CRYPTOGRAPHIC AGENTS

BY

SHASHANK AGRAWAL

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2015

Urbana, Illinois

Abstract

Over the last decade or so, thanks to remarkable breakthroughs in cryptographic techniques, a wave of "cryptographic objects"—identity-based encryption, fully-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 various security definitions, have become increasingly confusing.

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 *reduction* and *composition* theorems. We model the functionality desired from a cryptographic object via a *schema* in an ideal world. Our new security definition, indistinguishability preservation, is parametrized by a family of *test* functions. We say that a scheme securely implements a schema against a test family in the real world if for every test in the family, if test is able to hide some bit of information from all adversaries in the ideal world, then this bit should be hidden in the real world too. By choosing test families appropriately, we are able to place known security definitions (along with new ones) for a given object on the same canvas, enabling comparative analysis.

Next, we explore the implications of a meaningful relaxation of our security definition, the one obtained by considering all-powerful adversaries in the ideal world. Thanks to our framework, we are not only able to substantially generalize known results connecting two important flavors of security definitions (simulation and indistinguishability) in cryptography under this relaxation, but significantly simplify them too.

We also initiate a systematic study of the security of fundamental cryptographic primitives like public-key encryption under a new class of attacks that had not been considered so far in the literature. Once again, owing to the flexibility of our framework, we are able to model such attacks, along with existing ones, in a clean and satisfactory way.

Acknowledgments

First of all I would like to thank my advisor Manoj Prabhakaran. He is a brilliant, hard-working, innovative researcher, and at the same time, a really nice, humble, down-to-earth person. He has given me immense support throughout the PhD program. Inspite of his busy schedule, we have always had regular meetings. I have learnt from him how to be persistent, precise, and thorough.

I am extremely fortunate to have collaborated with some of the most talented and amazing people in the field of cryptography. In alphabetical order they are Divesh Aggarwal, Shweta Agrawal, Prabhanjan Ananth, Erman Ayday, Saikrishna Badrinarayanan, Melissa Chase, Vipul Goyal, Divya Gupta, Carl Gunter, Jean-Pierre Hubaux, Venkata Koppula, Hemanta K. Maji, Abishek Kumarasubramanian, Muhammad Naveed, Omkant Pandey, Alon Rosen, Amit Sahai, Xiaofeng Wang, David Wu, and Ching-Hua Yu. I would like to thank them all for the time they spent with me and the things that I learnt from them.

A special thank you to Vipul Goyal for my very first research internship experience, just after my first year of PhD. The two months I spent at Microsoft Research India were wonderful in several ways: the office space was fabulous, the food was awesome, and above all, I met a lot of people and made some good friends. A special thanks is also in order for Shweta Agrawal, with whom I started working at the beginning of my third year. Since then, we have collaborated on a number of projects, and have come to know each other very well. She is a very nice person, has a lot of ideas, and always willing to discuss them. I also want to thank her for inviting me to visit IIT Delhi.

Apart from my advisor, the person with whom I have worked most closely is Melissa Chase. We started working together in the summer of 2014 during my internship at Microsoft Research Redmond. Melissa is perhaps the nicest person I have ever met. Needless to say, she is really smart, focused, and hard-working. In the last one and a half years, our collaboration has been quite fruitful, and I hope it stays the same way. I want to thank her for inviting me to visit Microsoft Research, and offering me another internship, which I will be starting soon.

I also want to thank my PhD committee members: Carl Gunter, Nikita Borisov, Nitin Vaidya, and Vinod Vaikuntanathan, whose feedback has been invaluable.

It has also been a pleasure interacting with members of the theory group at UIUC. Hsien-Chih Chang, Shalmoli Gupta, Nirman Kumar, Vivek Madan, Kent Quanrud, Ben Raichel, Ali Vakilian, and others, have been very helpful and friendly. I had a wonderful time TA'ing CS 473 (Fundamental Algorithms) taught by Sariel Har-Peled. He is such an easy-going, fun person to work with. The same goes for my co-TA, Kent Quanrud—a delightful person, and so helpful.

I am very fortunate to have friends like Mayank Baranwal, Ankita Bhutani, Debashish Das, Piyush Deshpande, Pranav Garg, Dileep Kini, Vivek Kumar, Shikhar Mohan, Shishir Pandya, Advitya Patyal, Manila Sarangi, Piyush Singh, Prakalp Srivastava, and Ankur Taneja. They have made my time in Urbana-Champaign a truly memorable one. We have done so many things together from hanging out and making fun of each other, to playing tennis, squash, and table-tennis, to swimming, biking, and hiking.

Last but not the least I want to thank my family. My father has always believed in me and encouraged me to pursue my dreams. My mother has been very loving and supportive. And my wife Shikha—well, let me just say that it wouldn't have been possible without her.

Table of Contents

Chapter 1

Introduction

Over the last decade or so, thanks to remarkable breakthroughs in cryptographic techniques, a wave of "cryptographic objects"—identity-based encryption, fully-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 various security definitions, have become increasingly confusing.

A case in point is functional encryption (FE) [1]. FE comes in numerous flavors: public key or symmetric [2, 3], with or without function hiding [4, 5], public or private index [1], bounded or unbounded key [6, 7, 8]. Each flavor has several candidate security definitions: indistinguishability based [9, 2], adaptive simulation based [1], non-adaptive simulation [10], unbounded simulation [11], fully-adaptive security [12], black-box/non black-box simulation [13] to name a few. In addition, FE can be constructed from obfuscation [14] and can be used to construct property preserving encryption [15], each of which have numerous security definitions of their own [16, 17, 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, and 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

[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

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 (for example, the "virtual black-box" definition of obfuscation is unrealizable [17]). 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.

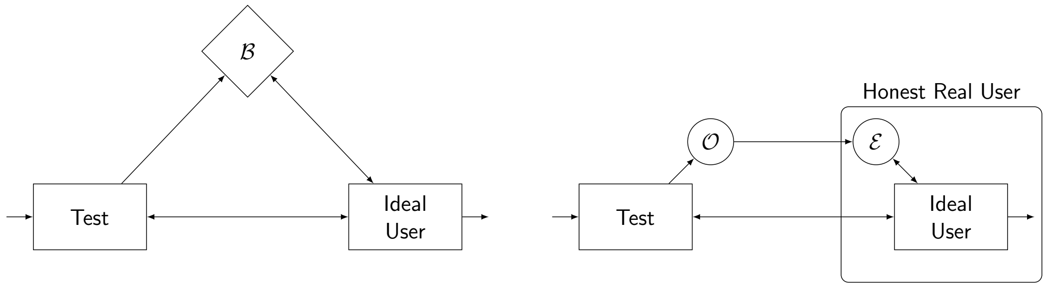

Figure 1.1: The ideal world (on the left) and the real world with an honest user.

[2]One exception to this is the "input-indistinguishable computation" framework of Micali, Pass and Rosen for secure function evaluation of deterministic functions [19]. 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.

- **Ideal Execution.** The ideal world consists of two (adversarially designed) entities — a User and a Test — who can freely interact with each other. (See the left-hand side of Figure 1.1.) 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

[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.

execute a session of agents, and thereby update their states) using an "execution" operation applied to the cryptographic agents. In Figure 1.1, $\mathcal{O}$ indicates the algorithm for encoding, and $\mathcal{E}$ indicates 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.

Chapter 2

Agents Framework

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.*[1] *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 $\mathbf{\Sigma} = (\mathcal{P}_{\mathsf{auth}}, \mathcal{P}_{\mathsf{user}})$ *(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.*

[1]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.

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 $|\Pr[X = 1] - \Pr[Y = 1]| \leq \text{negl}(\kappa)$. For two systems M and M', we say $M \approxeq M'$ if the two systems are indistinguishable to an interactive PPT distinguisher.

For two functions f and g, we write $f(g)$ to denote the function $f \circ g$, so that $f(g)(x) = f(g(x))$.

2.1 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 $\mathbf{\Sigma}$ consists of two parties Test and User and a fixed third party $\mathcal{B}[\mathbf{\Sigma}]$ (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 $\mathcal{B}[\mathbf{\Sigma}]$ can be summarized as follows:

- **Uploading agents.** Let $\mathbf{\Sigma} = (\mathcal{P}_{\mathsf{auth}}, \mathcal{P}_{\mathsf{user}})$ 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 $\mathcal{B}[\mathbf{\Sigma}]$. More precisely, User can send a string to $\mathcal{B}[\mathbf{\Sigma}]$, and $\mathcal{B}[\mathbf{\Sigma}]$ 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 [2]. Whenever an agent is instantiated, $\mathcal{B}[\mathbf{\Sigma}]$ sends a

[2] In fact, for convenience, we allow Test and User to specify multiple agents in a single

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 $\mathcal{B}[\mathbf{\Sigma}]$) to specify the configurations of the agents in the session, along with their inputs. $\mathcal{B}[\mathbf{\Sigma}]$ reports back the outputs from the session, and also gives new handles corresponding to the configurations of the agents when the session terminated.[3] If an agent halts in a session, no new handle is given for that agent.

Observe that only User receives any output from $\mathcal{B}[\mathbf{\Sigma}]$; the communication between Test and $\mathcal{B}[\mathbf{\Sigma}]$ is one-way. (See Figure 1.1.)

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

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

$$\text{IDEAL}\langle \mathsf{Test}(0) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle \approx \text{IDEAL}\langle \mathsf{Test}(1) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle.$$

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 1.1.) Test remains the same as

message to $\mathcal{B}[\mathbf{\Sigma}]$.

[3] 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.

in the ideal world, except that instead of sending an agent to $\mathcal{B}[\mathbf{\Sigma}]$, 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 $(\mathsf{MSK}, \mathsf{MPK})$;[4] 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 $P \in \mathcal{P}_{\mathsf{user}}$, 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.

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

Admissibility. We would only be interested in schemes that are admissible according to the following definition.

Definition 6 (Admissibility of schemes)**.** *A cryptographic agent scheme* $\Pi = (\mathcal{O}, \mathcal{E})$ *is said to be an* admissible scheme *for a schema* $\mathbf{\Sigma}$ *if the following conditions hold.*

- *Correctness.* $\forall$ PPT User *and* $\forall$ Test,

$$|\Pr[\text{IDEAL}\langle \mathsf{Test} \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle = 1] - \Pr[\text{REAL}\langle \mathsf{Test} \mid \mathcal{O} \mid \mathcal{E} \circ \mathsf{User}\rangle = 1]| \leq \text{negl}$$

  for some negligible function negl. *If the difference is 0,* $(\mathcal{O}, \mathcal{E})$ *is said to have perfect correctness.*

- *Efficiency. There exists a polynomial* poly *such that,* $\forall$ PPT User, $\forall$ Test,

$$\text{TIME}\langle \mathsf{Test} \mid \mathcal{O} \mid \mathcal{E} \circ \mathsf{User}\rangle \leq \text{poly}(\text{TIME}\langle \mathsf{Test} \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle, \kappa).$$

$\mathsf{IND\text{-}PRE}$ **Security.** Now we are ready to present the security definition of a cryptographic agent scheme $(\mathcal{O}, \mathcal{E})$ implementing a schema $\mathbf{\Sigma}$. 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 1.1.

Definition 7. *An admissible cryptographic agent scheme* $\Pi = (\mathcal{O}, \mathcal{E})$ *is said to be a* Γ-$\mathsf{IND\text{-}PRE}$-secure scheme *for a schema* $\mathbf{\Sigma}$ *if*

- *Indistinguishability Preservation.* $\forall \mathsf{Test} \in \Gamma$,

$$\mathsf{Test} \textit{ is hiding w.r.t. } \mathbf{\Sigma} \Rightarrow \mathsf{Test} \textit{ is hiding w.r.t. } \mathcal{O}.$$

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* $\mathbf{\Sigma}$.

