On the Achievability of Simulation-Based Security for Functional Encryption

Angelo De Caro[1], Vincenzo Iovino[2], Abhishek Jain[3⋆], Adam O'Neill[4] ⋆⋆, Omer Paneth[4] ⋆⋆⋆, and Giuseppe Persiano[2]

[1] NTT Secure Platform Laboratories, Japan, Angelo.Decaro@lab.ntt.co.jp
[2] Dipartimento di Informatica, University of Salerno, Italy, {iovino,giuper}@dia.unisa.it,
[3] MIT and Boston University, abhishek@csail.mit.edu,
[4] Boston University, {amoneill,omer}@bu.edu

Abstract. This work attempts to clarify to what extent simulation-based security (SIM-security) is achievable for functional encryption (FE) and its relation to the weaker indistinguishability-based security (IND-security). Our main result is a compiler that transforms any FE scheme for the general circuit functionality (which we denote by Circuit-FE) meeting indistinguishability-based security (IND-security) to a Circuit-FE scheme meeting SIM-security, where:

- In the random oracle model, the resulting scheme is secure for an unbounded number of encryption and key queries, which is the strongest security level one can ask for.
- In the standard model, the resulting scheme is secure for a bounded number of encryption and non-adaptive key queries, but an *unbounded* number of adaptive key queries. This matches known impossibility results and improves upon Gorbunov et al. [CRYPTO'12] (which is secure for a *bounded* number of adaptive key queries).

Our compiler is inspired by the celebrated Fiat-Lapidot-Shamir paradigm [FOCS'90] for obtaining zero-knowledge proof systems from witness-indistinguishable proof systems. As it is currently unknown whether Circuit-FE meeting IND-security exists, the purpose of this result is to establish that it remains a good target for future research despite known deficiencies of IND-security [Boneh et al. – TCC'11, O'Neill – ePrint '10]. We also give a tailored construction of SIM-secure hidden vector encryption (HVE) in composite-order bilinear groups. Finally, we revisit the known negative results for SIM-secure FE, extending them to natural weakenings of the security definition and thus providing essentially a full picture of the (in)achievability of SIM-secure FE.

Keywords. Functional Encryption, Hidden Vector Encryption, Simulation-Based Security.

⋆ Supported by NSF Contract CCF-1018064 and DARPA Contract Number: FA8750-11-2-0225. The author also thanks BU RISCS (Reliable Information Systems and Cyber Security) Institute.

⋆⋆ Supported by NSF grants CNS-1012910 and CNS-0546614. The author also thanks BU RISCS (Reliable Information Systems and Cyber Security) Institute.

⋆⋆⋆ Supported by the Simons award for graduate students in theoretical computer science and NSF award 1218461.

1 Introduction

Let $F : K \times M \to \Sigma$ be a functionality, where K is the *key space* and M is the *message space* and Σ is the *output space*. Then a *functional encryption scheme for* F (or F-FE scheme) [7] is a special encryption scheme in which, for every *key* $k \in K$, the owner of the master secret key Msk associated with the public key Pk can generate a special key or "token" Tok_k that allows the computation of $F(k, m)$ from a ciphertext of m computed under public key Pk. In other words, whereas in traditional encryption schemes decryption is an all-or-nothing affair, in FE it is possible to finely control the amount of information that is revealed by a ciphertext. This opens up exciting applications to access control, searching on encrypted data, and secure delegation of computation (cf. [21]), among others.

Unlike in the case of classical cryptosystems, a general study of the security of FE did not appear initially. Instead, progressively more expressive forms of FE were constructed in a series of works (see, e.g., [6, 8, 16, 17, 19, 22]) that adopted indistinguishability-based (IND) notions of security. The study of simulation-based (SIM) notions of security for FE were initiated only comparatively recently by Boneh, Sahai, and Waters [7] and O'Neill [20].[5] In particular, they show there exist clearly insecure FE schemes for certain functionalities that are nonetheless deemed secure by IND-security, whereas these schemes do not meet the stronger notion of SIM-security. On the other hand, negative results have also emerged showing SIM-security is not always achievable [7, 3, 1]. This leads to the main questions that we study in this work:

> To what extent is SIM-security for FE achievable? In particular, can schemes for IND-secure FE be "compiled" to ones meeting the stronger notion of SIM-security?

In order to make these questions more precise, let us call an F-FE scheme (q_1, ℓ, q_2)-SIM-secure (resp. (q_1, ℓ, q_2)-IND-secure) if it is secure under the respective security definition for adversaries making at most q_1 "non-adaptive" key queries (i.e., before seeing the challenge ciphertexts), q_2 "adaptive" key queries (i.e., after seeing the challenge ciphertexts), and at most ℓ encryption queries (i.e., the number of challenge ciphertexts). Note

[5] Very roughly, in both definitions the adversary makes key-derivation queries, then queries for challenge ciphertexts, then again makes key-derivation queries. IND-security asks that the adversary cannot distinguish between encryptions of messages that it cannot trivially distinguish using the keys. SIM-security asks that the "view" of the adversary can be simulated by simulator given neither ciphertexts nor keys but only the corresponding outputs of the functionality on the underlying plaintexts.

that these bounds are fixed *a priori* and do not vary per adversary. In the case that a parameter is unbounded we denote it by poly, so for example $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-SIM security means SIM-security where the number of encryption and key queries are all unbounded. Of particular interest is $F = \mathsf{Circuit}$, meaning the general circuit functionality.

A Compiler for General Functionalities. Our main result is a compiler that takes a IND-secure $\mathsf{Circuit}$-FE scheme and produces a SIM-secure $\mathsf{Circuit}$-FE scheme. More specifically, in the random oracle (RO) model [4], we show the existence of a $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-secure $\mathsf{Circuit}$-FE scheme implies the existence of a $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-SIM-secure $\mathsf{Circuit}$-FE scheme. In the standard (random oracle devoid) model, we show the existence of a $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-secure[6] $\mathsf{Circuit}$-FE scheme implies the existence of a $(q_1, \ell, \mathsf{poly})$-SIM-secure $\mathsf{Circuit}$-FE scheme for any polynomials q_1, ℓ. The result in the standard model is optimal in that it matches recent impossibility results [7, 1, 3] discussed later.

