# On the Power of Rewinding Simulators in Functional Encryption

Angelo De Caro[1] and Vincenzo Iovino[2]

[1] IBM Research Zurich, Switzerland
adc@zurich.ibm.com
[2] University of Luxembourg
vinciovino@gmail.com

**Abstract.** In a seminal work, Boneh, Sahai and Waters (BSW, for short) [TCC'11] showed that for functional encryption the indistinguishability notion of security (IND-Security) is weaker than simulation-based security (SIM-Security), and that SIM-Security is in general impossible to achieve. This has opened up the door to a plethora of papers showing feasibility and new impossibility results. Nevertheless, the quest for better definitions that (1) overcome the limitations of IND-Security and (2) the known impossibility results, is still open.

In this work, we explore the benefits and the limits of using *efficient rewinding black-box simulators* to argue security. To do so, we introduce a new simulation-based security definition, that we call *rewinding simulation-based security* (RSIM-Security), that is weaker than the previous ones but it is still sufficiently strong to not meet pathological schemes as it is the case for IND-Security (that is implied by the RSIM). This is achieved by retaining a strong simulation-based flavour but adding more rewinding power to the simulator having care to guarantee that it can not learn more than what the adversary would learn in any run of the experiment. What we found is that for RSIM the BSW impossibility result does not hold and that IND-Security is *equivalent* to RSIM-Security for *Attribute-Based Encryption* in the *standard model*. Nevertheless, we prove that there is a setting where rewinding simulators are of no help. The adversary can put in place a strategy that forces the simulator to rewind continuously.

**Keywords. Functional Encryption, Simulation-Based Security, Rewinding.**

# 1 Introduction

Functional encryption (FE, for short) is a sophisticated type of encryption that was first proposed by Sahai and Waters in 2005 [36] and formalized by Boneh, Sahai and Waters in 2011, [14]. Roughly speaking, in a functional encryption system, a decryption key allows a user to learn a *function* of the encrypted data. More specifically, in a functional encryption scheme for functionality $F : K \times X \rightarrow \Sigma$, defined over *key space* $K$, *message space* $X$ and *output space* $\Sigma$, for every *key* $k \in K$, the owner of the master secret key $\mathsf{Msk}$ associated with master public key $\mathsf{Mpk}$ can generate a secret key $\mathsf{Sk}_k$ that allows the computation of $F(k, x)$ from a ciphertext of $x$ computed under master public key $\mathsf{Mpk}$. In other words, a functional encryption scheme generalizes classical encryption schemes where the secret key allows to compute the entire plaintext. In recent breakthroughs, functional encryption schemes for general functionalities have been constructed by [27, 19, 16, 6].

A notable subclass of functional encryption is that of *predicate encryption* (PE, for short) which are defined for functionalities whose message space $X$ consists of two subspaces $I$ and $M$ called respectively *index space* and *payload space*. In this case, the functionality $F$ is defined in terms of a polynomial-time predicate $P : K \times I \rightarrow \{0, 1\}$ as follows: $F(k, (\mathsf{ind}, \mathsf{m})) = \mathsf{m}$ if $P(k, \mathsf{ind}) = 1$, $\perp$ otherwise, where $k \in K$, $\mathsf{ind} \in I$ and $\mathsf{m} \in M$. Those schemes are also called *predicate encryption with private-index.* Examples of such schemes are Anonymous Identity-Based Encryption (AIBE, for short) [13, 21], Inner-Product Encryption [15, 32–34] among others. On the other hand, when the index $\mathsf{ind}$ is easily readable from the ciphertext those schemes are called *predicate encryption with public-index* (PIPE, for short). Also for this specific subclass, the literature provides lots of constructions such that Identity-Based Encryption (IBE, for short) [37, 13, 17], Attribute-Based Encryption (ABE, for short) [36, 30, 20, 29], Functional Encryption for Regular Languages [38], among others.

A general study of the security of functional encryption did not appear initially. Instead, progressively more expressive forms of FE were constructed in a series of works that adopted indistinguishability-based (IND) notions of security, which requires that it is infeasible to distinguish encryption of any two messages without getting a secret key that decrypts the ciphertexts to distinct values. Only recently, papers studying simulation-based (SIM) notions of security for functional encryption were proposed by Boneh, Sahai, and Waters [14] and O'Neill [35] who explored security definitions for functional encryption that arise from the simulation paradigm [25, 26, 23]. The aim of these simulation-based definitions was to capture the most basic intuition about security for FE, namely that getting the secret key $\mathsf{Sk}_k$ corresponding to the key $k \in K$ should only reveal $F(k, x)$ when given an encryption of $x$.

*Previous Works.* Results about functional encryption now live in a high-dimensional space, where there are many parameters and several results ruling out or constructing schemes for certain parameters. Before presenting these results, to make things clear, following [18] notation, we define $(q_1, \ell, q_2)$-$\mathsf{atk}$-Security, where $q_1 = q_1(\lambda), \ell = \ell(\lambda), q_2 = q_2(\lambda)$ are either polynomials in the security parameter $\lambda$ that are fixed a priori or equal to the formal variable $\mathsf{poly}$, and $\mathsf{atk} \in \{\mathsf{IND}, \mathsf{SIM}\}$, as follows. Specifically, $\mathsf{atk}$-Security holds for adversaries $\mathcal{A}$ that issues at most $q_1$ *non-adaptive* key-generation queries, output *challenge message vectors* of length at most $\ell$, and furthermore issues at most $q_2$ *adaptive* key-generation queries, and in the case that a parameter equals the formal variable $\mathsf{poly}$ it is meant that there is no fixed bound (the only bound is the running time of the adversary that is polynomial). Thus, for example, if $q_1$ and $\ell$ are polynomials then $(q_1, \ell, \mathsf{poly})$-$\mathsf{SIM}$-Security means that the adversary in the $\mathsf{SIM}$-Security definition makes a $q_1(\lambda)$-bounded number of non-adaptive key-generation queries but an unbounded (i.e., bounded only by its running time) number of adaptive key-generation queries, and outputs a $\ell(\lambda)$-bounded challenge message vector, where $\lambda$ is the security parameter. If the parameters are not specified we intend them set to $\mathsf{poly}$. ($\mathsf{IND}$-Security is defined in Section 2, Definition 3. As reference for $\mathsf{SIM}$-Security, we take the definitions of [18] and [14].) We will also consider in our work the *selective security model* which is a weaker security model (see, e.g., [12, 30, 4]) in which the adversary must commit to its challenge messages before seeing the public parameters. Then, we will use the notation $\mathsf{sel}$-$\mathsf{atk}$ to mean $\mathsf{atk}$-Security in the selective model.

In the seminal work of Boneh, Sahai and Waters [14], it was shown that for FE, unlike classical encryption, $\mathsf{IND}$-Security is weaker than $\mathsf{SIM}$-Security. Indeed, the authors show a clearly insecure FE scheme that is provably IND-Secure. Moreover, in the same work Boneh *et al.* show that

$(0, \mathsf{poly}, 2)$-SIM-Security is *impossible* to achieve even for a simple functionality like IBE in the *non-programmable oracle model*, but prove, in the random oracle model, that $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-Security implies $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-SIM-Security for predicate encryption with public-index, and there exists an AIBE scheme that is $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-SIM-Secure.

At the same time, O'Neill [35] does similar considerations and shows that for *pre-image sampleable functionalities*, $(\mathsf{poly}, \mathsf{poly}, 0)$-IND-Security is equivalent to $(\mathsf{poly}, \mathsf{poly}, 0)$-SIM-Secure. Barbosa and Farshim [8] extended O'Neill's equivalence between indistinguishability and semantic security to the adaptive setting by restricting the adversary to issue adaptive key-generation queries for keys that are constant over the support of the message distribution. We will not consider any of such restrictions but we stress that our positive results are for a model that does not share these limitations.

Later, Bellare and O'Neill [11] showed that the impossibility result of [14] also extends to the standard model assuming the existence of collision resistant hash functions. Furthermore, they introduce new definitions to the aim of overcoming the impossibility results. Specifically, they define a new notion equivalent of IND-Security and thus incurring the same deficiency, and a new simulation-based definition for which a proof of security was only shown for functionalities with key space of polynomial size (and so not including basic functionalities like IBE). Recently, Iovino and Żebrowski [31] proved positive results for Simulation-based Security in the Random Oracle model.

In 2012, Gorbunov *et al.* [27] presented a construction of FE for general circuits that is $(q_1, \mathsf{poly}, 0)$-SIM-Secure. Following, Agrawal *et al.* [5] proved an impossibility result showing that it is *impossible* to achieve $(\mathsf{poly}, 1, 0)$-SIM-Security. Their result does not hold in the selective security model[1] and for public-index functionalities.

Furthermore, in the same paper, the authors prove that $(\mathsf{poly}, 1, \mathsf{poly})$-IND-Security implies $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-Security, and propose a simulation-based notion of security that considers computational *unbounded* simulator as a way to overcome current impossibility results, leaving many open problems about this definition. In 2013, Goldwasser *et al.* [24] presented an FE for general circuits with succinct ciphertexts (meaning that the size of the ciphertext does grow only with the respect of the depth of the circuits to be evaluated) provable $(q_1, \mathsf{poly}, 0)$-SIM-secure.

Later, De Caro *et al.* [18] presented a general compiler to transform any $(q_1, \ell, \mathsf{poly})$-IND-Secure FE scheme for circuits into one that is $(q_1, \ell, \mathsf{poly})$-SIM-Secure matching the known impossibility results. Finally, in recent breakthroughs, Gorbunov *et al.* and Garg *et al.* [29, 20] proposed $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-Secure constructions for predicate encryption with public-index for general circuits, and [19, 6, 16] proposed the first candidate constructions for a $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-IND-Secure[2] functional encryption scheme for general circuits from indistinguishable obfuscation and extractable obfuscation.

Concurrently and independently from our work, Agrawal *et al.* [2][3] studied new definitions for functional encryption. Although part of their work focuses on function privacy (another property not addressed in our work), one of the definitions it contains, therein called RELAX-AD-SIM, is similar in spirit to ours. Loosely speaking, in RELAX-AD-SIM, the simulator is allowed to run in unbounded time and make more queries than the adversary but in a controlled way. See the rest of the paper for a deeper discussion and comparison.

*Our Work: Rewinding Simulators.* Given the current state of the affair in functional encryption, as shown in the previous section, the reader can be then tempted to ask why a new definition should be considered in this already messy scenario. We believe then the quest for a reasonable simulation-based security definition is still open and that connections with secure computation and zero-knowledge are relevant to better understand, then clarify, what is happening in functional encryption.

For instance, in the context of secure computation, Backes *et al.* in [7] present a protocol that can be proven secure using a rewinding simulator and that is not secure for any non-rewinding simulator. More-

[1] [5] shows that their impossibility result holds in a variant of the selective security model, called by [18] *fully non-adaptive model*, where the adversary makes *simultaneous* key-generation and challenge message queries before seeing the public parameters.

[2] Precisely, the functional encryption scheme of [19] only achieves $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-sel-IND-Security but later [16] and [6] provided schemes that avoid the selective security model.

[3] Note that we do not refer to their latest eprint revision but at the specific version posted on 6 March 2014 that has been updated after and in the subsequent revisions represents an extended abstract of the paper appeared in [3].

over they show that stand-alone security (where rewinding simulators are allowed) do not coincide with the notion of security under concurrent composition whose security guarantees are relevant in practice.

With the above in mind, in this paper, we explore the benefits and the limits of using *efficient rewinding black-box simulators* to argue security for functional encryption as a way to overcome the known impossibility results.

Specifically, so far, all the known simulation-based security definitions for functional encryption share a common characteristic. They all constraint the simulator to learn exactly what the adversary learns in a single run of the experiment. This is enforced by requiring straight-line simulators and/or by having the challenger of the experiment tracing the queries issued by the adversary and reporting them in the output distribution of the experiment. This is true also for the BSW definition which nevertheless allows the simulator to rewind the adversary to reconstruct its view.

We, then, allow the simulator to learn not only what the adversary learns in a single run of the experiment but also what can be extracted by rewinding the adversary multiple times under the condition that: (1) the simulator must be efficient, (2) the simulator can not learn more than what the adversary would learn in any run of the experiment. All that is needed is for the simulator to present to the distinguisher, at the end of the interaction, a complete view of the adversary that is indistinguishable from the view the adversary produces in a single run of the experiment.

In particular, by rewinding, we mean that the simulator runs parts of the adversary during the simulation and produces a fragment of the conversation that has some desired property with a certain probability. For some functionalities, if the simulator fails then it possibly gains some additional information on the challenge messages useful to produce a successful simulation and then can rewind the adversary based on this new information.

*Does the rewinding simulator learn too much information?* A matter of concern regarding rewinding strategies could be that the simulator is leaking too much information. If the simulator could rewind the adversary to its liking, we would have the undesired situation that insecure schemes could be secure. Therefore, we have to constrain the power of the simulator: it must learn information but in a *controlled* way. We make this as follows. The simulator can rewind based on the adversary's queries. If those queries allow the adversary to learn information on the challenge messages, then the simulator learns this information by rewinding too. Otherwise, the simulator can simulate the view for the adversary easily, without learning much information.

We control the power of the simulator by allowing it to ask only queries that the adversary would ask during a valid run of the experiment. More concretely, consider the different constraints on the simulator in BSW and in our definition. In BSW, the simulator is given *direct* access to the functionality oracle and so to make the definition not trivial the list of the queries is put in the transcript (otherwise the simulator could just query the functionality oracle on the identity function to get the challenge message and simulate perfectly any scheme even *insecure* ones).

Instead, in our definition, when the adversary makes a query $k$, the simulator is invoked with the value $F(k, x)$, where $x$ is the challenge message, but the simulator can not ever ask a query for a key $k$ that the adversary would not ask in a run of the game. Is this sufficient? As sanity check, we show that, although the simulator has this extra power, the new definition still implies IND-Security. Nevertheless, it seems to not suffer from the problems of IND-Security (such as the existence of clearly insecure schemes that satisfy such definition).

