathsf{FE}_{p+1}}[\mathcal{A}] = 2 \cdot 3^p \epsilon_0 + 8 \cdot 3^p \mu = \log Q \cdot \epsilon_0 + \log Q \cdot \mu.$$

Running times and output lengths :

- PKey: $2t_0^{\mathsf{PKey}} + (p+1)t^{\mathsf{G}} = 2t_0^{\mathsf{PKey}} + \log\log Q \cdot t^{\mathsf{G}}$
- Enc : $2t_0^{\mathsf{Enc}}(n + p\kappa, \kappa, d) + (p+1)t^{\mathsf{G}} = 2t_0^{\mathsf{Enc}}(n + \kappa \log\log Q, \kappa, d) + \log\log Q \cdot t^{\mathsf{G}}$
- Dec : $2^p t_0^{\mathsf{Dec}} = \log Q \cdot t_0^{\mathsf{Dec}}$
- Fun : $2(2^p - 1)t_0^{\mathsf{PKey}} + 2^p t_0^{\mathsf{Fun}} + (\sum_{i=0}^{p}(p+2-i)2^i + 2^{p+1} - 1)t^{\mathsf{G}} = 2\log Q \cdot t_0^{\mathsf{PKey}} + \log Q \cdot t_0^{\mathsf{Fun}} + 6\log Q \cdot t^{\mathsf{G}}$
- $\ell_{p+1}^k = 2\ell_0^k$
- $\ell_{p+1}^c = \ell_0^c(n + p\kappa, \kappa, d) = \ell_0^c(n + \kappa\log\log Q, \kappa, d)$
- $\ell_{p+1}^{fk} = 2^p \ell_0^{fk} = \log Q \cdot \ell_0^{fk}$

Clearly FE_{p+1} is a secure FE scheme, and the transformation incurs only logarithmic (in terms of Q) security lost. 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 $\mathrm{poly}(\log Q, n, \kappa, d)$ for some fixed polynomial poly, and hence FE_{p+1} is weakly 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 Fun_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_p to derive FE_{p+1}, a functional key $fk = (fk_0, fk_1)$ for any function f consists of two keys under FE_p: fk_0 is for a "re-encryption" function $e_i^{(p)}[c, pk](\cdot,\cdot)$, and fk_1 is for f. Hence for the composition to go through, FE_p must be capable of generating functional keys for these two classes of functions, namely

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

Recall from Section 4 that $\mathbb{RE}_{\mathsf{FE}_p}$ is the class containing $e_i^{(p)}[c, pk]$ for all $c \in M, pk \in R$. Let $\mathbb{RE}_{\mathsf{FE}_0}^p = \cup_{h=1}^p \mathbb{RE}_{\mathsf{FE}_h}$. 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 $F_{p+1} \cup \mathbb{RE}_{\mathsf{FE}_0}^p$ and the PRG G.

Theorem 4. *Fix any polynomial $Q(\kappa)$, and let $p(\kappa) = \omega(\log\log Q(\kappa))$. Assume FE_0 is a succinct $(1, \epsilon_0)$-non-adaptive (or selective/adaptive) secure FE scheme for the function class F such that $\mathbb{RE}_{\mathsf{FE}_0}^p \subseteq F$ and $\mathsf{G} \in F$, where $\epsilon_0(\kappa)$ is some negligible function; and assume G is a secure PRG. Then FE_{p+1} defined in Equation 1 is a weakly succinct (Q, ϵ)-non-adaptive (or selective/adaptive, respectively) secure FE scheme for F, where $\epsilon(\kappa)$ is some negligible function.*

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 that only requires the base scheme is capable of functionals keys for its own encryption function.

The *iterated linear composition* is defined recursively by

$$\mathsf{FE}_1 = \mathsf{FE}_0 + \mathsf{FE}_0, \text{and for } p \geq 1, \mathsf{FE}_{p+1} = \mathsf{FE}_1 \times \mathsf{FE}_p. \tag{2}$$

Under this composition, FE_1 supports 2 functional keys, and for $p \geq 1$, FE_{p+1} supports 2^{p+1} functional keys. For FE_p to achieve $Q(\kappa)$ functional keys, we need $p \geq \log Q$. Then it is straightforward to get the following characteristics of FE_p:

Security : The advantage of FE_p over any efficient adversary is

$$\mathsf{Adv}_{\mathsf{FE}_p}[\mathcal{A}] = (2^p + 2)\epsilon_0 + (2^{p+2} + 2)\mu = Q\epsilon_0 + Q\mu.$$

Running times and output lengths :

- PKey: $2t_0^{\mathsf{PKey}} + 2t^{\mathsf{G}}$
- Enc : $2t_0^{\mathsf{Enc}}(n + \kappa, \kappa, d) + 2t^{\mathsf{G}}$
- Dec : $pt_0^{\mathsf{Dec}} = \log Q \cdot t_0^{\mathsf{Dec}}$
- Fun : $pt_0^{\mathsf{Fun}} + 2(p-1)t_0^{\mathsf{PKey}} + (6p-5)t^{\mathsf{G}} = 2\log Q \cdot t_0^{\mathsf{PKey}} + \log Q \cdot t_0^{\mathsf{Fun}} + 6\log Q \cdot t^{\mathsf{G}}$
- $\ell_p^k = 2\ell_0^k$
- $\ell_p^c = \ell_0^c(n + \kappa, \kappa, d) = 2\ell_0^c(n + \kappa, \kappa, d)$
- $\ell_p^{fk} = p\ell_0^{fk} = \log Q \cdot \ell_0^{fk}$

The FE scheme FE_p is also secure, and this transformation incurs linear (in terms of Q) security lost. 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 5. *Assume FE_0 is a succinct $(1, \epsilon_0)$-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 $pk \in R$, where $\epsilon_0(\kappa)$ is some negligible function, and assume G is a secure PRG. Then, for any polynomial $Q(\kappa)$, the FE scheme FE_p defined in Equation 2 for $p = \omega(\log Q)$ is a succinct (Q, ϵ)-non-adaptive (or selective/adaptive, respectively) secure FE scheme for F, for some negligible function $\epsilon(\kappa)$.*

Comparing with the iterated squaring composition to support Q functional key queries, one can see that the running times and key lengths of PKey and Fun are about the same, and the iterated linear composition gives better encryption performance: Enc runs slightly faster and the ciphertext is shorter, as they are independent of Q. The trade-off is that the security loss is worse with our linear composition: Although the advantage against any efficient adversary is still negligible with linear composition, it grows linearly in Q rather than in $\log Q$ as achieved with the iterated squaring composition.

5.3 On the implications of our reductions

So far we have obtained two transformations from a 1-secure succinct FE scheme to a (weakly) succinct FE scheme that supports polynomially many functional key queries. In this subsection we explore the implications of our reductions.

A (Q, ϵ)-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^{Enc}_{p+1} of FE_{p+1} obtained by our transformations on FE_0:

1. If FE_{p+1} is as in the iterated squaring composition, then $p = \omega(\log\log Q)$ and $t^{\mathsf{Enc}}_{p+1} = t^{\mathsf{Enc}}_0(n + \kappa \cdot \log\log Q, \kappa, d) + \log\log Q \cdot t^{\mathsf{G}}(\kappa)$. Clearly t^{Enc}_{p+1} is sub-linear in Q, and thus FE_{p+1} is weakly collusion-succinct.

2. If FE_{p+1} is as in the iterated linear composition, then $p = \omega(\log Q)$ and $t^{\mathsf{Enc}}_{p+1} = 2t^{\mathsf{Enc}}_0(n+\kappa, \kappa, d)+2t^{\mathsf{G}}(\kappa)$, which is independent of Q. So FE_{p+1} is succinct (hence collusion-succinct).

Bitansky and Vaikuntanathan [9] described a reduction from any (weakly) compact Q-secure FE scheme to a (weakly) collusion-succinct Q-secure FE scheme for the same class of functions. We note that, although in [9] 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 6. *For any polynomial $Q(\kappa)$:*

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

2. *If there exists a succinct 1-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 FE scheme for F supporting $Q(\kappa)$ functional key queries.*

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 [9, 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 1-secure FE scheme. Based on sub-exponential lattice assumption, Goldwasser et al. [24] showed that, for any polynomial $d(n)$, there exists a succinct 1-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.

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

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

[11] 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)

[12] 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)

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

[14] 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)

[15] 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)

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

[17] 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)

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

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

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

[21] 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)

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

[23] 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)

[24] 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)

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

[26] 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)

[27] 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)

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

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

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

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

[32] 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

Suppose $\mathsf{G} : R \to S$ is a $\mu(\kappa)$-secure PRG. Recall the following two 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* $|\Pr\{[\mathcal{H}_b^{(0)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_b^{(1)}, \mathcal{A}] = 1\}| \le 2\mu(\kappa)$.