We note that it is currently a central open question in FE to construct a $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-secure $\mathsf{Circuit}$-FE scheme.[7] If such a scheme is achieved, we will obtain interesting new results via our compiler. To compare, Boneh et al. [7] achieve $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-SIM-secure *identity-based encryption* (IBE) in the RO model; in fact, they explicitly raise the open question of constructing SIM-secure $\mathsf{Circuit}$-FE in the RO model. Gorbunov et al. [14] construct $\mathsf{Circuit}$-FE (in the standard model) which achieves only (q_1, ℓ, q_2)-SIM-security (rather than $q_2 = \mathsf{poly}$). See Table 1.

Our Techniques. Our compiler is inspired by the construction of zero-knowledge proof systems from witness indistinguishable proof systems, as studied in the celebrated work of Feige, Lapidot and Shamir [12]. Recall that in the FLS paradigm, the simulator operates the proof system in a "trapdoor" mode which is indistinguishable from the behavior of the honest party to the adversary. Adopting this paradigm to FE, our compiler produces "trapdoor circuits" which have additional "slots" in plaintext and keys that are used only by the simulator, not by the real system. To illustrate our techniques, consider the simpler case of a single challenge ciphertext and only *adaptive* key queries. Then, at a very high-level, instead of F we use a trapdoor circuit F_{trap} with an additional slot flag in the plaintext and an additional slot value in the key, namely:

[6] This can be relaxed to $(q_1, \ell, \mathsf{poly})$-IND-security.

[7] We emphasize that, since our transformation matches the known impossibility results, we do not obtain any impossibility result for $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-secure $\mathsf{Circuit}$-FE. Indeed, we believe $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-secure $\mathsf{Circuit}$-FE is possible.

$$F_{trap}((k, \mathsf{value}), (m, \mathsf{flag})) = \begin{cases} F(k, m) & \text{if } \mathsf{flag} = 0 \\ \mathsf{value} & \text{if } \mathsf{flag} = 1 \end{cases}$$

(This is not actually sufficient because a key may reveal value, but we are just trying to get a rough idea across; see Section 3 for the full constructions.) An honest encryptor will always set $\mathsf{flag} = 0$, but the simulator will set $\mathsf{flag} = 1$ and can then set value in the tokens it gives out to program the output F_{trap} appropriately. The proof of SIM-security is by reduction to IND-security of the underlying scheme, since the output of F_{trap} in the $\mathsf{flag} = 0$ and $\mathsf{flag} = 1$ cases will be the same.

Why is IND-security Enough? The above shows that, surprisingly, despite the weakness of IND-security for certain functionalities shown in [7, 20], an IND-secure FE scheme for general circuits is enough to go "all the way" to a SIM-secure one. To see how this can be possible, let us look at the counter-example functionality of [20], which is an f (which we think of here as a circuit) for which there is another function g such that g is "hard to compute" from f but is "isomorphic" to g, meaning f and g have the same equality pattern across the domain. In this case, despite IND-security, a token for f may also allow computing g. However, the corresponding trapdoor circuit produced by our compiler can be programmed to agree on f (via the additional slots) on all the challenge messages but *no longer agree on* g, because g is computed on a "dummy" message in the first plaintext slot. This means that if a token for computing f in the compiled scheme allowed computing g an adversary could indeed violate IND-security.

Simulation-Secure Hidden Vector Encryption. By using a similar high-level approach as in our general compiler, we also give a tailored construction of $(\mathsf{poly}, \ell, \mathsf{poly})$-SIM-secure HVE-FE, where "HVE" denotes *hidden vector encryption*, a generalization of anonymous IBE introduced by [8]. Again, these parameters are optimal in that they match known impossibility results [7, 3] discussed later. (Note that in this case we are able to achieve security for an unbounded number of non-adaptive key queries, which is impossible for the general Circuit functionality considered above [1].) The scheme is set in composite order bilinear groups and proven secure under the general subgroup decision assumption of [5]. In some sense, we compile existing IND-secure constructions of HVE-FE to a SIM-secure one in a "non-blackbox" way. Namely, our scheme mirrors

existing IND-secure constructions of HVE-FE [19, 11] except in some additional subgroups. The presence of an additional subgroup component in a simulated ciphertext acts as the "flag" that triggers an interaction with this additional subgroup component in the simulated keys.

Stronger Impossibility Results. As we mentioned, the positive results we obtain match the recent impossibility results for SIM-secure FE. Namely, Boneh et al. [7] show that $(0, \mathsf{poly}, 2)$-SIM-secure IBE is impossible (though in the "non-programmable" RO model; this was recently extended to the standard model in [3]). Agrawal et al. [1] show impossibility of $(\mathsf{poly}, 1, 0)$-SIM-security for a functionality that computes a weak pseudorandom function (wPRF-FE) and hence for Circuit-FE.

One mays wonder if there are weaker formulations of SIM-security under which these results might be circumvented. We identify two:

- *Fully Non-Adaptivity Adversaries.* The above results crucially rely on the fact that the experiment proceeds in distinct phases of non-adaptive key queries, encryption queries, then adaptive key queries. (In particular, the result of [1] for non-adaptive key queries crucially uses adaptivity of the encryption queries.) We thus ask whether these results can be circumvented for *fully non-adaptive adversaries* that must choose their encryption and key queries *simultaneously.*
- *Non-Blackbox Simulation.* The result of [1] requires that the simulator use the adversary as a black-box. We ask whether this result can be circumvented by using *non-blackbox simulation.*

Unfortunately, by building on the techniques of [3, 1], we go on to resolve these possibilities in the negative. See Table 2.

As a final contribution of independent interest, we show that in Circuit-FE the ciphertext length must grow with the output length of the functionality. Namely, we show impossibility of $(1, 1, 0)$-SIM-secure for a functionality that computes a pseudorandom generator (PRG); that is, the ciphertext length must be as long as the output length of the PRG. To the best of our knowledge, this is the first impossibility result for non-adaptive key queries and bounded collusion. Note that Goldwasser et al. [13] recently give a construction of FE for *boolean* functionalities with "succinct" ciphertexts, but for functionalities with longer output the ciphertext length in all existing constructions grows linearly with the output length of the functionality. Our result shows this is inherent.

Organization. In Section 2 we give the basic definitions for FE. In Section 3 we describe the transformations from IND-secure to SIM-secure

Work	Func	#Non-Adaptive Key Queries	#Encryption Queries	#Adaptive Key Queries	Assumptions
[BSW11]	IBE	UB	UB	UB	RO
Ours	HVE	UB	B	UB	Standard
[GVW12]	Circuit	B	UB	0	Standard
Ours	Circuit	UB	UB	UB	RO, IND-security
Ours	Circuit	B	B	UB	IND-security

Table 1. Positive results for SIM-secure FE. UB and B denote unbounded and bounded, respectively. Single-boxed entries are inherent (matching impossibility results below).

