Cryptographic Agents: Towards a Unified Theory of Computing on Encrypted Data

Shashank Agrawal[1⋆], Shweta Agrawal[2], and Manoj Prabhakaran[1⋆]

University of Illinois Urbana-Champaign
{sagrawl2,mmp}@illinois.edu
Indian Institute of Technology Delhi
shweta@cse.iitd.ac.in

Abstract. We provide a new framework of *cryptographic agents* that unifies various modern "cryptographic objects" — identity-based encryption, fully-homomorphic encryption, functional encryption, and various forms of obfuscation – similar to how the Universal Composition framework unifies various multi-party computation tasks like commitment, coin-tossing and zero-knowledge proofs. These cryptographic objects can all be cleanly modeled as "*schemata*" in our framework.
Highlights of our framework include the following:

- We use a new *indistinguishability preserving* (IND-PRE) definition of security that interpolates indistinguishability and simulation style definitions, which (often) sidesteps the known impossibilities for the latter. IND-PRE-security is parameterized by the choice of the "test" family, such that by choosing different test families, one can obtain different levels of security for the same primitive (including various standard definitions in the literature).
- We present a notion of *reduction* from one schema to another and a powerful *composition theorem* with respect to IND-PRE security. We show that obfuscation is a "complete" schema under this notion, under standard cryptographic assumptions. We also provide a stricter notion of reduction ($\varDelta$-reduction) that composes even when security is only with respect to certain restricted test families of importance.
- Last but not the least, our framework can be used to model abstractions like the generic group model and the random oracle model, letting one translate a general class of constructions in these heuristic models to constructions based on *standard model assumptions*.

We also illustrate how our framework can be applied to specific primitives like obfuscation and functional encryption. We relate our definitions to existing definitions and also give new constructions and reductions between different primitives.

1 Introduction

Over the last decade or so, thanks to remarkable breakthroughs in cryptographic techniques, a wave of "cryptographic objects" — identity-based encryption, fully-

⋆ Research supported in part by NSF grant 1228856.

homomorphic encryption, functional encryption, and most recently, various forms of obfuscation — have opened up exciting new possibilities for computing on encrypted data. Initial foundational results on this front consisted of strong impossibility results. Breakthrough constructions, as they emerged, often used specialized security definitions which avoided such impossibility results. However, as these objects and their constructions have become numerous and complex, often building on each other, the connections among these disparate cryptographic objects — and among their disparate security definitions — have become increasingly confusing.

A case in point is functional encryption (FE) [80]. FE comes in numerous flavors — public key or symmetric [80,83], with or without function hiding [22,1], public or private index [24], bounded or unbounded key [79,63,61]. Each flavor has several candidate security definitions — indistinguishability based [21,80], adaptive simulation based [24], non-adaptive simulation [76], unbounded simulation [6], fully-adaptive security [72], black-box/non black-box simulation [14] to name a few. In addition, FE can be constructed from obfuscation [52] and can be used to construct property preserving encryption [77], each of which have numerous security definitions of their own [66,10,18]. It is unclear how these definitions relate, particularly as primitives are composed, resulting in a landscape cluttered with similar yet different definitions, of different yet similar primitives.

The goal of this work is to provide a clean and unifying framework for diverse cryptographic objects and their various security definitions, equipped with powerful *reductions* and *composition theorems*. In our framework, security is parametrized by a family of "test" functions — by choosing the appropriate family, we are able to place known security definitions for a given object on the same canvas, enabling comparative analysis. Our framework is general enough to model abstractions like the generic group model, letting one translate a general class of constructions in these heuristic models to constructions based on *standard model assumptions*.

Why A Framework? A unifying framework like ours has significant potential for affecting the future course of development of the theory and practice of cryptographic objects. The most obvious impact is on the definitional aspects – both positive and negative results crucially hinge on the specifics of the definition. Our framework allows one to systematically explore different definitions obtained by instantiating each component in the framework differently. We can not only "rediscover" existing definitions in this way, but also discover new definitions, both stronger and weaker than the ones in the literature. As an example, we obtain a new notion of "adaptive differing-inputs obfuscation" that leads to significant simplifications in constructions using "differing-inputs obfuscation".

The framework offers a means to identify what is common to a variety of objects, to compare them against each other by reducing one to another, to build one from the other by using our composition theorems. In addition, one may more easily identify intermediate objects of appropriate functionality and security that can be used as part of a larger construction. Another important contribution of the framework is the ability to model computational assumptions suitable for these constructions at an appropriate level of abstraction [1].

Why A *New* Framework? One might wonder if an existing framework for secure multi-party computation (MPC) — like the Universal Composition (UC) framework — cannot be used, or repurposed, to handle cryptographic objects as well. While certain elements of these frameworks (like the real/ideal paradigm) are indeed relevant beyond MPC, there are several differences between MPC and cryptographic objects which complicates this approach (which indeed was the starting point for our framework). Firstly, there is a strict syntactic requirement on schemes implementing cryptographic objects — namely, that they are non-interactive — which is absent for MPC protocols; indeed, MPC frameworks typically do not impose any constraints on the number of rounds, let alone rule out interaction. Secondly, and more importantly, the security definition in general-purpose MPC frameworks typically follow a simulation paradigm[2]. Unfortunately, such a strong security requirement is well-known to be unrealizable — e.g., the "virtual black-box" definition of obfuscation is unrealizable [10]. To be relevant, it is very important that a framework for modeling obfuscation and other objects admits weaker security definitions.

Finally, a simple framework for cryptographic objects need not model various subtleties of protocol execution in a network that the MPC frameworks model. These considerations lead us to a bare-bones framework, which can model the basic security requirements of cryptographic objects (but little else).

Cryptographic Agents Framework. Our unifying framework, called the *Cryptographic Agents framework* models one or more (possibly randomized, stateful) objects that interact with each other, so that a user with access to their codes can only learn what it can learn from the output of these objects. As a running example, functional encryption schemes could be considered as consisting of "message agents" and "key agents."

To formalize the security requirement, we use a real-ideal paradigm, but at the same time rely on an indistinguishability notion (rather than a simulation-based security notion). We informally describe the framework below.

– **Ideal Execution.** The ideal world consists of two (adversarially designed) entities — a User and a Test — who can freely interact with each other. (See

[1] cf. in secure multi-party computation, the existence of a semi-honest OT protocol is a more appropriate assumption that the existence of an enhanced trapdoor one-way permutation

[2] One exception to this is the "input-indistinguishable computation" framework of Micali, Pass and Rosen for secure function evaluation of deterministic functions [74]. Unfortunately, this framework heavily relies on interactivity of protocols (an "implicit input" is defined by a transcript; but when a party interacts with an object it received, there is no well-defined transcript), and is unsuitable for modeling cryptographic objects.

the left-hand side of Figure 1.) User is given access, via handles, to a collection of "agents" (interactive Turing Machines), maintained by $\mathcal{B}$ (a "blackbox"). User and Test are both allowed to add agents to the collection maintained by $\mathcal{B}$, but the class of agents that they can add are restricted by a *schema*.[3] The User can feed inputs to these agents, and also allow a set of them to interact with each other, in a "session." At the end of this interaction, the user obtains all the outputs from the session, and also additional handles to the agents with updated states.

Example: In a schema capturing public-key functional encryption, there are two kinds of agents – "message agents" and "key agents." A message agent simply sends out (i.e., copies into its *communication tape*) an inbuilt message, every time it is invoked. A key agent reads a message from its incoming communication tape, applies an inbuilt function to it, and copies the result to its *output tape*. The user can add only message agents to the collection maintained by $\mathcal{B}$; Test can add key agents as well. Note that the outputs that the user receives from a session involving a message agent and a key agent is the output produced by the key agent (the message agent produces no output; it only communicates its message to the key agent). [4]

– **Real Execution.** The real execution also consists of two entities, the (real-world) user (or an adversary Adv) and Test. The latter is in fact the same as in the ideal world. But in the real world, when Test requests adding an agent to the collection of agents, the user is handed a cryptographically generated object – a "cryptographic agent" – instead of a handle to this agent. The *correctness requirement* is that an honest user should be able to perform all the operations any User can in the ideal world (i.e., add new agents to the collection, and execute a session of agents, and thereby update their states) using an "execution" operation applied to the cryptographic agents. In Figure 1, $\mathcal{O}$ indicates the algorithm for encoding, and $\mathcal{E}$ indicates

[3] Here, a *schema* is analogous to a *functionality* in UC security. Thus different primitives like functional encryption and fully-homomorphic encryption are specified by different schemata.

[4] For functional encryption, neither inputs to agents nor their states are relevant, as the message and key agents have all the relevant information built in. However, obfuscation is most directly modeled by non-interactive agents that take an input, and modeling fully homomorphic encryption requires agents that maintain state.

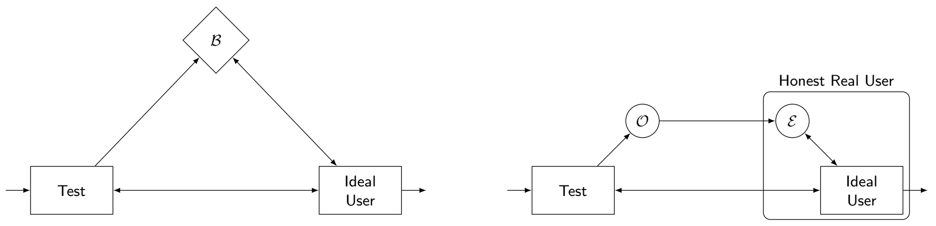

Fig. 1: The ideal world (on the left) and the real world with an honest user.

a procedure that applies an algorithm for session executions, as requested by the User. (However, an adversarial user Adv in the real world may analyze the cryptographic agents in anyway it wants.)

– **Security Definition.** We define IND-PRE (for indistinguishability preserving) security, which requires that *if* a Test is such that a certain piece of information about it (modeled as an input bit) remains hidden from every user in the ideal world, *then* that information should stay hidden from every user that interacts with Test in the real world as well. Note that we do not require that the view in the real world can be simulated in the ideal world. In the real world we require all entities to be computationally bounded. But in the *ideal* world, we may consider users that are computationally bounded or unbounded (possibly with a limit on the number of sessions it can invoke). Another variable in our definition is the family of tests: by default, we consider Tests that are PPT; but we may consider Tests from a family Γ, in which case the resulting security definition is termed Γ-IND-PRE security. These choices allow us to model different levels of security, which translate to various natural notions of security for specific schemata.

Our Contributions. Our main contribution is a new model of cryptographic computation, that unifies and extends primitives for computing on encrypted data such as obfuscation, functional encryption, fully homomorphic encryption, property preserving encryption, and such others. One can consider our framework analogous to the now-standard approach in secure multi-party computation (MPC) (e.g., following [58,39]) that uses *a common paradigm to abstract the security guarantees in a variety of different tasks* like commitments, zero-knowledge proofs, coin-flipping, oblivious-transfer, etc. While we anticipate several refinements and extensions to the framework presented here, we consider that, thanks to its simplicity, the current model already provides important insight about the "right" security notions for the primitives we capture, and opens up a wealth of new questions and connections for further investigation.

The list of technical results in this paper could be viewed in two parts: contributions to the foundational aspects of cryptographic objects, and contributions to specific objects of interest (mainly, obfuscation, functional encryption and assumptions related to (bi/multi-linear) groups). Some of our specific contributions to the foundational aspects of this area are as follows.

– We first define a general framework of cryptographic agents that can be instantiated for different primitives using different *schemata*. The resulting security definition, called Γ-IND-PRE-security is parameterized by a test family Γ.
For natural choices of Γ, these definitions tend to be not only stronger than standard definitions, but also easier to work with in larger constructions (see next). For some schemata, like obfuscation and functional encryption, choosing Γ to be the family of all PPT tests can lead to definitions that are known to be impossible to realize. But more restricted test families can be

used to capture existing definitions (with candidate constructions) exactly: we identify $\varDelta$, $\varDelta_{\mathsf{det}}$ and $\varDelta^*$ (defined later) as important test families that do this for obfuscation and/or functional encryption.