Proof of Lemma 1. 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.

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

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 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 $|\mathsf{Adv}[\mathcal{A}]_b^{(0)} - \mathsf{Adv}[\mathcal{A}]_b^{(1)}| = \mathsf{Adv}_{\mathsf{G}}[\mathcal{B}^{\mathcal{A}}] \leq 2\mu(\kappa)$. □ □

Lemma 2. *If FE_1 is a (q_1, ϵ_1)-non-adaptively secure FE scheme, then for any adversary $\mathcal{A}$ we have $|\Pr\{[\mathcal{H}_0^{(1)}, \mathcal{A}] = 1\} - \Pr\{[\mathcal{H}_0^{(2)}, \mathcal{A}] = 1\}| \ \leq \ \epsilon_1(\kappa)$.*

Proof of Lemma 2. We define the following adversary $\mathcal{B}$ using $\mathcal{A}$ as an oracle to attack the FE scheme FE_1.

$$\begin{aligned}&\mathcal{B}(pk_1, c_1, \{fk_1^{(1,i)}\}_{i \in I_1}) = ((m_0, m_1), \{f_{(1,i)}\}_{i \in I_1}) \\ &\quad \text{where } (pk, c, \{fk^{(h,i)}\}_{(h,i) \in I}) = \mathcal{H}_0^{(2)}[pk_1, c_1, \{fk_1^{(1,i)}\}_{(h,i) \in I_1}]((m_0, m_1), \{f_{(h,i)}\}_{(h,i) \in I}) \\ &\qquad\quad ((m_0, m_1), \{f_{(h,i)}\}_{(h,i) \in I}, b') = \mathcal{A}(pk, c, \{fk^{(h,i)}\}_{(h,i) \in I})\end{aligned}$$

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 $i \in I_1$; 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 $\mathsf{Adv}_{\mathsf{FE}_1}[\mathcal{B}] = |\mathsf{Adv}[\mathcal{A}]_0^{(1)} - \mathsf{Adv}[\mathcal{A}]_0^{(2)}| \leq \epsilon_1(\kappa)$. □ □

Next we prove lemmas in Section 4 that are used to establish security of the PRODUCT constructions. From now on, hybrids refer to those defined in Section 4.

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 $|\mathsf{Adv}[\mathcal{A}]_b^{(0)} - \mathsf{Adv}[\mathcal{A}]_b^{(1)}| \leq 4\mu(\kappa)$.*

Proof of Lemma 4. We build an adversary $\mathcal{B}$ using $\mathcal{A}$ as an oracle to attack the PRG G. Similar to previous proofs, by $\mathcal{H}_b^{(1)}[sk_0, r', r'', sk_1^1, \ldots, sk_1^{q_0}, c_1, \ldots, c_{q_0}]$ we mean the hybrid obtained by substituting $sk_0, r', r'', sk_1^1, \ldots, sk_1^{q_0}, c_1, \ldots, c_{q_0}$ with the given values. The adversary $\mathcal{B}$ is defined as:

$$\begin{aligned}&\mathcal{B}(x) = b' \\ &\quad \textbf{where } sk_0 \| r' \| r'' \| sk_1^1 \| \cdots \| sk_1^{q_0} \| c_1 \| \cdots \| c_{q_0} = x \\ &\qquad\quad (pk, c, \{fk^{i,j}\}_{(i,j) \in I}) = \mathcal{H}_b^{(1)}[sk_0, r', r'', sk_1^1, \ldots, sk_1^{q_0}, c_1, \ldots, c_{q_0}]((m_0, m_1), \{f_{i,j}\}_{(i,j) \in I}) \\ &\qquad\quad ((m_0, m_1), \{f_{i,j}\}_{(i,j) \in I}, b') = \mathcal{A}(pk, c, \{fk^{i,j}\}_{(i,j) \in I})\end{aligned}$$