Work	Func	#Key Queries	#Encryption Queries	Key Query Time	Non-Black-Box Simulation?
[BSW11,BO12]	IBE	2	UB	Post-Challenge	YES
[AGVW12]	wPRF	UB	1	Pre-Challenge	NO
Ours	wPRF	UB	1	Pre-Challenge	YES
Ours	wPRF	UB	UB	Simultaneous	YES
Ours	PRG^*	1	1	Pre-Challenge	NO

Table 2. Negative results for SIM-secure FE. Boxed entries are new to our work. UB denote unbounded. Pre-challenge and post-challenge key queries refer to non-adaptive and adaptive queries , respectively, while simultaneous queries are queried together with the challenge. Results using pre-challenge and post-challenge queries are incomparable, while results using simultaneous queries are stronger. PRG^* refers to a pseudorandom generator functionality whose output is longer than the ciphertext size.

FE, both in the random oracle model and in the plain model. Section 4 describes the construction of SIM-secure FE for the hidden vector encryption functionality. In Section 5 we present our negative results.

2 Definitions

A *negligible* function $\mathsf{negl}(k)$ is a function that is smaller than the inverse of any polynomial in k. If D is a probability distribution, "$x \leftarrow D$" denotes that x is chosen according to D. If D is a finite set, "$x \leftarrow D$" denotes that x is chosen according to uniform probability on D. If $q > 0$ is an integer then $[q]$ denotes the set $\{1, \ldots, q\}$. "PPT" stands for "probabilistic polynomial time" and "PT" stands for "polynomial time." Algorithms are PPT unless explicitly noted otherwise. If A and B are algorithms, we denote by "$y \leftarrow A^{B(\cdot)}(x)$" that y is assigned the output of A when run on input x with oracle access to B. If A or B are randomized this is done using fresh random coins. If a and b are strings, then $a|b$ denotes the string representing their delimited concatenation. We will make use of standard primitives such as pseudorandom functions and symmetric-key encryption; definitions are in the full version of this paper [10].

2.1 Functional Encryption

Functional encryption (FE) schemes [7] are encryption schemes for which the owner of the master secret can compute restricted keys (also called "tokens") that allow to compute a *functionality* on the plaintext associated with a ciphertext. A formal definition follows.

Syntax. A *functionality* $F = \{F_n\}_{n>0}$ is a family of PT functions $F_n : K_n \times M_n \to \Sigma$ where K_n is the *key space* for parameter n, M_n is the *message space* for parameter n and Σ is the *output space*. Sometimes we will refer to functionality F as a function from $F : K \times M \to \Sigma$ with $K = \cup_n K_n$ and $M = \cup_n M_n$.

A *functional encryption scheme for* F (an F-FE scheme) is a tuple $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval})$ of four algorithms with the following syntax: Algorithm $\mathsf{Setup}(1^\lambda, 1^n)$ outputs *public* and *master secret* keys $(\mathsf{Pk}, \mathsf{Msk})$ for *security parameter* λ and *length parameter* n that are polynomially related. Algorithm $\mathsf{KeyGen}(\mathsf{Msk}, k)$, on input a master secret key Msk outputs *secret key* (or *token*) Tok. Algorithm $\mathsf{Enc}(\mathsf{Pk}, m)$, on input public key Pk and *plaintext* $m \in M_n$ outputs *ciphertext* Ct. PT algorithm $\mathsf{Eval}(\mathsf{Pk}, \mathsf{Ct}, \mathsf{Tok})$ outputs $y \in \Sigma \cup \{\perp\}$.

For *correctness* we require for all $(\mathsf{Pk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, all $k \in K_n$ and $m \in M_n$, for $\mathsf{Tok} \leftarrow \mathsf{KeyGen}(\mathsf{Msk}, k)$ and $\mathsf{Ct} \leftarrow \mathsf{Enc}(\mathsf{Pk}, m)$, we have that $\mathsf{Eval}(\mathsf{Pk}, \mathsf{Ct}, \mathsf{Tok}) = F(k, m)$ whenever $F(k, m) \neq \perp$, except with negligible probability. (See [3] for a discussion about this condition.)

Functionalities of interest. In this paper, we we mainly be concerned with two specific functionalities.

First, the $\mathsf{Circuit}$ functionality has key space K_n equals to the set of all n-input Boolean circuits and message space M_n the set $\{0,1\}^n$ of n-bit strings. For $C \in K_n$ and $m \in M_n$, we have $\mathsf{Circuit}(C, m) = C(m)$. In the random oracle (RO) model [4] we *allow* the circuits in the $\mathsf{Circuit}$ functionality to have RO gates. This is because, in practice, we replace the random oracle invocations with computation of a cryptographic hash function having an explicit circuit description.

Second, the HVE functionality [8] has message space M_n equal to the set of length n Boolean vectors $\boldsymbol{x} = \langle x_1, \ldots, x_n \rangle$ and key space K_n equal to the set length n Boolean vectors $\boldsymbol{y} = \langle y_1, \ldots, y_n \rangle$ with $\star$'s ("don't-care" entries). $\mathsf{HVE}(\boldsymbol{x}, \boldsymbol{y})$ is equal to 1 iff, for all $1 \leq i \leq n$, $x_i = y_i$ or $y_i = \star$.

Security. We next define indistinguishability-based and simulation-based security for FE based on [7, 20]. Some remarks about the definitions follow them.