$\varDelta$-IND-PRE-security is of particular interest, because for each of the example primitives we consider in this paper — obfuscation, functional encryption, fully-homomorphic encryption and property-preserving encryption — $\varDelta$-IND-PRE-security for the corresponding schema implies the standard security definitions (that are not known to be impossible to realize) in the literature, and yet, is not known to be impossible to realize.

– We present a notion of reduction from one schema to another[5], and a composition theorem. This provides a modular means to build and analyze secure schemes for a complicated schema based on those for simpler schemata. Further, reduction provides a way to study, in abstract, relative complexity of different schemata: e.g., general purpose obfuscation turns out to be a "complete" schema under this notion.
– The notion of reduction mentioned above composes for $\varGamma_{\mathsf{ppt}}$-IND-PRE-security where $\varGamma_{\mathsf{ppt}}$ is the class of all probabilistic polynomial time (PPT) tests. Unfortunately, obfuscation (and hence, any other complete schema) can be shown to be unrealizable under this definition. Hence, we present a more structured notion of reduction, called $\varDelta$-reduction, that composes with respect to $\varDelta$-IND-PRE-security as well.

These basic results have several important implications to specific primitives of interest. In this paper, we initiate the study of a few such primitives in our framework (and leave others to future work).

– **Functional Encryption.** Our framework provides a unified method to capture all variants of FE using just a few basic schemata by employing different test families. For concreteness, below we focus on public-key FE.
  • *Defining FE With and Without Function-Hiding.* Function-hiding (public-key) FE had proved difficult to define satisfactorily [22,23,1]. The IND-PRE framework provides a way to obtain a natural and general definition of this primitive. We present a simple schema $\boldsymbol{\Sigma}_{\text{FH-FE}}$ to capture the security guarantees of function-hiding FE; a similar schema $\boldsymbol{\Sigma}_{\text{FE}}$ captures FE without function-hiding.
  • *Hierarchy of Security Requirements.* By using different test families, we obtain a hierarchy of security notions for FE (with and without function-hiding), $\varDelta_{\mathsf{det}}$-IND-PRE $\Leftarrow$ $\varDelta$-IND-PRE $\Leftarrow$ IND-PRE $\Leftarrow$ SIM (see Footnote 5). Of these, $\varDelta_{\mathsf{det}}$-IND-PRE security for FE without function-hiding is equivalent to the standard notion of security used currently [21,80]. The strongest one, SIM security, is impossible for general function families [24,14,6].

[5] Our reduction uses a simulation-based security requirement. Thus, among other things, it also provides a means for capturing simulation-based security definition: we say that a scheme $\varPi$ is a $\varGamma$-SIM-secure scheme for a schema $\boldsymbol{\Sigma}$ if $\varPi$ reduces $\boldsymbol{\Sigma}$ to the null-schema.

  - *Constructions.* We present new constructions for Δ-IND-PRE secure FE (both with and without function hiding) for all polynomial-time computable functions. We also present an IND-PRE secure FE for the inner product functionality. Two of these constructions are in the form of reductions (a Δ-reduction to an obfuscation schema, and a (standard) reduction to a "bilinear generic group" schema, which are described below). Also, the first two constructions crucially rely on Δ-IND-PRE-security of obfuscation (i.e., adaptive differing-inputs obfuscation), thereby considerably simplifying the constructions and the analysis compared to those in recent work [8,28] which use (non-adaptive) differing-inputs obfuscation.
- **Obfuscation.** We study in detail, the various notions of obfuscation in the literature, and relate them to Γ-IND-PRE-security for various test families Γ. Our strongest definition of this form, which considers the family of all PPT tests, turns out to be impossible. Our definition is conceptually "weaker" than the virtual black-box simulation definition (in that it does not require a simulator), but the impossibility result of Barak et al. [10] continues to apply to this definition. To circumvent the impossibility, we identify three test families, Δ, Δ^* and Δ_{det}, such that Δ_{det}-IND-PRE-security is *equivalent* to indistinguishability obfuscation, Δ^*-IND-PRE-security is equivalent to differing inputs obfuscation, and Δ-IND-PRE-security implies both the above. We state a new definition for the security of obfuscation – *adaptive differing-inputs obfuscation* – which is equivalent Δ-IND-PRE-security. Informally, it is the same as differing inputs obfuscation, but an adversary is allowed to *interact* with the "sampler" (which samples two circuits one of which will be obfuscated and presented to the adversary as a challenge), even after it receives the obfuscation. Such a notion was independently introduced in [7].
- **Using the Generic Group in the Standard Model.** One can model random oracles and the generic group model as schemata. An assumption that such a schema has an IND-PRE-secure scheme is a standard model assumption, and to the best of our knowledge, not ruled out by the techniques in the literature. This is because, IND-PRE-security captures only certain indistinguishability guarantees of the generic group model, albeit in a broad manner (by considering arbitrary tests). Indeed, for random oracles, such an assumption is implied by (for instance) virtual black-box secure obfuscation of point-functions, a primitive that has plausible candidates in the literature. The generic group schema (as well as its bilinear version) is a highly versatile resource used in several constructions, including that of cryptographic objects that can be modeled as schemata. Such constructions can be considered as *reductions* to the generic group schema. Combined with our composition theorem, this creates a recipe for standard model constructions under a strong, but simple to state, computational assumption.

  We give such an example for obtaining a standard model *function-hiding* public-key FE scheme for inner-product predicates (for which a satisfactory general security definition has also been lacking).
- **Other Primitives.** Our model is extremely flexible, and can easily capture most cryptographic objects for which an indistinguishability security notion

is required. This includes witness encryption, functional witness encryption, fully homomorphic encryption (FHE), property-preserving encryption (PPE) etc. We discuss a couple of them – FHE and PPE – to illustrate this. We can model FHE using (stateful) cryptographic agents. The resulting security definition, even with the test family Δ_{det}, implies the standard definition in the literature, with the additional requirement that a ciphertext does not reveal how it was formed, even given the decryption key. For PPE, we show that an Δ_{det}-$\mathsf{IND\text{-}PRE}$ secure scheme for the PPE schema is in fact equivalent to a scheme that satisfies the standard definition of security for PPE.

Related Work. Here, we provide a short (non exhaustive) summary of related work on the objects that are studied in this paper.

Program Obfuscation. Program Obfuscation is the task of garbling a given program so that the input-output behavior is retained, but everything else about the program is hidden. The formal study of program obfuscation was initiated by Barak et al. [10] who showed that the strongest possible notion of security, called *virtual black box* security was impossible to achieve for general circuits. To address this, they defined weaker notions of security, such as *indistinguishability obfuscation* (denoted by $\mathsf{I\text{-}Obf}$), which states that for two equivalent circuits C_0 and C_1, their obfuscations should be computationally indistinguishable. A related but stronger security notion defined by [10] was that of *differing input obfuscation* (denoted by $\mathsf{DI\text{-}Obf}$), which further requires that an adversary who can distinguish between C_0 and C_1 can be used to *extract* an input on which the two circuits differ.

Despite these weakenings, the area of program obfuscation was plagued by impossibilities [71,62,67] for a long time, with few positive results, often for very specialized classes of functions [38,42,86,43,68,40]. This state of affairs, however, has improved significantly in recent times. We now have program obfuscators for complex functionalities such as conjunctions [32], d-CNF formulas [33], circuits [52,34,44] and even Turing machines [8], in weaker models of computation such as the generic graded encoding scheme model [32,33,34,8], the generic colored matrix model [52] and the idealized pseudo free group model [44].

These constructions are proven secure under different notions of security: virtual black box, $\mathsf{I\text{-}Obf}$, $\mathsf{DI\text{-}Obf}$. Alongside, several new applications have been developed for $\mathsf{IP\text{-}Obf}$ [81] and $\mathsf{DI\text{-}Obf}$ [8,28]. There is a growing research effort in exploring alternate notions of obfuscation [19,54].

Functional Encryption. Functional encryption generalizes public key encryption to allow fine grained access control on encrypted data. In functional encryption, a user can be provided with a secret key corresponding to a function f, denoted by SK_f. Given SK_f and ciphertext $\mathsf{CT}_x = \mathsf{Encrypt}(x)$, the user may run the decryption procedure to learn $f(x)$. Security of the system guarantees that nothing beyond $f(x)$ can be learned from CT_x and SK_f. Functional encryption systems traditionally focused on restricted classes of functions such as the identity function [82,21,47,27,56,46,3,4], membership checking [26], boolean formulas

[65,16,70], inner product functions [69,70,5] and more recently, even regular languages [85]. Recent times saw constructions for more general classes of functions: Gorbunov et al. [64] and Garg et al. [53] provided the first constructions for an important subclass of FE called "public index FE" for all circuits, Goldwasser et al. [61] constructed succinct simulation-secure single-key FE scheme for all circuits, Garg et al. [52] constructed multi-key FE schemes for all circuits while Goldwasser et al. and Ananth et al. [60,8] constructed FE for Turing machines.

Fully homomorphic encryption. Fully homomorphic encryption allows a user to evaluate a circuit C on encrypted messages $\{\mathsf{CT}_i = \mathsf{Encrypt}(x_i)\}_{i\in[n]}$ so that $\mathsf{Decrypt}\big(C(\mathsf{CT}_1,\ldots,\mathsf{CT}_n)\big) = C(x_1,\ldots,x_n)$. Since the first breakthrough construction by Gentry [55], extensive research effort has been focused on providing improvements [49,35,36,31,30,37,57].

2 Preliminaries

To formalize the model of cryptographic agents, we shall use the standard notion of probabilistic interactive Turing Machines (ITM) with some modifications (see below). To avoid cumbersome formalism, we keep the description somewhat informal, but it is straightforward to fully formalize our model. We shall also not attempt to define the model in its most generality, for the sake of clarity.

In our case an ITM has separate tapes for input, output, incoming communication, outgoing communication, randomness and work-space.

Definition 1 (Agents and Family of Agents). *An agent is an interactive Turing Machine, with the following modifications:*

- *There is a special read-only parameter tape, which always consists of a security parameter κ, and possibly other parameters.*
- *There is an a priori restriction on the size of all the tapes other than the randomness tape (including input, communication and work tapes), as a function of the security parameter.*
- *There is a special* blocking state *such that if the machine enters such a state, it remains there if the input tape is empty. Similarly, there are blocking states which let the machine block if any combination of the communication tape and the input tape is empty.*

An agent family *is a maximal set of agents with the same program (i.e., state space and transition functions), but possibly different contents in their parameter tapes. We also allow an agent family to be the empty set $\emptyset$.*

We can allow *non-uniform agents* by allowing an additional advice tape. Our framework and basic results work in the uniform and non-uniform model equally well.

Note that an agent who enters a blocking state can move out of it if its configuration is changed by adding a message to its input tape and/or communication tape. However, if the agent enters a halting state, it will not move out of that state. An agent who never enters a blocking state is called a *non-reactive agent*. An agent who never reads or writes from a communication tape is called a *non-interactive agent*.

Definition 2 (Session). *A session maps a finite ordered set of agents, their configurations and inputs, to outputs and (updated) configurations of the same agents, as follows. The agents are initialized with the given inputs on their input tapes, and then executed together until they are deadlocked.*[6] *The result of applying the session is defined as the collection of outputs and configurations of the agents when the session terminates (if it terminates; if not, the result is left undefined).*

We shall be restricting ourselves to collections of agents such that sessions involving them are guaranteed to terminate. Note that we have defined a session to have only an initial set of inputs, so that the outcome of a session is well-defined (without the need to specify how further inputs would be chosen).

Next we define an important notion in our framework, namely that of an *ideal agent schema*, or simply, a schema. A schema plays the same role as a functionality does in the Universal Composition framework for secure multi-party computation. That is, it specifies what is legitimate for a user to do in a system. A schema defines the families of agents that a "user" and a "test" (or authority) are allowed to create.