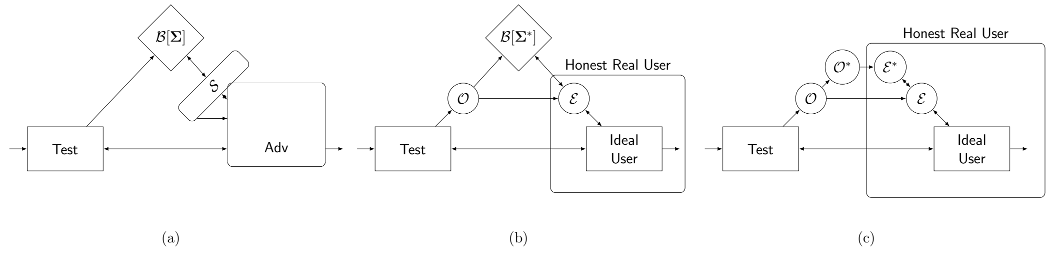

Figure 2.1: $(\mathcal{O}, \mathcal{E})$ in (b) is a reduction from schema $\mathbf{\Sigma}$ to $\mathbf{\Sigma}^*$. The security requirement is that no adversary Adv in the system (a) can distinguish that execution from an execution of the system in (b) (with Adv taking the place of honest real user). The correctness requirement is that the ideal User in (b) behaves the same as the ideal User interacting directly with $\mathcal{B}[\mathbf{\Sigma}]$ (as in Figure 1.1(a)). (c) shows the composition of the hybrid scheme $(\mathcal{O}, \mathcal{E})^{\mathbf{\Sigma}^*}$ with a scheme $(\mathcal{O}^*, \mathcal{E}^*)$ that $\mathsf{IND\text{-}PRE}$-securely implements $\mathbf{\Sigma}^*$.

2.2 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* $(\mathcal{O}, \mathcal{E})^{\mathbf{\Sigma}^*}$ is a cryptographic agent scheme in which $\mathcal{O}$ and $\mathcal{E}$ have access to $\mathcal{B}[\mathbf{\Sigma}^*]$, as shown in Figure 2.1 (in the middle), where $\mathbf{\Sigma}^* = (\mathcal{P}^*_{\mathsf{auth}}, \mathcal{P}^*_{\mathsf{user}})$. 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 $\text{IDEAL}\langle \mathsf{Test} \circ \mathcal{O} \mid \mathbf{\Sigma}^* \mid \mathsf{Adv}\rangle$, where $\mathsf{Test} \circ \mathcal{O}$ denotes the combination of Test and $\mathcal{O}$ as in Figure 2.1.

Definition 8 (Reduction)**.** *We say that a (hybrid) cryptographic agent scheme*

$\Pi = (\mathcal{O}, \mathcal{E})$ reduces $\mathbf{\Sigma}$ *to* $\mathbf{\Sigma}^*$ *with respect to* Γ*, if there exists a* PPT *simulator* $\mathcal{S}$ *such that* $\forall$ PPT User*,*

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

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

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

Figure 2.1 illustrates a reduction. It also shows how such a reduction can be composed with an IND-PRE-secure scheme for $\mathbf{\Sigma}^*$. 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 2.1(c).[5]

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

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

$$\begin{aligned}
\text{REAL}\langle \mathsf{Test} \mid \mathcal{O}' \mid \mathcal{E}' \circ \mathsf{User} \rangle &= \text{REAL}\langle \mathsf{Test}' \mid \mathcal{O}^* \mid \mathcal{E}^* \circ \mathsf{User}' \rangle \\
&\overset{(a)}{\approx} \text{IDEAL}\langle \mathsf{Test}' \mid \mathbf{\Sigma}^* \mid \mathsf{User}' \rangle \\
&= \text{IDEAL}\langle \mathsf{Test} \circ \mathcal{O} \mid \mathbf{\Sigma}^* \mid \mathcal{E} \circ \mathsf{User} \rangle \\
&\overset{(b)}{\approx} \text{IDEAL}\langle \mathsf{Test} \mid \mathbf{\Sigma} \mid \mathsf{User} \rangle
\end{aligned}$$

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 $\mathbf{\Sigma}$ to $\mathbf{\Sigma}^*$. (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. $\mathbf{\Sigma}$ then Test is hiding w.r.t. $\mathcal{O}'$.

[5] 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 $\mathcal{O}^*_{\mathsf{user}}$.

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

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

where the second implication is due to the fact that $(\mathcal{O}^*, \mathcal{E}^*)$ is an $\mathsf{IND\text{-}PRE}$ secure implementation of $\mathbf{\Sigma}^*$, 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, $\mathbf{\Sigma}$ and $\mathbf{\Sigma}^*$, if $(\mathcal{O}, \mathcal{E})$ reduces $\mathbf{\Sigma}$ to $\mathbf{\Sigma}^*$ and $(\mathcal{O}^*, \mathcal{E}^*)$ is a $(\Gamma \circ \mathcal{O})$-$\mathsf{IND\text{-}PRE}$ secure scheme for $\mathbf{\Sigma}^*$, then $(\mathcal{O} \circ \mathcal{O}^*, \mathcal{E}^* \circ \mathcal{E})$ is a Γ-$\mathsf{IND\text{-}PRE}$ secure scheme for $\mathbf{\Sigma}$.*

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

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 $\mathbf{\Sigma}_1$ to $\mathbf{\Sigma}_2$ and that of $\mathbf{\Sigma}_2$ to $\mathbf{\Sigma}_3$, 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 $\mathbf{\Sigma}_1$ to $\mathbf{\Sigma}_3$. 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 $\mathcal{S}_2 \circ \mathcal{S}_1$. □

2.3 Restricted Test Families

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 Section 3.2.2 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 $\mathcal{B}[\mathbf{\Sigma}]$, it picks two agents (P_0, P_1). Both the agents are sent to User, and P_b is sent to $\mathcal{B}[\mathbf{\Sigma}]$ (where b is the secret bit input to Test). Except for selecting the agent to be sent to $\mathcal{B}[\mathbf{\Sigma}]$, 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 $\mathcal{B}[\mathbf{\Sigma}]$ (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,

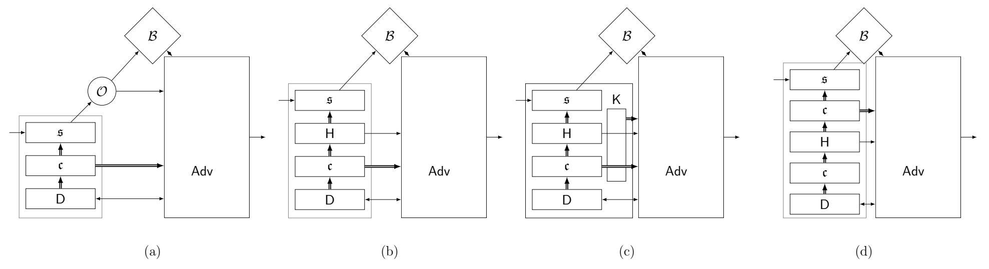

Figure 2.2: Illustration of Δ and the extra requirements on Δ-reduction. (a) illustrates the structure of a test in Δ; the double-arrows indicate messages consisting of a pair of agents. The first condition on H is that (a) and (b) are indistinguishable to Adv: i.e., H can mimic the message from $\mathcal{O}$ 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.

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 program $\mathcal{O}$, have three communication ports (in addition to the secret bit that $\mathfrak{s}$ receives): in terms of Figure 2.2, 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 9 (Δ-Reduction)**.** *We say that a (hybrid) obfuscated agent scheme* $\Pi = (\mathcal{O}, \mathcal{E})$ Δ*-reduces* $\mathbf{\Sigma}$ *to* $\mathbf{\Sigma}^*$ *if*

1. Π *reduces* $\mathbf{\Sigma}$ *to* $\mathbf{\Sigma}^*$ *with respect to* Δ *(as in Definition 8), 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.* $\mathbf{\Sigma}$*,* $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}(b) \circ \mathcal{O} \approxeq \mathsf{D} \circ \mathfrak{c} \circ \mathsf{H} \circ \mathfrak{s}(b)$*, 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* $\mathbf{\Sigma}$ *to* $\mathbf{\Sigma}^*$*, then we say* $\mathbf{\Sigma}$ Δ*-reduces to* $\mathbf{\Sigma}^*$*.*

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-$\mathsf{IND\text{-}PRE}$ security. We prove the following theorems now.

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

Proof sketch: Correctness and efficiency are easily confirmed. To prove security, we need to show that for every $\mathsf{Test} \in \Delta$, if Test is hiding w.r.t. $\mathbf{\Sigma}$, then it is hiding w.r.t. $\mathcal{O} \circ \mathcal{O}^*$. Since $\mathsf{Test} \in \Delta$, we can write it as $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}$. Let $\mathsf{Test}' \in \Delta$ be defined as $\mathsf{D} \circ \mathfrak{c} \circ \mathsf{H} \circ \mathfrak{c} \circ \mathfrak{s}$, where H is related to $\mathcal{O}$ as in Definition 9.

First we shall argue that Test' is hiding w.r.t. $\mathbf{\Sigma}^*$. Below, we shall also use K that relates to H as in Definition 9. For any PPT User, for each $b \in \{0, 1\}$, we have

$$\begin{aligned}
\mathsf{Test}'(b) &\equiv \mathsf{D} \circ \mathfrak{c} \circ \mathsf{H} \circ \mathfrak{c} \circ \mathfrak{s}(b) \\
&\approxeq \mathsf{D} \circ \mathfrak{c} \circ \mathsf{H} \circ \mathfrak{s}(b) \ / \ \mathsf{K} \qquad (2.1) \\
&\approxeq \mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}(b) \circ \mathcal{O} \ / \ \mathsf{K} \equiv \mathsf{Test}(b) \circ \mathcal{O} \ / \ \mathsf{K}.
\end{aligned}$$

So for any PPT User,

$$\begin{aligned}\textsc{ideal}\langle \mathsf{Test}'(b) \mid \mathbf{\Sigma}^* \mid \mathsf{User}\rangle &\approx \textsc{ideal}\langle \mathsf{Test}(b) \circ \mathcal{O} \;/\; \mathsf{K} \mid \mathbf{\Sigma}^* \mid \mathsf{User}\rangle \\ &= \textsc{ideal}\langle \mathsf{Test}(b) \circ \mathcal{O} \mid \mathbf{\Sigma}^* \mid \mathsf{User}'\rangle \\ &= \textsc{ideal}\langle \mathsf{Test}(b) \mid \mathbf{\Sigma} \mid \mathcal{S} \circ \mathsf{User}'\rangle,\end{aligned}$$

where User' incorporates K and User, and $\mathcal{S}$ is from Definition 8. Hence if Test is hiding w.r.t. $\mathbf{\Sigma}$, $\textsc{ideal}\langle \mathsf{Test}(0) \mid \mathbf{\Sigma} \mid \mathsf{User}''\rangle \approx \textsc{ideal}\langle \mathsf{Test}(1) \mid \mathbf{\Sigma} \mid \mathsf{User}''\rangle$, where User'' stands for $\mathcal{S} \circ \mathsf{User}'$, and hence $\textsc{ideal}\langle \mathsf{Test}'(0) \mid \mathbf{\Sigma}^* \mid \mathsf{User}\rangle \approx \textsc{ideal}\langle \mathsf{Test}'(1) \mid \mathbf{\Sigma}^* \mid \mathsf{User}\rangle$. Since this holds for all PPT User, Test' is hiding w.r.t. $\mathbf{\Sigma}^*$. Thus we have,

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

where the second implication follows from the fact that $(\mathcal{O}^*,\mathcal{E}^*)$ $\mathsf{IND\text{-}PRE}$ securely implements $\mathbf{\Sigma}^*$, and the third from Equation 2.1. Now, since K only provides extra information to User, if $\mathsf{Test} \circ \mathcal{O} \;/\; \mathsf{K}$ is hiding w.r.t. $\mathcal{O}^*$, then $\mathsf{Test} \circ \mathcal{O}$ is hiding w.r.t. $\mathcal{O}^*$. This is the same as saying that $\mathsf{Test} \circ \mathcal{O} \circ \mathcal{O}^*$ is hiding (w.r.t. a null scheme), as was required to be shown. $\square$

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

Proof sketch: Let $\Pi_1 = (\mathcal{O}_1, \mathcal{E}_1)$ and $\Pi_2 = (\mathcal{O}_2, \mathcal{E}_2)$ be the schemes that carry out the Δ-reduction of $\mathbf{\Sigma}_1$ to $\mathbf{\Sigma}_2$ and that of $\mathbf{\Sigma}_2$ to $\mathbf{\Sigma}_3$, respectively. We define the scheme $\Pi = (\mathcal{O}_1 \circ \mathcal{O}_2, \mathcal{E}_2 \circ \mathcal{E}_1)$. As in Theorem 3, we see that Π reduces $\mathbf{\Sigma}_1$ to $\mathbf{\Sigma}_3$ with respect to Δ. What remains to be shown is that Π also has associated machines (H, K) as required in Definition 9.