Notice that if x is generated by four calls to G then $\mathcal{B}$ is running $(\mathcal{H}_b^{(0)} \mid \mathcal{A})$, and if x is truly random then $\mathcal{B}$ is running $(\mathcal{H}_b^{(1)} \mid \mathcal{A})$. Since G is a μ-secure pseudorandom generator, we have

$$|\mathsf{Adv}[\mathcal{A}]_b^{(0)} - \mathsf{Adv}[\mathcal{A}]_b^{(1)}| = |\Pr\{\mathcal{B}(x) = 1 \mid x = \mathsf{G}(y) \text{ for some } y\} - \Pr\{\mathcal{B}(x) = 1 \mid x \leftarrow R\}| \leq 4\mu(\kappa).$$

□ □

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 of Lemma 5. Since c_i is sampled uniformly 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 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* $|\mathsf{Adv}[\mathcal{A}]_b^{(2)} - \mathsf{Adv}[\mathcal{A}]_b^{(3)}| \le \mu(\kappa)$.

Proof of Lemma 6. We build an adversary $\mathcal{B}$ using $\mathcal{A}$ as an oracle to attack G:

$$
\begin{aligned}
&\mathcal{B}(x) = b' \\
&\quad \textbf{where } s_1 \| \cdots \| s_{q_0} = x \\
&\qquad\quad (pk, c, \{fk^{i,j}\}_{(i,j)\in I}) = \mathcal{H}_b^{(1)}[s_1, \ldots, s_{q_0}]((m_0, m_1), \{f_{i,j}\}_{(i,j)\in I}) \\
&\qquad\quad ((m_0, m_1), \{f_{i,j}\}_{(i,j)\in I}, b') = \mathcal{A}(pk, c, \{fk^{i,j}\}_{(i,j)\in I})
\end{aligned}
$$

Notice that if x is chosen uniformly random then $\mathcal{B}$ is running $(\mathcal{H}_b^{(2)} \mid \mathcal{A})$, and if x is generated by G then $\mathcal{B}$ is running $(\mathcal{H}_b^{(3)} \mid \mathcal{A})$. Thus we have

$$
|\Pr\{\mathcal{B}(x) = 1 \mid x \leftarrow K\} - \Pr\{\mathcal{B}(x) = 1 \mid x = \mathsf{G}_i(s) \text{ for some } s \in K\}| = |\mathsf{Adv}[\mathcal{A}]_b^{(2)} - \mathsf{Adv}[\mathcal{A}]_b^{(3)}|.
$$

Since G is μ-secure, we get $|\mathsf{Adv}[\mathcal{A}]_b^{(2)} - \mathsf{Adv}[\mathcal{A}]_b^{(3)}| \le \mu(\kappa)$. □ □

Lemma 7. *If* FE_0 *is a* (q_0, ϵ_0)*-non-adaptive secure FE scheme for functions in the class* $\mathbb{RE}_{\mathsf{FE}_0}$*, then for any* $b \in \{0,1\}$ *and any efficient adversary* $\mathcal{A}$*, we have* $|\mathsf{Adv}[\mathcal{A}]_b^{(3)} - \mathsf{Adv}[\mathcal{A}]_b^{(4)}| \le \epsilon_0(\kappa)$.

Proof of Lemma 7. 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:

$$
\begin{aligned}
&\mathcal{B}(pk_0, c_0, \{fk_0^i\}_{i\in I_0}) = ((x_0, x_1), \{e_i[c_i, pk_1^i]\}_{i\in I_0}, b') \\
&\quad \textbf{where } pk = pk_0,\ c = c_0 \\
&\qquad\qquad r \leftarrow R,\ r' \leftarrow K,\ s \leftarrow K \\
&\qquad\qquad x_0 = (m_b, r'),\ x_1 = (\perp, s) \\
&\qquad\qquad \text{For all } i \in I_0, j \in I_1: \\
&\qquad\qquad\quad fk^{i,j} = \textbf{if } (f_{i,j}(m_0) = f_{i,j}(m_1) \ne \perp) \textbf{ then } (fk_0^i, fk_1^{i,j}) \\
&\qquad\qquad\qquad\quad \textbf{where } sk_1^i \leftarrow K \\
&\qquad\qquad\qquad\qquad\quad pk_1^i = \mathsf{PKey}_1(sk_1^i) \\
&\qquad\qquad\qquad\qquad\quad s_i = \mathsf{G}_i(s),\ \tilde{c}_1^i = \mathsf{Enc}_1(pk_1^i, m_b; \mathsf{G}_i(r')),\ c_i = s_i \oplus \tilde{c}_1^i \\
&\qquad\qquad\qquad\qquad\quad fk_1^{i,j} = \mathsf{Fun}_1(sk_1^i, f_{i,j}, j) \\
&\qquad\qquad ((m_0, m_1), \{f_{i,j}\}_{(i,j)\in I}, b') = \mathcal{A}(pk, c, \{fk^{i,j}\}_{(i,j)\in I})
\end{aligned}
$$