Definition 1 [Indistinguishability-based security.] We say that an F-FE scheme is (q_1, ℓ, q_2)-*IND-secure* if for every PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ where $\mathcal{A}_0$ makes at most q_1 oracle queries and $\mathcal{A}_1$ makes at most q_2 oracle queries, the advantage of $\mathcal{A}$ defined as

$$\mathsf{Adv}^{\mathsf{FE},\mathsf{IND}}_{\mathcal{A}}(1^\lambda, 1^n) = \left|\mathrm{Prob}[\mathsf{IND}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda, 1^n) = 1] - 1/2\right|$$

is negligible, where:

Experiment $\mathsf{IND}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda, 1^n)$:
$(\mathsf{Pk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$
$(\boldsymbol{m}_0, \boldsymbol{m}_1, \mathtt{st}) \leftarrow \mathcal{A}_0^{\mathsf{FE.KeyGen}(\mathsf{Msk},\cdot)}(\mathsf{Pk})$ where $\boldsymbol{m}_0, \boldsymbol{m}_1 \in M_n^\ell$
$b \leftarrow \{0,1\}$; $\mathsf{Ct}[i] \leftarrow \mathsf{FE.Enc}(\mathsf{Pk}, \boldsymbol{m}_b[i])$ for $i \in [\ell]$
$b' \leftarrow \mathcal{A}_1^{\mathsf{FE.KeyGen}(\mathsf{Msk},\cdot)}(\mathtt{st}, \mathsf{Ct})$
Output: $(b = b')$

Above we require that $F(k, \boldsymbol{m}_0[i]) = F(k, \boldsymbol{m}_1[i])$ for every $i \in [\ell]$ and every oracle query k made by either $\mathcal{A}_0$ or $\mathcal{A}_1$.

Definition 2 [Simulation-Based security.] We say that an F-FE scheme is (q_1, ℓ, q_2)-SIM-*secure* if for every PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ where $\mathcal{A}_0$ makes at most q_1 oracle queries and $\mathcal{A}_1$ makes at most q_2 oracle queries, there exists a PPT simulator $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ such that the outputs of the following two experiments are computationally indistinguishable:

Experiment $\mathsf{RealExp}^{\mathsf{FE},\mathcal{A}}(1^\lambda, 1^n)$:
$(\mathsf{Pk}, \mathsf{Msk}) \leftarrow \mathsf{FE.Setup}(1^\lambda, 1^n)$
$(\boldsymbol{m}, \mathtt{st}) \leftarrow \mathcal{A}_0^{\mathsf{FE.KeyGen}(\mathsf{Msk},\cdot)}(\mathsf{Pk})$
$\mathsf{Ct} \leftarrow \mathsf{Enc}(\mathsf{Pk}, \boldsymbol{m})$
$\alpha \leftarrow \mathcal{A}_1^{\mathsf{FE.KeyGen}(\mathsf{Msk},\cdot)}(\mathsf{Pk}, \mathsf{Ct}, \mathtt{st})$
Output: $(\mathsf{Pk}, \boldsymbol{m}, \{k_i\}, \alpha)$

Experiment $\mathsf{IdealExp}^{\mathsf{FE},\mathcal{A}}_{\mathsf{Sim}}(1^\lambda, 1^n)$:
$(\mathsf{Pk}, \mathsf{Msk}) \leftarrow \mathsf{FE.Setup}(1^\lambda, 1^n)$
$(\boldsymbol{m}, \mathtt{st}) \leftarrow \mathcal{A}_0^{\mathsf{FE.KeyGen}(\mathsf{Msk},\cdot)}(\mathsf{Pk})$
$(\mathsf{Ct}, \mathtt{st}') \leftarrow \mathsf{Sim}_0(\mathsf{Pk}, |\boldsymbol{m}|, \{k_i, \mathsf{Tok}_{k_i}, F(k_i, \boldsymbol{m})\})$
$\alpha \leftarrow \mathcal{A}_1^{\mathcal{O}(\cdot)}(\mathsf{Pk}, \mathsf{Ct}, \mathtt{st})$
Output: $(\mathsf{Pk}, \boldsymbol{m}, \{k_i\}, \alpha)$

Above we require $|\boldsymbol{m}| \leq \ell$. In the output of the experiments, $\{k_i\}$ contains the token queries of the adversary (i.e., the queries of $\mathcal{A}_0$ and $\mathcal{A}_1$ combined). The oracle $\mathcal{O}(\cdot)$ is the second stage of the simulator, namely algorithm $\mathsf{Sim}_1(\mathsf{Msk}, \mathtt{st}', \cdot, \cdot)$, which receives as its third argument a key k_j for which the adversary queries a token and as its fourth argument the output value $F(k_j, m)$. Further, note that the simulator algorithm Sim_1 is stateful in that after each invocation, it updates the state $\mathtt{st}'$ which is carried over to its next invocation.

We also note that above follows the security definitions of [20, 15] in that in the ideal experiment, the setup and non-adaptive token queries are handled honestly (not by the simulator). This is just for simplicity. Additionally, the challenge messages are selected by $\mathcal{A}_0$ in "one-shot" and not adaptively depending on previous challenge ciphertexts as in [3]. Again, this is just for simplicity.

Random oracle model. To lift our definition to the random oracle (RO model [4], the output of the real experiment *includes* the queries made by any algorithm (i.e., either those of the scheme or the adversary) to the RO and the responses. In the ideal experiment, the simulator provides responses to the queries made by any algorithms to the RO and the output of the experiment again *includes* all these queries and responses. This is analogous to the "explicitly" programmable RO model formalized by Wee [23] for zero-knowledge and seems to us to be the most natural formalization of security in the RO model in our context.

3 From Indistinguishability to Simulation-Based Security

In this section, we show that from an IND-secure Circuit-FE scheme one can construct a SIM-secure Circuit-FE scheme. We give two constructions: one in the RO model [4] and one in the standard model, which (necessarily) achieve security for different parameters.

3.1 Trapdoor Circuits

The idea of our transformations is to replace the original circuit with a "trapdoor" one that a simulator can use to program the output in some way. This will be done via interaction of additional "slots" in the plaintext and key that interact when a flag is set in the plaintext. This approach is inspired by the FLS paradigm introduced by Feige, Lapidot and Shamir [12] to obtain zero-knowledge proof systems from witness indistinguishable proof systems.