Let $(\mathsf{H}_1, \mathsf{K}_1)$ and $(\mathsf{H}_2, \mathsf{K}_2)$ be associated with Π_1 and Π_2 respectively, as in Definition 9. We let $\mathsf{H} \equiv \mathsf{H}_1 \circ \mathsf{H}_2$. To define K, consider the cascade $\mathsf{K}_1 \;/\; \mathsf{K}_2$: i.e., K_1 appends a message to the first part of the input to K (from $\mathfrak{c} \circ \mathsf{H}_1$) and passes it on to K_2, which also gets the second part of the input (from H_2), and appends another message of its own. K behaves as $\mathsf{K}_1 \;/\; \mathsf{K}_2$ but from the output, it removes the message added by K_1. We write this as

$\mathsf{K} \equiv \mathsf{K}_1 \ / \ \mathsf{K}_2 \ /\!\!/\mathsf{trim}$, where $/\!\!/\mathsf{trim}$ stands for the operation of redacting the appropriate part of the message. Note that K has the required format, in that it only appends to the entire message it receives.

We confirm that (H, K) satisfy the two required properties:

$$\mathfrak{s}(b) \circ \mathcal{O} \equiv \mathfrak{s}(b) \circ \mathcal{O}_1 \circ \mathcal{O}_2 \approxeq \mathsf{H}_1 \circ \mathfrak{s}(b) \circ \mathcal{O}_2 \approxeq \mathsf{H}_1 \circ \mathsf{H}_2 \circ \mathfrak{s}(b) \equiv \mathsf{H} \circ \mathfrak{s}(b)$$

$$\mathfrak{c} \circ \mathsf{H} \ / \ \mathsf{K} \equiv (\mathfrak{c} \circ \mathsf{H}_1 \ / \ \mathsf{K}_1) \circ \mathsf{H}_2 \ / \ \mathsf{K}_2 \ /\!\!/\mathsf{trim} \ \approxeq \mathfrak{c} \circ \mathsf{H}_1 \circ \mathfrak{c} \circ \mathsf{H}_2 \ / \ \mathsf{K}_2 \ /\!\!/\mathsf{trim}$$
$$\approxeq \mathfrak{c} \circ \mathsf{H}_1 \circ \mathfrak{c} \circ \mathsf{H}_2 \circ \mathfrak{c} \ /\!\!/\mathsf{trim} \ \equiv \mathfrak{c} \circ \mathsf{H}_1 \circ \mathsf{H}_2 \circ \mathfrak{c}$$

where the last identity follows from the fact that the operation $/\!\!/\mathsf{trim}$ removes the appropriate part of the outgoing message. □

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 $\mathfrak{c}$. The composition theorem for Δ, Theorem 4, extends to Δ^* as well.

We note the composition (and transitivity) extend to Δ^* as well. In particular, the following theorem can be proven by observing that in the proof of Theorem 4, if we consider $\mathsf{Test} \in \Delta^*$, then Test' defined in the proof belongs to Δ^*. (In contrast, this result does *not* extend to Δ_{det}, unless the notion of reduction is severely restricted, by requiring H and K to be deterministic.)

Theorem 6 (Δ^*-Composition)**.** *For any two schemata, $\mathbf{\Sigma}$ and $\mathbf{\Sigma}^*$, if $(\mathcal{O}, \mathcal{E})$ Δ-reduces $\mathbf{\Sigma}$ to $\mathbf{\Sigma}^*$ and $(\mathcal{O}^*, \mathcal{E}^*)$ is a* Δ^*-IND-PRE *secure implementation of $\mathbf{\Sigma}^*$, then $(\mathcal{O} \circ \mathcal{O}^*, \mathcal{E}^* \circ \mathcal{E})$ is a* Δ^*-IND-PRE *secure implementation of $\mathbf{\Sigma}$.*

Note that here the notion of reduction is still the same as in Theorem 4, namely Δ-reduction.

Chapter 3

Applications

Recent times have seen a fantastic boom in the area of *computing with encrypted information.* Several exciting primitives supporting advanced functionalities, such as fully homomorphic encryption [20], functional encryption [1], property preserving encryption [15] have been constructed. Some functionalities require the data to be hidden but permit the function to be public, while others, most notably program obfuscation [17], permit the data to be public but the function to be hidden. Let us review the state of the art in these fields.

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. [17] who showed that the strongest possible notion of security—a simulation-based notion—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 [17] 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 [21, 22, 23] for a long time, with few positive results, often for very specialized classes of functions [24, 25, 26, 27, 28, 29]. This state of affairs however, has improved significantly in recent times, when constructions of graded encoding schemes [30] were leveraged to build program obfuscators for complex functionalities, such as conjunctions [31], d-CNF formulas [32], circuits [14, 33, 34] and even Turing machines [35] in weaker models of computation such as the generic graded encoding scheme model [31, 32, 33, 35], the generic colored matrix model [14] and the idealized pseudo free group model [34].

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 $\mathsf{IP\text{-}Obf}$ [36] and $\mathsf{DI\text{-}Obf}$ [35, 37]. There is a growing research effort in exploring the plausibility and connections between different notions of obfuscation [38, 39] . A better understanding of various notions of obfuscation and connections with various related notions such as functional encryption is slowly emerging [40, 41, 42].

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 [43, 9, 44, 45, 46, 47, 48, 49], membership checking [50], boolean formulas [51, 52, 53], inner product functions [54, 53, 55], and more recently, even regular languages [56]. Recent times saw constructions for more general classes of functions: Gorbanov et al. [57] and Garg et al. [58] provided the first constructions for an important subclass of FE called "public index FE" for all circuits, Goldwasser et al. [8] constructed succinct simulation-secure single-key FE scheme for all circuits, Garg et al. [14] constructed multi-key FE schemes for all circuits while Goldwasser et al. and Ananth et al. [59, 35] also constructed FE for Turing machines.

Functional encryption and obfuscation are not just powerful cryptographic primitives in their own right, but are also intimately related objects – for example, it was shown in [14] that indistinguishability obfuscation implies functional encryption. Recently, differing input obfuscation has been used to construct FE for Turing machines [35].

In [1], Boneh et al. initiated the study of FE under simulation based definitions, wherein (informally) the view of any adversary attacking an FE scheme can also be produced with access to only the information we would like it to learn in an ideal sense. They not only show that game based security definitions—under which the above schemes are proven secure—are inadequate for certain functionalities, but unfortunately, they also prove that their stronger simulation based definition cannot be realized even for the most basic type of functional encryption. This led to a series of works that studied variants of simulation-based definitions, proved more impossibility results, gave transformations, etc. [10, 13, 60, 11]. The general consensus that emerged was that simulation-based security is too strong a requirement for functional encryption.

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 [20], extensive research effort has been focused on providing improvements [61, 62, 63, 64, 65, 66, 67].

Recently, Alwen et al. [68] explored the connections between FHE, FE and obfuscation. In [68], the authors introduce the notion of randomized FE which can be used to construct FHE. In addition, they explore the problem of obfuscating specific re-encryption functionalities, introducing new notions extending those proposed in earlier works on re-encryption [28]. They also develop techniques to use obfuscated re-encryption circuits to construct FE schemes.

Property Preserving Encryption. The notion of property preserving encryption (PPE) was introduced in a recent work by Pandey and Rouselakis [15]. Property preserving encryption is a symmetric key primitive, which permits some pre-determined property $P(x_1, x_2)$ to be publicly tested given only the ciphertexts $\mathsf{CT}(x_1), \mathsf{CT}(x_2)$. In [15], the authors formalize the notion of PPE, provide definitions of security and provide a candidate construction for *inner product* PPE in the generic group model. Subsequently, [69] demonstrated an attack against the construction in [15], which was fixed in [5]. Agrawal et al. [5] also provide the first standard model construction of PPE.

This rich body of primitives is interdependent not only in terms of philosophy and techniques, but also in terms of *non-interaction.* Unlike the case of multi-party computation, where a user (in general) continues to send and receive messages throughout the protocol, the above primitives do not permit users to “keep playing”. A user may create an *obfuscated agent* once and for all, and then release it into the wild. This agent is expected to reveal nothing other than what is permitted by its functionality, but must interface in a well defined manner with other agents or expected inputs.

Another aspect to note, is that many of the above primitives are known to be impossible to instantiate under the strong *simulation based security* desired by MPC. Indeed, positive results often settle for a weaker *indistinguishability based security*, which is also the focus of this work.

We now study various cryptographic primitives, including the above, in our framework. But first we investigate how a popular abstraction, the generic group model, used in many cryptographic constructions can be captured in our model.

3.1 Generic Group

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 [70] or the (bilinear) generic group model [71, 72], 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 $\mathbf{\Sigma}$ is secure in the generic group model typically amounts to a *reduction from* $\mathbf{\Sigma}$ *to a “generic group schema”* $\mathbf{\Sigma}_{\text{GG}}$.
- The assumption that there is an IND-PRE-secure scheme Π_{GG} for $\mathbf{\Sigma}_{\text{GG}}$ is a standard-model assumption (that does not appear to be ruled out by known results or techniques).

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

Above, the generic group schema $\mathbf{\Sigma}_{\text{GG}}$ 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-$\mathsf{IND}\text{-}\mathsf{PRE}$*-secure scheme for the generic group schema* $\mathbf{\Sigma}_{\text{GG}}$.

Similarly, we put forward the Γ-*Bilinear* Generic Group Agent Assumption, where $\mathbf{\Sigma}_{\text{GG}}$ is replaced by $\mathbf{\Sigma}_{\text{BGG}}$ 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, $\mathsf{IND}\text{-}\mathsf{PRE}$ security is weaker than simulation based security, and its achievability is not easily ruled out (see discussion in Section 3.6). 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, $\mathbf{\Sigma}_{\text{RO}}$, in which the agents encode elements in a large set and interact with each other to carry out equality checks. Then, a Δ_{det}-$\mathsf{IND}\text{-}\mathsf{PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{RO}}$ 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 [24, 25, 26, 29]. This fits into a broader theme of research that attempts to capture several features of the random oracle using standard model assumptions (e.g., [73, 74]). 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 $\mathbf{\Sigma}_{\mathrm{GG}}$ or $\mathbf{\Sigma}_{\mathrm{BGG}}$ by a multi-linear generic group schema $\mathbf{\Sigma}_{\mathrm{MGG}}$, which permits black box computation of multilinear map operations [75, 30]. Interestingly, this assumption is provably false if we consider Γ to be Γ_{ppt}, since there exists a reduction of obfuscation schema $\mathbf{\Sigma}_{\mathrm{OBF}}$ to $\mathbf{\Sigma}_{\mathrm{MGG}}$ [33, 76], and we have seen that there is no $\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\mathrm{OBF}}$. On the other hand, for Γ being Δ or Δ^*, 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 [77]. 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 $\mathsf{IND\text{-}PRE}$ security, we used a modified notion of security $\mathsf{IND\text{-}PRE}'$, which requires that every test which is *efficiently provably* ideal-hiding is real-hiding. We note that $\mathsf{IND\text{-}PRE}'$ 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.

3.2 Obfuscation

In this section we define and study the obfuscation schema $\mathbf{\Sigma}_{\mathrm{OBF}}$. 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.