Definition 3 (Ideal Agent Schema). *A (well-behaved)* ideal agent schema $\boldsymbol{\Sigma} = (\mathcal{P}_{\mathsf{auth}}, \mathcal{P}_{\mathsf{user}})$ *(or simply schema) is a pair of agent families, such that there is a polynomial* poly *such that for any session of agents belonging to* $\mathcal{P}_{\mathsf{auth}} \cup \mathcal{P}_{\mathsf{user}}$ *(with any inputs and any configurations, with the same security parameter κ), the session terminates within* $\mathrm{poly}(\kappa, t)$ *steps, where t is the number of agents in the session.*

Other Notation. If X and Y are a family of binary random variables (one for each value of κ), we write $X \approx Y$ if there is a negligible function negl such that $|\Pr[X = 1] - \Pr[Y = 1]| \leq \mathrm{negl}(\kappa)$. For two systems M and M', we say $M \approxeq M'$ if the two systems are indistinguishable to an interactive PPT distinguisher.

[6] More precisely, the first agent is executed till it enters a blocking or halting state, and then the second and so forth, in a round-robin fashion, until all the agents remain in blocking or halting states for a full round. After each execution of an agent, the contents of its outgoing communication tape are interpreted as an ordered sequence of messages to each of the other agents in the session (some or all of them possibly being empty messages), and copied over to the respective agents' incoming communication tapes.

3 Defining Cryptographic Agents

In this section we define what it means for a cryptographic agent scheme to securely implement a given ideal agent schema. Intuitively, the security notion is of *indistinguishability preservation*: if two executions using an ideal schema are indistinguishable, we require them to remain indistinguishable when implemented using a cryptographic agent scheme. While it consists of several standard elements of security definitions, indistinguishability preservation as defined here is novel, and potentially of broader interest.

Ideal World. The ideal system for a schema $\boldsymbol{\Sigma}$ consists of two parties Test and User and a fixed third party $\mathcal{B}[\boldsymbol{\Sigma}]$ (for "black-box"). All three parties are probabilistic polynomial time (PPT) ITMs, and have a security parameter κ built-in. We shall explicitly refer to their random-tapes as r, s and t. Test receives a "secret bit" b as input and User produces an output bit b'. The interaction between User, Test and $\mathcal{B}[\boldsymbol{\Sigma}]$ can be summarized as follows:

- **Uploading agents.** Let $\boldsymbol{\Sigma} = (\mathcal{P}_{\mathsf{auth}}, \mathcal{P}_{\mathsf{user}})$ where we associate $\mathcal{P}_{\mathsf{test}} := \mathcal{P}_{\mathsf{auth}} \cup \mathcal{P}_{\mathsf{user}}$ with Test and $\mathcal{P}_{\mathsf{user}}$ with User. Test and User can, at any point, choose an agent from its agent family and send it to $\mathcal{B}[\boldsymbol{\Sigma}]$. More precisely, User can send a string to $\mathcal{B}[\boldsymbol{\Sigma}]$, and $\mathcal{B}[\boldsymbol{\Sigma}]$ will instantiate an agent $\mathcal{P}_{\mathsf{user}}$, with the given string (along with its own security parameter) as the contents of the parameter tape, and all other tapes being empty. Similarly, Test can send a string and a bit indicating whether it is a parameter for $\mathcal{P}_{\mathsf{auth}}$ or $\mathcal{P}_{\mathsf{user}}$, and it is used to instantiate an agent $\mathcal{P}_{\mathsf{auth}}$ or $\mathcal{P}_{\mathsf{user}}$, accordingly [7]. Whenever an agent is instantiated, $\mathcal{B}[\boldsymbol{\Sigma}]$ sends a unique handle (a serial number) for that agent to User; the handle also indicates whether the agent belongs to $\mathcal{P}_{\mathsf{auth}}$ or $\mathcal{P}_{\mathsf{user}}$.
- **Request for Session Execution.** At any point in time, User may request an execution of a session, by sending an ordered tuple of handles $(h_1, \ldots, h_t)$ (from among all the handles obtained thus far from $\mathcal{B}[\boldsymbol{\Sigma}]$) to specify the configurations of the agents in the session, along with their inputs. $\mathcal{B}[\boldsymbol{\Sigma}]$ reports back the outputs from the session, and also gives new handles corresponding to the configurations of the agents when the session terminated.[8] If an agent halts in a session, no new handle is given for that agent.

Observe that only User receives any output from $\mathcal{B}[\boldsymbol{\Sigma}]$; the communication between Test and $\mathcal{B}[\boldsymbol{\Sigma}]$ is one-way. (See Figure 1.)

[7] In fact, for convenience, we allow Test and User to specify multiple agents in a single message to $\mathcal{B}[\boldsymbol{\Sigma}]$.

[8] Note that if the same handle appears more than once in the tuple $(h_1, \ldots, h_t)$, it is interpreted as multiple agents with the same configuration (but possibly different inputs). Also note that after a session, the old handles for the agents are not invalidated; so a User can access a configuration of an agent any number of times, by using the same handle.

We define the random variable $\textsc{ideal}\langle \mathsf{Test}(b) \mid \boldsymbol{\Sigma} \mid \mathsf{User}\rangle$ to be the output of User in an execution of the above system, when Test gets b as input. We write $\textsc{ideal}\langle \mathsf{Test} \mid \boldsymbol{\Sigma} \mid \mathsf{User}\rangle$ in the case when the input to Test is a uniformly random bit. We also define $\textsc{Time}\langle \mathsf{Test} \mid \boldsymbol{\Sigma} \mid \mathsf{User}\rangle$ as the maximum number of steps taken by Test (with a random input), $\mathcal{B}[\boldsymbol{\Sigma}]$ and User in total.

Definition 4. *We say that* Test *is* hiding w.r.t. $\boldsymbol{\Sigma}$ *if* $\forall$ *PPT party* User*,*

$$\textsc{ideal}\langle \mathsf{Test}(0) \mid \boldsymbol{\Sigma} \mid \mathsf{User}\rangle \approx \textsc{ideal}\langle \mathsf{Test}(1) \mid \boldsymbol{\Sigma} \mid \mathsf{User}\rangle.$$

When the schema is understood, we shall refer to the property of being hiding w.r.t. a schema as simply being ideal-hiding.

Real World. A *cryptographic scheme* (or simply scheme) consists of a pair of (possibly stateful and randomized) programs $(\mathcal{O}, \mathcal{E})$, where $\mathcal{O}$ is an encoding procedure for agents in $\mathcal{P}_{\mathsf{test}}$ and $\mathcal{E}$ is an execution procedure. The real world execution for a scheme $(\mathcal{O}, \mathcal{E})$ consists of Test, a user that we shall generally denote as Adv and the encoder $\mathcal{O}$. ($\mathcal{E}$ features as part of an honest user in the real world execution: see Figure 1.) Test remains the same as in the ideal world, except that instead of sending an agent to $\mathcal{B}[\boldsymbol{\Sigma}]$, it sends it to the encoder $\mathcal{O}$. In turn, $\mathcal{O}$ encodes this agent and sends the resulting cryptographic agent to Adv.

We define the random variable $\textsc{real}\langle \mathsf{Test}(b) \mid \mathcal{O} \mid \mathsf{Adv}\rangle$ to be the output of Adv in an execution of the above system, when Test gets b as input; as before, we omit b from the notation to indicate a random bit. Also, as before, $\textsc{Time}\langle \mathsf{Test} \mid \mathcal{O} \mid \mathsf{User}\rangle$ is the maximum number of steps taken by Test (with a random input), $\mathcal{O}$ and User in total.

Definition 5. *We say that* Test *is* hiding w.r.t. $\mathcal{O}$ *if* $\forall$ *PPT party* Adv*,*

$$\textsc{real}\langle \mathsf{Test}(0) \mid \mathcal{O} \mid \mathsf{Adv}\rangle \approx \textsc{real}\langle \mathsf{Test}(1) \mid \mathcal{O} \mid \mathsf{Adv}\rangle.$$

Note that $\textsc{real}\langle \mathsf{Test} \mid \mathcal{O} \mid \mathsf{Adv}\rangle = \textsc{real}\langle \mathsf{Test} \circ \mathcal{O} \mid \emptyset \mid \mathsf{Adv}\rangle$ where $\emptyset$ stands for the null implementation. Thus, instead of saying Test is hiding w.r.t. $\mathcal{O}$, we shall sometimes say $\mathsf{Test} \circ \mathcal{O}$ is hiding (w.r.t. $\emptyset$). Also, when $\mathcal{O}$ is understood, we may simply say that Test is real-hiding.

Syntactic Requirements on $(\mathcal{O}, \mathcal{E})$. $(\mathcal{O}, \mathcal{E})$ may or may not use a "setup" phase. In the latter case we call it a *setup-free cryptographic agent scheme*, and $\mathcal{O}$ is required to be a memory-less program that takes an agent $P \in \mathcal{P}_{\mathsf{test}}$ as input and outputs a cryptographic agent that is sent to Adv. If the scheme has a setup phase, $\mathcal{O}$ consists of a triplet of memory-less programs $(\mathcal{O}_{\mathsf{setup}}, \mathcal{O}_{\mathsf{auth}}, \mathcal{O}_{\mathsf{user}})$: in the real world execution, first $\mathcal{O}_{\mathsf{setup}}$ is run to generate a secret-public key pair $(\mathsf{MSK}, \mathsf{MPK})$;[9] MPK is sent to Adv. Subsequently, when $\mathcal{O}$ receives an agent $P \in \mathcal{P}_{\mathsf{auth}}$ it will invoke $\mathcal{O}_{\mathsf{auth}}(P, \mathsf{MSK})$, and when it receives an agent $P \in \mathcal{P}_{\mathsf{user}}$,

[9] For "master" secret and public-keys, following the terminology in some of our examples.

it will invoke $\mathcal{O}_{\mathsf{user}}(P, \mathsf{MPK})$, to obtain a cryptographic agent that is then sent to Adv.

$\mathcal{E}$ is required to be memoryless as well, except that when it gives a handle to a User, it can record a string against that handle, and later when User requests a session execution, $\mathcal{E}$ can access the string recorded for each handle in the session. There is a *compactness requirement* that the size of this string is *a priori* bounded (note that the state space of the ideal agents are also *a priori* bounded). If there is a setup phase, $\mathcal{E}$ can also access MPK each time it is invoked.

$\mathsf{IND\text{-}PRE}$ Security. Now we are ready to present the security definition of a cryptographic agent scheme $(\mathcal{O}, \mathcal{E})$ implementing a schema $\boldsymbol{\Sigma}$. Below, the *honest real-world user*, corresponding to an ideal-world user User, is defined as the composite program $\mathcal{E} \circ \mathsf{User}$ as shown in Figure 1.

Definition 6. *A cryptographic agent scheme $\Pi = (\mathcal{O}, \mathcal{E})$ is said to be a Γ-*$\mathsf{IND\text{-}PRE}$-secure scheme *for a schema $\boldsymbol{\Sigma}$ if the following conditions hold.*

- *Correctness.* $\forall$ PPT User *and* $\forall$ $\mathsf{Test} \in \Gamma$, $\textsc{ideal}\langle \mathsf{Test} \mid \boldsymbol{\Sigma} \mid \mathsf{User} \rangle \approx \textsc{real}\langle \mathsf{Test} \mid \mathcal{O} \mid \mathcal{E} \circ \mathsf{User} \rangle$. *If equality holds, $(\mathcal{O}, \mathcal{E})$ is said to have perfect correctness.*
- *Efficiency. There exists a polynomial* poly *such that, $\forall$ PPT* User, $\forall \mathsf{Test} \in \Gamma$,
$$\textsc{Time}\langle \mathsf{Test} \mid \mathcal{O} \mid \mathcal{E} \circ \mathsf{User} \rangle \leq \mathrm{poly}(\textsc{Time}\langle \mathsf{Test} \mid \boldsymbol{\Sigma} \mid \mathsf{User} \rangle, \kappa).$$
- *Indistinguishability Preservation.* $\forall \mathsf{Test} \in \Gamma$,
$$\mathsf{Test} \textit{ is hiding w.r.t. } \boldsymbol{\Sigma} \Rightarrow \mathsf{Test} \textit{ is hiding w.r.t. } \mathcal{O}.$$

When Γ is the family of all PPT tests – denoted by Γ_{ppt}, we simply say that Π is an $\mathsf{IND\text{-}PRE}$*-secure scheme for $\boldsymbol{\Sigma}$.*

4 Reductions and Compositions

A fundamental question regarding (secure) computational models is that of reduction: which tasks can be reduced to which others. In the context of cryptographic agents, we ask which schemata can be reduced to which other schemata. We shall use a strong *simulation-based* notion of reduction. While a simulation-based security notion for general cryptographic agents or even just obfuscations (i.e., virtual black-box obfuscation) is too strong to exist, it is indeed possible to meet a simulation-based notion for reductions between schemata. This is *analogous to the situation in Universally Composable security*, where sweeping impossibility results exist for UC secure realizations in the plain model, but there is a rich structure of UC secure reductions among functionalities.