In an independent and concurrent work, Agrawal *et al.* [2] formulated a new definition called RELAX-AD-SIM to the scope of bypassing the impossibility results for previous SIM-Security and of not being vulnerable to the weakness of IND-Security. Interestingly, both our definition and RELAX-AD-SIM share the same intuition and spirit. In RELAX-AD-SIM, the simulator can learn more information than the adversary but this leak is controlled in the following way (this is an oversimplification for the scope of our presentation, see their paper for details). Fix a value $\epsilon$ and consider the set of queries $Q_\epsilon$ that the adversary would ask with probability greater than $\epsilon$. Then, the simulator of RELAX-AD-SIM can ask any query in $Q_\epsilon$. Moreover, their simulator is allowed to run in time inversely proportional to $1/\epsilon$ and it is only required that the distinguisher can not have distinguishing advantage (between the real and ideal world) greater than $\epsilon$.

The reader may notice that this mechanism of giving extra power to the simulator in a constrained way is similar to ours. In fact, if our *efficient* simulator can learn some extra query by means of rewinding then it means that the adversary is likely to ask such query, and their simulator could make the same query as well.

*Efficient simulation with non-negligible distinguishing advantage.* A technical difference between our work and Agrawal *et al.* [2] is that Agrawal *et al.* allows the simulator to run in time polynomial in $1/\epsilon$ and thus it would run in super-polynomial time when $\epsilon$ is smaller than the inverse of any polynomial, whereas we stick to *efficient* simulation and impose a distinguishing advantage at most inverse of any polynomial. We remark that both works do not allow simulators that work for all $\epsilon$. Notwithstanding, in our work efficient simulation is sufficient to bypass the impossibility result of BSW.

*Our Results and Roadmap.* In Section 2 we present our notation and general definitions for FE. In Section 3, we put forth a weaker notion of simulation-based security that we call *rewinding simulation-based security* (RSIM, for short), that lies between SIM-Security and IND-Security. Our definition is a weakening of previous definitions proposed in literature (see [18]).

For completeness, in Section 4 we review the insufficiency of IND-Security as shown by [14].

In Section 5, we show that in the *standard model* for *efficient rewinding black-box simulators*, $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-IND-Security implies $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-RSIM-Security, for predicate encryption with public-index. This establishes an equivalence between $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-IND-Security and $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-RSIM-Security for predicate encryption with public-index. We can also show that in this case *composition* holds, meaning that single-message security implies many-massage security which is relevant in real scenarios.

To complete our analysis, we seek for settings where rewinding simulators are of no help. Thus, we answer the question of whether RSIM-Security *with negligible advantage* is achievable for general functionalities in the negative.

Specifically, in Section 6, we establish a lower bound showing that $(0,\mathsf{poly},1)$-RSIM-Security with negligible advantage can not be achieved for general functionalities.[4] No lower bounds were known in this setting. Our result, as that of [14, 11], is a trade-off.

It shows that RSIM-Security requires long secret keys, meaning that the total number of bits in messages securely encrypted must be bounded by the length of a secret key.

In Appendix C we also show positive results for RSIM-Security for some classes (not all) of Predicate Encryption with Private-Index, in particular in Theorem 10 we establish equivalence between $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-IND-Security and $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-RSIM-Security for the functionality Inner-Product over $\mathbb{Z}_2$.

## 2 Definitions

*Notation.* A *negligible* function $\mathsf{neg}(\lambda)$ is a function that is smaller than the inverse of any polynomial in $\lambda$. If $x_1$ and $x_2$ are binary strings, we denote by $x_1||x_2$ or $(x_1, x_2)$ their concatenation. If $X$ and $Y$ are two ensembles of random variables indexed by the security parameter $\lambda$, we say that $X \approx_\epsilon Y$ if no PPT distinguisher can distinguish them with advantage greater than $\epsilon(\lambda)$. We denote by $[n]$ the set $\{1,\ldots,n\}$. If $x$ is a binary string we denote by $|x|$ the bit length of $x$, we denote by $x_i$ the $i$-th bit of $x$, $1 \leq i \leq |x|$. PPT is a shorthand for Probabilistic Polynomial-Time. We denote by $A(x;r)$ the execution of a PPT algorithm $A$ with input $x$ and randomness $r$. Sometimes we simply write $A(x)$ instead of $A(x;r)$ when it is clear from the context. If $B$ is an algorithm and $A$ is an algorithm with access to an oracle then $A^B(\cdot)$ denotes the execution of $A$ with oracle access to $B(\cdot)$.

Following Boneh *et al.* [14], we start by defining the notion of functionality and then that of functional encryption scheme FE for functionality $F$.

**Definition 1** [Functionality] A *functionality* $F$ defined over $(K, X)$ is a function $F : K \times X \to \Sigma \cup \{\perp\}$ where $K$ is the *key space*, $X$ is the *message space* and $\Sigma$ is the *output space* and $\perp$ is a special string not contained in $\Sigma$. Notice that the functionality is undefined for when either the key is not in the key space or the message is not in the message space. Furthermore we require that there are efficient procedures to check membership of a string in the message space and key space and to sample from these spaces.

[4] Precisely, we show a stronger result that $(0,\mathsf{poly},1)$-RSIM-Security with negligible advantage is not achievable in the standard model in the auxiliary input setting (see Section 3). The auxiliary input setting has been already used by [11] in the same context.

**Definition 2** [Functional Encryption Scheme] A *functional encryption* (FE) scheme $\mathsf{FE}$ for functionality $F$ is a tuple $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ of 4 algorithms:

1. $\mathsf{Setup}(1^\lambda)$ outputs *public* and *master secret* keys $(\mathsf{Mpk}, \mathsf{Msk})$ for *security parameter* $\lambda$.
2. $\mathsf{KeyGen}(\mathsf{Msk}, k)$, on input a master secret key $\mathsf{Msk}$ and *key* $k \in K$ outputs *secret key* $\mathsf{Sk}_k$.
3. $\mathsf{Enc}(\mathsf{Mpk}, x)$, on input public key $\mathsf{Mpk}$ and *message* $x \in X$ outputs *ciphertext* $\mathsf{Ct}$;
4. $\mathsf{Dec}(\mathsf{Mpk}, \mathsf{Ct}, \mathsf{Sk}_k)$ outputs $y \in \Sigma \cup \{\perp\}$.

In addition we make the following *correctness* requirement: for all $(\mathsf{Mpk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^n)$, all $k \in K_n$ and $m \in M_n$, for $\mathsf{Sk} \leftarrow \mathsf{KeyGen}(\mathsf{Msk}, k)$ and $\mathsf{Ct} \leftarrow \mathsf{Enc}(\mathsf{Mpk}, m)$, we have that $\mathsf{Dec}(\mathsf{Mpk}, \mathsf{Ct}, \mathsf{Sk}) = F(k, m)$ whenever $F(k, m) \neq \perp$[5], except with negligible probability.

*The empty key.* For any functionality, we also assume that the key space contains a special *empty key* $\epsilon$ such that $F(\epsilon, x)$ gives the length of $x$ and (depending on the functionality) some intentionally leaked information on $x$ that can be easily extracted from an encryption of $x$. When $\boldsymbol{x} = (x_1, \ldots, x_\ell)$ is a vector of messages, for any $k \in K \cup \{\epsilon\}$, we denote by $F(k, \boldsymbol{x})$ the vector of evaluations $(F(k, x_1), \ldots, F(k, x_\ell))$.

*Secret-key length.* We say that a functional encryption scheme $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ has secret-key length $kl(\cdot)$ if $|\mathsf{Sk}| \leq kl(\lambda)$ for all $k \in K_\lambda$, $X \in X_\lambda$, all $(\mathsf{Mpk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda)$, and all $\mathsf{Sk} \leftarrow \mathsf{KeyGen}(\mathsf{Msk}, k)$. Note that every FE scheme must have some polynomial $kl(\cdot)$ secret-key length in order to be efficient.

*Indistinguishability-based Security.* The indistinguishability-based notion of security for functional encryption scheme $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for functionality $F$ defined over $(K, X)$ is formalized by means of the following game $\mathsf{IND}^{\mathsf{FE}}_{\mathcal{A}}$ between an adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ and a *challenger* $\mathcal{C}$.

1. $\mathcal{C}$ generates $(\mathsf{Mpk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ and runs $\mathcal{A}_0$ on input $\mathsf{Mpk}$;
2. $\mathcal{A}_0$, during its computation, issues $q_1$ *non-adaptive key-generation queries.* $\mathcal{C}$ on input key $k \in K$ computes $\mathsf{Sk} = \mathsf{KeyGen}(\mathsf{Msk}, k)$ and sends it to $\mathcal{A}_0$. When $\mathcal{A}_0$ stops, it outputs two *challenge messages vectors*, of length $\ell$, $\boldsymbol{x}_0, \boldsymbol{x}_1 \in X^\ell$ and its internal state $\mathtt{st}$.
3. $\mathcal{C}$ picks $b \in \{0, 1\}$ at random, and, for $i \in \ell$, computes the *challenge ciphertexts* $\mathsf{Ct}_i = \mathsf{Enc}(\mathsf{Mpk}, x_b[i])$. Then $\mathcal{C}$ sends $(\mathsf{Ct}_i)_{i \in [\ell]}$ to $\mathcal{A}_1$ that resumes its computation from $\mathtt{st}$.
4. $\mathcal{A}_1$, during its computation, issues $q_2$ *adaptive key-generation queries.* $\mathcal{C}$ on input key $k \in K$ computes $\mathsf{Sk} = \mathsf{KeyGen}(\mathsf{Msk}, k)$ and sends it to $\mathcal{A}_1$.
5. When $\mathcal{A}_1$ stops, it outputs $b'$.
6. **Output:** if $b = b'$, $F(\epsilon, \boldsymbol{x}_0) = F(\epsilon, \boldsymbol{x}_1)$, and $F(k, \boldsymbol{x}_0) = F(k, \boldsymbol{x}_1)$ for each $k$ for which $\mathcal{A}$ has issued a key-generation query, then output 1 else output 0.

The advantage of adversary $\mathcal{A}$ is: $\mathbf{Adv}^{\mathsf{FE},\mathsf{IND}}_{\mathcal{A}}(1^\lambda) = \mathrm{Prob}[\mathsf{IND}^{\mathsf{FE}}_{\mathcal{A}}(1^\lambda) = 1] - 1/2$

**Definition 3** We say that $\mathsf{FE}$ is $(q_1, \ell, q_2)$-*indistinguishably secure* ($(q_1, \ell, q_2)$-$\mathsf{IND}$-$\mathsf{Secure}$, for short) where $q_1 = q_1(\lambda), \ell = \ell(\lambda), q_2 = q_2(\lambda)$ are polynomials in the security parameter $\lambda$ that are fixed a priori, if all probabilistic polynomial-time adversaries $\mathcal{A}$ issuing at most $q_1$ non-adaptive key queries, $q_2$ adaptive key queries and output challenge message vectors of length and most $\ell$, have at most negligible advantage in the above game. (Notice that, if $q_1$ (resp. $q_2$) is equal to $\mathsf{poly}$, then the interpretation is that there is no bound on the number of non-adaptive (resp. adaptive) key-generation queries and if $\ell = \mathsf{poly}$ there is no bound on the length of the challenge message vector).

*Predicate Encryption (PE, for short).* Those schemes are defined for functionalities whose message space $X$ consists of two sub-spaces $I$ and $M$ called respectively *index space* and *payload space.* In this case, the functionality $F$ is defined in terms of a polynomial-time predicate $P : K \times I \to \{0, 1\}$ as follows: $F(k, (\mathsf{ind}, \mathsf{m})) = \mathsf{m}$ if $P(k, \mathsf{ind}) = 1$, $\perp$ otherwise, where $k \in K$, $\mathsf{ind} \in I$ and $\mathsf{m} \in M$. In particular, for the $\epsilon$ key, we have $F(\epsilon, (\mathsf{ind}, \mathsf{m})) = (|\mathsf{ind}|, |\mathsf{m}|)$. As for general functionalities, a predicate $P$ can be a family of predicates and in this case the functionality $F$ defined in terms of $P$ is a family of functions. Indistinguishable Security for PE is defined analogously to Definition 3.

[5] See [11, 1] for a discussion about this condition.

*Anonymous IBE (AIBE, for short).* Let the key space $K_n = \{0,1\}^n$, index space $I_n = \{0,1\}^n$ and payload space $M_n = \{0,1\}^n$ the payload space for $n \in \mathbb{N}$. The predicate family $\mathsf{IBE} = \{\mathsf{IBE}_n : K_n \times I_n \rightarrow \{0,1\}\}_{n \in \mathbb{N}}$ is defined so that for any $k \in K_n, \mathsf{ind} \in I_n$, $\mathsf{IBE}(k, \mathsf{ind}) = 1$ if and only if $k = \mathsf{ind}$. We call a predicate encryption scheme (with private-index) for this predicate *Anonymous IBE.*

*Predicate Encryption with Public-Index (a.k.a. ABE) (PIPE, for short).* In this type of FE the empty key $\epsilon$ explicitly reveals the index $\mathsf{ind}$, namely $F(\epsilon, (\mathsf{ind}, \mathsf{m})) = (\mathsf{ind}, |\mathsf{m}|)$. Indistinguishable security is defined again analogously to Definition 3, with the main difference being in the adversary's challenge, namely it consists of two *payloads* $\mathsf{m}_0, \mathsf{m}_1$ and an index $\mathsf{ind}$. An example of PIPE is *Identity-based Encryption.*

An example of PIPE is IBE that is identical to AIBE except that the "identity is in clear".

## 3 Rewinding Simulation-based Security

In this section, we present our *rewinding simulation-based security* definition.

**Definition 4** [Rewinding Simulation-based Security] Let $q_1 = q_1(\lambda), \ell = \ell(\lambda), q_2 = q_2(\lambda)$ be specific polynomials in the security parameter $\lambda$ that are fixed a priori or be equal to the formal variable $\mathsf{poly}$.