We begin by showing that the obfuscation schema is "complete" under reduction as defined in Definition 8. Thus, there is an IND-PRE secure implementation $(\mathcal{O}, \mathcal{E})$ which reduces any general $\mathbf{\Sigma}_F$ to $\mathbf{\Sigma}_{\text{OBF}}$. This means that if there is an IND-PRE secure implementation of $\mathbf{\Sigma}_{\text{OBF}}$, say using secure hardware, then this implementation can be used in a modular way to build an IND-PRE secure schema for any general functionality.

Next, we show there cannot exist an IND-PRE secure schema for general functionalities. Thus, we exhibit a class of programs $\mathcal{F}$ such that Test is hiding in the ideal world but for any real world cryptographic scheme $(\mathcal{O}, \mathcal{E})$, Test is not hiding in the real world. Our impossibility follows the broad outline of the impossibility of virtual black box obfuscation by Barak et al. [17]. Since our definition of obfuscation schema is implied by virtual black box obfuscation, we obtain a potential[1] strengthening of the result of Barak et al. [17].

Finally, we relate the notion of IND-PRE obfuscation to standard notions of obfuscation such as indistinguishability obfuscation and differing inputs obfuscation. We show that the former is equivalent to IND-PRE restricted to the Δ_{det} family of tests, while the latter is is equivalent to IND-PRE restricted to the Δ^* family of tests. We define a new notion of obfuscation corresponding to IND-PRE restricted to the Δ family of tests, namely *adaptive differing inputs obfuscation.*

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

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

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 $\mathbf{\Sigma}_{\text{OBF}(\mathcal{F})}$.

A special case of $\mathbf{\Sigma}_{\text{OBF}(\mathcal{F})}$ 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 $C(x)$

[1]we do not have a separation between IND-PRE and VBB obfuscation so far

(padding or truncating x as necessary). We define the "general" obfuscation schema

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

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

3.2.1 Completeness of Obfuscation

We show that $\mathbf{\Sigma}_{\text{OBF}}$ is a complete schema with respect to schematic reduction (Definition 8). That is, *every schema* (including possibly randomized, interactive, and stateful agents) can be reduced to $\mathbf{\Sigma}_{\text{OBF}}$. 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 $\mathbf{\Sigma}_{\text{OBF}}$, as described in Section 3.2.2. However, if there is, say, a hardware-based IND-PRE secure implementation of $\mathbf{\Sigma}_{\text{OBF}}$, then this implementation can be used in a modular way to build an IND-PRE secure schema for any general functionality.

We show two kinds of reductions. They use only standard cryptographic primitives: CCA secure public-key encryption and digital signatures.

1. Any schema $(\emptyset, \mathcal{P})$ in which the agents are *non-interactive* (but possibly randomized and reactive) has a reduction $(\mathcal{O}, \mathcal{E})$ to $\mathbf{\Sigma}_{\text{OBF}}$ in which $\mathcal{O}$ is setup-free.

2. Any schema $\mathbf{\Sigma} = (\mathcal{P}_{\mathsf{auth}}, \mathcal{P}_{\mathsf{user}})$ (possibly with $\mathcal{P}_{\mathsf{test}} \neq \mathcal{P}_{\mathsf{user}}$ and containing possibly randomized, reactive, interactive agents) has a reduction $(\mathcal{O}, \mathcal{E})$ to $\mathbf{\Sigma}_{\text{OBF}}$ in which $\mathcal{O}$ has setup.

We point out that if $\mathcal{P}_{\mathsf{test}} \neq \mathcal{P}_{\mathsf{user}}$, in general it is necessary that $\mathcal{O}$ has setup, as otherwise an adversarial user can create obfuscations of programs in $\mathcal{P}_{\mathsf{auth}}$ itself.

We sketch each of these reductions below. The security of these reductions only depend on standard symmetric-key and public-key cryptography primitives. The proofs are conceptually clean and simple, as the reductions between schemata occur in an idealized world. However, the detailed descriptions of the reductions and the simulator tend to be somewhat long. We provide some of the details to clarify subtleties and also to illustrate the nature of the reductions and proofs.

We carry out the reduction in two steps: first we show how to reduce any schema with non-interactive agents to $\mathbf{\Sigma}_{\text{OBF}}$, and then build on it to reduce all schemata (including those with interactive agents) to $\mathbf{\Sigma}_{\text{OBF}}$.

Construction for Non-Interactive Agents

In this section we reduce any schema of the form $\mathbf{\Sigma}_{RR} = (\emptyset, \mathcal{P})$, in which the agents are randomized and reactive, but non-interactive, to $\mathbf{\Sigma}_{\text{OBF}}$. For this we define an intermediate schema, $\mathbf{\Sigma}_R$ of randomized, but non-reactive, non-interactive agents, and give two reductions: we reduce $\mathbf{\Sigma}_{RR}$ to $\mathbf{\Sigma}_R$ and $\mathbf{\Sigma}_R$ to $\mathbf{\Sigma}_{\text{OBF}}$. These can then be composed together using the transitivity of reducibility (Theorem 3) to obtain our reduction from $\mathbf{\Sigma}_{RR}$ to $\mathbf{\Sigma}_{\text{OBF}}$.[2]

Below, we will write $\mathbf{\Sigma}_0$ for $\mathbf{\Sigma}_{\text{OBF}}$, $\mathbf{\Sigma}_1$ for $\mathbf{\Sigma}_R$, and $\mathbf{\Sigma}_2$ for $\mathbf{\Sigma}_{RR}$.

$(\mathcal{O}_1, \mathcal{E}_1)$ to reduce $\mathbf{\Sigma}_1$ to $\mathbf{\Sigma}_0$. On receiving a randomized agent P_1 from Test, $\mathcal{O}_1$ uploads the following (deterministic) agent P_0 to $\mathcal{B}[\mathbf{\Sigma}_0]$: P_0 has the parameters of P_1 as well as a freshly chosen seed s for a pseudorandom function (PRF) built-in as its parameters; when invoked it interprets its input as (i, x), generates a random tape for P_1 using the PRF applied to (i, x), as $r = \mathsf{PRF}_s(i, x)$, and executes $P_1(x; r)$. (The κ-bit index i is used to implement multiple independent executions of the randomized agent with the same input.)

$\mathcal{E}_1$ translates User's interaction with $\mathcal{B}[\mathbf{\Sigma}_1]$ to an interaction with $\mathcal{B}[\mathbf{\Sigma}_0]$: when User requests to upload a randomized agent to $\mathcal{B}[\mathbf{\Sigma}_1]$, $\mathcal{E}_1$ will upload to $\mathcal{B}[\mathbf{\Sigma}_0]$ an agent as created by $\mathcal{O}_1$. When $\mathcal{B}[\mathbf{\Sigma}_0]$ sends $\mathcal{E}_1$ a handle, it forwards it to User. When User sends an execution command with a handle h and an input x to $\mathcal{B}[\mathbf{\Sigma}_1]$, $\mathcal{E}_1$ translates it to the handle h and input (i, x) for $\mathcal{B}[\mathbf{\Sigma}_0]$, where i is a randomly chosen κ-bit index. The correctness of the reduction follows directly from the security of the PRF, and the fact that it is unlikely that $\mathcal{E}_1$ will choose the same value for i in two different sessions.

[2]The tape and time bounds for the agents in $\mathbf{\Sigma}_{\text{OBF}}$ will depend on the tape and time bounds for the schema $\mathbf{\Sigma}_{RR}$. For simplicity, we leave this bound to be only implicitly specified by our reductions.

The simulator $\mathcal{S}_1$, which translates Adv's interaction with $\mathcal{B}[\mathbf{\Sigma}_0]$ to an interaction with $\mathcal{B}[\mathbf{\Sigma}_1]$, behaves as follows: it passes on handles it receives from $\mathcal{B}[\mathbf{\Sigma}_1]$ as handles from $\mathcal{B}[\mathbf{\Sigma}_0]$. If the user sends an upload command, $\mathcal{S}_1$ will upload the agent as it is (since $\mathbf{\Sigma}_1$ allows deterministic agents as well). $\mathcal{S}_1$ also maintains a list of the form (h, i, x, y) where h is a handle obtained that does not correspond to an agent uploaded by Adv, (i, x) is an input for h from a session execution command given by Adv, and y is the output it reported back to Adv for that session. On receiving a new session request $h(z)$, i.e., for an agent handle h with input z, $\mathcal{S}_1$ behaves differently depending on whether h is a handle that corresponds to an agent uploaded by Adv, or not. In the former case, $\mathcal{S}_1$ simply forwards the request $h(z)$ to $\mathcal{B}[\mathbf{\Sigma}_1]$ and returns the response from $\mathcal{B}[\mathbf{\Sigma}_1]$ back to Adv. In the latter case, $\mathcal{S}_1$ interprets z as (i, x); then, if there is an entry of the form (h, i, x, y) in its list, $\mathcal{S}_1$ returns y to Adv; else it forwards the session request (h, x) to $\mathbf{\Sigma}_0$, and gets back (a fresh) output y, records (h, i, x, y) in its list, and sends y to User. It is easy to show, from the security of the PRF, that $\mathcal{S}_1$ satisfies the correctness requirements.

$(\mathcal{O}_2, \mathcal{E}_2)$ to reduce $\mathbf{\Sigma}_2$ to $\mathbf{\Sigma}_1$. We omit the detailed description of $\mathcal{O}_2, \mathcal{E}_2$ and the simulator $\mathcal{S}_2$ associated with this reduction, but instead just describe the behavior of the non-reactive agent P_1 that $\mathcal{O}_2$ sends to $\mathcal{B}[\mathbf{\Sigma}_1]$, when given a reactive agent P_2 of schema $\mathbf{\Sigma}_2$.

The idea is that the reactive agent P_2 can be implemented by a non-reactive agent P_1 which outputs an encrypted configuration of P_2 that can then be fed back as input to P_1. More precisely, P_1 will contain the parameters of P_2 and keys for a semantically secure symmetric-key encryption scheme and a message authentication code (MAC) built-in as its own parameters. If invoked with just an input for P_2, P_1 considers this an invocation of P_2 from its start configuration. In this case, P_1 uses its internal randomness to initialize a random-tape for P_2, and executes P_2 on the given input until it blocks or halts. Then (using fresh randomness) it produces an authenticated ciphertext of the resulting configuration of P_2. It outputs this encrypted configuration along with the (unencrypted) contents of the output tape of P_2. P_1 can also be invoked with an encrypted configuration and an input: in this case, it checks the authentication, decrypts the configuration (which contains the random tape for P_2) and executes P_2 starting from this configuration, with the given input added to the input-tape.

The security of this reduction follows from the semantic security of the encryption and the existential unforgeability of the MAC.

General Construction for Interactive Agents

In this section, we shall reduce a general schema $\mathbf{\Sigma} = (\mathcal{P}_{\mathsf{auth}}, \mathcal{P}_{\mathsf{user}})$ to the schema $\mathbf{\Sigma}_R$ from Section 3.2.1, which consists of arbitrary randomized (non-reactive, non-interactive) agents. Combined with the first of two reductions from the previous section, using Theorem 3, this gives a reduction of $\mathbf{\Sigma}$ to $\mathbf{\Sigma}_{\text{OBF}}$.

Our reduction $(\mathcal{O}, \mathcal{E})$ is fairly simple. At a high-level, $\mathcal{O}$ will upload an agent called P_{run} to $\mathcal{B}[\mathbf{\Sigma}_R]$, which will be used as a key for carrying out all sessions. The agents in the sessions are maintained as encrypted and authenticated configurations, which the P_{run} will decrypt, execute and update, and then reencrypt and sign. Note that for this P_{run} needs to be a randomized agent (hence the reduction to $\mathbf{\Sigma}_R$ rather than $\mathbf{\Sigma}_{\text{OBF}}$).

More precisely, during setup, $\mathcal{O}_{\mathsf{setup}}$ will pick a secret-key and public-key pair (SK, PK) for a CCA2-secure public-key encryption, and a pair of signing and verification keys (Sig, Ver) for a digital signature scheme, and sets $MPK = PK$ and $MSK = (SK, PK, Sig, Ver)$. It will also upload the following randomized agent P_{run} to $\mathcal{B}[\mathbf{\Sigma}_R]$: the parameters of P_{run} include MSK.