Random oracle model construction. Here, a plaintext will have four slots and a key will have two. In the plaintext, the slots will be: (1) actual message m (2) a bit flag to indicate trapdoor mode, (3) a random string x, and (4) a seed r for a pseudorandom function (PRF). In the key, the slots will be (1) the actual circuit C and (2) a random string y. For evaluation, in non-trapdoor mode we simply evaluate the original circuit C on m. in

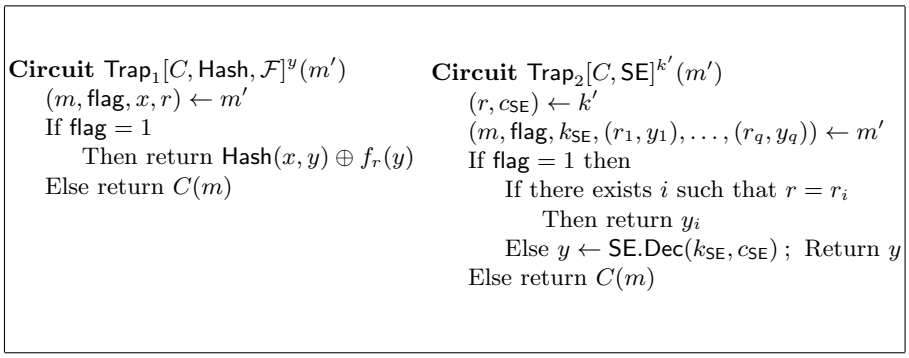


Fig. 1. Construction of trapdoor circuit in the RO model (left) and standard model (right) from a given circuit C.

trapdoor mode, the output is instead "programmed" as $\mathsf{Hash}(x, y) \oplus f_r(y)$, where Hash is a RO and f_r is the PRF keyed by r.

Formally, let C be a circuit on n-bits. Let $\mathsf{Hash}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a hash function and $\mathcal{F} = \{f_s : s \in \{0,1\}^k\}_{k \in \mathbb{N}}$ be a PRF. For $y \in \{0,1\}^n$ define the corresponding *RO-based trapdoor circuit* $\mathsf{Trap}_1[C, \mathsf{Hash}, \mathcal{F}]^y$ on $(3n+1)$-bits as in Figure 1.

Standard model construction. Here, a plaintext will have $3+2q$ slots (for a polynomial q) and a key will three two. In the plaintext, the slots will be: (1) the actual message m, (2) a bit flag to indicate trapdoor mode, (3) a key $\mathsf{sk}_{\mathsf{SE}}$ for a symmetric-key encryption scheme SE, and finally the last $2q$ slots will be pairs (r_i, z_i), where r_i is a random string and z_i is a desired output value. (Looking ahead, the third slot will be used to handle adaptive key queries and the last $2q$ slots will be used to handle up to q non-adaptive key queries.) On the other hand, in the key the slots will be: (1) the actual circuit C, (2) a random string r, and (3) a ciphertext c_{SE} under SE. For evaluation, in non-trapdoor mode we simply evaluate the original circuit C on m. In in trapdoor mode, if $r = r_i$ for some $i \in [q]$ then the output is "programmed"' as z_i, and otherwise as $\mathsf{SE.Dec}(\mathsf{sk}_{\mathsf{SE}}, \mathsf{c})$ where $\mathsf{SE.Dec}$ is the decryption algorithm of SE.

Formally, let C be a circuit with n-bit inputs and n-bit outputs, and let $\mathsf{SE} = (\mathsf{SE.Enc}, \mathsf{SE.Dec})$ be a symmetric-key encryption scheme with key-space $\{0,1\}^s$, message-space $\{0,1\}^n$, and ciphertext-space $\{0,1\}^\nu$. For $k' \in \{0,1\}^{n+\nu}$ define the corresponding *standard-model trapdoor circuit* $\mathsf{Trap}_2[C, \mathsf{SE}]^{k'}$ with $((2q+1)n + 1 + s)$-bit inputs and n-bit outputs as in Figure 1.

3.2 Random Oracle Model Transformation

Let $\mathsf{IndFE} = (\mathsf{IndFE.Setup}, \mathsf{IndFE.Enc}, \mathsf{IndFE.KeyGen}, \mathsf{IndFE.Eval})$ be a functional encryption scheme for the functionality $\mathsf{Circuit}$. Let $\mathsf{Hash}\colon \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a hash function (which will be modeled as a random oracle) and $\mathcal{F} = \{f_s : s \in \{0,1\}^k\}_{k\in\mathbb{N}}$ be a PRF. We define a new FE scheme $\mathsf{SimFE}_1[\mathsf{Hash}, \mathcal{F}] = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval})$ for $\mathsf{Circuit}$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^n)$: returns the output of $\mathsf{IndFE.Setup}(1^\lambda, 1^{3n+1})$ as its own output.
- $\mathsf{Enc}(\mathsf{Pk}, m)$: on input Pk and $m \in \{0,1\}^n$, the algorithm chooses x at random from $\{0,1\}^n$, sets $m' = (m, 0, x, 0^n)$ and returns $\mathsf{IndFE.Enc}(\mathsf{Pk}, m')$ as its own output.
- $\mathsf{KeyGen}(\mathsf{Msk}, C)$: on input Msk and a n-input Boolean circuit C, the algorithm chooses random $y \in \{0,1\}^n$ and returns (y, Tok) where $\mathsf{Tok} \leftarrow \mathsf{IndFE.KeyGen}(\mathsf{Msk}, \mathsf{Trap}_1[C, \mathsf{Hash}, \mathcal{F}]^y)$.
- $\mathsf{Eval}(\mathsf{Pk}, \mathsf{Ct}, \mathsf{Tok})$: on input Pk, Ct and Tok, returns the output $\mathsf{IndFE.Eval}(\mathsf{Pk}, \mathsf{Ct}, \mathsf{Tok})$.

Theorem 3 Suppose IndFE is $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-Secure. Then SimFE_1 is $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-SIM-secure in the random oracle model.

We defer the proof to the full version of this paper [10] and give some intuition here. it is instructive to consider a simpler system where $f_r(y)$ in the evaluation is simply replaced by r. In this case, the fourth slot in the plaintext acts as an encryption under Nielsen's RO-based non-committing encryption scheme [18], whose decryption can be adaptively programmed. However, this approach does not work for multiple tokens, since then the simulator would need to program two hash outputs to $r \oplus C_1(m)$ and $r \oplus C_2(m)$, which would not look independently random to the distinguisher. Since the number of tokens is unbounded, we need to generate more randomness than can be contained in the plaintext slot, and thus we use a PRF to generate a "fresh" ciphertext for each token.

A note on uninstantiability. We notice that, due to the result of Bellare and O'Neill [3], our construction in the RO model cannot be proven SIM-secure when implemented with any function ensemble in place of the RO. However, we stress that unlike other some other "uninstantiable" schemes (e.g., those of Canetti et al. [9]) which are *clearly* insecure (in the standard model) when implemented with any function ensemble, our construction does not seem to suffer any real-world attack. In this sense,

we still view it as a good heuristic for our scheme to have a proof of security in the RO model.