A functional encryption scheme $\mathsf{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ for functionality $F$ defined over $(K, X)$ is $(q_1, \ell, q_2)$-*rewinding simulation-secure* ($(q_1, \ell, q_2)$-RSIM-Secure, for short), if for any polynomial $\nu(\lambda)$ there exists a PPT *simulator* algorithm $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ such that for all PPT *adversary* algorithms $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$, issuing at most $q_1$ non-adaptive key-generation queries, $q_2$ adaptive key-generation queries and output challenge message vector of length and most $\ell$, no PPT distinguisher can distinguish the outputs of the following two experiments $\mathsf{RealExp}^{\mathsf{FE}, \mathcal{A}}$ and $\mathsf{IdealExp}^{\mathsf{FE}, \mathcal{A}}$ with advantage greater than $1/\nu(\lambda)$.

(Note that, if $q_1$ (resp. $q_2$) is set to $\mathsf{poly}$, then the interpretation is that there is no bound on the number of non-adaptive (resp. adaptive) key-generation queries and if $\ell = \mathsf{poly}$ there is no bound on the length of the challenge message vector).

```
$\mathsf{RealExp}^{\mathsf{FE},\mathcal{A}}(1^\lambda)$

$(\mathsf{Mpk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda);$
$(\boldsymbol{x}, \mathtt{st}) \leftarrow \mathcal{A}_0^{\mathsf{KeyGen}(\mathsf{Msk},\cdot)}(\mathsf{Mpk});$
$(\mathsf{Ct}_i \leftarrow \mathsf{Enc}(\mathsf{Mpk}, x[i]))_{i \in \ell};$
$\alpha \leftarrow \mathcal{A}_1^{\mathsf{KeyGen}(\mathsf{Msk},\cdot)}(\mathsf{Mpk}, (\mathsf{Ct}_i), \mathtt{st});$
**Output:** $(\mathsf{Mpk}, \boldsymbol{x}, \alpha)$

$\mathsf{IdealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{A}}(1^\lambda)$

$(\mathsf{Mpk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda);$
$(\boldsymbol{x}, \mathtt{st}) \leftarrow \mathcal{A}_0^{\mathsf{KeyGen}(\mathsf{Msk},\cdot)}(\mathsf{Mpk});$
Let $\mathcal{Q} = (k_i, \mathsf{Sk}_{k_i}, F(k_i, \boldsymbol{x}))_{i \in [q_1]}.$
$\alpha \leftarrow \mathsf{Sim}_0^{\mathcal{A}_1^{\mathcal{O}}(\mathsf{Mpk},\cdot,\mathtt{st})}(\mathsf{Mpk}, \mathsf{Msk}, \mathcal{Q}, F(\epsilon, \boldsymbol{x}));$
**Output:** $(\mathsf{Mpk}, \boldsymbol{x}, \alpha)$
```

Here, the $(k_i \in K)_{i \in [q_1]}$'s are the $q_1$ keys for which $\mathcal{A}_0$ has issued a *non-adaptive* key-generation query to its key-generation oracle. In the ideal experiment $\mathcal{A}_1$ is provided with a special oracle $\mathcal{O}$ for adaptive key-generation queries. The oracle $\mathcal{O}$ takes in input a key $k \in K$ and answers the query in the following way. The oracle invokes the simulator $\mathsf{Sim}_1$ on input $(k, F(k, \boldsymbol{x}))$. $\mathsf{Sim}_1$ outputs a secret key for $k$ that the oracle returns to $\mathcal{A}_1$. We require the simulator $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ to be stateful and allow $\mathsf{Sim}_0$ and $\mathsf{Sim}_1$ to communicate by means of a shared memory. We remark that each time $\mathsf{Sim}_0$ runs the adversary $\mathcal{A}_1$ on some input $(\mathsf{Ct}_i)$, $\mathcal{A}_1$ is executed with input $(\mathsf{Mpk}, (\mathsf{Ct}_i), \mathtt{st})$ and *fresh* randomness.

$\mathsf{RSIM}$-*Security with negligible advantage.* With the obvious meaning, we say that $\mathsf{FE}$ is $\mathsf{RSIM}$-Secure with negligible advantage if in the above definition the two experiments are computationally indistinguishable, i.e. whether the function $\nu(\lambda)$ is negligible. Moreover, the definition could be generalized making it parameterized by a generic function $\nu(\lambda)$, but for our scopes this is not possible.[6] In fact, we focus on efficient simulation, and for this reason the function $\nu(\lambda)$ can *not* be set to a negligible function (see the paragraph 'The actual simulation' in theorem for an explanation). Instead, if the function $\nu(\lambda)$ is the inverse of an arbitrary polynomial, we can achieve efficient simulation. As said in the introduction, simulators with non-negligible advantage are also used in [2].

*Auxiliary Inputs.* Our definition can be extended naturally to the auxiliary input setting, as in Bellare and O'Neill [11]. An auxiliary input generator algorithm $Z$ outputs $z$ which is given to the adversary and simulator, and included in the output distribution of security game. Notice that, the simulator is not allowed to pick $z$. As in [11], the auxiliary input setting will be used in our impossibility result in Section 6, where $z$ will contain a key for a collision-resistant hash function.

*Selective Security.* The selective security model is a weaker model in which the adversary must commit to its challenge messages before seeing the public parameters. In particular, for $\mathsf{RSIM}$, in the ideal experiment the simulator will simulate also the answers to the non-adaptive key queries. Due to space constraints, we defer the selective $\mathsf{RSIM}$-Security definition to the full version of this work at the IACR eprint archive.

*Relations among Definitions.* Our $\mathsf{RSIM}$ definition stands in between $\mathsf{SIM}$ and $\mathsf{IND}$ security. Specifically, it is easy to see that $\mathsf{SIM}$ implies $\mathsf{RSIM}$ because the $\mathsf{RSIM}$ simulator simply runs the $\mathsf{SIM}$ simulator. Moreover, we show in Appendix A that $\mathsf{RSIM}$-Security implies $\mathsf{IND}$-Security.

*Composition.* Despite the fact that the $\mathsf{RSIM}$ simulator can rewind the adversary $\mathcal{A}_1$ to reconstruct its view (this in general is problematic for composition), we can show that for the class of functionalities for which we prove the equivalence between $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-$\mathsf{IND}$-Security and $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-$\mathsf{RSIM}$-Security, single-message $\mathsf{RSIM}$-Security implies multiple-message $\mathsf{RSIM}$-Security, namely $(\mathsf{poly},1,\mathsf{poly})$-$\mathsf{RSIM}$-Security implies $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-$\mathsf{RSIM}$-Security. This is because, $(\mathsf{poly},1,\mathsf{poly})$-$\mathsf{RSIM}$-Security implies $(\mathsf{poly},1,\mathsf{poly})$-$\mathsf{IND}$-Security (by Theorem 7 in Appendix A) and $(\mathsf{poly},1,\mathsf{poly})$-$\mathsf{IND}$-Security implies $(\mathsf{poly},\mathsf{poly},\mathsf{poly})$-$\mathsf{IND}$-Security (this was shown by [28]).

*Relations with BSW [14]* First of all, in BSW the simulator is allowed to pick its own simulated public-key and non-adaptive secret keys, whereas in our definition this information is generated honestly. Notice that the main aim of BSW was to prove *impossibility* results and such results are stronger if they hold with respect to more powerful simulators (i.e., simulators that can also simulate the public- and secret- keys). On the other hand, we are mainly interested to prove *positive* results and so our choice of the definition (i.e. the fact that the public- and non-adaptive secret- keys are generated honestly) makes our results stronger.

To clarify the difference with the [14] definition, we recall the [14, 11]'s impossibility result and show why it does not hold for $\mathsf{RSIM}$.

Specifically, consider the following adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ for the IBE functionality. $\mathcal{A}_0$ returns as challenge messages the vector $((0, r_i))_{i\in[\ell]}$, where $\ell = kl(\lambda) + \lambda$, $kl$ is a polynomial bounding the length of secret keys, 0 is the identity and $r_i$ is a random bit for each $i \in [\ell]$.

Then, $\mathcal{A}_1$ invokes its key-generation oracle on input identity $w = \mathsf{CRHF}(\mathsf{Mpk}||\mathsf{Ct}_1|| \cdots ||\mathsf{Ct}_\ell)$ for some collision-resistant hash function $\mathsf{CRHF}$, and then asks to see a secret key for identity 0. The output of $\mathcal{A}_1$ is the transcript of its entire computation including its inputs. Thus, the strategy of the above adversary forces the simulator to commit to the challenge ciphertexts he has generated (through the query on identity $w$) before seeing the evaluation of IBE functionalities on the key for identity 0 and so learning the bit $r_i$'s.

Then, the challenge ciphertexts can not be reprogrammed and by choosing the number of encrypted bits to be larger than the length of the secret key the simulator is forced to achieve an information theoretic compression of random bits which is in turn impossible. Notice that, even though the simulator

[6] Precisely, it would be possible at the cost of non-efficient simulation.

in BSW definition is formally allowed to rewind the adversary, the same simulator is not allowed to learn more information than what is learnt by the adversary in a single run of experiment.

This, in turn, means that the only way for the simulator to reconstruct the view of the adversary is to break the collision resistant hash function. The strategy of this adversary is clearly not successful with the respect to $\mathsf{RSIM}$ because an $\mathsf{RSIM}$ simulator once obtained the $r_i$'s can simply generates new ciphertexts encrypting them and rewind the adversary. In the new run, the $\mathsf{RSIM}$ simulator can answer all the key-generation queries by simply generating honest secret keys.

Observe that the BSW definition forbids this kind of simulation since: (1) the simulator is given direct access to the functionality oracle and (2) the key-generation queries issued by the simulator are given as input to the distinguisher. So according to the BSW definition, the distinguisher would see 4 key-generation queries, and thus it could tell apart the real experiment where the adversary always asks 2 secret keys from the ideal experiment.

The same holds for the [8] definition. On the other hand, in $\mathsf{RSIM}$ the distinguisher would only see the *last* transcript. The definitions of [5, 18] also forbid this kind of simulation since their simulator is straight-line.

## 4 Insufficiency of Indistinguishable-Based Security for PE

In this section, we review the insufficiency of IND-Security shown by [14]. Precisely, we show that the *equivalence* between indistinguishable-based security and simulation-based security in the standard model for *public-index predicate encryption* is the best one can hope for.

Namely, we will show a complex predicate for which is possible to construct a *private-index predicate encryption* scheme that is $\mathsf{IND}$-Secure, but it is clearly insecure in practice. For completeness, we slightly adapt the example of [14] extend it to a PE scheme (with private-index) that also encrypts a payload message. Specifically, let $\pi$ be a *one-way permutation* and consider the predicate $P$ defined as follows: $P(k, \mathsf{ind}) = 1$ if $k = \pi(\mathsf{ind})$, 0 otherwise. Now, consider the following implementation based on an Anonymous IBE scheme $\mathsf{AIBE}$. The ciphertext for the pair $(\mathsf{ind}, \mathsf{m})$ is an $\mathsf{AIBE}$'s ciphertext for identity $\pi(\mathsf{ind})$ and payload $\mathsf{m}||\mathsf{ind}$, namely $\mathsf{AIBE.Enc}(\mathsf{AIBE.Mpk}, \pi(\mathsf{ind}), \mathsf{m}||\mathsf{ind})$. and a secret key for $k$ is an $\mathsf{AIBE}$'s secret key for identity $k$, namely $\mathsf{AIBE.KeyGen}(\mathsf{AIBE.Msk}, k)$.

Clearly, given ciphertext for pair $(\mathsf{ind}, \mathsf{m})$ and secret key for $k$ such that $\pi(\mathsf{ind}) = k$ reveal more information than need about the index $\mathsf{ind}$.

Nevertheless, the new scheme can be proved $\mathsf{IND}$-Secure. In fact, according to the constraints of the $\mathsf{IND}$-Security, for challenge indices $\mathsf{ind}_0, \mathsf{ind}_1$, $P(k, \mathsf{ind}_0) = P(k, \mathsf{ind}_1) = 1$ if and only if $\pi(\mathsf{ind}_0) = k$ and $\pi(\mathsf{ind}_1) = k$ but this happens if and only if $\mathsf{ind}_0 = \mathsf{ind}_1$. Thus, the adversary can only issue secret key queries for $k$ such that $P(k, \mathsf{ind}_0) = P(k, \mathsf{ind}_1) = 0$. Under this constraint it is easy to reduce the $\mathsf{IND}$-Security of the new scheme to that of the $\mathsf{AIBE}$ scheme. On the other hand, the new scheme is not $\mathsf{SIM}$-Secure (with the respect to any simulation-based security definition for FE known in the literature). Specifically, consider an adversary for the SIM-Security that chooses as challenge a pair $(\mathsf{ind}, \mathsf{m})$ for random index $\mathsf{ind}$ and random message $\mathsf{m}$, and then asks the secret key for $k = \pi(\mathsf{ind})$, decrypts the challenge ciphertext using this secret key and outputs a transcript of its computation. The simulator has to generate an indistinguishable transcript only given as input $F(\epsilon, (\mathsf{ind}, \mathsf{m})) = (|\mathsf{ind}|, |\mathsf{m}|)$, $k = \pi(\mathsf{ind})$, the secret for $k$ and the evaluation of the functionality on the challenge message, that in this case is just $\mathsf{m}$ being $\pi(\mathsf{ind}) = k$. Since $\mathsf{ind}$ is chosen at random and independently from the view of the simulator, the probability of successful simulation reduces to inverting $\pi$.

## 5 An Equivalence for Public-Index Schemes

In this section, we show that for public-index schemes $\mathsf{IND}$-Security is equivalent to $\mathsf{RSIM}$-Security. In Appendix A, we have already shown that $\mathsf{RSIM}$-Security implies $\mathsf{IND}$-Security, therefore, the main theorem of this section is the following.

**Theorem 5** Let $\mathsf{PIPE}$ be an $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{IND}$-$\mathsf{Secure}$ PE scheme with public-index for predicate $P : K \times I \to \{0, 1\}$. Then, $\mathsf{PIPE}$ is $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{RSIM}$-Secure as well.