1. P_{run} takes as input $((C_1, \sigma_1, x_1), \cdots, (C_t, \sigma_t, x_t))$ for $t \geq 1$, where C_i are encrypted configurations of agents in $\mathcal{P}_{\mathsf{auth}} \cup \mathcal{P}_{\mathsf{user}}$, σ_i are signatures on C_i, and x_i are inputs for the agents.

2. It decrypts each C_i using SK. It also checks the signatures on all the ciphertexts using Ver, except for the ciphertexts that contain a start configuration[3] of an agent in $\mathcal{P}_{\mathsf{user}}$.

3. If all the configurations and signatures are valid, then first P_{run} chooses a seed for a pseudorandom generator (PRG) to define the random-tape of each agent in a start configuration.[4]

[3]A start configuration has all the tapes, except the parameter tape, empty. The configuration also contains information about the agent family that the agent belongs to.

[4]We use the PRG in a stream-cipher mode: it produces a stream of pseudorandom bits,

4. Then P_{run} copies the inputs x_i to input tapes of the respective agents and carries out a session execution.

5. When the session terminates, P_{run} encrypts each agent's configuration (along with the updated seed of the PRG that defines its random tape), to obtain ciphertexts C'_i; it signs them using Sig to get signatures σ'_i; finally, it halts after outputting $((C'_1, \sigma'_1, y_1), \cdots, (C'_t, \sigma'_t, y_t))$, where y_i are the contents of the output tapes of the agents.

After setup, when $\mathcal{O}_{\mathsf{auth}}$ is given an agent in $\mathcal{P}_{\mathsf{auth}}$ by Test, it simply encrypts (the start configuration of) the agent, signs it, and outputs the resulting pair (C, σ) as the obfuscation of the given agent. $\mathcal{O}_{\mathsf{user}}$ only encrypts the agent and outputs $(C, \perp)$ as the obfuscation; it is important that the encryption scheme used is CCA2 secure.

$\mathcal{E}$ behaves as follows. During setup, $\mathcal{E}$ receives PK from $\mathcal{O}$ and a handle from $\mathcal{B}[\mathbf{\Sigma}_R]$. $\mathcal{E}$ sends User the handles corresponding to agents which are uploaded by User, or received (as cryptographically encoded agents) from $\mathcal{O}$, or (as part of session output) from P_{run}. For each agent uploaded by User, $\mathcal{E}$ stores its parameters (i.e., start configuration) encrypted with PK, indexed by its handle. For cryptographically encoded agents received from $\mathcal{O}$ or P_{run}, it stores the obfuscation (C, σ), indexed by its handle. When given a session execution command, $\mathcal{E}$ retrieves the cryptographically encoded agents stored for each handle (with an empty signature if it is an agent in $\mathcal{P}_{\mathsf{user}}$ with start configuration) and sends them to $\mathcal{B}[\mathbf{\Sigma}_R]$, along with the handle for P_{run}. It gets back $((C'_1, \sigma'_1, y_1), \cdots, (C'_t, \sigma'_t, y_t))$ as the output from the session. It stores each (C'_i, σ'_i) received with a new handle, and sends these handles along with the outputs y_i to User.

The correctness of this reduction is straightforward, depending only on the security of the PRG (and only the correctness of the encryption and signature schemes). To prove the security property, we sketch a simulator $\mathcal{S}$. It internally runs $\mathcal{O}_{\mathsf{setup}}$ to produce (MSK, MPK) and sends the latter to Adv. It also sends Adv a handle, to simulate the handle for P_{run} it would receive during the setup phase. Subsequently, when $\mathcal{S}$ receives handles from $\mathcal{B}[\mathbf{\Sigma}]$ for agents uploaded by Test, it simulates the output of $\mathcal{O}_{\mathsf{auth}}$ or $\mathcal{O}_{\mathsf{user}}$

such that at any point there is an updated seed that can be used to continue extracting more bits from the PRG.

(depending on whether the handle is for $\mathcal{P}_{\mathsf{auth}}$ or $\mathcal{P}_{\mathsf{user}}$) by encrypting a dummy configuration for an agent. Note that the ciphertexts produced by $\mathcal{O}_{\mathsf{user}}$ are not signed. Also, when $\mathcal{S}$ receives new handles from a session executed by $\mathcal{B}[\mathbf{\Sigma}]$, it simulates the output of P_{run}, again by encrypting dummy configurations (these are signed ciphertexts). $\mathcal{S}$ hands over all such simulated ciphertexts to Adv, and also records them along with the corresponding handles it received from $\mathcal{B}[\mathbf{\Sigma}]$. When Adv sends a session execution command for P_{run}, with an input of the form $((C_1, \sigma_1, x_1), \cdots, (C_t, \sigma_t, x_t))$, $\mathcal{S}$ attempts to find a handle for each C_i as follows: first, $\mathcal{S}$ looks up the handles for the ciphertexts, if any, that it has recorded already. Note that if a ciphertext has a valid signature, it must have been generated and recorded by $\mathcal{S}$. But if there is any ciphertext which is not signed, and which does not appear in $\mathcal{S}$'s table, then $\mathcal{S}$ will decrypt the ciphertext; if that gives a valid start configuration for an agent in $\mathcal{P}_{\mathsf{user}}$, then $\mathcal{S}$ will upload that agent to $\mathcal{B}[\mathbf{\Sigma}]$, and obtains a handle for it. As we shall see, this is where the CCA2 security is crucial, as explained below. (If any of the above steps fail (invalid signature, invalid ciphertext or invalid decrypted configuration), $\mathcal{S}$ can simply simulate an empty output from P_{run}.) Once it has a handle h_i for every C_i, $\mathcal{S}$ asks $\mathcal{B}[\mathbf{\Sigma}]$ to execute a session with those handles and inputs $x_1, \cdots, x_t$. It returns the resulting outputs as well as dummy ciphertexts (as already described) as the output from P_{run}.

The proof that the simulation is good relies on the CCA2 security of the encryption scheme (as well as the unforgeability of the signatures, and the security of the PRG). Note that on obtaining handles for various agents from $\mathcal{B}[\mathbf{\Sigma}]$, $\mathcal{S}$ hands over dummy ciphertexts to Adv, and if Adv gives them back to $\mathcal{S}$, it translates them back to the handles. Every other ciphertext is decrypted by $\mathcal{S}$ and used to create an agent that it uploads. However, if the encryption scheme were malleable, Adv could generate such a ciphertext by malleating one of the ciphertexts it received (from $\mathcal{S}$ or from, say, $\mathcal{O}_{\mathsf{user}}$). Thus in the real execution, the agent created by Adv would be related to an agent created by $\mathcal{O}_{\mathsf{user}}$, where as in the simulation it would be related to dummy agent created by $\mathcal{S}$, leading to a distinguishing attack. CCA2 security prevents this: one can translate a distinguishing attack (Test, Adv and $\mathcal{S}$ together) to an adversary in the CCA2 security experiment, in which, though the adversary does not have access to the decryption keys as $\mathcal{S}$ would, it can still carry out the decryptions carried out by $\mathcal{S}$ using the decryption oracle in the CCA2 experiment. The details of this reduction are fairly routine, and hence omitted.

3.2.2 Impossibility of $\mathsf{IND\text{-}PRE}$ obfuscation for general functionalities

In this section we exhibit a class of programs $\mathcal{F}$ such that Test is hiding w.r.t $\mathbf{\Sigma}_{\text{OBF}(\mathcal{F})}$ but for any real world cryptographic scheme $(\mathcal{O}, \mathcal{E})$, Test is not hiding w.r.t. $\mathcal{O}$. The idea for our impossibility follows the broad outline of the impossibility of general virtual black box (VBB) obfuscation demonstrated by Barak et al. [17]. Intuitively the impossibility of VBB obfuscation by Barak et al. follows the following broad outline: consider a program P which expects code C as input. If the input code responds to a secret challenge α with a secret response β, then P outputs a secret bit b. Barak et al. show that using the code of P, one can construct nontrivial input code C that can be fed back to P forcing it to output the bit b. On the other hand, a simulator given oracle access to P cannot use it to construct a useful input code C and has negligible probability of guessing an input that will result in P outputting the secret b. For more details, we refer the reader to [17].

At first glance, it is not clear if the same argument can be used to rule out $\mathsf{IND\text{-}PRE}$ secure obfuscation schema. The argument by Barak et al. seems to rely crucially on simulation based security, whereas ours is an indistinguishability style definition. Indeed, other indistinguishability style definitions such as indistinguishability obfuscation ($\mathsf{I\text{-}Obf}$) and differing input obfuscation ($\mathsf{DI\text{-}Obf}$) are conjectured to exist for all functions. However, our notion of indistinguishability preserving obfuscation is too strong to be achieved, as the following informal argument shows. Consider the same class of functions $\mathcal{F}$ as in [17], with the bit b as the secret. We construct Test which expects b as external input, and uploads agents from the function family $\mathcal{F}$. In the ideal world, it is infeasible to distinguish between $\mathsf{Test}(0)$ and $\mathsf{Test}(1)$ since it is infeasible to recover b from black box access. In the real world however, a user may execute a session in which the agent for P is executed to produce an agent for C, following which P may be run on C to output the secret bit.

3.2.3 Indistinguishability and Differing Inputs Obfuscation

First we provide formal definitions for indistinguishability obfuscation and differing-inputs obfuscation.

Definition 10 (Indistinguishability Obfuscation). *A uniform* PPT *machine* $\textsc{obf}(\cdot)$ *is called an indistinguishability obfuscator for a circuit family* $\mathcal{F} = \{\mathcal{F}_\kappa\}$ *if it probabilistically maps circuits to circuits such that the following conditions are satisfied:*

- **Correctness:** $\forall \kappa \in \mathbb{N}$, $\forall C \in \mathcal{F}_\kappa$, *and* $\forall$ *inputs* x *we have that*

$$\Pr\left[C'(x) = C(x) : C' \leftarrow \textsc{obf}(1^\kappa, C)\right] = 1.$$

- **Relaxed Polynomial Slowdown:** *There exists a universal polynomial* p *such that for any circuit* C*, we have* $|C'| \leq p(|C|, \kappa)$ *where* $C' \leftarrow \textsc{obf}(1^\kappa, C)$.

- **Indistinguishability:** *For every pair of circuits* $C_0, C_1 \in \mathcal{F}_\kappa$*, such that* $\forall x,\ C_0(x) = C_1(x)$*, we have that for all* PPT *distinguishers* $\mathcal{D}$

$$\mathcal{D}\big(1^\kappa, \textsc{obf}(1^\kappa, C_0)\big) \approx \mathcal{D}\big(1^\kappa, \textsc{obf}(1^\kappa, C_1)\big).$$

Multiple obfuscated circuits: Using a hybrid argument, one can show that if $\textsc{obf}(\cdot)$ is an indistinguishability obfuscator, then security also holds against distinguishers who have access to multiple obfuscated circuits. More formally, let $(\mathcal{C}_0, \mathcal{C}_1)$ be a pair of sequence of circuits where $\mathcal{C}_b = \{C_{b,1}, C_{b,2}, \ldots, C_{b,\ell}\}$ for $b \in \{0,1\}$ (and ℓ is some polynomial in κ). Suppose for every $i \in [1, \ell]$ and for all x, $C_{0,i}(x) = C_{1,i}(x)$. Then, for all PPT distinguishers $\mathcal{D}$ we have that

$$\mathcal{D}\big(1^\kappa, \textsc{obf}(1^\kappa, C_{0,1}), \ldots, \textsc{obf}(1^\kappa, C_{0,\ell})\big) \approx \mathcal{D}\big(1^\kappa, \textsc{obf}(1^\kappa, C_{1,1}), \ldots, \textsc{obf}(1^\kappa, C_{1,\ell})\big).$$

Definition 11 (Differing Inputs Obfuscation). *A uniform* PPT *machine* $\textsc{obf}(\cdot)$ *is called a differing inputs obfuscator for a circuit family* $\mathcal{F} = \{\mathcal{F}_\kappa\}$ *if it probabilistically maps circuits to circuits such that it satisfies the correctness and relaxed polynomial slowdown conditions as in Definition 10 and also:*

- **Differing Inputs:** *For every algorithm* $\mathsf{Sampler}$ *which takes* 1^κ *as input and outputs* (C_0, C_1, aux)*, where* $C_0, C_1 \in \mathcal{F}_\kappa$*, if for all* PPT $\mathcal{A}$

  $$\Pr\big[C_0(x) \neq C_1(x) : (C_0, C_1, \mathsf{aux}) \leftarrow \mathsf{Sampler}(1^\kappa); \\ x \leftarrow \mathcal{A}(1^\kappa, C_0, C_1, \mathsf{aux})\big] \leq \mathrm{negl}(\kappa),$$

  then for all PPT *distinguishers* $\mathcal{D}$*,*

  $$\mathcal{D}\big(1^\kappa, \textsc{obf}(1^\kappa, C_0), \mathsf{aux}\big) \approx \mathcal{D}\big(1^\kappa, \textsc{obf}(1^\kappa, C_1), \mathsf{aux}\big).$$

We call the $\mathsf{Sampler}$ whose output satisfies the condition given above (against all PPT $\mathcal{A}$) a *good* sampler. Note that differing inputs obfuscation requires indistinguishability to hold for good samplers only.