3.3 Standard Model Transformation

Let $\mathsf{IndFE} = (\mathsf{IndFE.Setup}, \mathsf{IndFE.Enc}, \mathsf{IndFE.KeyGen}, \mathsf{IndFE.Eval})$ be a functional encryption scheme for the functionality $\mathsf{Circuit}$. Let $\mathsf{SE} = (\mathsf{SE.Enc}, \mathsf{SE.Dec})$ be a symmetric-key encryption scheme with key-space $\{0,1\}^s$, message-space $\{0,1\}^n$, and ciphertext-space $\{0,1\}^\nu$. We define a new FE scheme $\mathsf{SimFE}_2[\mathsf{SE}] = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Eval})$ for $\mathsf{Circuit}$ as follows:

- $\mathsf{Setup}(1^\lambda, 1^n)$: returns the output of $\mathsf{IndFE.Setup}(1^\lambda, 1^{n(2q+1)+s+1})$ as its own output. In addition the algorithm picks a random key $\mathsf{sk_{SE}} \in \{0,1\}^{\mathsf{s}}$ and keeps it in the master secret key Msk.
- $\mathsf{Enc}(\mathsf{Pk}, m)$: on input Pk and $m \in \{0,1\}^n$, the algorithm sets $m' \leftarrow (m, 0, 0^s, (0^n, 0^n), \ldots, (0^n, 0^n))$ and returns the output of $\mathsf{IndFE.Enc}(\mathsf{Pk}, m')$ as its own output.
- $\mathsf{KeyGen}(\mathsf{Msk}, C)$: on input Msk and a n-input Boolean circuit C, the algorithm chooses random $r \in \{0,1\}^\lambda$ and $c \in \{0,1\}^\nu$, and returns (r, c, Tok) where it computes $\mathsf{Tok} \leftarrow \mathsf{IndFE.KeyGen}(\mathsf{Msk}, \mathsf{Trap}_2[C, \mathsf{SE}]^{k'})$ and sets $k' \leftarrow r \| c$.
- $\mathsf{Eval}(\mathsf{Pk}, \mathsf{Ct}, \mathsf{Tok})$: on input Pk, ciphertext Ct and token (r, c, Tok), returns the output of $\mathsf{IndFE.Eval}(\mathsf{Pk}, \mathsf{Ct}, \mathsf{Tok})$.

Theorem 4 Suppose IndFE is $(q_1, 1, \mathsf{poly})$-IND-secure, $\mathcal{F}$ is a PRF, and SE has pseudorandom ciphertexts. Then SimFE_2 is $(q_1, 1, \mathsf{poly})$-SIM-secure.

Again, we defer the proof to the full version [10] and give some intuition here. The intuition is very similar in spirit to that of Theorem 3. First, consider a simpler system where the $2q$ pairs (r_i, z_i) are replaced by a single pair $(\tilde{r}, \tilde{z})$. This approach does not work for multiple non-adaptive tokens, since then the simulator would need to program $\tilde{z}$ to be $C_1(m)$ and $C_2(m)$ at the same time. To solve this problem, we add additional pairs in the ciphertext, one for each non-adaptive query. This is also the reason why we need an *a priori* bound on the number of non-adaptive key queries. For adaptive key queries, the simulator can instead program c in the token to be an encryption of the desired output.

We note that it is straightforward to extend our construction to achieve $(q_1, \ell, \mathsf{poly})$-SIM-security for any polynomial ℓ (where now we need to assume the starting scheme is $(q_1, \ell, \mathsf{poly})$-IND-secure). Note that by [7, 3], the restriction to a bounded ℓ is necessary in the standard model. Moreover, by [1], for the $\mathsf{Circuit}$ functionality the restriction to a bounded number of non-adaptive key queries q_1 is also necessary.

An instantiation for polynomial evaluation. In the full version of this paper [10], we show how to adapt our standard model transformation to the polynomial evaluation functionality [16], for which which we have efficient constructions from bilinear maps and lattices.

4 Simulation-Secure Hidden Vector Encryption

In the section we present a SIM-secure HVE-FE scheme whose whose security can be proved under static assumptions in the bilinear pairing setting in the standard model. We use composite order bilinear groups whose order is the product of five distinct primes (see the full version for the standard definition of such groups [10]).

The Scheme. We now describe our HVE scheme. To make our description and proofs simpler, we add to all vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ two dummy components and set both of them equal to 0. We can thus assume that all vectors have at least two non-star positions.

- $\mathsf{Setup}(1^\lambda, 1^\ell)$: The setup algorithm chooses a description of a bilinear group $\mathcal{I} = (N = p_1p_2p_3p_4p_5, \mathbb{G}, \mathbb{G}_T, \mathbf{e}) \leftarrow \mathcal{G}(1^\lambda)$ with known factorization, and random $g_1 \in \mathbb{G}_{p_1}$, $g_2 \in \mathbb{G}_{p_2}$, $g_3 \in \mathbb{G}_{p_3}$, $g_4 \in \mathbb{G}_{p_4}$, and, for $i \in [\ell]$ and $b \in \{0,1\}$, random $t_{i,b} \in Z_N$ and random $R_{i,b} \in \mathbb{G}_{p_3}$ and sets $T_{i,b} = g_1^{t_{i,b}} \cdot R_{i,b}$. The public key is $\mathsf{Pk} = [\mathcal{I}, g_3, (T_{i,b})_{i\in[\ell],b\in\{0,1\}}]$, and the master secret key is $\mathsf{Msk} = [g_{12}, g_4, (t_{i,b})_{i\in[\ell],b\in\{0,1\}}]$, where $g_{12} = g_1 \cdot g_2$. The algorithm returns $(\mathsf{Pk}, \mathsf{Msk})$.
- $\mathsf{KeyGen}(\mathsf{Msk}, \boldsymbol{y})$: Let $S_{\boldsymbol{y}}$ be the set of indices i such that $y_i \neq \star$. The key generation algorithm chooses random $a_i \in \mathbb{Z}_N$ for $i \in S_{\boldsymbol{y}}$ under the constraint that $\sum_{i\in S_{\boldsymbol{y}}} a_i = 0$. For $i \in S_{\boldsymbol{y}}$, the algorithm chooses random $W_i \in \mathbb{G}_{p_4}$ and sets $Y_i = g_{12}^{a_i/t_{i,y_i}} \cdot W_i$. The algorithm returns ciphertext $\mathsf{Ct} = (Y_i)_{i\in S_{\boldsymbol{y}}}$. Here we use the fact that $S_{\boldsymbol{y}}$ has size at least 2.
- $\mathsf{Enc}(\mathsf{Pk}, \boldsymbol{x})$: The encryption algorithm chooses random $s \in \mathbb{Z}_N$. For $i \in [\ell]$, the algorithm chooses random $Z_i \in \mathbb{G}_{p_3}$ and sets $X_i = T_{i,x_i}^s \cdot Z_i$, and returns the token $\mathsf{Tok}_{\boldsymbol{y}} = (X_i)_{i\in[\ell]}$.
- $\mathsf{Eval}(\mathsf{Pk}, \mathsf{Ct}, \mathsf{Tok}_{\boldsymbol{y}})$: The test algorithm computes $T = \prod_{i\in S_{\boldsymbol{y}}} \mathbf{e}(X_i, Y_i)$. It returns TRUE if $T = 1$, FALSE otherwise.