*Overview.* To give some intuitions on the proof strategy, let us start by considering a weak adversary that issues only key-generation queries for keys that can not be used to decrypt any of the challenge ciphertexts. In such a case, the simulator will generates challenge ciphertexts for random payloads and for the indices that the simulator gets in input (recall that we are considering public-index functionalities).

It is clear that under the IND-Security of the PIPE scheme the adversary can not notice any difference given the fact that all the requested secret keys can not decrypt any of the challenge ciphertexts. Now, let consider an adversary that issue, after having seen the challenge ciphertexts, a key-generation query for a key that decrypts one of the these challenge ciphertexts, let say $\mathsf{Ct}_i$ (Notice that up to this moment the simulation is perfect under the IND-Security of the PIPE scheme).

Because the challenge ciphertexts were made for random payloads, the output of the Dec algorithm will be different from what the adversary expects, meaning that the simulation of current run is not successful. But now notice that the simulator learns the payload corresponding to $\mathsf{Ct}_i$, $\mathsf{m}_i$. Then, the simulator can executes a new run of the adversary generating $\mathsf{Ct}_i$ for the correct payload. Notice that, from now, no key-generation query for a key that decrypts $\mathsf{Ct}_i$ will not cause a rewind anymore.

*Remark.* The above is an oversimplified sketch. In fact, if the simulator follows this strategy it produces a biased output. Anyway, henceforth, we prefer to first present the simplified simulation and then explain why is biased and finally we proceed to fix it. We think that presenting the security reductions in this way helps the reader in understating the need of all the details. We will follow this approach for the rest of the work.

We prove Theorem 5 in Appendix B. In Appendix C.1 we also extend such theorem to the case of Anonymous IBE.

## 6 Impossibility of RSIM for FE for General Circuits

In this section, we show that RSIM-Security with negligible advantage can not be achieved in the adaptive setting for general circuits.

**Theorem 6** Assuming the existence of collision resistance hash functions and pseudo-random functions, there exists a family of circuits for which there are no functional encryption schemes that are $(0, \mathsf{poly}, 1)$-RSIM-Secure with negligible advantage in the auxiliary input setting (for the standard model).

*Overview.* To prove the theorem, it is enough to present an adversary whose strategy is such that at any run the simulator is forced to rewind, meaning that the information gathered in any run are useless to successfully simulate any other run. To force the rewind, our adversary will use a [14, 11]-like strategy. Namely, our adversary will first force the simulator to commit to the challenge ciphertexts he has generated by using a collision resistant hash function. Then, our adversary will request to see a secret key that extracts from the challenge ciphertexts a (pseudo-)random string whose length is much larger then the length of the secret key itself. Because it is information-theoretical impossible to compress such (pseudo-)random string in the space provided by the secret key, the simulator will rewind hoping to use the information gathered so far to successfully simulate the next run.

Now notice that in the [14, 11]'s impossibility results for the IBE functionality, only the first run can not be successfully simulated. In fact, in the the same run the simulator learns the challenge messages, which remains the same in all the runs, and can successfully simulate the next run. Thus, the IBE functionality is of limited use. Therefore, we have to consider a functionality that let the adversary extracts a pseudo-random string from the challenge ciphertexts, this is to invoke the information-theoretical argument that will force the simulator to rewind, and at the same time makes this string useless to simulate the next run, meaning that the output of the functionality crucially depends on the challenge ciphertexts generated by the simulator. Here is where the pseudo-random functions come in.

In more details, we consider an adversary that issues a suitable number of challenge messages, let us say $kl(\lambda) + \lambda$, where $kl(\cdot)$ is the polynomial bounding the length of the secret keys, of the type $(s||r_i)_{i\in[\ell]}$ where $s$ will be the seed of the pseudo-random function and $r_i$ a random value that will be part of the input on which the pseudo-random function will be evaluated. Then the adversary, on input $\mathsf{Mpk}$ and the ciphertexts $(\mathsf{Ct}_i)_{i\in[\ell]}$ for the challenge messages, issues a single adaptive key-generation query to its oracle for the circuit $C^{\mathsf{PRF},w}$ that computes the pseudo-random function on input seed $s$ and value $r||w$, where $w = \mathsf{CRHF}(\mathsf{Mpk}||\mathsf{Ct}_1||\cdots||\mathsf{Ct}_\ell)$ is hardwired in $C^{\mathsf{PRF},w}$ and is used to commit the simulator to the ciphertexts it has generated. Crucial is the fact that the output of $C^{\mathsf{PRF},w}$ on the challenge messages depends on the $\mathsf{Ct}_i$'s.

*Proof.* Le $\mathsf{FE}$ be a $(0, 1, \mathsf{poly})$-$\mathsf{RSIM}$-Secure with negligible advantage functional encryption scheme for circuits with secret-key length $kl(\cdot)$. Let $\mathsf{PRF} = \{\mathsf{PRF}_\lambda : \{0,1\}^\lambda \times \{0,1\}^{2 \cdot m(\lambda)} \to \{0,1\}\}_{\lambda \in \mathbb{N}}$ a circuit family of pseudo-random functions. Let $\mathsf{CRHF}$ be the collision resistance hash function with range $m(\lambda)$ whose key $\mathsf{hk}$ has been chosen by the auxiliary input generator. We omit $\mathsf{hk}$ in the notation just for the sake of simplicity.

For ease of presentation, henceforth, we simply talk about $\mathsf{RSIM}$-Security without specifying that it is with respect with negligible advantage. Consider the following adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ and distinguisher $\mathcal{D}$ in the $(0, 1, \mathsf{poly})$-$\mathsf{RSIM}$ security experiment. Specifically, $\mathcal{A}$ works as follows:

- $\mathcal{A}_0$ returns $\ell = kl(\lambda) + \lambda$ challenge messages of the form $(s||r_i)$ for random $s \in \{0,1\}^\lambda$ and $r_i \in \{0,1\}^{m(\lambda)}$.
- $\mathcal{A}_1$, on input $\mathsf{Mpk}$, $(\mathsf{Ct}_i)_{i\in[\ell]}$ and $\mathtt{st}$, sets $w = \mathsf{CRHF}(\mathsf{Mpk}||\mathsf{Ct}_1||\cdots||\mathsf{Ct}_\ell)$ and invokes the key-generation oracle on input the circuit $C^{\mathsf{PRF},w}(s, r) := \mathsf{PRF}(s, r||w)$ , and obtains secret key $\mathsf{Sk}$ for it. Finally, $\mathcal{A}_1$ outputs $\alpha = ((\mathsf{Ct}_i)_{i\in[\ell]}, w, \mathsf{Sk})$.

Instead, the distinguisher $\mathcal{D}$ does the following:

- $\mathcal{D}$, on input $\mathsf{Mpk}$, the challenge messages $(s||r_i)_{i\in[\ell]}$ and $\alpha$, interprets $\alpha$ as $\alpha = ((\mathsf{Ct}_i)_{i\in[\ell]}, w, \mathsf{Sk})$ and checks that (1) $w$ is equal to $\mathsf{CRHF}(\mathsf{Mpk}||\mathsf{Ct}_1||\cdots||\mathsf{Ct}_\ell)$, and (2) $\mathsf{Dec}(\mathsf{Mpk}, \mathsf{Ct}_i, \mathsf{Sk}) = \mathsf{PRF}(s, r_i||w)$ for each $i \in [\ell]$. $\mathcal{D}$ returns 1 if all the checks passed, 0 otherwise.

Because we assumed $\mathsf{FE}$ to be $(0, 1, \mathsf{poly})$-$\mathsf{RSIM}$-Secure, it means there exists a simulator $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ that generates a view indistinguishable to that of $\mathcal{A}$ when it plays in the real game. Given this simulator, we now construct an adversary $\mathcal{A}$ against the security of the pseudo-random function. Specifically, $\mathcal{A}$ on input the security parameter $1^\lambda$ and given access to oracle $\mathcal{O}$ does the following:

$\mathcal{A}^{\mathcal{O}}(1^\lambda)$:

1. $\mathcal{A}$ invokes the setup algorithm of $\mathsf{FE}$ to generate master public and secret key. Namely, $(\mathsf{Mpk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda)$.
2. Let $\ell = kl(\lambda) + \lambda$. Then $\mathcal{A}$ chooses random $r_i \in \{0,1\}^{m(\lambda)}$ for $i \in [\ell]$, as $\mathcal{A}_0$ does.
3. $\mathcal{A}$ runs $\mathsf{Sim}_0$ on input $(\mathsf{Mpk}, \mathsf{Msk}, \mathcal{Q}, (F(\epsilon, (s||r_i)))_{i\in[\ell]})$, where $\mathcal{Q}$ is empty because $\mathcal{A}_0$ does not issue any key-generation query. When $\mathcal{A}_1$ invokes its key-generation oracle on input circuit $C^{\mathsf{PRF},w}$, $\mathcal{A}$ invokes $\mathsf{Sim}_1$ on input $C^{\mathsf{PRF},w}$ and $(\mathcal{O}(r_i||w))_{i\in[\ell]}$ as input.
   At some point $\mathsf{Sim}_0$ returns $\alpha$.
4. Finally, $\mathcal{A}$ does the same checks as $\mathcal{D}$. Namely, $\mathcal{A}$ interprets $\alpha$ as $\alpha = ((\mathsf{Ct}_i)_{i\in[\ell]}, w, \mathsf{Sk})$ and checks that
   (a) $w$ is equal to $\mathsf{CRHF}(\mathsf{Mpk}||\mathsf{Ct}_1||\cdots||\mathsf{Ct}_\ell)$, and
   (b) $\mathsf{Dec}(\mathsf{Mpk}, \mathsf{Ct}_i, \mathsf{Sk}) = \mathcal{O}(r_i||w)$ for each $i \in [\ell]$.
   $\mathcal{A}$ returns 1 if all the checks passed, 0 otherwise.

Now observe that, $\mathcal{D}$ outputs 1 with overwhelming probability when given the output of adversary $\mathcal{A}$ in the $(0, 1, \mathsf{poly})$-$\mathsf{RSIM}$ real experiment. Moreover, by the $(0, 1, \mathsf{poly})$-$\mathsf{RSIM}$-Security of $\mathsf{FE}$, $\mathcal{D}$ also output 1 with overwhelming probability when given the output of the simulator $\mathsf{Sim}$. Then, if $\mathcal{O}$ is the pseudo-random oracle for random seed $s$, $\mathcal{A}$ perfectly simulates the output of $\mathsf{Sim}$ in the $(0, 1, \mathsf{poly})$-$\mathsf{RSIM}$ ideal experiment and thus $\mathcal{A}$ gives in output 1 with high probability.

Suppose now that $\mathcal{O}$ is a truly random oracle. Let $\alpha = ((\mathsf{Ct}_i)_{i\in[\ell]}, w, \mathsf{Sk})$ be the output of $\mathsf{Sim}$ during the execution of $\mathcal{A}$ (see point (3) in the description of $\mathcal{A}$). We distinguish two mutually exclusive cases.

1. $\mathcal{A}_1$ *has never ever issued a key-generation query for circuit* $C^{\mathsf{PRF},w}$. In this case the probability that $\mathcal{A}$ outputs 1 is negligible since the output of the simulator is independent from $\mathcal{O}(r_i||w)$ for each $i \in [\ell]$ and these values are random being $\mathcal{O}$ a truly random oracle.
2. $\mathcal{A}_1$ *invoked its key-generation oracle on input the circuit* $C^{\mathsf{PRF},w}$ *at least one time.* First, notice that $\mathcal{A}$ implements the interface between $\mathcal{A}_1$ and $\mathsf{Sim}$. Precisely, when $\mathsf{Sim}_0$ invokes its oracle on some input, then $\mathcal{A}$ invokes $\mathcal{A}_1$ on the same input. Then, when $\mathcal{A}_1$ issues a key-generation query for a circuit $C^{\mathsf{PRF},w}$, $\mathcal{A}$ sees the value $w$ and answers such query as described above.

   Let $p(\lambda)$ be the running time of $\mathsf{Sim}$. Therefore, the execution of $\mathsf{Sim}$ can be divided in at most $p(\lambda)$ *runs*, where for $j = 1, \ldots, p(\lambda)$, in the $j$-th run $\mathsf{Sim}_0$ invokes its oracle on input $(\mathsf{Ct}_i^j)_{i\in[\ell]}$ that corresponds to a key-generation query for circuit $C^{\mathsf{PRF},w_j}$, where $w_j = \mathsf{CRHF}(\mathsf{Mpk}||\mathsf{Ct}_1^j|| \ldots ||\mathsf{Ct}_\ell^j)$. Now notice that there exists some index $k \leq p(\lambda)$ such that $w = w_k$ and $k$ is the first index for which $w = w_k$. From this fact and from the fact that $\mathcal{A}$ checks whether $w = \mathsf{CRHF}(\mathsf{Mpk}||\mathsf{Ct}_1|| \ldots ||\mathsf{Ct}_\ell)$, it follows that with all but negligible probability $(\mathsf{Ct}_i) = (\mathsf{Ct}_i^k)$. Indeed, suppose towards a contradiction that with non-negligible probability $q$ it holds that $(\mathsf{Ct}_i) \neq (\mathsf{Ct}_i^k)$. Then, $\mathcal{A}_1$ and $\mathsf{Sim}$ can be used to build an adversary $\mathcal{B}$ for $\mathsf{CRHF}$ as follows. $\mathcal{B}$ on input the security parameter $1^\lambda$ and the hash key $\mathsf{hk}$ does the following:

   $\mathcal{B}(\mathsf{hk})$:

   (a) $\mathcal{B}$ invokes the setup algorithm of $\mathsf{FE}$ to generate master public and secret key, namely $(\mathsf{Mpk}, \mathsf{Msk}) \leftarrow \mathsf{Setup}(1^\lambda)$, Then, $\mathcal{B}$ initializes a list $L$ to empty and set a global index $j$ to zero. The list $L$ is used by $\mathcal{B}$ to trace the invocations to $\mathcal{A}_1$ made by $\mathsf{Sim}_0$.