Multiple obfuscated circuits : Like in the case of indistinguishability obfuscation, we can show that if a differing inputs obfuscator $\textsc{obf}(\cdot)$ exists, then it is also secure against the following more general class of sampling functions. Let $\mathsf{Sampler}_\ell$ be an algorithm that on input 1^κ outputs a pair of sequence of circuits $(\mathcal{C}_0, \mathcal{C}_1)$ and aux, where $\mathcal{C}_b = \{C_{b,1}, C_{b,2}, \ldots, C_{b,\ell}\}$ for $b \in \{0,1\}$ (and ℓ is some polynomial in κ). We claim that if $\mathsf{Sampler}_\ell$ is *good*, i.e., for all PPT $\mathcal{A}$,

$$\Pr\big[C_{0,i}(x) \neq C_{1,i}(x) : (\mathcal{C}_0, \mathcal{C}_1, \mathsf{aux}) \leftarrow \mathsf{Sampler}(1^\kappa); \\ (x, i) \leftarrow \mathcal{A}(1^\kappa, \mathcal{C}_0, \mathcal{C}_1, \mathsf{aux})\big] \leq \mathrm{negl}(\kappa),$$

then for all PPT distinguishers $\mathcal{D}$,

$$\mathcal{D}\big(1^\kappa, \textsc{obf}(1^\kappa, C_{0,1}), \ldots, \textsc{obf}(1^\kappa, C_{0,\ell}), \mathsf{aux}\big) \approx \\ \mathcal{D}\big(1^\kappa, \textsc{obf}(1^\kappa, C_{1,1}), \ldots, \textsc{obf}(1^\kappa, C_{1,\ell}), \mathsf{aux}\big).$$

‘

We can prove this claim via a hybrid argument. For $i \in [0, \ell]$, let $\mathcal{H}_i$ be the hybrid consisting of $\textsc{obf}(C_{1,1}), \ldots, \textsc{obf}(C_{1,i})$, $\textsc{obf}(C_{0,i+1}), \ldots, \textsc{obf}(C_{0,\ell})$ and aux (κ has been omitted for convenience). In order to show that $\mathcal{H}_0$ is indistinguishable from $\mathcal{H}_\ell$, it is sufficient to show that for every $i \in [0, \ell - 1]$, $\mathcal{H}_i$ is indistinguishable from $\mathcal{H}_{i+1}$. Both $\mathcal{H}_i$ and $\mathcal{H}_{i+1}$ have $\textsc{obf}(C_{1,1}), \ldots, \textsc{obf}(C_{1,i})$,

$\text{OBF}(C_{0,i+2}), \ldots, \text{OBF}(C_{0,\ell})$ and aux in common. The only difference is that while the former has $\text{OBF}(C_{0,i+1})$, the latter has $\text{OBF}(C_{1,i+1})$.

Consider an algorithm $\mathsf{Sampler}$ which on input 1^κ, runs $\mathsf{Sampler}_\ell(1^\kappa)$ and outputs $(C_{0,i+1}, C_{1,i+1}, \mathsf{aux}')$, where $\mathsf{aux}' = (\mathcal{C}_0, \mathcal{C}_1, \mathsf{aux})$. We can easily show that $\mathsf{Sampler}$ is a good sampling algorithm if $\mathsf{Sampler}_\ell$ is good. Hence, $(\text{OBF}(C_{0,i+1}), \mathsf{aux}')$ is indistinguishable from $(\text{OBF}(C_{1,i+1}), \mathsf{aux}')$. This implies that $\mathcal{H}_i$ is indistinguishable from $\mathcal{H}_{i+1}$.

3.2.4 Relation to existing notions of Obfuscation

In this section we relate the security notions for obfuscation schema to the standard notions of obfuscation known in literature, such as indistinguishability obfuscation and differing inputs obfuscation. These relaxations of the VBB definition were first proposed by Barak et al. [17], but no constructions were discovered for a long time. Recently, Garg et al. [14] proposed the first candidate for indistinguishability obfuscation. Later works assume that this candidate is also a differing inputs obfuscator [35].

Conversion: Firstly, we note that a (set-up free) obfuscation scheme $(\mathcal{O}, \mathcal{E})$ in our framework can be easily mapped to the syntax of an obfuscation scheme in the traditional sense. It is easy to see that the efficiency requirement of $(\mathcal{O}, \mathcal{E})$ implies a pre-determined poly(κ) upperbound on the execution time of $\mathcal{E}$ on a single invocation (because, the agents in $\mathbf{\Sigma}_{\text{OBF}}$ have a pre-determined poly(κ) upperbound on their running time). Hence we can define a circuit $\mathcal{E}[O]$ (with a built-in string O) of a pre-determined poly(κ) size that carries out the following computation: on input x, it interacts with an internal copy of $\mathcal{E}$, first simulating to it the message O from $\mathcal{O}$ (upon which $\mathcal{E}$ will output a handle), followed by a request from User to execute a session with that handle and input x; $\mathcal{E}[O]$ outputs whatever $\mathcal{E}$ outputs. Let $\mathcal{O} \circ \mathcal{E}$ denote a program which, on input a circuit C (of size at most $s(\kappa)$), invokes $\mathcal{O}$ on C to obtain a string O, and then outputs the program $\mathcal{E}[O]$.

Indistinguishability Obfuscation. We now show that if we restrict our attention to the family of tests $\Delta_{\mathsf{det}} \subset \Delta$ where D is a deterministic party, then a secure scheme for this family exists iff an indistinguishability obfuscator does. Formally,

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

Proof. Suppose $(\mathcal{O}, \mathcal{E})$ is a set-up free Δ_{det}-$\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{OBF}}$, then $\mathcal{O} \circ \mathcal{E}$ is an indistinguishability obfuscator, where $\mathcal{O} \circ \mathcal{E}$ is defined as discussed before. By construction, $\mathcal{O} \circ \mathcal{E}$ satisfies the correctness and polynomial slowdown requirements. So suppose $\mathcal{O} \circ \mathcal{E}$ does not satisfy the indistinguishability preservation property. Then there exists two circuits C_0 and C_1 which have identical input-output behavior, but there exists a PPT algorithm $\mathcal{D}$ which distinguishes between of $\mathcal{O} \circ \mathcal{E}(C_0)$ and $\mathcal{O} \circ \mathcal{E}(C_1)$. Now, define a simple $\mathsf{Test} \in \Delta_{\mathsf{det}}$ which on input b, uploads C_b. It is easy to see that Test is hiding w.r.t. $\mathbf{\Sigma}_{\text{OBF}}$ as the User gets only black-box access to C_0 or C_1. On the other hand, we argue that Test is not hiding w.r.t. $\mathcal{O}$. For this consider an adversary Adv which, on obtaining a string O from $\mathcal{O}$ constructs the program $Z := \mathcal{E}[O]$ and invokes $\mathcal{D}(Z)$. Then $\text{REAL}\langle \mathsf{Test}(0) \mid \mathcal{O} \mid \mathsf{Adv}\rangle \not\approx \text{REAL}\langle \mathsf{Test}(1) \mid \mathcal{O} \mid \mathsf{Adv}\rangle$ follows from the fact that Z is distributed as $\mathcal{O} \circ \mathcal{E}(C_b)$, where b is the input to Test, and the distinguishing advantage of $\mathcal{D}$.

We now show how $\text{OBF}(.)$, an indistinguishability obfuscator, yields a (perfectly correct) set-up free Δ_{det}-$\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{OBF}}$. Together with the observation above, this will prove the lemma. We know that OBF maps circuits to circuits. Hence, $\mathcal{O}$ on input a circuit C runs OBF on the same input to obtain another circuit C'. This latter circuit is forwarded to $\mathcal{E}$. When $\mathcal{E}$ receives a circuit, it forwards a handle to the User; and when it receives a handle h and an input x from the User, it executes the circuit C' corresponding to h on x, and returns $C'(x)$. Correctness easily follows from the construction of $\mathcal{O}$ and $\mathcal{E}$.

Now, suppose that $(\mathcal{O}, \mathcal{E})$ is not a secure implementation. This implies that there exists a $\mathsf{Test} \in \Delta_{\mathsf{det}}$ which is hiding w.r.t $\mathbf{\Sigma}_{\text{OBF}}$ but not w.r.t. $\mathcal{O}$. Hence, there exists an adversary Adv which can distinguish between $\mathsf{Test}(0)$ and $\mathsf{Test}(1)$ in the real world. Using Test and Adv, we construct a distinguisher $\mathcal{D}$ as follows. Recall that $\mathsf{Test}(b)$ can be represented as $\mathsf{D} \circ \mathfrak{cs}(b)$, where D is a deterministic party. $\mathcal{D}$ internally simulates a real world set-up with D, $\mathcal{O}$ and Adv. Note that every pair of circuits (C_0, C_1) that D sends to $\mathfrak{c}$ must be equivalent, otherwise Test would not be hiding w.r.t. $\mathbf{\Sigma}_{\text{OBF}}$. When D uploads

(C_0, C_1), $\mathcal{D}$ forwards them to Adv and the challenger. Let us say that the challenger picks a bit $b \in \{0,1\}$. When $\mathcal{D}$ receives $\text{OBF}(C_b) = \mathcal{O}(C_b)$ from the challenger, he forwards it to Adv. Finally, $\mathcal{D}$ outputs the view of the adversary. Since the view of Adv in the experiment where challenger picks b is identical to its view in the real world when Test has input b, $\mathcal{D}$ succeeds in distinguishing between the case where challenger picks 0 from the case where it picks 1. $\square$

Differing Inputs Obfuscation. Next we show that if we consider the family of tests which do not receive any input from the user, then a secure scheme for this family exists iff a differing input obfuscator does. Formally,

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

Proof. We first show the only if direction. Let $(\mathcal{O}, \mathcal{E})$ be a Δ^*-IND-PRE-secure scheme for $\mathbf{\Sigma}_{\text{OBF}}$. We claim that $\mathcal{O} \circ \mathcal{E}$ is a differing inputs obfuscator, where $\mathcal{O} \circ \mathcal{E}$ is defined as discussed before. Towards this, let $\mathcal{S}$ be a good sampling algorithm which takes 1^κ as input and outputs (C_0, C_1, aux). We define a $\mathsf{Test}_{\mathcal{S}} \in \Delta^*$ as follows: D runs $\mathcal{S}$ to obtain (C_0, C_1, aux); it sends aux to the User and (C_0, C_1) to $\mathfrak{c}$. The only way an adversary can distinguish between the case where $\mathfrak{s}$ uploads C_0 from the case where it uploads C_1 is if it queries $\mathcal{B}[\mathbf{\Sigma}_{\text{OBF}}]$ with an input x s.t. $C_0(x) \neq C_1(x)$. But this is not possible because $\mathcal{S}$ is a good sampler. Therefore, $\mathsf{Test}_{\mathcal{S}}$ is hiding w.r.t. $\mathbf{\Sigma}_{\text{OBF}}$. This implies that $\mathsf{Test}_{\mathcal{S}}$ is hiding w.r.t. $\mathcal{O}$ as well. It is now easy to show that $(\mathcal{O} \circ \mathcal{E}(C_0), \mathsf{aux})$ is indistinguishable from $(\mathcal{O} \circ \mathcal{E}(C_1), \mathsf{aux})$.