We need to argue that $\mathcal{B}$ is a valid adversary for the FE game, that is, the functions $e_i[c_i, pk_1^i]$ 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 $i \in I_0$. Since $x_0 = (m_b, r')$ and $x_1 = (\perp, s)$, by definition of $e_i[c_i, pk_1^i]$ we have

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

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 $c_0 = \mathsf{Enc}_0(pk_0, (\perp, s); r'')$ 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 (q_0, ϵ_0)-non-adaptively secure, we have that $|\mathsf{Adv}[\mathcal{A}]_b^{(3)} - \mathsf{Adv}[\mathcal{A}]_b^{(4)}| \le \epsilon_0(\kappa)$. □ □

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 of Lemma 8. We build an adversary $\mathcal{B}$ to attack G using $\mathcal{A}$ as an oracle.

$$\begin{aligned}
&\mathcal{B}(x) = b' \\
&\quad \textbf{where } r_1 \| \cdots \| r_{q_0} = x \\
&\qquad\qquad (pk, c, \{fk^{i,j}\}_{(i,j)\in I}) = \mathcal{H}_b^{(1)}[r_1, \ldots, r_{q_0}]((m_0, m_1), \{f_{i,j}\}_{(i,j)\in I}) \\
&\qquad\qquad ((m_0, m_1), \{f_{i,j}\}_{(i,j)\in I}, b') = \mathcal{A}(pk, c, \{fk^{i,j}\}_{(i,j)\in I})
\end{aligned}$$

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

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

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

Lemma 9. *If* FE_1 *is a* (q_1, ϵ_1)*-non-adaptive secure FE scheme, then for any efficient adversary* $\mathcal{A}$ *we have* $|\mathsf{Adv}[\mathcal{A}]_0^{(5)} - \mathsf{Adv}[\mathcal{A}]_1^{(5)}| \leq \epsilon_1(\kappa)$.

Proof of Lemma 9. We build an adversary $\mathcal{B}$ to attack the FE scheme FE_1 using $\mathcal{A}$ as an oracle. Let $I_0 = \{0\}$ be a singleton set. Then $\mathcal{B}$ is defined as:

$$\begin{aligned}
&\mathcal{B}(pk_1, c_1, \{fk_1^{0,j}\}_{j\in I_1}) = ((m_0, m_1), \{f_{0,j}\}_{j\in I_1}, b') \\
&\quad \textbf{where } pk_1^0 = pk_1,\ \tilde{c}_1^0 = c_1 \\
&\qquad\qquad (pk, c, \{fk^{i,j}\}_{(i,j)\in I}) = \mathcal{H}_b^{(5)}[pk_1^0, \tilde{c}_1^0]((m_0, m_1), \{f_{i,j}\}_{(i,j)\in I}) \\
&\qquad\qquad ((m_0, m_1), \{f_{i,j}\}_{(i,j)\in I}, b') = \mathcal{A}(pk, c, \{fk^{i,j}\}_{(i,j)\in I})
\end{aligned}$$

Notice that if $c_1 = \mathsf{Enc}_1(pk_1, m_0; \tilde{r})$ for some randomness $\tilde{r}$ then $\mathcal{B}$ is running $(\mathcal{H}_0^{(5)} \mid \mathcal{A})$, and if $c_1 = \mathsf{Enc}_1(pk_1, m_1; \tilde{r})$ then $\mathcal{B}$ is running $(\mathcal{H}_1^{(5)} \mid \mathcal{A})$. So the advantage of $\mathcal{B}$ in winning the FE game for the FE_1 scheme is $|\mathsf{Adv}[\mathcal{A}]_0^{(5)} - \mathsf{Adv}[\mathcal{A}]_1^{(5)}|$. Since FE_1 is (q_1, ϵ_1)-non-adaptively secure, we have that $|\mathsf{Adv}[\mathcal{A}]_0^{(5)} - \mathsf{Adv}[\mathcal{A}]_1^{(5)}| \leq \epsilon_1(\kappa)$. □ □