   (b) Let $\ell = kl(\lambda) + \lambda$. Then $\mathcal{B}$ chooses random $r_i \in \{0,1\}^{m(\lambda)}$ for $i \in [\ell]$, as $\mathcal{A}_0$ does.

   (c) $\mathcal{B}$ runs $\mathsf{Sim}_0$ on input $(\mathsf{Mpk}, \mathsf{Msk}, \mathcal{Q}, (F(\epsilon, (s||r_i)))_{i\in[\ell]})$, where $\mathcal{Q}$ is empty because $\mathcal{A}_0$ does not issue any key-generation query. When $\mathcal{A}_1$ is invoked on input ciphertexts $(\mathsf{Ct}_i^j)_{i\in[\ell]}$ then $\mathcal{B}$ put an entry in the list $L$ corresponding to

   $$\left((\mathsf{Ct}_i^j)_{i\in[\ell]}, w_j = \mathsf{CRHF}(\mathsf{Mpk}||\mathsf{Ct}_1^j|| \ldots ||\mathsf{Ct}_\ell^j)\right),$$

   and increment the global index $j$ by one. Then, when $\mathcal{A}_1$ invokes its key-generation oracle on input circuit $C^{\mathsf{PRF},w_j}$, $\mathcal{B}$ invokes $\mathsf{Sim}_1$ on input $C^{\mathsf{PRF},w_j}$ and $(\mathsf{PRF}(s, r_i||w_j))_{i\in[\ell]}$ as input.
   At some point $\mathsf{Sim}_0$ returns $\alpha$.

   (d) At this point, $\mathcal{B}$ interprets $\alpha$ as $\alpha = ((\mathsf{Ct}_i)_{i\in[\ell]}, w, \mathsf{Sk})$ and looks up in the list $L$ for the first index $k$ such that $w_k = w$. If $\mathcal{B}$ does not find this index it aborts, otherwise $\mathcal{B}$ returns the pair $((\mathsf{Mpk}||\mathsf{Ct}_1^k|| \ldots ||\mathsf{Ct}_\ell^k),\ (\mathsf{Mpk}||\mathsf{Ct}_1|| \ldots ||\mathsf{Ct}_\ell))$ as its collision.

   It is easy to see that the probability that $\mathcal{B}$ finds a collision is exactly $q$.

   Finally, notice that, when $\mathsf{Sim}_0$ invokes $\mathcal{A}_0$, its view is independent from the values $\mathcal{O}(r_i||w)$'s. This is because, being $\mathcal{O}$ a truly random oracle, for any $j < k$, $w_j \neq w_k = w$ and thus the values $\mathcal{O}(r_i||w_j)$'s are randomly and independently chosen from the values $\mathcal{O}(r_i||w)$'s. Thus, the tuple of ciphertexts $(\mathsf{Ct}_i)_{i\in[\ell]}$ is independent from the tuple $(\mathcal{O}(r_i||w))_{i\in[\ell]}$: we call this Fact 1.

   We now bound the probability of the following event $\mathsf{E}$ which is defined to be the event that for any $i \in [\ell]$, $\mathsf{Dec}(\mathsf{Mpk}, \mathsf{Ct}_i, \mathsf{Sk}) = \mathcal{O}(r_i||w)$, where the probability is taken over the random choices of $\mathcal{A}$ (and thus of $\mathcal{A}_1$ and $\mathsf{Sim}$) and of the oracle $\mathcal{O}$. In what follows we define $Y$ to be the event that

$\forall i \in [\ell]$ $\mathsf{Dec}(\mathsf{Mpk}, \mathsf{Ct}_i, \mathsf{Sk}) = \mathcal{O}(r_i||w)$ and we call fact 2 the fact that $\mathcal{O}$ is an oracle implementing a truly random function.

$$\begin{aligned}\Pr[\,\mathsf{E}\,] &\le \Pr[\,\exists\ \mathsf{Sk} : |\mathsf{Sk}| = kl(\lambda) \text{ and } Y\,] \\ &\le \sum_{\mathsf{Sk}\in\{0,1\}^{kl(\lambda)}} \Pr[\,Y\,] \text{ (by the union bound)} \\ &\le \sum_{\mathsf{Sk}\in\{0,1\}^{kl(\lambda)}} 2^{-\ell} \text{ (by Fact 1 and 2)} \\ &\le 2^{kl(\lambda)-\ell} = 2^{-\lambda} \text{ (since } \ell = kl(\lambda)+\lambda).\end{aligned}$$

Then, it follows that when $\mathcal{O}$ is a truly random oracle, the probability that $\mathcal{A}^{\mathcal{O}}$ outputs 1 is negligible in the security parameter. Therefore, $\mathcal{A}^{\mathcal{O}}$ can tell apart a pseudorandom oracle from a truly random oracle with non-negligible probability. This concludes the proof.

## Acknowledgments

Vincenzo Iovino is supported by the Luxembourg National Research Fund (FNR grant no. 7884937). Part of this work was made while Vincenzo Iovino was at University of Warsaw supported by the WELCOME/2010-4/2 grant founded within the framework of the EU Innovative Economy Operational Programme. We thank Abhishek Jain for helpful discussions and pointing out a bug in an earlier version of this manuscript. Vincenzo Iovino thanks Yu Li for his precious comments and Sadeq Dousti for invaluable discussions and for suggesting him this line of research.

## References

1. M. Abdalla, M. Bellare, and G. Neven. Robust encryption. In D. Micciancio, editor, *TCC 2010: 7th Theory of Cryptography Conference*, volume 5978 of *Lecture Notes in Computer Science*, pages 480–497, Zurich, Switzerland, Feb. 9–11, 2010. Springer, Berlin, Germany.
2. S. Agrawal, S. Agrawal, S. Badrinarayanan, A. Kumarasubramanian, M. Prabhakaran, and A. Sahai. Function private functional encryption and property preserving encryption: New definitions and positive results. Cryptology ePrint Archive, Report 2013/744, Version posted on 6 March 2014. `http://eprint.iacr.org/2013/744/20140306:053744`.
3. S. Agrawal, S. Agrawal, S. Badrinarayanan, A. Kumarasubramanian, M. Prabhakaran, and A. Sahai. On the practical security of inner product functional encryption. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, pages 777–798, 2015.
4. S. Agrawal, D. M. Freeman, and V. Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In D. H. Lee and X. Wang, editors, *Advances in Cryptology – ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*, pages 21–40, Seoul, South Korea, Dec. 4–8, 2011. Springer, Berlin, Germany.
5. S. Agrawal, S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption: New perspectives and lower bounds. In R. Canetti and J. A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 500–518, Santa Barbara, CA, USA, Aug. 18–22, 2013. Springer, Berlin, Germany.
6. P. Ananth, D. Boneh, S. Garg, A. Sahai, and M. Zhandry. Differing-inputs obfuscation and applications. Cryptology ePrint Archive, Report 2013/689, 2013. `http://eprint.iacr.org/2013/689`.
7. M. Backes, J. Müller-Quade, and D. Unruh. On the necessity of rewinding in secure multiparty computation. In S. P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 157–173, Amsterdam, The Netherlands, Feb. 21–24, 2007. Springer, Berlin, Germany.
8. M. Barbosa and P. Farshim. On the semantic security of functional encryption schemes. In K. Kurosawa and G. Hanaoka, editors, *PKC 2013: 16th International Workshop on Theory and Practice in Public Key Cryptography*, volume 7778 of *Lecture Notes in Computer Science*, pages 143–161, Nara, Japan, Feb. 26 – Mar. 1, 2013. Springer, Berlin, Germany.
9. O. Barkol and Y. Ishai. Secure computation of constant-depth circuits with applications to database search problems. In V. Shoup, editor, *Advances in Cryptology – CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 395–411, Santa Barbara, CA, USA, Aug. 14–18, 2005. Springer, Berlin, Germany.

10. M. Bellare, R. Dowsley, B. Waters, and S. Yilek. Standard security does not imply security against selective-opening. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 645–662, Cambridge, UK, Apr. 15–19, 2012. Springer, Berlin, Germany.
11. M. Bellare and A. O'Neill. Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition. In *Cryptology and Network Security - 12th International Conference, CANS 2013, Paraty, Brazil, November 20-22. 2013. Proceedings*, pages 218–234, 2013.
12. D. Boneh and X. Boyen. Efficient selective identity-based encryption without random oracles. *Journal of Cryptology*, 24(4):659–693, Oct. 2011.
13. D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. In J. Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229, Santa Barbara, CA, USA, Aug. 19–23, 2001. Springer, Berlin, Germany.
14. D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In Y. Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273, Providence, RI, USA, Mar. 28–30, 2011. Springer, Berlin, Germany.
15. D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In S. P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554, Amsterdam, The Netherlands, Feb. 21–24, 2007. Springer, Berlin, Germany.
16. E. Boyle, K.-M. Chung, and R. Pass. On extractability obfuscation. In Y. Lindell, editor, *TCC 2014: 11th Theory of Cryptography Conference*, volume 8349 of *Lecture Notes in Computer Science*, pages 52–73, San Diego, CA, USA, Feb. 24–26, 2014. Springer, Berlin, Germany.
17. C. Cocks. An identity based encryption scheme based on quadratic residues. In B. Honary, editor, *8th IMA International Conference on Cryptography and Coding*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363, Cirencester, UK, Dec. 17–19, 2001. Springer, Berlin, Germany.
18. A. De Caro, V. Iovino, A. Jain, A. O'Neill, O. Paneth, and G. Persiano. On the achievability of simulation-based security for functional encryption. In R. Canetti and J. A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 519–535, Santa Barbara, CA, USA, Aug. 18–22, 2013. Springer, Berlin, Germany.
19. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *54th Annual Symposium on Foundations of Computer Science*, pages 40–49, Berkeley, CA, USA, Oct. 26–29, 2013. IEEE Computer Society Press.
20. S. Garg, C. Gentry, S. Halevi, A. Sahai, and B. Waters. Attribute-based encryption for circuits from multilinear maps. In R. Canetti and J. A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 479–499, Santa Barbara, CA, USA, Aug. 18–22, 2013. Springer, Berlin, Germany.
21. C. Gentry. Practical identity-based encryption without random oracles. In S. Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Berlin, Germany.
22. O. Goldreich and A. Kahan. How to construct constant-round zero-knowledge proof systems for NP. *Journal of Cryptology*, 9(3):167–190, 1996.
23. O. Goldreich, S. Micali, and A. Wigderson. Proofs that yield nothing but their validity and a methodology of cryptographic protocol design (extended abstract). In *27th Annual Symposium on Foundations of Computer Science*, pages 174–187, Toronto, Ontario, Canada, Oct. 27–29, 1986. IEEE Computer Society Press.
24. S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich. Reusable garbled circuits and succinct functional encryption. In D. Boneh, T. Roughgarden, and J. Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 555–564, Palo Alto, CA, USA, June 1–4, 2013. ACM Press.
25. S. Goldwasser and S. Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
26. S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof-systems (extended abstract). In *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 291–304, 1985.
27. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In R. Safavi-Naini and R. Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 162–179, Santa Barbara, CA, USA, Aug. 19–23, 2012. Springer, Berlin, Germany.
28. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions via multi-party computation. In R. Safavi-Naini and R. Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 162–179. Springer, 2012.
29. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits. In D. Boneh, T. Roughgarden, and J. Feigenbaum, editors, *STOC*, pages 545–554. ACM, 2013.

30. V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In A. Juels, R. N. Wright, and S. Vimercati, editors, *ACM CCS 06: 13th Conference on Computer and Communications Security*, pages 89–98, Alexandria, Virginia, USA, Oct. 30 – Nov. 3, 2006. ACM Press. Available as Cryptology ePrint Archive Report 2006/309.
31. V. Iovino and K. Żebrowski. Simulation-based secure functional encryption in the random oracle model. In *Progress in Cryptology - LATINCRYPT 2015 - 4th International Conference on Cryptology and Information Security in Latin America, Guadalajara, Mexico, August 23-26, 2015, Proceedings*, pages 21–39, 2015.
32. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In N. P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162, Istanbul, Turkey, Apr. 13–17, 2008. Springer, Berlin, Germany.
33. 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 H. Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 62–91, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany.
34. T. Okamoto and K. Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 591–608, Cambridge, UK, Apr. 15–19, 2012. Springer, Berlin, Germany.
35. A. O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. `http://eprint.iacr.org/`.
36. A. Sahai and B. R. Waters. Fuzzy identity-based encryption. In R. Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany.
37. A. Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and D. Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53, Santa Barbara, CA, USA, Aug. 19–23, 1984. Springer, Berlin, Germany.
38. B. Waters. Functional encryption for regular languages. In R. Safavi-Naini and R. Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 218–235, Santa Barbara, CA, USA, Aug. 19–23, 2012. Springer, Berlin, Germany.

## A RSIM-Security $\Longrightarrow$ IND-Security

**Theorem 7** Let FE be a functional encryption scheme that is RSIM-Secure, then FE is IND-Secure as well.

*Proof.* Suppose towards a contradiction that there exists an adversary $\mathcal{A} = (\mathcal{A}_0, \mathcal{A}_1)$ that breaks the IND-Security of FE. Consider the following adversary $\mathcal{B}^b = (\mathcal{B}_0^b, \mathcal{B}_1^b)$, for $b \in \{0, 1\}$, and distinguisher $\mathcal{D}$, for the RSIM-Security of FE.