It easy to see that the scheme is correct. Regarding security, we show:

Theorem 5 Under the General Subgroup Decision Assumption [5] the HVE scheme described is $(\mathsf{poly}, 1, \mathsf{poly})$-SIM-secure.

The proof is in the full version of this paper [10]. Informally, we simulate the flag used in the trapdoor circuits by means of the presence of the $\mathbb{G}_{p_5}$ subgroup. Specifically, if the $\mathbb{G}_{p_5}$ part is absent the ciphertext is in normal mode, otherwise it acts in trapdoor mode. The simulator then modifies the distributions of the adaptive queries, adding a $\mathbb{G}_{p_5}$ part, to interact with the trapdoor mechanism of the ciphertext when needed.

We note that one can easily extend our construction to meet $(\mathsf{poly}, \ell, \mathsf{poly})$-SIM security for polynomial ℓ. The idea is simply to use a different subgroup for each message in the "trapdoor" mode. By [7, 3], the restriction to a bounded number of challenge ciphertexts ℓ is necessary. On the other hand, the impossibility result of [1] does not apply to HVE, so there is no contradiction with the fact that our result here has $q_1 = \mathsf{poly}$ (instead of bounded q_1 as for our standard-model construction of $\mathsf{Circuit}$-FE in Section 3, which is necessary).

5 Impossibility Results

In the section we present new negative results for simulation-based secure FE. We refer the reader to Section 1 for a background on the previously known impossibility results. All of our negative results build on ideas from the impossibility result of [1] for $(\mathsf{poly}, 1, 0)$-SIM-secure wPRF-FE w.r.t. black-box simulation. Below, we first describe the weak PRF functionality (that will be used in our negative results as well) and recall the impossibility result of [1]. We then proceed to discuss our new results.

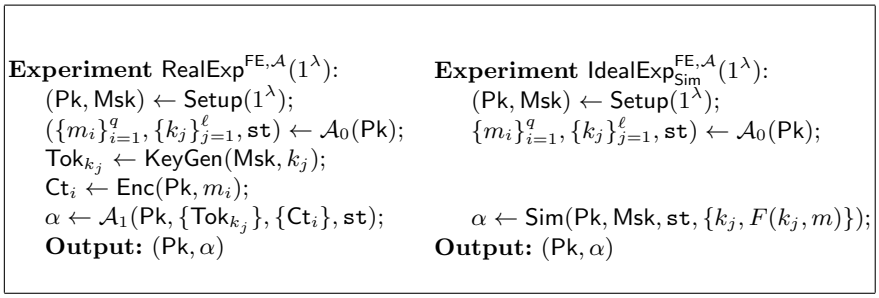
Impossibility of Agrawal et al. [1]. Let $\{F\}$ be a weak pseudo-random function family on domain K and key space M. The wPRF functionality on key $k \in K$ and input $m \in M$ outputs $F_m(k)$. Let $l - 1$ be an upper bound on the ciphertext size of the wPRF-FE scheme. The adversary asks tokens for l random inputs $x_1, \ldots, x_l$ in the domain of F, and for an encryption of a random k from the key space of F. The simulator needs to produce tokens $\{\mathsf{Tok}_i\}_{i\in[l]}$, and then it is given the functionality's outputs $\{F_k(x_i)\}_{i\in[l]}$. Now the simulator has to produce a ciphertext Ct such that for every $i \in [l]$, $F_k(x_i) = \mathsf{Eval}(\mathsf{Pk}, \mathsf{Ct}, \mathsf{Tok}_i)$. Now, on the one hand, the simulator needs to "encode" all of the functionality's outputs into Ct. On the other hand, the functionality's outputs are l pseudo-random bits, while $|\mathsf{Ct}| < l - 1$. Since a pseudo-random string cannot be efficiently compressed we get a contradiction. (Note that a *black-box* simulator cannot encode the functionality's outputs into the tokens $\{\mathsf{Tok}_i\}$ since these are fixed before the simulator learns the outputs.)

5.1 Fully Non-Adaptive Adversaries

In this section we give an impossibility result for a natural relaxation of the simulation-security considering only adversaries that are *fully non-adaptive*. In particular, we consider adversaries who make *simultaneous* token and ciphertext queries in the SIM-security game for FE.

Below, we formally define security against fully non-adaptive adversaries. Our definition allows for non-black-box simulation.

Definition 6 [Fully Non-Adaptive Security] We say that an F-FE scheme is (q, ℓ)-*fully non-adaptively* SIM-secure if every PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ there exists a PPT simulator Sim such that the outputs of the following two experiments are computationally indistinguishable:



Theorem 7 Assuming the existence of a collision-resistant hash function family, there does not exist a (poly, poly)-fully-non-adaptively SIM-secure wPRF-FE.

We prove the above theorem by extending the impossibility of [1]. Roughly speaking, the central idea is to use many ciphertext queries instead of one. The intuition is that in the non-adaptive case, the simulator can encode information about the function outputs in the tokens that might be long; however, by making many ciphertext queries, the same tokens can be used to decrypt many ciphertexts, making the length of the tokens insignificant. Indeed, the same idea can be used in the impossibility for given in the next subsections (at the cost of an increase in a number of ciphertext queries). We defer details to the full version [10].

5.2 Non-Black-Box Simulation

The impossibility of [1] rules out SIM-security against adversaries who make an unbounded number of non-adaptive token queries assuming the simulator is using the code of the adversary as black-box. In this section we extend their result to non-black-box simulators using the techniques from [3].