A *hybrid scheme* $(\mathcal{O}, \mathcal{E})^{\boldsymbol{\Sigma}^*}$ is a cryptographic agent scheme in which $\mathcal{O}$ and $\mathcal{E}$ have access to $\mathcal{B}[\boldsymbol{\Sigma}^*]$, as shown in Figure 2 (in the middle), where $\boldsymbol{\Sigma}^* =$

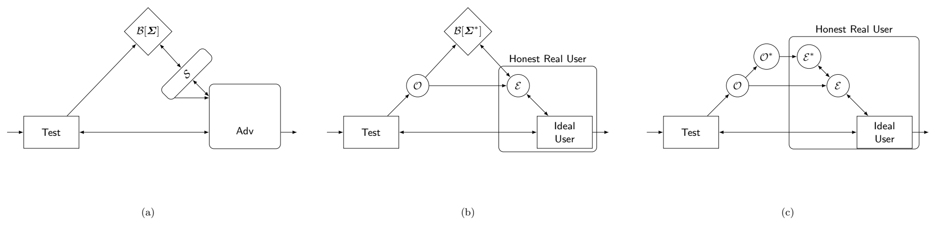

Fig. 2: $(\mathcal{O}, \mathcal{E})$ in (b) is a reduction from schema $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$. The security requirement is that no adversary Adv in the system (a) can distinguish that execution from an execution of the system in (b) (with Adv taking the place of honest real user). The correctness requirement is that the ideal User in (b) behaves the same as the ideal User interacting directly with $\mathcal{B}[\boldsymbol{\Sigma}]$ (as in Figure 1(a)). (c) shows the composition of the hybrid scheme $(\mathcal{O}, \mathcal{E})^{\boldsymbol{\Sigma}^*}$ with a scheme $(\mathcal{O}^*, \mathcal{E}^*)$ that $\mathsf{IND\text{-}PRE}$-securely implements $\boldsymbol{\Sigma}^*$.

$(\mathcal{P}^*_{\mathsf{auth}}, \mathcal{P}^*_{\mathsf{user}})$. If $\mathcal{O}$ has a setup phase, we require that $\mathcal{O}_{\mathsf{user}}$ uploads agents only in $\mathcal{P}^*_{\mathsf{user}}$ (but $\mathcal{O}_{\mathsf{auth}}$ can upload any agent in $\mathcal{P}^*_{\mathsf{auth}} \cup \mathcal{P}^*_{\mathsf{user}}$). In general, the honest user would be replaced by an adversarial user Adv. Note that the output bit of Adv in such a system is given by the random variable $\text{IDEAL}\langle \mathsf{Test} \circ \mathcal{O} \mid \boldsymbol{\Sigma}^* \mid \mathsf{Adv} \rangle$, where $\mathsf{Test} \circ \mathcal{O}$ denotes the combination of Test and $\mathcal{O}$ as in Figure 2.

Definition 7 (Reduction). *We say that a (hybrid) cryptographic agent scheme $\Pi = (\mathcal{O}, \mathcal{E})$ reduces $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$ with respect to Γ, if there exists a* PPT *simulator $\mathcal{S}$ such that $\forall$* PPT User*,*

1. *Correctness:* $\forall \mathsf{Test} \in \Gamma_{\mathsf{ppt}}$, $\text{IDEAL}\langle \mathsf{Test} \mid \boldsymbol{\Sigma} \mid \mathsf{User} \rangle \approx \text{IDEAL}\langle \mathsf{Test} \circ \mathcal{O} \mid \boldsymbol{\Sigma}^* \mid \mathcal{E} \circ \mathsf{User} \rangle$.
2. *Simulation:* $\forall \mathsf{Test} \in \Gamma$, $\text{IDEAL}\langle \mathsf{Test} \mid \boldsymbol{\Sigma} \mid \mathcal{S} \circ \mathsf{User} \rangle \approx \text{IDEAL}\langle \mathsf{Test} \circ \mathcal{O} \mid \boldsymbol{\Sigma}^* \mid \mathsf{User} \rangle$.

If $\Gamma = \Gamma_{\mathsf{ppt}}$, we simply say Π reduces $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$. If there exists a scheme that reduces $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$, then we say $\boldsymbol{\Sigma}$ reduces to $\boldsymbol{\Sigma}^*$. (Note that correctness is required for all PPT Test*, and not just in Γ.)*

Figure 2 illustrates a reduction. It also shows how such a reduction can be composed with an $\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}^*$. Below, we shall use $(\mathcal{O}', \mathcal{E}') = (\mathcal{O} \circ \mathcal{O}^*, \mathcal{E}^* \circ \mathcal{E})$ to denote the composed scheme in Figure 2(c).[10]

Theorem 1 (Composition). *For any two schemata, $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}^*$, if $(\mathcal{O}, \mathcal{E})$ reduces $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$ and $(\mathcal{O}^*, \mathcal{E}^*)$ is an* $\mathsf{IND\text{-}PRE}$ *secure scheme for $\boldsymbol{\Sigma}^*$, then $(\mathcal{O} \circ \mathcal{O}^*, \mathcal{E}^* \circ \mathcal{E})$ is an* $\mathsf{IND\text{-}PRE}$ *secure scheme for $\boldsymbol{\Sigma}$.*

[10] If $(\mathcal{O}, \mathcal{E})$ and $(\mathcal{O}^*, \mathcal{E}^*)$ have a setup phase, then it is implied that $\mathcal{O}'_{\mathsf{auth}} = \mathcal{O}_{\mathsf{auth}} \circ \mathcal{O}^*_{\mathsf{auth}}$, $\mathcal{O}'_{\mathsf{user}} = \mathcal{O}_{\mathsf{user}} \circ \mathcal{O}^*_{\mathsf{user}}$; invoking $\mathcal{O}'_{\mathsf{setup}}$ invokes both $\mathcal{O}_{\mathsf{setup}}$ and $\mathcal{O}^*_{\mathsf{setup}}$, and may in addition invoke $\mathcal{O}^*_{\mathsf{auth}}$ or $\mathcal{O}^*_{\mathsf{user}}$.

Proof sketch: Let $(\mathcal{O}', \mathcal{E}') = (\mathcal{O} \circ \mathcal{O}^*, \mathcal{E}^* \circ \mathcal{E})$. Also, let $\mathsf{Test}' = \mathsf{Test} \circ \mathcal{O}$ and $\mathsf{User}' = \mathcal{E} \circ \mathsf{User}$. To show correctness, note that for any User, we have

$$\begin{aligned}
\textsc{real}\langle \mathsf{Test} \mid \mathcal{O}' \mid \mathcal{E}' \circ \mathsf{User} \rangle &= \textsc{real}\langle \mathsf{Test}' \mid \mathcal{O}^* \mid \mathcal{E}^* \circ \mathsf{User}' \rangle \\
&\overset{(a)}{\approx} \textsc{ideal}\langle \mathsf{Test}' \mid \boldsymbol{\Sigma}^* \mid \mathsf{User}' \rangle \\
&= \textsc{ideal}\langle \mathsf{Test} \circ \mathcal{O} \mid \boldsymbol{\Sigma}^* \mid \mathcal{E} \circ \mathsf{User} \rangle \\
&\overset{(b)}{\approx} \textsc{ideal}\langle \mathsf{Test} \mid \boldsymbol{\Sigma} \mid \mathsf{User} \rangle
\end{aligned}$$

where (a) follows from the correctness guarantee of IND-PRE security of $(\mathcal{O}^*, \mathcal{E}^*)$, and (b) follows from the correctness guarantee of $(\mathcal{O}, \mathcal{E})$ being a reduction of $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$. (The other equalities are by regrouping the components in the system.)

It remains to prove that for all PPT Test, if Test is hiding w.r.t. $\boldsymbol{\Sigma}$ then Test is hiding w.r.t. $\mathcal{O}'$.

Firstly, we argue that Test is hiding w.r.t. $\boldsymbol{\Sigma} \Rightarrow \mathsf{Test}'$ is hiding w.r.t. $\boldsymbol{\Sigma}^*$. Suppose Test' is not hiding w.r.t. $\boldsymbol{\Sigma}^*$. This implies that there is some User such that $\textsc{ideal}\langle \mathsf{Test}'(0) \mid \boldsymbol{\Sigma}^* \mid \mathsf{User} \rangle \not\approx \textsc{ideal}\langle \mathsf{Test}'(1) \mid \boldsymbol{\Sigma}^* \mid \mathsf{User} \rangle$. But, by security of the reduction $(\mathcal{O}, \mathcal{E})$ of $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$, $\textsc{ideal}\langle \mathsf{Test}'(b) \mid \boldsymbol{\Sigma}^* \mid \mathsf{User} \rangle \approx \textsc{ideal}\langle \mathsf{Test}(b) \mid \boldsymbol{\Sigma} \mid \mathcal{S} \circ \mathsf{User} \rangle$, for $b = 0, 1$. Then, $\textsc{ideal}\langle \mathsf{Test}(0) \mid \boldsymbol{\Sigma} \mid \mathcal{S} \circ \mathsf{User} \rangle \not\approx \textsc{ideal}\langle \mathsf{Test}(1) \mid \boldsymbol{\Sigma} \mid \mathcal{S} \circ \mathsf{User} \rangle$, showing that Test is not hiding w.r.t. $\boldsymbol{\Sigma}$. Thus we have,

$$\begin{aligned}
\mathsf{Test} \text{ is hiding w.r.t. } \boldsymbol{\Sigma} &\Rightarrow \mathsf{Test}' \text{ is hiding w.r.t. } \boldsymbol{\Sigma}^* \\
&\Rightarrow \mathsf{Test}' \text{ is hiding w.r.t. } \mathcal{O}^* \\
&\Rightarrow \mathsf{Test} \text{ is hiding w.r.t. } \mathcal{O}',
\end{aligned}$$

where the second implication is due to the fact that $(\mathcal{O}^*, \mathcal{E}^*)$ is an IND-PRE secure implementation of $\boldsymbol{\Sigma}^*$, and the last implication follows by observing that for any Adv, we have $\textsc{real}\langle \mathsf{Test}' \mid \mathcal{O}^* \mid \mathsf{Adv} \rangle = \textsc{real}\langle \mathsf{Test} \mid \mathcal{O}' \mid \mathsf{Adv} \rangle$ (by regrouping the components). □

Note that in the above proof, we invoked the security guarantee of $(\mathcal{O}^*, \mathcal{E}^*)$ only with respect to tests of the form $\mathsf{Test} \circ \mathcal{O}$. Let $\Gamma \circ \mathcal{O} = \{\mathsf{Test} \circ \mathcal{O} | \mathsf{Test} \in \Gamma\}$. Then we have the following generalization.