– $\mathcal{B}_0^b$, on input master public key $\mathsf{Mpk}$ and having oracle access to the key-generation oracle, invokes $\mathcal{A}_0$ on input $\mathsf{Mpk}$ and emulates $\mathcal{A}_0$'s key-generation oracle by using its own oracle.
  When $\mathcal{A}_0$ stops, it outputs two *challenge messages vectors*, of length $\ell$, $\boldsymbol{x}_0, \boldsymbol{x}_1 \in X^\ell$ and its internal state $\mathtt{st}$.
  hen, $\mathcal{B}_0^b$ outputs $(\boldsymbol{x}_b, \mathtt{st}' = (\mathtt{st}, b, \boldsymbol{x}_{1-b}))$.
– $\mathcal{B}_1^b$, on input master public key $\mathsf{Mpk}$, challenge ciphertexts $(\mathsf{Ct}_i)_{i \in [\ell]}$ and state $\mathtt{st}' = (\mathtt{st}, b, \boldsymbol{x}_{1-b})$ and having oracle access to the key-generation oracle, invokes $\mathcal{A}_1$ on input the challenge ciphertexts and state $\mathtt{st}$ and emulates $\mathcal{A}_1$'s key-generation oracle by using its own oracle.
  At some point $\mathcal{A}_1$ stops giving in output a bit $b'$. Then, $\mathcal{B}_1$ outputs $(b', b, \boldsymbol{x}_{1-b}, \mathcal{Q})$ as its own output, where $\mathcal{Q} = (\mathsf{k}_i)$ is the list of keys for which $\mathcal{A}$ has issued a key-generation query.

$\mathcal{D}(\mathsf{Mpk}, \boldsymbol{x}, \alpha)$:

- $\mathcal{D}$ interprets $\alpha$ as $\alpha = (b', b, \boldsymbol{x}_{1-b}, \mathcal{Q})$ and returns 1 if $b = b'$ and for any $k \in \mathcal{Q}$ $F(k, \boldsymbol{x}) = F(k, \boldsymbol{x}_{1-b})$, 0 otherwise.

Let $\mathsf{IND}_{\mathcal{A}}^{\mathsf{FE},b}$ be an experiment identical to the $\mathsf{IND}$-Security experiment except that the challenger always encrypts challenge vector $\boldsymbol{x}_b$ (instead of choosing one of the two challenges at random). Then, it holds that for any function $\epsilon(\lambda)$ that is inverse of a polynomial:

$$\mathsf{IND}_{\mathcal{A}}^{\mathsf{FE},0} = \mathsf{RealExp}^{\mathsf{FE},\mathcal{B}^0} \approx_\epsilon \mathsf{IdealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^0} = \mathsf{IdealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^1},$$

and

$$\mathsf{IdealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^1} \approx_\epsilon \mathsf{RealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^1} = \mathsf{IND}_{\mathcal{A}}^{\mathsf{FE},1},$$

where, more specifically:

1. $\mathsf{IND}_{\mathcal{A}}^{\mathsf{FE},0} = \mathsf{RealExp}^{\mathsf{FE},\mathcal{B}^0}$ (i.e., the probability that $\mathcal{A}$ wins in experiment $\mathsf{IND}_{\mathcal{A}}^{\mathsf{FE},0}$ equals the probability that $D$ outputs 1 on input the output of $\mathsf{RealExp}^{\mathsf{FE},\mathcal{B}^0}$) holds by definition of $\mathcal{B}^0$ and $\mathcal{D}$.
2. $\mathsf{RealExp}^{\mathsf{FE},\mathcal{B}^0} \approx_\epsilon \mathsf{IdealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^0}$. This holds by the $\mathsf{RSIM}$-Security of $\mathsf{FE}$.
3. $\mathsf{IdealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^0} = \mathsf{IdealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^1}$ holds because if $\mathcal{A}$ breaks the $\mathsf{IND}$-Security of $\mathsf{FE}$, then with all but negligible probability, the queries issued by $\mathcal{A}$ (and thus by $\mathcal{B}$) are such that $F(k, \boldsymbol{x}_0) = F(k, \boldsymbol{x}_1)$ for any key $k$ for which $\mathcal{A}$ has issued a key-generation query.
4. $\mathsf{IdealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^1} \approx_\epsilon \mathsf{RealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^1}$ holds again by the $\mathsf{RSIM}$-Security of $\mathsf{FE}$.
5. Finally, $\mathsf{RealExp}_{\mathsf{Sim}}^{\mathsf{FE},\mathcal{B}^1} = \mathsf{IND}_{\mathcal{A}}^{\mathsf{FE},1}$ (i.e., the probability that $\mathcal{A}$ wins in experiment $\mathsf{IND}_{\mathcal{A}}^{\mathsf{FE},1}$ equals the probability that $D$ outputs 1 on input the output of $\mathsf{RealExp}^{\mathsf{FE},\mathcal{B}^1}$) holds by definition of $\mathcal{B}^1$ and $\mathcal{D}$.

But, if for any $\epsilon$, $\mathsf{IND}_{\mathcal{A}}^{\mathsf{FE},0} \approx_\epsilon \mathsf{IND}_{\mathcal{A}}^{\mathsf{FE},1}$, then $\mathcal{A}$ does not break the $\mathsf{IND}$-Security of $\mathsf{FE}$, a contradiction.

## B Proof of Theorem 5

*Proof.* (**Simplified simulation.**) As explained before, for purposes of exposition, we first present a simplified simulation strategy in which the ouput of the simulator is "biased" (i.e., it has different distribution than the output of the adversary in the real experiment) and then we illustrate how to remove such restriction.

Our simulator $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ works as follows. $\mathsf{Sim}_0$ takes in input the master public and secret key, the list $\mathcal{Q} = (k_i, \mathsf{Sk}_{k_i}, F(k_i, \boldsymbol{x}))_{i \in [q_1]}$, and the intentionally leaked information about the challenge messages[7] $F(\epsilon, \boldsymbol{x}) = (\mathsf{ind}_j, |\mathsf{m}_j|)_{j \in [\ell]}$. Then, for each $i \in [q_1]$, $\mathsf{Sim}_0$ checks whether $P(k_i, \mathsf{ind}_j) = 1$ for some $j \in [\ell]$. If it is the case, then $\mathsf{Sim}_0$ learns $\mathsf{m}_j$. Furthermore, let $\mathcal{X}$ the tuple of messages (indices with the relative payloads) learnt by $\mathsf{Sim}_0$. Then, for each pair in $\mathcal{X}$, $\mathsf{Sim}_0$ generates a normal ciphertext by invoking the encryption algorithm. For all the other indices for which $\mathsf{Sim}_0$ was not able to learn the corresponding payload, $\mathsf{Sim}_0$ generates ciphertexts for those indices having a random payload. Let $\boldsymbol{x}^\star$ be the resulting message vector that the simulator used to produce the challenge ciphertexts.

Then, $\mathsf{Sim}_0$ executes $\mathcal{A}_1$ on input the so generated challenge ciphertexts. When $\mathcal{A}_1$ invokes its key-generation oracle on input key $k$, $\mathsf{Sim}_1$ is asked to generate a corresponding secret key given $k$ and $F(k, \boldsymbol{x})$. Now we have two cases:

1. $P(k, \mathsf{ind}_j) = 1$ for some $j \in [\ell]$: Then, $\mathsf{Sim}$ learns $\mathsf{m}_j$. If $\mathsf{m}_j$ was already known by $\mathsf{Sim}$, it means that the corresponding challenge ciphertext was well formed when $\mathsf{Sim}_0$ invoked $\mathcal{A}_1$. Then $\mathsf{Sim}_1$ generates the secret key for $k$ (using the master secret key) and the simulation continues. On the other hand, if $\mathsf{Sim}_0$ didn't know $\mathsf{m}_j$ then the ciphertext corresponding to $\mathsf{ind}_j$ was for a random message. Therefore, $\mathsf{Sim}_0$ must halt $\mathcal{A}_1$ and rewinds it. $\mathsf{Sim}_0$ adds $(\mathsf{ind}_j, \mathsf{m}_j)$ to $\mathcal{X}$ (and thus updates $\boldsymbol{x}^\star$) and with this new knowledge $\mathsf{Sim}_0$ rewinds $\mathcal{A}_1$ on input the encryption of the new ciphertexts (i.e., the encryption of the new $\boldsymbol{x}^\star$). The above reasoning easily extends to the case that $P(k, \mathsf{ind}_j) = 1$ for more than one $j$.

[7] Recall that $\boldsymbol{x}$ is a vector of challenge messages in which, for $j \in [\ell]$, the $j$-th component consists of a pair $(\mathsf{ind}_j, \mathsf{m}_j)$, where $\mathsf{ind}_j$ is the "index" and $\mathsf{m}_j$ is the "payload".

2. $P(k, \mathsf{ind}_j) = 0$ for all $j \in [\ell]$: In this case, a secret key for $k$ can not be used to decrypt any of the challenge ciphertexts. Then, $\mathsf{Sim}_1$ generates the secret key for $k$ (using the master secret key) and the simulation continues.

If at some point the adversary halts giving some output the simulator outputs what the adversary outputs. This conclude the description of the simulator $\mathsf{Sim}$.

It remains to show that the simulated challenge ciphertexts does not change $\mathcal{A}_1$'s behavior significantly. We call a key-generation query *good* if the simulator can answer such query without rewinding the adversary according to the previous rules. We call a completed execution of the adversary between two rewinds of the adversary a *run*. First, notice that the number of runs, meaning the number of times the simulator rewinds, is upper-bounded by the number of challenge messages $\ell$ that is polynomial in the security parameter. In fact, each time that a query is not good and the simulator needs to rewind then the simulator learn a new pair $(\mathsf{ind}_j, \mathsf{m}_j)$, for some $j \in [\ell]$ and the same query will never cause a rewind anymore. In the last run, that in which all the key-generation queries are good, the view of the adversary is indistinguishable from that in the real game. This follows from the $\mathsf{IND}$-Security of $\mathsf{PIPE}$. In fact, the evaluations of the secret keys on the challenge ciphertexts in the real experiment give the same values than the evaluation of the simulated secret keys on the simulated ciphertexts in the ideal experiment since the secret keys are generated honestly. Therefore, the $\mathsf{IND}$-Security guarantees that in this case the view in the real experiment is indistinguishable from that in the ideal experiment.

*The actual simulation.* The previous simulation incurs the following problem: the output of the simulator could be biased. Consider for example an adversary that with probability $1/3$ does not ask any query and with probability $2/3$ asks a query that triggers a rewind, and outputs its computation. In the real experiment the transcript contains zero queries with probability $1/3$ whereas the output of the ideal experiment contains zero queries with probability much larger than $1/3$.[8]. Above, we have shown that the *last* transcript of the simulator would be indistinguishable from the transcript of the adversary in the real experiment but this final output could be biased and corresponds to different runs of the adversary. Thus, we need the following smarter strategy. First, recall that by standard use of Chernoff's bound we can estimate a $(\beta, \gamma)$-approximation of a random variable, where the estimate is $\beta$-close with probability $1-\gamma$. Moreover, this can be made by sampling the random variable a number of times that is polynomial in $1/\beta$ and logarithmic in $1/\gamma$. Let $\mu$ be some fixed negligible function and $\nu$ be the the distinguishing advantage we wish to achieve (see Definition 3). Let $i = 0$ to $\ell$, the simulator makes the following.
Consider the experiment $X_i$ in which the simulator executes the adversary in a run where the information it learnt consists of the pairs $(\mathsf{ind}_j, \mathsf{m}_j)$ for $j = 1, \ldots, i$, and we assume that for $i = 0$ the simulator starts the run with random pairs. The run is executed as described in the simplified simulation, where if the adversary triggers a rewind then the simulator outputs a dummy value, otherwise the simulator outputs what the adversary outputs.
We denote by $p_i$ the probability that in experiment $X_i$ the adversary triggers a rewind. Setting $\nu' = \nu^{1/2}/\ell$, the simulator computes a $(\nu', \delta)$-estimate $\tilde{p}_i$ for $p_i$ for some negligible function $\delta$ (the reason for setting $\nu'$ to such value will be clear at the end of the analysis). If the estimate $\tilde{p}_i \leq \mu$, then the simulator executes the adversary in experiment $X_i$ and if the adversary terminates without triggering a rewind, the simulator outputs what the adversary outputs, otherwise the simulator outputs a dummy value. Instead, if the estimate is greater than $\mu$, then simulator increments $i$ and proceeds to next step.
Let us compute the advantage of a PPT distinguisher in telling apart the real from the ideal experiment. By assumption on the estimate and by construction of the simulator, the output of the simulator is the output of the adversary in experiment $X_1$ with probability at most $w_1 = (1-\delta)(\mu+\nu')$ and is the output of the adversary in experiment $X_2$ with probability at most $a_2(1-\delta)(\mu+\nu')$, where $a_2 = 1 - q_1 < 1$, and so forth. Therefore, the output of the simulator is the output of the adversary in experiment $X_i$ with probability at most $(1-\delta)(\mu+\nu')$.
If the output of the simulator equals the output of the adversary in experiment $X_i$, then the distinguishing advantage is at most $\nu'$ up to some negligible factor. Indeed, if the adversary does not trigger a rewind the two experiment are computationally indistinguishable by the $\mathsf{IND}$-Security and in experiment $X_i$ the adversary triggers a rewind with probability at most $\mu + \nu'$ and $\mu$ is negligible. By definition of $\nu'$, it follows that the overall advantage is at most $\ell\nu'^2 = \nu$ up to a negligible factor.

[8] A similar problem arises in the context of rewinding simulators for constant-round zero-knowledge as in [22]

# C Positive Results for PE with Private-Index

In this section we go further showing equivalences for PE with *private-index* for several functionalities including Anonymous IBE, inner-product over $\mathbb{Z}_2$, monotone conjunctive Boolean formulae, and the existence of $\mathsf{RSIM}$-Secure schemes for all classes of $\mathsf{NC}_0$ circuits.

As before, because in Appendix A, we show that $\mathsf{RSIM}$-Security implies $\mathsf{IND}$-Security, to establish the equivalence for the functionalities we study, it is enough to prove the other direction, namely that $\mathsf{IND}$-Security implies $\mathsf{RSIM}$-Security.

*Abstracting the properties needed by the simulator.* A closer look at the proof of theorem 5 hints some abstract properties that a predicate has to satisfy in order for the simulator to be able to produce an indistinguishable view. We identify the following two properties.