Below, we give a non-black-box definition of SIM-security, which is similar to that of [7] except that we only consider an unbounded number of non-adaptive token queries and one ciphertext query (corresponding to $(\mathsf{poly}, 1, 0)$-SIM-security). Further, following [3] we let the adversary and the simulator use an auxiliary input sampled from some distribution. In our negative result, we use this auxiliary input to store a random key of a collision-resistant hash function.[8]

Definition 8 [Non-Black-Box Simulation] We say that an F-FE scheme SIM-secure with *non-black-box* simulator if for every distribution on auxiliary input Z, and every PPT adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, there exists a PPT simulator $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ such that the outputs of the following two experiments are computationally indistinguishable:

Experiment $\mathsf{RealExp}^{\mathsf{FE},\mathcal{A}}(1^\lambda)$:
$(\mathsf{Pk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda)$; $z \leftarrow Z$;
$(M, \mathtt{st}) \leftarrow \mathcal{A}_0^{\mathsf{KeyGen}(\mathsf{Msk},\cdot)}(\mathsf{Pk}, z)$;
$m \leftarrow M, \mathsf{Ct} \leftarrow \mathsf{Enc}(\mathsf{Pk}, m)$;
$\alpha \leftarrow \mathcal{A}_1(\mathsf{Ct}, \mathtt{st})$;
Let $\{k_i\}$ be the queries of $\mathcal{A}_0$ to KeyGen;
Output: $(z, M, m, \alpha, \{k_i\})$

Experiment $\mathsf{IdealExp}^{\mathsf{FE}}_{\mathsf{Sim}}(1^\lambda)$:
$z \leftarrow Z$;
$(M, \mathtt{st}) \leftarrow \mathsf{Sim}_0(z)$;
$m \leftarrow M$;
$\alpha \leftarrow \mathsf{Sim}_1^{F(\cdot,m)}(\mathtt{st})$;
Let $\{k_i\}$ be the queries of Sim_1 to F;
Output: $(z, M, m, \alpha, \{k_i\})$

where the output of $\mathcal{A}_0$ and Sim_0 consists of an arbitrary state $\mathtt{st}$ and a description of a distribution over messages M.

We now state our result:

Theorem 9 Assuming collision-resistant hash functions, there does not exist a SIM-secure wPRF-FE with non-black-box simulator.

In the non-black-box simulation definition the real and the simulated outputs may contain the generated tokens and ciphertext. However, the simulator is only required to produce the simulated tokens and ciphertext after receiving the functionality's outputs. Since the tokens may encode a lot of information (at least as much as the functionality's outputs), the impossibility of [1] is not applicable here. To commit the simulator to the tokens before learning the functionality's outputs we use technique of [2]. This technique was recently used by [3] to extend the impossibility of

[8] A stronger variant of FE with non-black-box simulation is defined in [3] and our negative result holds also for their definition.

[7] to hold without a non-programmable random oracle. The main idea is to consider an adversary that computes a collision-resistant hash of the tokens, and selects the message distribution based on the hash value. Intuitively, this commits the simulator to the tokens before it learns the functionality's outputs. We defer details to the full version [10].

5.3 FE for Multi-bit Outputs with Succinct Ciphertexts

Finally, we show that it is impossible to construct FE schemes where the ciphertext length is *independent* of the output length of the functionality. Recently, Goldwasser et al. [13] construct a SIM-secure FE scheme with "succinct" ciphertexts, improving on Gorbunov et al. [15] (in which the ciphertext size depends on the size of the circuit computing the functionality). However, [13] is only for functionalities with *boolean* output; for functionalities with longer output, the ciphertexts in both of these constructions grows linearly with the output length of the functionality. Our result shows this dependency is *inherent*.

To prove this result we consider the functionality that computes a pseudorandom generator, and we set the output length of the generator to be longer then the size of the FE ciphertext. The proof uses an incompressibility argument similar to the one used in [1]. However, unlike in [1] we consider only one token query and one ciphertext query and do not rely on an unbounded collusion. Due to space constraints, formal definitions and details are deferred to the full version [10].

Acknowledgements

We thank the anonymous reviewers of Crypto 2013 for very helpful comments regarding the presentation.

References

1. S. Agrawal, S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption: New perspectives and lower bounds. Cryptology ePrint Archive, Report 2012/468, 2012. `http://eprint.iacr.org/`.
2. M. Bellare, R. Dowsley, B. Waters, and S. Yilek. Standard security does not imply security against selective-opening. In *EUROCRYPT 2012*, pages 645–662.
3. M. Bellare and A. O'Neill. Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition. Cryptology ePrint Archive, Report 2012/515, 2012. `http://eprint.iacr.org/`.
4. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM CCS 93*, pages 62–73.

5. M. Bellare, B. Waters, and S. Yilek. Identity-based encryption secure against selective opening attack. In *TCC 2011*, pages 235–252.
6. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano. Public key encryption with keyword search. In *EUROCRYPT 2004*, pages 506–522.
7. D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In *TCC 2011*, pages 253–273.
8. D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC 2007*, pages 535–554.
9. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited (preliminary version). In *30th ACM STOC*, pages 209–218. Full version avaiable at Cryptology ePrint Archive, Report 1998/011.
10. A. D. Caro, V. Iovino, A. Jain, A. O'Neill, O. Paneth, and G. Persiano. On the achievability of simulation-based security for functional encryption. *IACR Cryptology ePrint Archive*, 2013.
11. A. De Caro, V. Iovino, and G. Persiano. Fully secure hidden vector encryption. In *PAIRING 2012*, pages 102–121.
12. U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract). In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume I*, pages 308–317.
13. S. Goldwasser, Y. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich. Reusable garbled circuits and succinct functional encryption. In *45th ACM STOC*, pages 555–564.
14. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In *CRYPTO 2012*, pages 162–179.
15. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In *CRYPTO*.
16. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT 2008*, pages 146–162.
17. A. B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT 2010*, pages 62–91.
18. J. B. Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *CRYPTO 2002*, pages 111–126.
19. T. Okamoto and K. Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In *EUROCRYPT 2012*, pages 591–608.
20. A. O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. `http://eprint.iacr.org/`.
21. B. Parno, M. Raykova, and V. Vaikuntanathan. How to delegate and verify in public: Verifiable computation from attribute-based encryption. In *TCC 2012*, pages 422–439.
22. B. Waters. Functional encryption for regular languages. In *CRYPTO*.
23. H. Wee. Zero knowledge in the random oracle model, revisited. In *ASIACRYPT 2009*, pages 417–434.