We now show the if direction of the lemma. Suppose $\text{OBF}(\cdot)$ is a differing inputs obfuscator. Using OBF we can define a scheme $(\mathcal{O}, \mathcal{E})$ in the natural way (see the proof of the previous lemma for details). We claim that $(\mathcal{O}, \mathcal{E})$ is an Δ^*-IND-PRE-secure scheme for $\mathbf{\Sigma}_{\text{OBF}}$. Correctness easily follows from construction. In order to prove indistinguishability preservation, consider a $\mathsf{Test} \in \Delta^*$. Let $(\mathcal{C}_0, \mathcal{C}_1)$ denote the sequence of pairs of circuits uploaded by D, where $\mathcal{C}_b = \{C_{b,1}, C_{b,2}, \ldots, C_{b,\ell}\}$ for $b \in \{0,1\}$, and aux be the messages sent to the User. Observe that for both $b = 0$ and 1, any adversary Adv receives $\mathcal{C}_0, \mathcal{C}_1, \mathsf{aux}$, and a sequence of handles $h_1, \ldots, h_\ell$. If Test is hiding w.r.t $\mathbf{\Sigma}_{\text{OBF}}$, then the probability that Adv queries with h_i and input x such that $C_{0,i}(x) = C_{1,i}(x)$ is negligible.

Hence an algorithm Sampler which runs Test and outputs $(\mathcal{C}_0, \mathcal{C}_1, \mathsf{aux})$ is a good sampling algorithm. Therefore, $(\text{OBF}(C_{0,1}), \ldots, \text{OBF}(C_{0,\ell}), \mathsf{aux})$ cannot be distinguished from $(\text{OBF}(C_{1,1}), \ldots, \text{OBF}(C_{1,\ell}), \mathsf{aux})$. We can now show that Test is hiding w.r.t. $\mathcal{O}$ in a manner similar to the previous lemma. □

3.2.5 Adaptive Differing Inputs Obfuscation

Earlier, we saw that indistinguishability obfuscation is equivalent to Δ_{det}-IND-PRE and differing inputs obfuscation is equivalent to Δ^*-IND-PRE. In Section 3.2.2, we saw that IND-PRE secure obfuscation is impossible for general functionalities. It is natural to ask what happens "in-between", i.e. for Δ family of tests?

To this end, we state a 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. We define it formally below. An equivalent notion was defined in [68].

Good sampler : Let $\mathcal{F} = \{\mathcal{F}_\kappa\}$ be a circuit family. Let Sampler be a PPT stateful oracle which takes 1^κ as input, and upon every invocation outputs two circuits $C_0, C_1 \in \mathcal{F}_\kappa$ and some auxiliary information aux. We call this oracle *good* if for every PPT adversary $\mathcal{A}$ with oracle access to Sampler, the probability that $\mathcal{A}$ outputs an x such that $C_0(x) \neq C_1(x)$ for some C_0, C_1 given by Sampler, is negligible in κ.

Definition 12 (Adaptive Differing Inputs Obfuscation). *A uniform* PPT *machine* $\text{OBF}(\cdot)$ *is called an adaptive differing inputs obfuscator for a circuit family* $\mathcal{F} = \{\mathcal{F}_\kappa\}$ *if it probabilistically maps circuits to circuits such that it satisfies the following conditions:*

- **Correctness:** $\forall \kappa \in \mathbb{N}$, $\forall C \in \mathcal{F}_\kappa$, *and* $\forall$ *inputs* x *we have that*

$$\Pr\left[C'(x) = C(x) : C' \leftarrow \text{OBF}(1^\kappa, C)\right] = 1.$$

- **Relaxed Polynomial Slowdown:** *There exists a universal polynomial*

p such that for any circuit C, we have $|C'| \leq p(|C|, \kappa)$ where $C' \leftarrow$ $\textsc{obf}(1^\kappa, C)$.

- **Adaptive Indistinguishability:** *Let* $\mathsf{Sampler}$ *be a stateful oracle as described above. Define* $\mathsf{Sampler}_b$ *to be an oracle that simulates* $\mathsf{Sampler}$ *internally, and when* $\mathsf{Sampler}$ *outputs C_0, C_1 and* aux, $\mathsf{Sampler}_b$ *additionally outputs $\textsc{obf}(1^\kappa, C_b)$. We require that for every good* $\mathsf{Sampler}$, *for all* PPT *distinguishers $\mathcal{D}$*

$$\mathcal{D}^{\mathsf{Sampler}_0}(1^\kappa) \approx \mathcal{D}^{\mathsf{Sampler}_1}(1^\kappa).$$

As we shall see, this notion of obfuscation is very useful and we will be able to construct Δ-$\mathsf{IND\text{-}PRE}$ FE schema by providing a Δ reduction to a Δ-$\mathsf{IND\text{-}PRE}$ secure obfuscation schema (see Section 3.3 for more details).

3.3 Functional Encryption

In this section, we present a schema $\mathbf{\Sigma}_{\text{FE}}$ for Functional Encryption. Although all variants of FE can [5] 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 3.3.1 we introduce the schema $\mathbf{\Sigma}_{\text{FE}}$ for FE without function-hiding, and in Section 3.3.3 we introduce the schema $\mathbf{\Sigma}_{\text{FH-FE}}$ for function-hiding FE.

3.3.1 Functional Encryption without Function Hiding

Traditional Definition. The following definition of functional encryption is from [1, 10]. It corresponds to non-function-hiding, public-key functional encryption.

Syntax. A functional encryption scheme $\mathcal{FE}$ for a circuit family $\mathcal{F} = \{\mathcal{F}_\kappa\}$ over a message space $\mathcal{X} = \{\mathcal{X}_\kappa\}$ consists of four PPT algorithms:

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

- $\mathsf{Setup}(1^\kappa)$ takes as input the unary representation of the security parameter, and outputs the master public and secret keys $(\mathsf{MPK}, \mathsf{MSK})$;
- $\mathsf{KeyGen}(\mathsf{MSK}, C)$ takes as input the master secret key MSK and a circuit $C \in \mathcal{F}_\kappa$, and outputs a corresponding secret key SK_C;
- $\mathsf{Encrypt}(\mathsf{MPK}, x)$ takes as input the master public key MPK and a message $x \in \mathcal{X}_\kappa$, and outputs a ciphertext CT_x;
- $\mathsf{Decrypt}(\mathsf{SK}_C, \mathsf{CT}_x)$ takes as input a key SK_C and a ciphertext CT_x, and outputs a value.

These algorithms must satisfy the following *correctness* property for all $\kappa \in \mathbb{N}$, all $C \in \mathcal{F}_\kappa$ and all $x \in \mathcal{X}_\kappa$,

$$\Pr\left[\begin{array}{l}(\mathsf{MPK}, \mathsf{MSK}) \leftarrow \mathsf{Setup}(1^\kappa); \\ \qquad \mathsf{Decrypt}(\mathsf{KeyGen}(\mathsf{MSK}, C), \mathsf{Encrypt}(\mathsf{MPK}, x)) \neq C(x)\end{array}\right] = \mathrm{negl}(\kappa),$$

where the probability is taken over their coin tosses.

Indistinguishability Security. The standard indistinguishability based security definition for functional encryption is defined as a game between a challenger and an adversary $\mathcal{A}$ as follows.

- **Setup**: The challenger runs $\mathsf{Setup}(1^\kappa)$ to obtain $(\mathsf{MPK}, \mathsf{MSK})$, and gives MPK to $\mathcal{A}$.
- **Key queries**: $\mathcal{A}$ sends a circuit $C \in \mathcal{C}_\kappa$ to the challenger, and receives $\mathsf{SK}_C \leftarrow \mathsf{KeyGen}(\mathsf{MSK}, C)$ in return. This step can be repeated any polynomial number of times.
- **Challenge**: $\mathcal{A}$ submits two messages x_0 and x_1 such that $C(x_0) = C(x_1)$ for all C queried by $\mathcal{A}$ in the previous step. Challenger sends $\mathsf{Encrypt}(\mathsf{MPK}, x_b)$ to $\mathcal{A}$.
- **Adaptive key queries**: $\mathcal{A}$ continues to send circuits to the challenger subject to the restriction that any C queried must satisfy $C(x_0) = C(x_1)$.
- **Guess**: $\mathcal{A}$ outputs a bit b'.

The advantage of $\mathcal{A}$ in this security game is given by

$$\left|\mathsf{Pr}[b' = 1 \mid b = 1] - \mathsf{Pr}[b' = 1 \mid b = 0]\right|.$$

We say that a functional encryption scheme $\mathcal{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is *indistinguishability secure* if for all PPT adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the security game described above is negligible in κ.

Multiple challenge phases : One can show via a hybrid argument that if no adversary has a significant advantage in the above security game, then the same holds for a generalized game where there are multiple challenge phases interspersed with key query phases. In the generalized game, it is required that for every (x_0, x_1) submitted in a challenge phase, and every circuit C queried in any key query phase, $C(x_0) = C(x_1)$.

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 $\mathbf{\Sigma}_{\text{FE}} = (\mathcal{P}^{\text{FE}}_{\mathsf{auth}}, \mathcal{P}^{\text{FE}}_{\mathsf{user}})$ 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 [14, 35, 78, 37, 79], 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 $\mathbf{\Sigma}_{\text{FE}}$ be reduced to $\mathbf{\Sigma}_{\text{OBF}}$? For this reduction to translate to an $\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{FE}}$, we will need an $\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{OBF}}$, and a composition theorem.

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

Δ-$\mathsf{IND\text{-}PRE}$ secure implementation of $\mathbf{\Sigma}_{\text{FE}}$, thanks to Theorem 4. [6]

Before explaining our reduction, we compare it with the results in [14, 35, 37]. At a high-level, these works could be seen as giving "$(\Gamma_{\text{FE}}, \Gamma_{\text{OBF}})$-reductions" from $\mathbf{\Sigma}_{\text{FE}}$ to $\mathbf{\Sigma}_{\text{OBF}}$ 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 $\mathbf{\Sigma}_{\text{OBF}}$ one gets a Γ_{FE}-$\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{FE}}$. For example, in [14], $\Gamma_{\text{OBF}} = \Delta_{\mathsf{det}}$ (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 [14, 35, 37]. 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 $\mathbf{\Sigma}_{\text{FE}}$ to $\mathbf{\Sigma}_{\text{OBF}}$.

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 [37] as 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 $\mathbf{\Sigma}_{\text{FE}}$ to $\mathbf{\Sigma}_{\text{OBF}}$ and prove that it is in fact a Δ-reduction. Let $\mathbf{\Sigma}_{\text{FE}} = (\mathcal{P}^{\text{FE}}_{\mathsf{auth}}, \mathcal{P}^{\text{FE}}_{\mathsf{user}})$ and $\mathbf{\Sigma}_{\text{OBF}} = (\emptyset, \mathcal{P}^{\text{OBF}}_{\mathsf{user}})$. 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 $\mathcal{B}[\mathbf{\Sigma}_{\text{OBF}}]$, which behaves as follows: on input (f, σ)

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

$P_{m,\mathsf{MPK}}$ 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 $\perp$.

To show that this is a valid Δ-reduction, apart from verifying correctness, we need to demonstrate $\mathcal{S}$, H and K as required in Definition 8 and Definition 9. We describe these below.