The execution of the simulator is divided in *runs*. At run $j$, the simulator invokes the adversary on input a ciphertext for message $x_j$, whereas the adversary chose $x$, and keeps the invariant that $x_j$ gives the same results than $x$ respect to the queries asked by the adversary until that run.

At some point the adversary asks a query $k$ for which $F(k, x) \neq F(k, x_j) \neq \perp$ thus the simulator is not able to answer the query in this run. But if the functionality has the property (1) that it is easy to *pre-sample* a new value $x_{j+1}$ that satisfies all queries including the new one, the simulator can rewind the adversary this time on input an encryption of value $x_{j+1}$.

This is still not sufficient since there is no bound on the maximum number of rewinds needed by the simulator so we have to require the property (2) to force the simulation progresses towards a maximum.

To give a clear example, consider how a simulator could work for Anonymous IBE. Suppose that the adversary chooses as challenge identity `crypto` and the simulator chooses `aaaaa` as simulated identity for the ciphertext the simulator will pass to the adversary. Then, the adversary issues a query for identity `bbbbb` and the simulator learns that the predicate is not satisfied against, so the query gives the same evaluation on both the challenge identity and the simulated identity. This is coherent with the query, so the simulator can continue the simulation.

Now, suppose that the adversary issues the query for identity `crypto`. Then, the simulated identity is no more compatible with the new query and the simulator has to rewind the adversary but, since the simulator has learnt the challenge identity *crypto* and the corresponding payload exactly, in the next run the simulator is able to finish the simulation perfectly. This simulation strategy is simplified, and as we explained in Section 5 the simulator also need to guarantee that the output is not biased. In Section C.2, we show how to implement a more complicated strategy for the predicate inner-product over $\mathbb{Z}_2$.

## C.1 Equivalence for Anonymous IBE

The following theorem is an extension of Theorem 5.

**Theorem 8** Let $\mathsf{AIBE}$ be an Anonymous IBE scheme $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{IND}$-$\mathsf{Secure}$. Then, $\mathsf{AIBE}$ is $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{RSIM}$-Secure as well.

*Intuition.* Notice that, in an Anonymous IBE scheme the ciphertext does not leak the identity for which it has been generated and thus the special key $\epsilon$ does not provide this information as for a public-index scheme. Despite this, when the adversary issues a key-generation query for a key $k$ such that $F(k, x) \neq \perp$, then the simulator *learns* that $x$ is a message for index (or identity for the case of AIBE) $k$ and payload $F(k, x)$. Thus, the simulator rewinds the adversary on input a freshly generated ciphertext for that pair and can safely generate an *honest* secret key for $k$ upon request.

Another important difference with the proof of Theorem 5 is that the simulator could be forced to rewind without gaining any new knowledge and this could result in a never ending simulation. This happens for example in the following case: Let $x$ a challenge message chosen by the adversary and let $x^\star$ the message chosen by the simulator to simulate the ciphertext for $x$. Then, if the adversary issues a key-generation query for key k such that $F(k, x) = \perp$ but $F(k, x^\star) \neq \perp$, then the simulator is forced to rewind without gaining any new knowledge and this could happen indefinitely. But, the $\mathsf{IND}$-Security of $\mathsf{AIBE}$ scheme guarantees that such situation can happen only with negligible probability, and thus the simulator can just abort in such cases.

*Proof.* (**Simplified simulation.**) Our simulator $\mathsf{Sim} = (\mathsf{Sim}_0, \mathsf{Sim}_1)$ works as follows. $\mathsf{Sim}_0$ takes in input the master public and secret key, the list $\mathcal{Q} = (k_i, \mathsf{Sk}_{k_i}, F(k_i, \boldsymbol{x}))_{i\in[q_1]}$, and the intentionally leaked information about the challenge messages $F(\epsilon, \boldsymbol{x}) = (|\mathsf{ind}_j|, |\mathsf{m}_j|)_{j\in[\ell]}$. Then, for each $i \in [q_1]$, $\mathsf{Sim}_0$ checks whatever $F(k_i, x_j) \neq \perp$ for some $j \in [\ell]$. If it is the case, then $\mathsf{Sim}_0$ learns that message $x_j$ is for identity $\mathsf{ind}_j = k_i$ and payload $\mathsf{m}_j = F(k_i, x_j)$.

Let $\mathcal{X}$ the set of tuple of the following form $(j, \mathsf{ind}_j, \mathsf{m}_j)$ learnt by $\mathsf{Sim}_0$. Then, for each pair in $\mathcal{X}$, $\mathsf{Sim}_0$ generates a normal ciphertext for message $x^\star_{\mathsf{j}} = (\mathsf{ind}^\star_j, \mathsf{m}^\star_j)$, with $\mathsf{ind}^\star_j = \mathsf{ind}_j$ and $\mathsf{m}^\star_j = \mathsf{m}_j$, by invoking the encryption algorithm. For all the other positions $k$ for which $\mathsf{Sim}_0$ was not able to learn the corresponding index and payload, $\mathsf{Sim}_0$ generate a ciphertext for random $x^\star_k = (\mathsf{ind}^\star_k, \mathsf{m}^\star_k)$.

Then, $\mathsf{Sim}_0$ executes $\mathcal{A}_1$ on input the challenge ciphertexts $(\mathsf{Ct}^\star_j)_{j\in[\ell]}$, where $\mathsf{Ct}^\star_j$ is for message $x^\star_{\mathsf{j}} = (\mathsf{ind}^\star_j, \mathsf{m}^\star_j)$ as described above. When $\mathcal{A}_1$ invokes its key-generation oracle on input key $k$, $\mathsf{Sim}_1$ is asked to generate a corresponding secret key given $k$ and $F(k, \boldsymbol{x})$. Now we have the following cases:

1. *If for each $j \in [\ell]$ such that $F(k, x_j) \neq\perp$, $(j, k, F(k, x_j)) \in \mathcal{X}$*: Then we have two sub-cases:
   (a) If there exists and index $\mathsf{j} \in [\ell]$ such that $F(k, x_{\mathsf{j}}) =\perp$ but $F(k, x^\star_{\mathsf{j}}) \neq\perp$ then $\mathsf{Sim}_0$ aborts.
   (b) Otherwise, $\mathsf{Sim}_1$ honestly generates a secret key $\mathsf{Sk}_k$ for key $k$. Notice that it holds that $F(k, x^\star_{\mathsf{j}}) = F(k, x_j)$ for all $j \in [\ell]$.
2. *If there exists an index $j \in [\ell]$ such that $F(k, x_j) \neq\perp$ but $(j, k, F(k, x_j)) \notin \mathcal{X}$*: Then $F(k, x^\star_{\mathsf{j}}) \neq F(k, x_j)$ with high probability. Thus $\mathsf{Sim}_0$ adds $(j, k, F(k, x_j))$ to $\mathcal{X}$ and rewinds $\mathcal{A}_1$ on freshly generated ciphertexts based on the information $\mathsf{Sim}_0$ has collected in $\mathcal{X}$ so far.
3. *If for all $j \in [\ell]$, $F(k, x_j) =\perp$*: Then we have two sub-cases:
   (a) If there exists and index $\mathsf{j} \in [\ell]$ such that $F(k, x_{\mathsf{j}}) =\perp$ but $F(k, x^\star_{\mathsf{j}}) \neq\perp$ then $\mathsf{Sim}_0$ aborts.
   (b) Otherwise, $\mathsf{Sim}_1$ honestly generates a secret key $\mathsf{Sk}_k$ for key $k$. Notice that it holds that $F(k, x^\star_{\mathsf{j}}) = F(k, x_j) =\perp$ for all $j \in [\ell]$.

If after a query the simulator has got to rewind the adversary, we say that such query triggered a rewind. If at some point the adversary halts giving some output, then the simulator outputs what the adversary outputs. This conclude the description of the simulator $\mathsf{Sim}$.

Let us first bound the probability that the simulator aborts during its simulation, this happens in cases $1.(a)$ or $3.(a)$. Let us focus on case $1.(a)$, the other one is symmetric. Notice that when case $1.(a)$ happens then $F(k, x_{\mathsf{j}}) =\perp$ but $F(k, x^\star_{\mathsf{j}}) \neq\perp$, meaning that $\mathsf{ind}_{\mathsf{j}} \neq k$ and $\mathsf{ind}^\star_{\mathsf{j}} = k$, and that all the previous key-generation queries are good, meaning that no rewind has been triggered. Therefore, if this event happens with non-negligible probability, $\mathcal{A}$ can be used to build another adversary $\mathcal{B}$ that distinguishes between the encryption of $x_{\mathsf{j}}$ and $x^\star_{\mathsf{j}}$ with the same probability, thus contradicting the $\mathsf{IND}$-Security of the scheme. Precisely, $\mathcal{B}$ simulates the view to $\mathcal{A}$ as described before (i.e., simulating the interface with the simulator) and returns as its challenges two messages with indices $\mathsf{ind}_0 = \mathsf{ind}_j$ and $\mathsf{ind}_1 = \mathsf{ind}^\star_j$, where the two indices are as before. Then, $\mathcal{B}$ runs $\mathcal{A}$ on some ciphertext that is identical to that described before except that $\mathsf{Ct}^\star_j$ is set to the challenge ciphertext received from the challenger of the $\mathsf{IND}$-Security game. If at some point $\mathcal{A}$ asks a query for identity $\mathsf{ind}^\star_j$, then $\mathcal{B}$ outputs 1 as its guess, otherwise $\mathcal{B}$ outputs 0 as its guess. Notice that if the challenge ciphertext for $\mathcal{B}$ is for the challenge message with identity $\mathsf{ind}_1 = \mathsf{ind}^\star_j$, $\mathcal{B}$ perfectly simulated the view of $\mathcal{A}$ when interacting with the above simulator, and thus, by hypothesis on the non-negligible probability of occurrence of the case $1.(a)$, $\mathcal{B}$ outputs 1 with non-negligible probability. On the other hand, if the challenge ciphertext is for the challenge message with identity $\mathsf{ind}_0 = \mathsf{ind}_j$, then the view of $\mathcal{A}$ is completely independent from $\mathsf{ind}^\star_j$, so the probability that $\mathcal{A}$ asks a query for such identity is negligible and thus $\mathcal{B}$ outputs 0 with overwhelming probability.

Finally, notice that the number of runs, meaning the number of times the simulator makes a rewind (a rewind happens when case 2. occurs), is upper-bounded by the number of challenge messages $\ell$ that is polynomial in the security parameter. In fact, every time that a query is not good and the simulator needs to rewind the adversary, the simulator learns a new pair $(\mathsf{ind}_j, \mathsf{m}_j)$, for some $j \in [\ell]$, and the same query will never cause a rewind anymore. In the last run, that in which all the key-generation queries are good, the view of the adversary is indistinguishable from that in the real game. This follows from the $\mathsf{IND}$-Security of $\mathsf{AIBE}$ by noting that the evaluations of the secret keys on the challenge ciphertexts in the real experiment give the same values than the evaluation of the simulated secret keys on the simulated ciphertexts in the ideal experiment since the secret keys are generated honestly. Therefore, the $\mathsf{IND}$-Security guarantees that in this case the view in the real experiment is indistinguishable from that in the ideal experiment.

**Non-biased simulation.** We stress that this is a simplified simulation and the simulator also needs to guarantee that the output is not biased. This can be made as explained in the security reduction of theorem 5.

### C.2 Equivalence for Inner-Product over $\mathbb{Z}_2$

The functionality inner-product over $\mathbb{Z}_2$ ($\mathsf{IP}$, for short)[9] is defined in the following way. It is a family of predicates with key space $K_n$ and index space $I_n$ consisting of binary strings of length $n$, and for any $k \in K_n, x \in I_n$ the predicate $\mathsf{IP}(k,x) = 1$ if and only if $\sum_{i \in [n]} k_i \cdot x_i = 0 \mod 2$.

Henceforth, we assume that the reader is familiar with the notion of pre-image samplability introduced by O'Neill [35].

In our positive results for $\mathsf{IP}$ over $\mathbb{Z}_2$ we use the following theorem.

**Theorem 9** [35] The functionality $\mathsf{IP}$ over $\mathbb{Z}_2$ is pre-image samplable.

**Theorem 10** If a predicate encryption scheme $\mathsf{PE}$ for $\mathsf{IP}$ is $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{IND}$-$\mathsf{Secure}$ then $\mathsf{PE}$ is $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{RSIM}$-$\mathsf{Secure}$ as well.

*Proof.* (**Simplified simulation.**) The proof follows the lines of the Theorem 5. For simplicity we assume that the adversary outputs a challenge message with the payload set to 1, i.e., the functionality returns values in $\{0,1\}$, but this can be easily generalized by handling the payload as in the proof of theorem 5.

Let $x = (x_1, \ldots, x_\ell) \in \{0,1\}^{n \cdot \ell}$ be the challenge index [10] output by the adversary $\mathcal{A}_0$ and let $(w_i)_{i=1}^{q_1}$ be the queries asked by $\mathcal{A}_0$ (i.e. the queries asked before seeing the challenge ciphertexts).

As usual we divide the execution of the simulator in runs and in any run the simulator keeps an index $x^0 = (x_1^0, \ldots, x_\ell^0) \in \{0,1\}^{n \cdot \ell}$ that uses to encrypt the simulated ciphertext given to the adversary in that run.

Let $Y_i$ be a matrix in $\{0,1\}^{(q_1+i-i) \times n}$ where the rows $y_1, \ldots, y_{q_1+i-1}$ of $Y_i$ are such that the first $q_1$ rows $y_1, \ldots, y_{q_1}$ consist of the vectors $w_1, \ldots, w_{q_1}$ (i.e., $y_1 = w_1, \ldots, y_{q_1} = w_{q_1}$) and for each $j = 1, \ldots, i-1$ the row $y_{q_1+j}$ of $Y_i$ corresponds to the last query asked by $\mathcal{A}_1$ in run $j$ (as it will be clear soon, in any run $i$, if the last query asked by the adversary in such run will trigger a rewind, then only such query is put in the matrix, and not any other previous query asked by the adversary in run $i$).