Theorem 2 (Generalized Composition). *For any two schemata, $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}^*$, if $(\mathcal{O}, \mathcal{E})$ reduces $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$ and $(\mathcal{O}^*, \mathcal{E}^*)$ is a $(\Gamma \circ \mathcal{O})$-IND-PRE secure scheme for $\boldsymbol{\Sigma}^*$, then $(\mathcal{O} \circ \mathcal{O}^*, \mathcal{E}^* \circ \mathcal{E})$ is a Γ-IND-PRE secure scheme for $\boldsymbol{\Sigma}$.*

Theorem 3 (Transitivity of Reduction). *For any three schemata, $\boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \boldsymbol{\Sigma}_3$, if $\boldsymbol{\Sigma}_1$ reduces to $\boldsymbol{\Sigma}_2$ and $\boldsymbol{\Sigma}_2$ reduces to $\boldsymbol{\Sigma}_3$, then $\boldsymbol{\Sigma}_1$ reduces to $\boldsymbol{\Sigma}_3$.*

Proof sketch: If $\Pi_1 = (\mathcal{O}_1, \mathcal{E}_1)$ and $\Pi_2 = (\mathcal{O}_2, \mathcal{E}_2)$ are schemes that carry out the reduction of $\boldsymbol{\Sigma}_1$ to $\boldsymbol{\Sigma}_2$ and that of $\boldsymbol{\Sigma}_2$ to $\boldsymbol{\Sigma}_3$, respectively, we claim that the scheme $\Pi = (\mathcal{O}_1 \circ \mathcal{O}_2, \mathcal{E}_2 \circ \mathcal{E}_1)$ is a reduction of $\boldsymbol{\Sigma}_1$ to $\boldsymbol{\Sigma}_3$. The correctness of this reduction follows from the correctness of the given reductions. Further, if $\mathcal{S}_1$ and $\mathcal{S}_2$ are the simulators associated with the two reductions, we can define a simulator $\mathcal{S}$ for the composed reduction as $\mathcal{S}_2 \circ \mathcal{S}_1$. $\square$

5 Restricted Test Families: $\boldsymbol{\Delta}$, $\boldsymbol{\Delta}^*$ and $\boldsymbol{\Delta}_{\mathsf{det}}$

In order to capture various notions of security, we define various corresponding families of test functions. For some schemata of interest, such as obfuscation, there exist no $\mathsf{IND\text{-}PRE}$ secure schemes (see the full version [2] for details). Restricted test families are also useful to bypass these impossibilities.

We remark that one could define test families specifically adapted to the existing security definitions of various primitives, but our goal is to provide general test families *that apply meaningfully to all primitives*, and also, would support a composable notion of reduction. Towards this we propose the following sub-class of PPT tests, called Δ. Intuitively Δ is a set of tests that reveal everything about the agents it sends to the user except for one bit b. This exactly captures indistinguishability style definitions such as indistinguishability obfuscation, differing inputs obfuscation, indistinguishability style FE and such others.

We formalize this intuition as follows: for $\mathsf{Test} \in \Delta$, each time Test sends an agent to $\mathcal{B}[\boldsymbol{\Sigma}]$, it picks two agents (P_0, P_1). Both the agents are sent to User, and P_b is sent to $\mathcal{B}[\boldsymbol{\Sigma}]$ (where b is the secret bit input to Test). Except for selecting the agent to be sent to $\mathcal{B}[\boldsymbol{\Sigma}]$, Test is oblivious to the bit b. It will be convenient to represent $\mathsf{Test}(b)$ (for $b \in \{0,1\}$) as $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}(b)$, where D is a PPT party which communicates with User, and outputs pairs of the form (P_0, P_1) to $\mathfrak{c}$; $\mathfrak{c}$ sends both the agents to User, and also forwards them to $\mathfrak{s}$; $\mathfrak{s}(b)$ forwards P_b to $\mathcal{B}[\boldsymbol{\Sigma}]$ (and sends nothing to User).

As we shall see, for both obfuscation and functional encryption, Δ-$\mathsf{IND\text{-}PRE}$-security is indeed stronger than all the standard indistinguishability based security definitions in the literature.

But a drawback of restricting to a strict subset of all PPT tests is that the composition theorems (Theorem 1 and Theorem 3) do not hold any more. This is because, these composition theorems crucially relied on being able to define $\mathsf{Test}' = \mathsf{Test} \circ \mathcal{O}$ as a member of the test family, where $\mathcal{O}$ was defined by the reduction (see Theorem 2). Nevertheless, as we shall see, analogous composition theorems do exist for Δ, if we enhance the definition of a reduction. At a high-level, we shall require $\mathcal{O}$ to have some natural additional properties that would let us convert $\mathsf{Test} \circ \mathcal{O}$ back to a test in Δ, if Test itself belongs to Δ.

Combining Machines: Some Notation. Before defining Δ-reduction and proving the related composition theorems, it will be convenient to introduce some additional notation. Note that the machines $\mathfrak{c}$ and $\mathfrak{s}$ above, as well as the

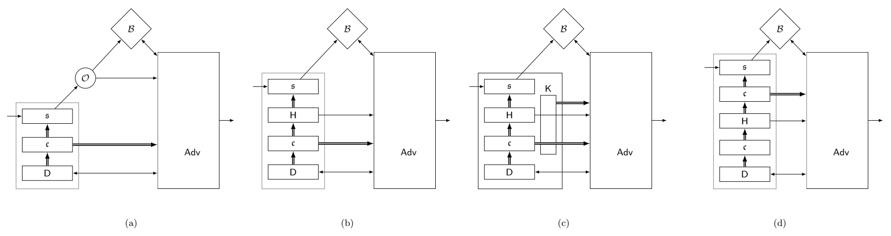

Fig. 3: Illustration of Δ and the extra requirements on Δ-reduction. (a) illustrates the structure of a test in Δ; the double-arrows indicate messages consisting of a pair of agents. The first condition on H is that (a) and (b) are indistinguishable to Adv: i.e., H can mimic the message from $\mathcal{O}$ without knowing the input bit to $\mathfrak{s}$. The second condition is that (c) and (d) are indistinguishable: i.e., K should be able to simulate the pairs of agents produced by H, based only on the input to H (copied by $\mathfrak{c}$ to Adv) and the messages from H to Adv.

program $\mathcal{O}$, have three communication ports (in addition to the secret bit that $\mathfrak{s}$ receives): in terms of Figure 3, there is an input port below, an output port above and another output port on the right, to communicate with User. (D is also similar, except that it has no input port below, and on the right, it can interact with User by sending and receiving messages.) For such machines, we use $M_1 \circ M_2$ to denote connecting the output port above M_1 to the input port of M_2. The message from $M_1 \circ M_2$ to User is defined to consist of the pair of messages from M_1 and M_2 (formatted into a single message).

We shall also consider adding machines to the right of such a machine. Specifically, we use $M \,/\, K$ to denote modifying M using a machine K that takes as input the messages output by M to User (i.e., to its right), and to each such message may *append* an additional message of its own. Recall that for two systems M and M', we say $M \approxeq M'$ if the two systems are indistinguishable to an interactive PPT distinguisher. Using this notation, we define Δ-reduction.

Definition 8 (Δ-Reduction). *We say that a (hybrid) obfuscated agent scheme $\Pi = (\mathcal{O}, \mathcal{E})$ Δ-reduces $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$ if*

1. *Π reduces $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$ with respect to Δ (as in Definition 7), and*
2. *there exists PPT H and K such that*
   (a) *for all D such that $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}$ is hiding w.r.t. $\boldsymbol{\Sigma}$, $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}(b) \circ \mathcal{O} \approxeq \mathsf{D} \circ \mathfrak{c} \circ \mathsf{H} \circ \mathfrak{s}(b)$, for $b \in \{0,1\}$;*
   (b) $\mathfrak{c} \circ \mathsf{H} \circ \mathfrak{c} \approxeq \mathfrak{c} \circ \mathsf{H} \,/\, \mathsf{K}$.

If there exists a scheme that Δ-reduces $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$, then we say $\boldsymbol{\Sigma}$ Δ-reduces to $\boldsymbol{\Sigma}^*$.

Informally, condition (a) allows us to move $\mathcal{O}$ "below" $\mathfrak{s}(b)$: note that H will need to send any messages $\mathcal{O}$ used to send to User, without knowing b. Condition

(b) requires that sending a copy of the pairs of agents output by H (by adding $\mathfrak{c}$ "above" H) is "safe": it can be simulated by K, which only sees the pair of agents that are given as input to H. Δ-reduction allows us to extend the composition theorem to Δ-IND-PRE security. We prove the following theorems in the full version of this paper [2].

Theorem 4 (Δ-Composition). *For any two schemata, $\boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}^*$, if $(\mathcal{O}, \mathcal{E})$ Δ-reduces $\boldsymbol{\Sigma}$ to $\boldsymbol{\Sigma}^*$ and $(\mathcal{O}^*, \mathcal{E}^*)$ is a Δ-IND-PRE secure implementation of $\boldsymbol{\Sigma}^*$, then $(\mathcal{O} \circ \mathcal{O}^*, \mathcal{E}^* \circ \mathcal{E})$ is a Δ-IND-PRE secure implementation of $\boldsymbol{\Sigma}$.*

Theorem 5 (Transitivity of Δ-Reduction). *For any three schemata, $\boldsymbol{\Sigma}_1$, $\boldsymbol{\Sigma}_2$, $\boldsymbol{\Sigma}_3$, if $\boldsymbol{\Sigma}_1$ Δ-reduces to $\boldsymbol{\Sigma}_2$ and $\boldsymbol{\Sigma}_2$ Δ-reduces to $\boldsymbol{\Sigma}_3$, then $\boldsymbol{\Sigma}_1$ Δ-reduces to $\boldsymbol{\Sigma}_3$.*

Other Restricted Test Families. We define two more restricted test families, Δ^* and Δ_{det}, which are of great interest for the obfuscation and functional encryption schemata. Both of these are subsets of Δ.

The family Δ_{det} simply consists of all deterministic tests in Δ. Equivalently, Δ_{det} is the class of all tests of the form $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}$, where D is a deterministic polynomial time party which communicates with User, and outputs pairs of the form (P_0, P_1) to $\mathfrak{c}$.

The family Δ^* consists of all tests in Δ which do not read any messages from User. Equivalently, Δ^* is the class of all tests of the form $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}$, where D is a PPT party which may send messages to User but does not accept any messages from User, and outputs pairs of the form (P_0, P_1) to $\mathfrak{c}$. As stated in the full version [2], the composition theorem for Δ, Theorem 4, extends to Δ^* as well.

6 Generic Group Schema

Our framework provides a method to convert a certain class of constructions — i.e., secure schemes for primitives that can be modeled as schemata — that are proven secure in heuristic models like the random oracle model [15] or the (bilinear) generic group model [84,73], into secure constructions in the standard model.

To be concrete, we consider the case of the generic group model. There are two important observations we make:

– Proving that a cryptographic scheme for a given schema $\boldsymbol{\Sigma}$ is secure in the generic group model typically amounts to a *reduction from $\boldsymbol{\Sigma}$ to a "generic group schema" $\boldsymbol{\Sigma}_{\mathrm{GG}}$*.
– The assumption that there is an IND-PRE-secure scheme Π_{GG} for $\boldsymbol{\Sigma}_{\mathrm{GG}}$ is a standard-model assumption (that does not appear to be ruled out by known results or techniques).

Combined using the composition theorem (Theorem 1), these two observations yield a standard model construction for an IND-PRE-secure scheme for $\boldsymbol{\Sigma}$.

Above, the generic group schema $\boldsymbol{\Sigma}_{\text{GG}}$ is defined in a natural way: the agents (all in $\mathcal{P}_{\mathsf{user}}$, with $\mathcal{P}_{\mathsf{auth}} = \emptyset$) are parametrized by elements of a large (say cyclic) group, and interact with each other to carry out group operations; the only output the agents produce for a user is the result of checking equality with another agent.

We formally state the assumption mentioned above:

Assumption 1 (Γ-Generic Group Agent Assumption) *There exists a Γ-*IND-PRE*-secure scheme for the generic group schema $\boldsymbol{\Sigma}_{\text{GG}}$.*

Similarly, we put forward the Γ-*Bilinear* Generic Group Agent Assumption, where $\boldsymbol{\Sigma}_{\text{GG}}$ is replaced by $\boldsymbol{\Sigma}_{\text{BGG}}$ which has three groups (two source groups and a target group), and allows the bilinear pairing operation as well.

The most useful form of these assumptions (required by the composition theorem when used with the standard reduction) is when Γ is the set of all PPT tests. However, weaker forms of this assumption (like Δ-GGA assumption, or Δ^*-GGA assumption) are also useful, if a given construction could be viewed as a stronger form of reduction (like Δ-reduction).

While this assumption may appear too strong at first sight – given the impossibility results surrounding the generic group model – we argue that it is plausible. Firstly, observe that primitives that can be captured as schemata are somewhat restricted: primitives like zero knowledge that involve simulation based security, CCA secure encryption or non-committing encryption and such others do not have an interpretation as a secure schema. Secondly, IND-PRE security is weaker than simulation based security, and its achievability is not easily ruled out (see discussion in Section 10). Also we note that such an assumption already exists in the context of another popular idealized model: the random oracle model (ROM). Specifically, consider a natural definition of the random oracle schema, $\boldsymbol{\Sigma}_{\text{RO}}$, in which the agents encode elements in a large set and interact with each other to carry out equality checks. Then, a Δ_{det}-IND-PRE-secure scheme for $\boldsymbol{\Sigma}_{\text{RO}}$ is equivalent to a point obfuscation scheme, which hides everything about the input except the output. The assumption that such a scheme exists is widely considered plausible, and has been the subject of prior research [38,42,86,40]. This fits into a broader theme of research that attempts to capture several features of the random oracle using standard model assumptions (e.g., [20,13]). The GGA assumption above can be seen as a similar approach to the generic group model, that captures only some of the security guarantees of the generic group model so that it becomes a plausible assumption in the standard model, yet is general enough to be of use in a broad class of applications.

One may wonder if we could use an even stronger assumption, by replacing the (bilinear) generic group schema $\boldsymbol{\Sigma}_{\text{GG}}$ or $\boldsymbol{\Sigma}_{\text{BGG}}$ by a multi-linear generic group schema $\boldsymbol{\Sigma}_{\text{MGG}}$, which permits black box computation of multilinear map operations [25,51]. Interestingly, this assumption is provably false if we consider $\varGamma$ to be $\varGamma_{\mathsf{ppt}}$, since there exists a reduction of obfuscation schema $\boldsymbol{\Sigma}_{\text{OBF}}$ to $\boldsymbol{\Sigma}_{\text{MGG}}$ [34,9], and we have seen that there is no IND-PRE-secure scheme for $\boldsymbol{\Sigma}_{\text{OBF}}$. On the other hand, for $\varGamma$ being $\varDelta$ or $\varDelta^*$, say, it remains a plausible assumption. Indeed, as mentioned earlier, Pass et al. introduced a computational assumption on multi-linear maps – called "semantic security" – and showed that the security of candidate constructions for indistinguishability obfuscation (aftersome modifications) can be based on semantically secure multi-linear groups [78]. We note that their assumption can be stated similar to Assumption 1, but using a multi-linear map schema and an appropriate test-family.

Falsifiability. Note that the above assumption as stated is not necessarily falsifiable, since there is no easy way to check that a given PPT test is hiding. However, it becomes falsifiable if instead of IND-PRE security, we used a modified notion of security IND-PRE′, which requires that every test which is *efficiently provably* ideal-hiding is real-hiding. We note that IND-PRE′ security suffices for all practical purposes as a security guarantee, and also suffices for the composition theorem. With this notion, to falsify the assumption, the adversary can (and must) provide a proof that a test is ideal-hiding and also exhibit a real world adversary who breaks its hiding when using the scheme.

7 Obfuscation Schema

In this section we define and study the obfuscation schema $\boldsymbol{\Sigma}_{\text{OBF}}$. In the obfuscation schema, agents are deterministic, non-interactive and non-reactive: such an agent behaves as a simple Turing machine, that reads an input, produces an output and halts.

Definition. Below, we formally define the obfuscation schema. If $\mathcal{F}$ is a family of deterministic, non-interactive and non-reactive agents, we define

$$\boldsymbol{\Sigma}_{\text{OBF}(\mathcal{F})} := (\emptyset, \mathcal{F}).$$

That is, in the ideal execution User obtains handles for computing $\mathcal{F}$. We shall consider setup-free, IND-PRE secure implementations $(\mathcal{O}, \mathcal{E})$ of $\boldsymbol{\Sigma}_{\text{OBF}(\mathcal{F})}$.

A special case of $\boldsymbol{\Sigma}_{\text{OBF}(\mathcal{F})}$ corresponds to the case when $\mathcal{F}$ is the class of all functions that can be computed within a certain amount of time. More precisely, we can define the agent family $\mathcal{U}_s$ (for *universal* computation) to consist of agents of the following form: the parameter tape, which is at most $s(\kappa)$ bits long is taken to contain (in addition to κ) the description of an arbitrary binary circuit C; on input x, $\mathcal{U}_s$ will compute and output $C(x)$ (padding or truncating x as necessary). We define the "general" obfuscation schema

$$\boldsymbol{\Sigma}_{\text{OBF}} := (\emptyset, \mathcal{P}^{\text{OBF}}_{\mathsf{user}}) := \boldsymbol{\Sigma}_{\text{OBF}(\mathcal{U}_s)},$$

for a given polynomial s. Here we have omitted s from the notation $\boldsymbol{\Sigma}_{\text{OBF}}$ and $\mathcal{P}^{\text{OBF}}_{\mathsf{user}}$ for simplicity, but it is to be understood that whenever we refer to $\boldsymbol{\Sigma}_{\text{OBF}}$ some polynomial s is implied.

Completeness of Obfuscation. We show that $\boldsymbol{\Sigma}_{\text{OBF}}$ is a complete schema with respect to schematic reduction (Definition 7). That is, *every schema* (including possibly randomized, interactive, and stateful agents) can be reduced to $\boldsymbol{\Sigma}_{\text{OBF}}$. We stress that this does not yield an IND-PRE-secure scheme for every schema (using composition), since there does not exist an IND-PRE-secure scheme for $\boldsymbol{\Sigma}_{\text{OBF}}$, as described in the full version [2]. However, if there is, say, a hardware-based IND-PRE secure implementation of $\boldsymbol{\Sigma}_{\text{OBF}}$, then this implementation can be used in a modular way to build an IND-PRE secure schema for any general functionality.

The reduction uses only standard cryptographic primitives: CCA secure public-key encryption and digital signatures. We present the full construction and proof in [2].

Relation to existing notions of Obfuscation. By using the test-families Δ_{det} and Δ^* in our framework, we can recover the notions of indistinguishability obfuscation and differing inputs obfuscation [10,11] exactly. We prove the following in the full version [2].

Lemma 1. *A set-up free* Δ_{det}-IND-PRE-*secure scheme for* $\boldsymbol{\Sigma}_{\text{OBF}}$ *(with perfect correctness) exists if and only if there exists an indistinguishability obfuscator.*

Lemma 2. *A set-up free* Δ^*-IND-PRE-*secure scheme for* $\boldsymbol{\Sigma}_{\text{OBF}}$ *(with perfect correctness) exists if and only if there exists a differing-inputs obfuscator.*

A Δ-IND-PRE secure scheme for $\boldsymbol{\Sigma}_{\text{OBF}}$ is a stronger notion than the above two notions of obfuscations (because Δ is a superset of Δ_{det} as well as Δ^*). One can give a definition of obfuscation in the traditional style, which exactly corresponds to this stronger notion. In the full version [2] we do exactly this, and term this adaptive differing inputs obfuscation. Independently, in [59] an equivalent definition appeared under the name of strong differing inputs obfuscation. Also, we note that we can model *Virtual Grey-Box Obfuscation* [17] in our framework, using an appropriate test-family and a statistical notion of hiding in Definition 4. This relies on an equivalence proven in [18] who give an indistinguishability based security definition for VGB security.

8 Functional Encryption

In this section, we present a schema $\boldsymbol{\Sigma}_{\text{FE}}$ for Functional Encryption. Although all variants of FE can [11] be captured as schemata secure against different families of test programs, we focus on adaptive secure, indistinguishability-based, public-key FE (with and without function-hiding). In Section 8.1 we introduce the schema $\boldsymbol{\Sigma}_{\text{FE}}$ for FE without function-hiding, and in Section 8.2 we introduce the schema $\boldsymbol{\Sigma}_{\text{FH-FE}}$ for function-hiding FE.

[11] Simulation-based definitions can be captured in terms of reduction to the null schema.

8.1 Functional Encryption without Function Hiding

Public-key FE without function-hiding is the most well-studied variant of FE. **Definition.** For a circuit family $\mathcal{C} = \{\mathcal{C}_\kappa\}$ and a message space $\mathcal{X} = \{\mathcal{X}_\kappa\}$, we define the schema $\boldsymbol{\Sigma}_{\text{FE}} = (\mathcal{P}^{\text{FE}}_{\mathsf{auth}}, \mathcal{P}^{\text{FE}}_{\mathsf{user}})$ as follows:

- $\mathcal{P}^{\text{FE}}_{\mathsf{user}}$: An agent $P_x \in \mathcal{P}^{\text{FE}}_{\mathsf{user}}$ simply sends x to the first agent in the session, where $x \in \mathcal{X}$ is a parameter of the agent, and halts. We will often refer to such an agent as a *message agent*.
- $\mathcal{P}^{\text{FE}}_{\mathsf{auth}}$: An agent $P_C \in \mathcal{P}^{\text{FE}}_{\mathsf{auth}}$, when invoked with input 0, outputs C (where $C \in \mathcal{C}$ is a parameter of the agent) and halts. If invoked with input 1, it reads a message $\tilde{x}$ from its incoming communication tape, writes $C(\tilde{x})$ on its output tape and halts. We will often refer to such an agent as a *function agent*.

Reducing Functional Encryption to Obfuscation. In a sequence of recent results [52,8,1,59,28,29], it was shown how to obtain various flavors of FE from various flavors of obfuscation. We investigate this connection in terms of schematic reducibility: can $\boldsymbol{\Sigma}_{\text{FE}}$ be reduced to $\boldsymbol{\Sigma}_{\text{OBF}}$? For this reduction to translate to an $\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}_{\text{FE}}$, we will need an $\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}_{\text{OBF}}$, and a composition theorem.

Our main result in this section is a Δ-reduction of $\boldsymbol{\Sigma}_{\text{FE}}$ to $\boldsymbol{\Sigma}_{\text{OBF}}$. Then, combined with a Δ-$\mathsf{IND\text{-}PRE}$ secure implementation of $\boldsymbol{\Sigma}_{\text{OBF}}$, we obtain a Δ-$\mathsf{IND\text{-}PRE}$ secure implementation of $\boldsymbol{\Sigma}_{\text{FE}}$, thanks to Theorem 4. [12]

Before explaining our reduction, we compare it with the results in [52,8,28]. At a high-level, these works could be seen as giving "$(\Gamma_{\text{FE}}, \Gamma_{\text{OBF}})$-reductions" from $\boldsymbol{\Sigma}_{\text{FE}}$ to $\boldsymbol{\Sigma}_{\text{OBF}}$ for some pair of test families Γ_{FE} and Γ_{OBF}, such that when it is composed with a Γ_{OBF}-$\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}_{\text{OBF}}$ one gets a Γ_{FE}-$\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}_{\text{FE}}$. For example, in [52], $\Gamma_{\text{OBF}} = \Delta_{\mathsf{det}}$ (corresponding to indistinguishability obfuscation); there Γ_{FE} is a test-family that captures *selective-secure* functional encryption. We do not define such $(\Gamma_{\text{FE}}, \Gamma_{\text{OBF}})$-reductions formally in this work, as they are specific to the test-families used in [52,8,28]. Instead, we propose Δ-$\mathsf{IND\text{-}PRE}$-security as a natural security notion for both obfuscation and functional encryption schemata, and provide a simpler Δ-reduction from $\boldsymbol{\Sigma}_{\text{FE}}$ to $\boldsymbol{\Sigma}_{\text{OBF}}$.