Furthermore, for any $i \geq 1$ and any $j \in [\ell]$, let $b_{i,j} \in \{0,1\}^{q_1+i-1}$ be the column vector such that $b_{i,j}[k] = \mathsf{IP}(y_k, x_j), k = 1, \ldots, q_1 + i - 1$. During the course of the simulation, the simulator will guarantee the following invariant: at the beginning of any run $i \geq 1$, for any $j \in [\ell]$, $Y_i \cdot x_j^0 = b_{i,j}$.

In the first run the simulator runs the adversary with input a ciphertext that encrypts an index $x^0 = (x_1^0, \ldots, x_\ell^0) \in \{0,1\}^{n \cdot \ell}$ such that for any $j \in [\ell]$, $Y_1 \cdot x_j^0 = b_{1,j}$. The simulator can efficiently find such vector by using the $\mathsf{PS}$ of $\mathsf{IP}$ guaranteed by Theorem 9. When in a run $i \geq 1$ the adversary makes a query for a vector $y \in \{0,1\}^n$ we distinguish two mutually exclusive cases. executed).

1. *The vector $y$ is a linear combination of the rows of $Y_i$.* Then, by the invariant property it follows that for any $j \in [\ell]$, $\mathsf{IP}(y, x_j) = \mathsf{IP}(y, x_j^0)$, and the simulator continues the simulation answering the query as usual (i.e., by giving to the adversary $\mathcal{A}_1$ the secret key for $y$ generated honestly).
2. *The vector $y$ is not a linear combination of the rows of $Y_i$.* Then, the simulator could not be able to answer this query. In this case, we say that the query triggered a rewind and the simulator rewinds the adversary $\mathcal{A}_1$ as follows. The simulator updates $Y_{i+1}$ by adding the new row $y$ to $Y_i$ and uses the $\mathsf{PS}$ of $\mathsf{IP}$ guaranteed by Theorem 9 to efficiently find a new vector $x' = (x'_1, \ldots, x'_\ell) \in \{0,1\}^{n \cdot \ell}$ such that for any $j \in [\ell]$, $Y_{i+1} \cdot x'_j = b_{i+1,j}$ (i.e., the $\mathsf{PS}$ algorithm is invoked independently for each equation $Y_{i+1} \cdot x'_j = b_{i+1,j}$). Finally, the simulator rewinds the adversary by invoking it with input the encryption of $x'$ and updates $x^0$ setting it to $x'$. Notice that at the beginning of run $i+1$ the invariant is still satisfied.

[9] We remark that our inner-product is defined over $\mathbb{Z}_2$ so the predicate is different from that of [32].

[10] The challenge index output by the adversary consists of a tuple $(x_1, \ldots, x_\ell)$ of vectors where each element $x_i \in \{0,1\}^n$ for $i = 1, \ldots, \ell$. For simplicity, henceforth we interpret such challenges as vectors in $\{0,1\}^{n \cdot \ell}$.

At the end of the last run, the simulator outputs what the adversary outputs. It is easy to see that the simulator executes at most $n$ runs since in any run $i > 2$ the rank $Y_i$ is greater than the rank of $Y_{i-1}$ and for any $i \geq 1$ the rank of $Y_i$ is at most $n$.

Finally, notice that at the beginning of the last run the invariant guarantees that for any query $y$ asked by $\mathcal{A}_0$ and for any $j \in [\ell]$ $\mathsf{IP}(y, x_j) = \mathsf{IP}(y, x_j^0)$. Furthermore, since in the last run no query has triggered a rewind, then any query asked by $\mathcal{A}_1$ in the last run still satisfies this property. Therefore, by the IND-Security of the scheme, it follows that the output of the simulator is indistinguishable from that of the adversary in the real game.

**Non-biased simulation.** We stress that this is a simplified simulation and the simulator also needs to guarantee that the output is not biased. This can be made as explained in the security reduction of theorem 5.

RSIM-*Security for* $\mathsf{NC}_0$ *circuits.* Recall that $\mathsf{NC}_0$ is the class of all family of Boolean circuits of polynomial size and constant depth with AND, OR, and NOT gates of fan-in at most 2. It is a known fact that circuits in $\mathsf{NC}_0$ with $n$-bits input and one-bit output can be expressed as multivariate polynomials $p(x_1, \ldots, x_n)$ over $\mathbb{Z}_2$ of constant degree.

Furthermore, you can encode such polynomials as vectors in $\mathbb{Z}_2^{n^m}$ for some constant $m$ and evaluate them at any point using the inner-product predicate. Therefore, it is easy to see that the previous proof implies naturally the existence of a RSIM-Secure FE scheme for any family of circuits in $\mathsf{NC}_0$ but we omit the details.

**Theorem 11** If there exists predicate encryption scheme for IP that is (poly, poly, poly)-IND-Secure then there exists a predicate encryption scheme PE for any family of circuits in $\mathsf{NC}_0$ that is (poly, poly, poly)-RSIM-Secure.

Despite their weakness, $\mathsf{NC}_0$ circuits can be employed for many practical applications (see [9]).

### C.3 Equivalence for Monotone Conjunctive Boolean Formulae

The functionality Monotone Conjunctive Boolean Formulae (MCF, for short) is defined in the following way. It is a family of predicates with key space $K_n$ consisting of monotone (i.e., without negated variables) conjunctive Boolean formulae over $n$ variables (i.e., a subset of indices in $[n]$) and index space $I_n$ consisting of assignments to $n$ Boolean variables (i.e., binary strings of length $n$), and for any $\phi \in K_n, x \in I_n$ the predicate $\mathsf{MCF}(\phi, x) = 1$ if and only if the assignment $x$ satisfies the formula $\phi$. If a formula $\phi \subseteq [n]$ contains the index $i$, we say that $\phi$ has the $i$-th formal variable set.

The reader may have noticed that PE for MCF is a special case of PE for the family of all conjunctive Boolean formulae introduced by [15]. Though the monotonicity weakens the power of the primitive, it has still interesting applications like PE for subset queries as shown by [15]. We point out that the monotonicity is fundamental to implement our rewinding strategy. In fact, (under some complexity assumption) the functionality that computes the family of all conjunctive Boolean formulae is not PS[11], so it is not clear whether an equivalence between (poly, poly, poly)-IND-Security and (poly, poly, poly)-RSIM-Security can be established for this primitive. On the other hand, weakening the functionality allowing only monotone formulae, we are able to prove the following theorem.

**Theorem 12** If a predicate encryption scheme PE for MCF is (poly, poly, poly)-IND-Secure then PE is (poly, poly, poly)-RSIM-Secure as well.

Proof Sketch. (**Simplified simulation.**) The proof follows the lines of the previous equivalence theorems and is only sketched outlining the differences. Let $x = (x_1, \ldots, x_\ell)$ be the challenge index (i.e., assignment) vector chosen by the adversary $\mathcal{A}_0$ that the simulator does not know.

The simulator can easily sample an index vector $x^0 = (x_1^0, \ldots, x_\ell^0)$ such that for any $i \in [\ell]$, $x_i^0$ satisfies the equations: $\mathsf{MCF}(\phi, x_i^0) = \mathsf{MCF}(\phi, x_i)$ for any query $\phi$ asked by $\mathcal{A}_0$ before seeing the challenge ciphertexts.

[11] The authors of [18] proved this fact that will appear in the full version of their paper.

This can be done by the simulator in the following way just having the evaluations of the assignments on the formulae. In full generality, fix an arbitrary set of formulae $A = \{\phi_i\}_{i\in[q]}$ and their evaluations over some (hidden) assignment $x = (x_1, \ldots, x_\ell)$. For any $j \in [\ell]$ and any position $k \in [n]$, the simulator sets the $k$-th bit of $x^0_j$ to be 1 or 0 according to the following rules.

If there exists some $\phi \in A$ that has the $k$-th formal variable set *and* $x_j$ satisfies $\phi$ (the simulator has this information because it knows the evaluation of $\phi$ on $x_j$), then the $k$-th bit of $x^0_j$ is set to 1, otherwise (i.e., whether either the $k$-th formal variable of $\phi$ is not set or $x_j$ does not satisfy $\phi$) it is set to 0.

It is easy to see that $x^0$ satisfies the previous equations with respect to the set of formulae $A$ and thus is a valid pre-image of $x$. As usual, we divide the execution of the simulation in runs.

During the course of the simulation, the simulator will guarantee the invariant that at the beginning of any run, the index vector $x^0$ satisfies all equations with respect to the (hidden) vector $x$ and to all queries asked by the adversary. If a new query does not satisfy such equations, then the simulator has to find a new pre-image that satisfies all the equations including the new one.

This is done as before by pre-sampling according to the above rules. Notice that once a bit in some index $x^0_j$ is set to 1, it is not longer changed. Thus, it follows that the number of runs is upper bounded by the bit length of $x$. Therefore, if $\mathsf{PE}$ is $\mathsf{IND\text{-}Secure}$, the simulator can conclude the simulation and produce an output indistinguishable from that of the adversary as desired.

**Non-biased simulation.** We stress that this is a simplified simulation and the simulator also needs to guarantee that the output is not biased. This can be made as explained in the security reduction of Theorem 5. □

### C.4 Predicates with Polynomial Size Key Space

Boneh *et al.* [14] (see also [15]) presented a generic construction for functional encryption for any functionality $F$ where the key space $K$ has polynomial size that can be proven $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{IND}$-Secure in the standard model and a modification that can be proven $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{SIM}$-Secure in the random oracle model.

Bellare and O'Neill [11] proved the $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{SIM}$-Security of their scheme assuming that the underlying PKE scheme is secure against key-revealing selective opening attack (SOA-K) [10]. On the other hand we prove that the construction is $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{RSIM}$-Secure assuming only IND-CPA PKE that is a weaker assumption than SOA-K PKE needed in [11].

The construction of Boneh *et al.* is the following. Let $s = |K| - 1$ and $K = (k_0 = \epsilon, k_1, \ldots, k_s)$.[12]

The brute force functional encryption scheme realizing $F$ uses a semantically secure public-key encryption scheme $\mathcal{E} = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ and works as follows:

1. $\mathsf{Setup}(1^\lambda)$: for $i = 1, \ldots, s$, run $(\mathcal{E}.\mathsf{pk}_i, \mathcal{E}.\mathsf{sk_i}) \leftarrow \mathcal{E}.\mathsf{KeyGen}(1^\lambda)$ and output $\mathsf{Mpk} = (\mathcal{E}.\mathsf{pk}_1, \ldots, \mathcal{E}.\mathsf{pk}_s)$ and $\mathsf{Msk} = (\mathcal{E}.\mathsf{sk}_1, \ldots, \mathcal{E}.\mathsf{sk_s})$.
2. $\mathsf{KeyGen}(\mathsf{Msk}, k_i)$: output $\mathsf{sk_i} := \mathcal{E}.\mathsf{sk_i}$.
3. $\mathsf{Enc}(\mathsf{Msk}, x)$: output $\mathsf{Ct} := (F(\epsilon, x), \mathcal{E}.\mathsf{Enc}(\mathcal{E}.\mathsf{pk}_1, F(k_1, x)), \ldots, \mathcal{E}.\mathsf{Enc}(\mathcal{E}.\mathsf{pk}_s, F(k_s, x)))$.
4. $\mathsf{Dec}(\mathsf{sk_i}, \mathsf{Ct})$: output $\mathsf{Ct}[0]$ if $\mathsf{sk_i} = \epsilon$, and output $\mathcal{E}.\mathsf{Dec}(\mathcal{E}.\mathsf{sk_i}, \mathsf{Ct[i]})$ otherwise.

**Theorem 13** Let $\mathsf{FE}$ be the above $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{IND\text{-}Secure}$ functional encryption scheme for the functionality $F$. Then, $\mathsf{FE}$ is $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{RSIM}$-Secure as well.

Proof Sketch. (**Simplified simulation.**) The security reduction uses the same ideas of those in the Sections 5 and C. Roughly, the strategy of the simulator is the following. Again, we divide the execution of the simulator in runs.

Let $(x_1, \ldots, x_\ell)$ be the vector of challenge messages chosen by the adversary and unknown to the simulator. At the beginning of the first run, the simulator executes the adversary on input ciphertexts $(\mathsf{Ct}_1, \ldots, \mathsf{Ct}_\ell)$ that encrypt dummy values.

Recall that for any $i \in [\ell]$, $\mathsf{Ct}_i[j]$ is supposed to encrypt $F(k_j, x_i)$. When the adversary issue a key-generation query $k_j$, the simulator learns $(F(k_j, x_1), \ldots, F(k_j, x_\ell))$. Then, the simulator rewinds the adversary executing it with input a new tuple of ciphertexts $(\mathsf{Ct}'_1, \ldots, \mathsf{Ct}'_n)$ where for each $i \in [\ell], j = 1, \ldots, s$, $\mathsf{Ct}'_i[j]$ encrypts $F(k_j, x_i)$.

[12] For sake of simplicity we implicitly assume that the functionality is not parameterized by the security parameter but this can be generalized easily.

After at most $s+1$ runs, the simulated ciphertext encrypts the same values as in the real game, and the simulator terminates returning the output of the adversary. This concludes the proof.

**Non-biased simulation.** We stress that this is a simplified simulation and the simulator also needs to guarantee that the output is not biased. This can be made as explained in the security reduction of Theorem 5. □

*FE with multi-bit output.* Notice that a predicate encryption scheme for predicate $P$ implies a predicate encryption scheme for the same predicate where the payload is fixed to 1 (meaning that the predicate is satisfied). This in turn implies a functional encryption for the functionality $P$ (where the evaluation algorithm of the FE scheme runs the evaluation algorithm of the PE scheme and outputs 0 if the PE scheme returns $\perp$ and 1 otherwise).

Finally, the latter implies a functional encryption scheme for the class of circuits with multi-bit output that extends $P$ in the obvious way. These implications preserve the $(\mathsf{poly}, \mathsf{poly}, \mathsf{poly})$-$\mathsf{RSIM}$- Security.