Our Construction. We shall use a simple and natural functional encryption scheme: the key for a function f is simply a description of f with a signature on it; a ciphertext of a message m is an obfuscation of a program which when given as input a signed description of a function f, returns $f(m)$ if the signature verifies (and $\perp$ otherwise). Essentially the same construction was used in [28] as

[12] Given a Δ^*-$\mathsf{IND\text{-}PRE}$ secure implementation of $\boldsymbol{\Sigma}_{\text{OBF}}$, we could obtain a Δ^*-$\mathsf{IND\text{-}PRE}$ secure implementation of $\boldsymbol{\Sigma}_{\text{FE}}$ using the same reduction. This follows from the fact that the composition theorem for Δ, Theorem 4, extends to Δ^* as well. See the full version [2] for details.

well, but they rely on "functional signatures" in which it is possible to derive keys for signing only messages satisfying an arbitrary relation. In our construction, we need only a standard digital signature scheme.

Below we describe our construction more formally, as a reduction from $\boldsymbol{\Sigma}_{\text{FE}}$ to $\boldsymbol{\Sigma}_{\text{OBF}}$ and prove that it is in fact a Δ-reduction. Let $\boldsymbol{\Sigma}_{\text{FE}} = (\mathcal{P}^{\text{FE}}_{\mathsf{auth}}, \mathcal{P}^{\text{FE}}_{\mathsf{user}})$ and $\boldsymbol{\Sigma}_{\text{OBF}} = (\emptyset, \mathcal{P}^{\text{OBF}}_{\mathsf{user}})$. We shall only describe $\mathcal{O} = (\mathcal{O}_{\mathsf{setup}}, \mathcal{O}_{\mathsf{auth}}, \mathcal{O}_{\mathsf{user}})$; $\mathcal{E}$ is naturally defined, and correctness is verified easily.

- $\mathcal{O}_{\mathsf{setup}}$ picks a pair of signing and verification keys $(\mathsf{SK}, \mathsf{VK})$ for the signature scheme as $(\mathsf{MSK}, \mathsf{MPK})$.
- $\mathcal{O}_{\mathsf{auth}}$, when given a function agent $P_f \in \mathcal{P}^{\text{FE}}_{\mathsf{auth}}$, outputs (f, σ) to be sent to $\mathcal{E}$, where f is the parameter of P_f and σ is a signature on it.
- $\mathcal{O}_{\mathsf{user}}$, when given an agent $P_m \in \mathcal{P}^{\text{FE}}_{\mathsf{user}}$ as input, uploads an agent $P_{m,\mathsf{MPK}} \in \mathcal{P}^{\text{OBF}}_{\mathsf{user}}$ to $\mathcal{B}[\boldsymbol{\Sigma}_{\text{OBF}}]$, which behaves as follows: on input (f, σ) $P_{m,\mathsf{MPK}}$ verifies that σ is a valid signature on f with respect to the signature verification key MPK; if so, it outputs $f(m)$, and else $\perp$.

In the full version [2] we show that this is indeed a Δ reduction from $\boldsymbol{\Sigma}_{\text{FE}}$ to $\boldsymbol{\Sigma}_{\text{OBF}}$.

Relation with known definitions. We examine the relation between $\mathsf{IND\text{-}PRE}$-secure Functional Encryption with standard notions of security, such as indistinguishability based security. Firstly, we show that Δ_{det}-$\mathsf{IND\text{-}PRE}$-secure is equivalent to indistinguishability secure FE.

Lemma 3. *$\exists$ a Δ_{det}-$\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}_{\text{FE}}$ iff $\exists$ an indistinguishability secure FE scheme.*

Note that an $\mathsf{IND\text{-}PRE}$ security implies Δ_{det}-$\mathsf{IND\text{-}PRE}$ security (for any schema). On the other hand, we show a strict separation between $\mathsf{IND\text{-}PRE}$ and Δ_{det}-$\mathsf{IND\text{-}PRE}$ security for FE.

Lemma 4. *$\exists$ a Δ_{det}-$\mathsf{IND\text{-}PRE}$ secure scheme for $\boldsymbol{\Sigma}_{\text{FE}}$ which is not an $\mathsf{IND\text{-}PRE}$ secure scheme for $\boldsymbol{\Sigma}_{\text{FE}}$.*

We prove these results in the full version [2].

8.2 Function-Hiding Functional Encryption

Now we turn our attention to *function-hiding* FE (with public-keys). This a significantly more challenging problem, both in terms of construction and even in terms of definition [22,23,1]. The difficulty in definition stems from the public-key nature of the encryption which allows the adversary to evaluate the function encoded in a key on arbitrary inputs of its choice: hence a security definition cannot insist on indistinguishability between two arbitrary functions. In prior work, this is often handled by restricting the security definition to involve functions that are chosen from a restricted class of distributions, such that the adversary's queries cannot reveal anything about the functions so chosen. The definition arising from our framework naturally generalizes this, as the security requirement applies to all hiding tests and thereby removes the need of specifying *ad hoc* restrictions. We only need to specify a schema for function-hiding FE, and the rest of the security definition follows from the framework.

The definition of the schema corresponding to function-hiding FE, $\boldsymbol{\Sigma}_{\text{FH-FE}} = (\mathcal{P}^{\text{FH-FE}}_{\mathsf{auth}}, \mathcal{P}^{\text{FH-FE}}_{\mathsf{user}})$, is identical to that of $\boldsymbol{\Sigma}_{\text{FE}}$, except that a function agent $P_C \in \mathcal{P}^{\text{FH-FE}}_{\mathsf{auth}}$ does not take any input, but always reads an input x from its communication tape and outputs $C(x)$. That is, the function agents do not reveal the function now.

Constructions. We present two constructions for function-hiding FE – an IND-PRE-secure scheme for the class of inner-product predicates, and a Δ-IND-PRE-secure scheme for all function families.

– The first construction is in fact an information-theoretic reduction of the schema $\boldsymbol{\Sigma}_{\text{FH-FE}(\mathsf{IP})}$ (where IP denotes the class of inner-product predicates) to the schema $\boldsymbol{\Sigma}_{\text{BGG}}$. Thus under the assumption that there is an IND-PRE secure scheme for $\boldsymbol{\Sigma}_{\text{BGG}}$, we obtain a scheme for $\boldsymbol{\Sigma}_{\text{FH-FE}}$, using Theorem 1. This construction is essentially the same as a construction in the recent work of [1], which was presented in the generic group model. Intuitively, the simulation based proof in [1] may be interpreted as a simulation based reduction from $\boldsymbol{\Sigma}_{\text{FH-FE}(\mathsf{IP})}$ to $\boldsymbol{\Sigma}_{\text{GG}}$ satisfying Definition 7.
– The second construction is for general function-hiding FE: a Δ-IND-PRE-secure scheme for $\boldsymbol{\Sigma}_{\text{FH-FE}}$, based on the assumption that a Δ-secure scheme for $\boldsymbol{\Sigma}_{\text{OBF}}$ exists. We mention that this construction is *not* a Δ-reduction. It relies on applying a signature to an obfuscation, and hence our framework cannot be used to model this as a black-box reduction (indeed, we cannot model the unforgeability requirement of signatures in our framework).

Further details of these constructions and their proofs are given in the full version [2].

9 Fully Homomorphic Encryption

In this section, we present a cryptographic agent schema $\boldsymbol{\Sigma}_{\text{FHE}}$ for Fully Homomorphic Encryption (FHE). This schema consists of *reactive agents* (i.e., agents which maintain state across invocations). For a message space $\mathcal{X} = \{\mathcal{X}\}_\kappa$ and a circuit family $\mathcal{F} = \{\mathcal{F}\}_\kappa$, we define the schema $\mathbb{P}_{\text{FHE}} = (\mathcal{P}^{\text{FHE}}_{\mathsf{test}}, \mathcal{P}^{\text{FHE}}_{\mathsf{user}})$ as follows:

– An agent $P^{\mathsf{Msg}} \in \mathcal{P}^{\text{FHE}}_{\mathsf{user}}$ is specified as follows: Its parameter tape consists of an initial value x. When invoked with an input C on its input tape, it reads a set of messages $x_2, x_3, \dots, x_t$ from its communication tapes. Then it computes $C(x_1, .., x_t)$ where x_1 is its own value (either read from the work-tape, or if the work-tape is empty, from its parameter tape). Then it updates its work-tape with this value. When invoked without an input, it sends its message to the first program in the session.

– An agent $P^{\mathsf{Dec}} \in \mathcal{P}_{\mathsf{auth}}^{\text{FHE}}$ is defined as follows: when executed with an agent P^{Msg} it reads from its communication tape a single message from P^{Msg} and outputs it.[13]

In the full version [2] we show that a semantically secure FHE scheme $\mathbb{S}_{\text{FHE}} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$ can be naturally constructed from a Δ_{det}-$\mathsf{IND\text{-}PRE}$ secure scheme for $\boldsymbol{\Sigma}_{\text{FHE}}$.

Other Examples. Several examples that we have not discussed, such as witness encryption and other flavors of FE, can also be naturally modeled as schemata. We present one more example — namely, property preserving encryption — in the full version [2], and leave the others to the full version and future work on these objects.

10 On Bypassing Impossibilities

An important aspect of our framework is that it provides a clean mechanism to tune the level of security for each primitive to a "sweet spot." The goal of such a definition is that it should imply prevalent achievable definitions while bypassing known impossibilities. The tuning is done by defining the family of tests, Γ with respect to which $\mathsf{IND\text{-}PRE}$ security is required. Below we discuss a few schemata and the definitions we recommend for them, based on what is known to be impossible.

Obfuscation. As we show in Section 7, an $\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}_{\text{OBF}}$ cannot exist. The impossibility proof relies on the fact that the test can upload an agent with (long) secrets in them. However, this argument stops applying when we restrict ourselves to tests in Δ: a test in Δ has the structure $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}$ and $\mathfrak{c}$ will reveal the agent to User. Note that then there could be at most one bit of uncertainty as to which agent was uploaded.

We point out that Δ-$\mathsf{IND\text{-}PRE}$-security is much stronger than the prevalent notions of indistinguishability obfuscation and differing inputs obfuscation, introduced by Barak et al. [10]. Indeed, to the best of our knowledge, it would be the strongest definition of obfuscation known that can plausibly exist for all functions. We also observe that Δ-$\mathsf{IND\text{-}PRE}$-secure obfuscation [14] is easier

[13] Note that there is no parameter to a $\mathcal{P}_{\mathsf{auth}}$ agent as there is only one of its kind. However, we can allow a single schema to capture multiple FHE schemes with independent keys, in which case an index for the key would be the parameter for $\mathcal{P}_{\mathsf{auth}}$ agents.

[14] or equivalently, adaptive differing-inputs obfuscation

to use in constructions than differing-inputs obfuscation, as exemplified by our constructions in the full version [2].

Functional Encryption. Public-key function-hiding FE, as modeled by $\boldsymbol{\Sigma}_{\text{FH-FE}}$, is a stronger primitive than obfuscation (for the same class of functions), as the latter can be easily reduced to the former. This means that there is no $\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}_{\text{FH-FE}}$ for general functions. We again consider Δ-$\mathsf{IND\text{-}PRE}$ security as a sweet-spot for defining function-hiding functional encryption. Indeed, prior to this definition, arguably there was no satisfactory definition for this primitive. Standard indistinguishability based definitional approaches (which typically specify an explicit test that is ideal-hiding) run into the problem that if the user is allowed to evaluate a given function on any inputs of its choice, there is no one natural ideal-hiding test. Prior works have proposed different approaches to this problem: by restricting to only a specific test [22,23], or using a relaxed simulation-based definition [1,45]. Δ-$\mathsf{IND\text{-}PRE}$ security implies the definitions of Boneh et al. [22,23], but is in general incomparable with the simulation-based definitions in [1,45]. These latter definitions can be seen as using a test in the ideal world that allows the adversary to learn more information than in the real world. Our definition does not suffer from such information leakage.

For non-function-hiding FE (captured by the schema $\boldsymbol{\Sigma}_{\text{FE}}$) too, there are many known impossibility results, when simulation-based security definitions are used [24,14,6]. At a high-level, these impossibilities followed a "compression" argument – the decryption of the challenge CT with the queried keys comprise a pseudorandom string R, but the adversary's key queries and challenge message are sequenced in such a way that to simulate its view, the simulator must somehow compress R significantly. These arguments do not apply to $\mathsf{IND\text{-}PRE}$-security simply for the reason that there is no simulator implied by it. We do not have any candidate constructions for $\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}_{\text{FE}}$, for general functions, but we leave open the possibility that it exists. We do however, provide a construction for a Δ-$\mathsf{IND\text{-}PRE}$-secure scheme for $\boldsymbol{\Sigma}_{\text{FE}}$, assuming one for $\boldsymbol{\Sigma}_{\text{OBF}}$.

Generic Group and Random Oracle. It is well known that a proof of security in the generic group or the random oracle model provides only a heuristic security guarantee. Several works have shown that these oracles are "uninstantiable," and further there are uninstantiable primitives that can be implemented in the models with such oracles [41,50,75,48,12]. These results do not contradict Assumption 1, however, because the primitives in question, like non-commiting encryptions, zero-knowledge proofs and even signature schemes, do not fit into our framework of schemata. In other words, despite its generality, schemata can be used to model only certain kind of primitives, which seem insufficient to imply such separations between the generic group model and the standard model. As such, we propose Assumption 1, with $\Gamma = \Gamma_{\mathsf{ppt}}$, the family of all PPT tests, as an assumption worthy of investigation. However, the weaker assumption, with $\Gamma = \Delta$ suffices for our construction in the full version [2], if we settle for Δ-$\mathsf{IND\text{-}PRE}$ security for the resulting scheme.

References

1. S. Agrawal, S. Agrawal, S. Badrinarayanan, A. Kumarasubramanian, M. Prabhakaran, and A. Sahai. On the practical security of inner product functional encryption. To appear in PKC 2015.
2. S. Agrawal, S. Agrawal, and M. Prabhakaran. Cryptographic agents: Towards a unified theory of computing on encrypted data. Cryptology ePrint Archive, Report 2014/480, 2014. http://eprint.iacr.org/.
3. S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT*, 2010.
4. S. Agrawal, D. Boneh, and X. Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In *CRYPTO*, 2010.
5. S. Agrawal, D. M. Freeman, and V. Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In *Asiacrypt*, 2011.
6. S. Agrawal, S. Gurbanov, V. Vaikuntanathan, and H. Wee. Functional encryption: New perspectives and lower bounds. In *Crypto*, 2013.
7. J. Alwen, M. Barbosa, P. Farshim, R. Gennaro, S. D. Gordon, S. Tessaro, and D. A. Wilson. On the relationship between functional encryption, obfuscation, and fully homomorphic encryption. In *IMA Int. Conf.*, 2013.
8. P. Ananth, D. Boneh, S. Garg, A. Sahai, and M. Zhandry. Differing-inputs obfuscation and applications. Cryptology ePrint Archive, 2013.
9. B. Barak, S. Garg, Y. T. Kalai, O. Paneth, and A. Sahai. Protecting obfuscation against algebraic attacks. In *Eurocrypt*, 2014.
10. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *CRYPTO*, 2001.
11. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2), May 2012.
12. M. Bellare, A. Boldyreva, and A. Palacio. An uninstantiable random-oracle-model scheme for a hybrid-encryption problem. In *EUROCRYPT*, pages 171–188, 2004.
13. M. Bellare, V. T. Hoang, and S. Keelveedhi. Instantiating random oracles via uces. In *Crypto*, 2013.
14. M. Bellare and A. O'Neill. Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition. In *CANS*, 2013.
15. M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the First Annual Conference on Computer and Communications Security*. ACM, November 1993.
16. J. Bethencourt, A. Sahai, and B. Waters. Ciphertext-policy attribute-based encryption. In *IEEE Symposium on Security and Privacy*, pages 321–334, 2007.
17. N. Bitansky and R. Canetti. On strong simulation and composable point obfuscation. In *CRYPTO*, 2010.
18. N. Bitansky, R. Canetti, Y. T. Kalai, and O. Paneth. On virtual grey box obfuscation for general circuits. In *CRYPTO*, pages 108–125, 2014.
19. N. Bitansky, R. Canetti, O. Paneth, and A. Rosen. Indistinguishability obfuscation vs. auxiliary-input extractable functions: One must fall. Cryptology ePrint Archive, Report 2013/641, 2013. http://eprint.iacr.org/.
20. A. Boldyreva, D. Cash, M. Fischlin, and B. Warinschi. Foundations of non-malleable hash and one-way functions. In *In ASIACRYPT*, 2009.
21. D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, pages 213–229, 2001.

22. D. Boneh, A. Raghunathan, and G. Segev. Function-private identity-based encryption: Hiding the function in functional encryption. In *CRYPTO*, 2013.
23. D. Boneh, A. Raghunathan, and G. Segev. Function-private subspace-membership encryption and its applications. In *Asiacrypt*, 2013.
24. D. Boneh, A. Sahai, and B. Waters. Functional encryption: Definitions and challenges. In *TCC*, 2011.
25. D. Boneh and A. Silverberg. Applications of multilinear forms to cryptography. *IACR Cryptology ePrint Archive*, 2002.
26. D. Boneh and B. Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, pages 535–554, 2007.
27. X. Boyen and B. Waters. Anonymous hierarchical identity-based encryption (without random oracles). In *CRYPTO*, pages 290–307, 2006.
28. E. Boyle, K.-M. Chung, and R. Pass. On extractability obfuscation. In *TCC*, 2014.
29. E. Boyle, S. Goldwasser, and I. Ivan. Functional signatures and pseudorandom functions. In *PKC*, 2014.
30. Z. Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In *CRYPTO*, 2012.
31. Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In *ITCS*, 2012.
32. Z. Brakerski and G. N. Rothblum. Obfuscating conjunctions. In *CRYPTO*, pages 416–434, 2013.
33. Z. Brakerski and G. N. Rothblum. Black-box obfuscation for d-cnfs. In *ITCS*, 2014.
34. Z. Brakerski and G. N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In *TCC*, 2014.
35. Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO*, pages 501–521, 2011.
36. Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *FOCS*, 2011.
37. Z. Brakerski and V. Vaikuntanathan. Lattice-based FHE as secure as PKE. In *ITCS*, 2014.
38. R. Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In *CRYPTO*, 1997.
39. R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science*, FOCS '01, 2001.
40. R. Canetti and R. R. Dakdouk. Obfuscating point functions with multibit output. In *EUROCRYPT*, 2008.
41. R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. In *STOC*, 1998.
42. R. Canetti, D. Micciancio, and O. Reingold. Perfectly one-way probabilistic hash functions (preliminary version). In *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, STOC '98, 1998.
43. R. Canetti, G. Rothblum, and M. Varia. Obfuscation of hyperplane membership. In *TCC*, 2010.
44. R. Canetti and V. Vaikuntanathan. Obfuscating branching programs using black-box pseudo-free groups. Cryptology ePrint Archive, Report 2013/500, 2013. http://eprint.iacr.org/.
45. A. D. Caro and V. Iovino. On the power of rewinding simulators in functional encryption. Cryptology ePrint Archive, Report 2013/752, 2013. http://eprint.iacr.org/.

46. D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, 2010.
47. C. Cocks. An identity based encryption scheme based on quadratic residues. In *IMA Int. Conf.*, pages 360–363, 2001.
48. A. W. Dent. Adapting the weaknesses of the random oracle model to the generic group model. In *Advances in CryptologyASIACRYPT 2002*, pages 100–109. Springer, 2002.
49. M. Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 24–43, 2010.
50. M. Fischlin. A note on security proofs in the generic model. In *ASIACRYPT*, 2000.
51. S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, 2013.
52. S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, 2013.
53. S. Garg, C. Gentry, S. Halevi, A. Sahai, and B. Waters. Attribute-based encryption for circuits from multilinear maps. In *CRYPTO*, 2013.
54. S. Garg, C. Gentry, S. Halevi, and D. Wichs. On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input, 2014.
55. C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.
56. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, 2008.
57. C. Gentry, A. Sahai, and B. Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO*, 2013.
58. O. Goldreich, S. Micali, and A. Wigderson. How to play ANY mental game. In *STOC*, 1987.
59. S. Goldwasser, S. D. Gordon, V. Goyal, A. Jain, J. Katz, F.-H. Liu, A. Sahai, E. Shi, and H.-S. Zhou. Multi-input functional encryption. In *Eurocrypt*, 2014.
60. S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich. How to run turing machines on encrypted data. In *CRYPTO (2)*, pages 536–553, 2013.
61. S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich. Reusable garbled circuits and succinct functional encryption. In *STOC*, pages 555–564, 2013.
62. S. Goldwasser and G. N. Rothblum. On best-possible obfuscation. In *TCC*, 2007.
63. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Functional encryption with bounded collusions from multiparty computation. In *CRYPTO*, 2012.
64. S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute based encryption for circuits. In *STOC*, 2013.
65. V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *ACM Conference on Computer and Communications Security*, pages 89–98, 2006.
66. S. Hada. Zero-knowledge and code obfuscation. In *ASIACRYPT*, 2000.
67. D. Hofheinz, J. Malone-lee, and M. Stam. Obfuscation for cryptographic purposes. In *In TCC*, pages 214–232, 2007.
68. S. Hohenberger, G. N. Rothblum, A. Shelat, and V. Vaikuntanathan. Securely obfuscating re-encryption. In *Proceedings of the 4th Conference on Theory of Cryptography*, TCC'07, 2007.
69. J. Katz, A. Sahai, and B. Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT*, pages 146–162, 2008.

70. A. B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT*, pages 62–91, 2010.
71. B. Lynn, M. Prabhakaran, and A. Sahai. Positive results and techniques for obfuscation. In *EUROCRYPT*, 2004.
72. C. Matt and U. Maurer. A constructive approach to functional encryption. Cryptology ePrint Archive, Report 2013/559, 2013. http://eprint.iacr.org/.
73. U. Maurer. Abstract models of computation in cryptography. In *IMA Int. Conf.*, 2005.
74. S. Micali, R. Pass, and A. Rosen. Input-indistinguishable computation. In *FOCS*, 2006.
75. J. B. Nielsen. Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In *CRYPTO*, pages 111–126, 2002.
76. A. O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. http://eprint.iacr.org/.
77. O. Pandey and Y. Rouselakis. Property preserving symmetric encryption. In *EUROCRYPT*, 2012.
78. R. Pass, K. Seth, and S. Telang. Indistinguishability obfuscation from semantically-secure multilinear encodings. In *CRYPTO*, pages 500–517, 2014.
79. A. Sahai and H. Seyalioglu. Worry-free encryption: Functional encryption with public keys. In *CCS*, 2010.
80. A. Sahai and B. Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473, 2005.
81. A. Sahai and B. Waters. How to use indistinguishability obfuscation: Deniable encryption, and more. In *Crypto*, 2013.
82. A. Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53, 1984.
83. E. Shen, E. Shi, and B. Waters. Predicate privacy in encryption systems. In *TCC*, pages 457–473, 2009.
84. V. Shoup. Lower bounds for discrete logarithms and related problems. In *Eurocrypt*, pages 256–266, 1997.
85. B. Waters. Functional encryption for regular languages. In *Crypto*, 2012.
86. H. Wee. On obfuscating point functions. In *STOC*, pages 523–532, 2005.