- $\mathcal{S}$ will first simulate $\mathcal{O}_{\mathsf{setup}}$, by picking a signing and verification key pair itself, and publishing the latter as MPK. On obtaining a handle h_f for an agent in $\mathcal{P}^{\text{FE}}_{\mathsf{auth}}$, it runs the agent with no input to recover f, and then simulates $\mathcal{O}_{\mathsf{auth}}$ by outputting (f, σ) where σ is a signature on f. On obtaining a handle h for an agent in $\mathcal{P}^{\text{FE}}_{\mathsf{user}}$, it outputs a simulated handle h' from $\mathcal{B}[\mathbf{\Sigma}_{\text{OBF}}]$ (for the agent P_m uploaded by $\mathcal{O}_{\mathsf{user}}$), and internally keeps a record of the pair (h, h'). Subsequently, on receiving a session execution request for a simulated handle h' with some input, first $\mathcal{S}$ checks if the input is of the form (f, σ) and σ is a valid signature on f. If so, it looks for a handle h_f corresponding to f that it received from $\mathcal{B}[\mathbf{\Sigma}_{\text{FE}}]$; if no such handle exists, it aborts the simulation. Else it requests an execution of $\mathcal{B}[\mathbf{\Sigma}_{\text{FE}}]$ session involving two handles h_f and h, where (h, h') was the pair it had recorded when issuing the simulated handle h'. It returns the output from this $\mathcal{B}[\mathbf{\Sigma}_{\text{FE}}]$ session as the outcome of the execution of the simulated $\mathcal{B}[\mathbf{\Sigma}_{\text{OBF}}]$ session.

The probability $\mathcal{S}$ aborts is negligible, since any PPT adversary will have negligible probability of producing an f with a valid signature, if it was not given out by $\mathcal{S}$. Conditioned on the adversary never creating a forged signature, the simulated and real executions are identical.

- We can define H as follows. It implements $\mathcal{O}_{\mathsf{setup}}$ faithfully. When it is given a pair of agents in $\mathcal{P}^{\text{FE}}_{\mathsf{user}}$, it simply forwards both of them (to $\mathfrak{s}$). When it receives a pair of agents in $\mathcal{P}^{\text{FE}}_{\mathsf{auth}}$ from T, if they are not identical, H aborts; otherwise H will simulate the effect of $\mathcal{O}_{\mathsf{auth}}$ by signing the function f in (both) the agents, and forwards it to User. Now, conditioned on D never outputting a pair of distinct agents in $\mathcal{P}^{\text{FE}}_{\mathsf{auth}}$, we have $\mathsf{D} \circ \mathfrak{c} \circ \mathsf{H} \circ \mathfrak{s} \equiv \mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s} \circ \mathcal{O}$.

Now, if $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}$ is hiding w.r.t. $\mathbf{\Sigma}_{\text{FE}}$, then it must be the case that the probability of D outputting a pair of distinct agents is negligible. This is because, $\mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s}$ will forward the two agents to User, and if the two agents are not identical, the function-revealing nature of the schema, will let the User learn the secret bit b.

- We define K as follows. It observes the inputs sent to H (as reported to User by $\mathfrak{c}$), and whenever it sees a pair of agents in $\mathcal{P}^{\text{FE}}_{\mathsf{user}}$, it appends a copy of those two agents (as if it was reported the second instance of $\mathfrak{c}$ in $\mathfrak{c} \circ \mathsf{H} \circ \mathfrak{c}$). This in fact ensures that $\mathfrak{c} \circ \mathsf{H} \circ \mathfrak{c} \equiv \mathfrak{c} \circ \mathsf{H} \mathbin{/} \mathsf{K}$.

3.3.2 Indistinguishability Secure FE vs. Secure Schemes for FE Schema.

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. 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 3. *A Δ_{det}-$\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{FE}}$ exists if and only if there exists an indistinguishability secure FE scheme.*

Proof. We first prove the easier side. Let $(\mathcal{O}, \mathcal{E})$ be a Δ_{det}-$\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{FE}}$, where $\mathcal{O} = (\mathcal{O}_{\mathsf{setup}}, \mathcal{O}_{\mathsf{auth}}, \mathcal{O}_{\mathsf{user}})$. We construct an FE scheme $\mathbb{S}_{\text{FE}}$ using $(\mathcal{O}, \mathcal{E})$ as follows.

- $\mathsf{Setup}(1^{\kappa})$**:** Run $\mathcal{O}_{\mathsf{setup}}$ to obtain a master secret key MSK and a public key MPK.

- $\mathsf{KeyGen}(\mathsf{MSK}, C)$**:** Output $\mathsf{SK}_C \leftarrow \mathcal{O}_{\mathsf{auth}}(C; \mathsf{MSK})$ (where C is passed to $\mathcal{O}_{\mathsf{auth}}$ as the parameter for the agent $P^{\mathsf{Fun}}_C \in \mathcal{P}^{\mathsf{PubFE}}_{\mathsf{auth}}$).

- $\mathsf{Encrypt}(\mathsf{MPK}, x)$**:** Output $\mathsf{CT}_x \leftarrow \mathcal{O}_{\mathsf{user}}(x; \mathsf{MPK})$ (where x is passed to $\mathcal{O}_{\mathsf{user}}$ as the parameter for the agent $P^{\mathsf{Msg}}_x \in \mathcal{P}^{\mathsf{PubFE}}_{\mathsf{user}}$).

- $\mathsf{Decrypt}(\mathsf{SK}_C, \mathsf{CT}_x)$**:** Run a copy of $\mathcal{E}$ as follows: first feed it SK_C and CT_x as messages from $\mathcal{O}$, and obtain agent handles h_C and h_x; then request it for a session execution with handles (h_C, h_x) (and no input). Return the output for the agent h_C as reported by $\mathcal{E}$.

In order to show that $\mathbb{S}_{\text{FE}}$ is an indistinguishability secure FE scheme, we consider the following $\mathsf{Test} \in \Delta_{\mathsf{det}}$. Upon receipt of a circuit C from User, Test

uploads C and adds C to a list L. Upon receipt of a pair of inputs (x_0, x_1), if for every $C \in L$, $C(x_0) = C(x_1)$, Test uploads x_b. After this, if User sends a circuit C', Test uploads C' iff $C'(x_0) = C'(x_1)$. (If User sends any other type of message, it is ignored.)

Now suppose there is an adversary $\mathcal{A}$ who breaks the security of $\mathbb{S}_{\text{FE}}$. Then we show that the above Test is hiding w.r.t. $\mathbf{\Sigma}_{\text{FE}}$ but not w.r.t. $\mathcal{O}$. To see this, firstly note that Test is hiding w.r.t. $\mathbf{\Sigma}_{\text{FE}}$ by design: there is no way an adversary can learn whether Test uploaded x_0 or x_1 in the ideal world. Now, consider an adversary Adv who runs $\mathcal{A}$ internally: first it forwards MPK received from $\mathcal{O}_{\mathsf{setup}}$ to $\mathcal{A}$; then it forwards $\mathcal{A}$'s requests to the challenger (in the IND security game) to Test; the outputs received from $\mathcal{O}$ are forwarded to $\mathcal{A}$. Finally Adv outputs $\mathcal{A}$'s output bit. It is straightforward to see that the advantage Adv has in distinguishing interaction with $\mathsf{Test}(0)$ and $\mathsf{Test}(1)$ is exactly the advantage $\mathcal{A}$ has in the IND security experiment.

We now prove the other side of the lemma. Let $\mathcal{FE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ be an indistinguishability secure FE scheme. We can construct a Δ_{det}-IND-PRE-secure scheme $(\mathcal{O}, \mathcal{E})$ for $\mathbf{\Sigma}_{\text{FE}}$ using the scheme $\mathcal{FE}$ in a way analogous to how an IND secure FE scheme is constructed from an IND-PRE secure scheme above. We now show that if $(\mathcal{O}, \mathcal{E})$ is not a secure scheme then neither is $\mathcal{FE}$. That is, if there exists a $\mathsf{Test} \in \Delta_{\mathsf{det}}$ such that Test is hiding in the ideal world, but there exists a PPT adversary Adv which can distinguish between $\mathsf{Test}(0)$ and $\mathsf{Test}(1)$ in the real world, then there exists an adversary $\mathcal{A}$ which can break the security of $\mathcal{FE}$ in the generalized IND game.

Recall that $\mathsf{Test}(b)$ can be represented as $\mathsf{D} \circ \mathfrak{cs}(b)$, where D is a deterministic party. $\mathcal{A}$ internally simulates a real world set-up with D, $\mathcal{O}$ and Adv, and externally participates in the indistinguishability game with a challenger. We will show that for $b \in \{0, 1\}$, if challenger picks the bit b, then Adv's view is identically distributed to its view in the real world when Test gets input b. This will complete the proof.

At any point during a run of the real world, D either uploads a pair of function agents (C_0, C_1) or a pair of message agents (x_0, x_1) to $\mathfrak{c}$. We can see that C_0 and C_1 must be the same circuits, otherwise Test would not be hiding in the ideal world. Similarly, for every function agent $C = C_0 = C_1$

ever uploaded, it must be the case that $C(x_0) = C(x_1)$, for every (x_0, x_1). It is now easy to simulate the view of Adv. When a function agent C is uploaded by D, $\mathcal{A}$ sends C to the challenger, and forwards the key obtained to Adv (along with (C_0, C_1)). When D uploads (x_0, x_1), $\mathcal{A}$ forwards it to the challenger. The ciphertext returned by the challenger is forwarded to Adv (along with (x_0, x_1)). $\square$

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

Proof. The idea of the separation follows that in [1]. Let $\beta : \{0,1\}^n \to \{0,1\}^n$ be a one-way permutation, and h be its hard-core predicate. Consider a function family which has only one function f. For all $x \in \{0,1\}^n$, define $f(x) := \beta(x)$. Consider an FE scheme $\mathcal{FE}$ where $\mathsf{Encrypt}(x)$ is simply a public-key encryption (PKE) of x, and the secret key for f is the secret key of the PKE scheme. ($\mathsf{Decrypt}$ first runs the decryption algorithm of PKE to obtain x, and then outputs $\beta(x)$.) In the indistinguishability game, if the adversary doesn't ask for any key, then clearly he cannot distinguish. On the other hand, if he does request a key for f, he can only send identical messages to the challenger (β is a permutation), and therefore has no advantage. Hence, $\mathcal{FE}$ is secure under the standard indistinguishability based security definition.

On the other hand, if we transform $\mathcal{FE}$ to a scheme $(\mathcal{O}, \mathcal{E})$ in the schemata framework, we show that the latter is not secure. Consider a Test algorithm which on input a bit b, chooses an n-bit string x uniformly at random, uploads message agent x and sends $b \oplus h(x)$ to the User. It also uploads a function agent corresponding to f. Thus, the ideal user sees $\beta(x)$ and $b \oplus h(x)$. Clearly, in the ideal world a PPT adversary cannot distinguish between $\mathsf{Test}(0)$ and $\mathsf{Test}(1)$, since doing so would imply guessing the hard-core bit. However, in the real world distinguishing between $\mathsf{Test}(0)$ and $\mathsf{Test}(1)$ is trivial because decryption reveals x.

Finally, one can see that if $\mathcal{O}_{\mathsf{user}}$ simply outputs $\beta(x)$ on input x, then we get a secure $\mathsf{IND\text{-}PRE}$ scheme. $\square$

3.3.3 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 [4, 80, 5]. 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, $\mathbf{\Sigma}_{\text{FH-FE}} = (\mathcal{P}^{\text{FH-FE}}_{\mathsf{auth}}, \mathcal{P}^{\text{FH-FE}}_{\mathsf{user}})$, is identical to that of $\mathbf{\Sigma}_{\text{FE}}$, 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 $\mathsf{IND\text{-}PRE}$-secure scheme for the class of inner-product predicates, and a Δ-$\mathsf{IND\text{-}PRE}$-secure scheme for all function families.

- The first construction is in fact an information-theoretic reduction of the schema $\mathbf{\Sigma}_{\text{FH-FE}(\mathsf{IP})}$ (where IP denotes the class of inner-product predicates) to the schema $\mathbf{\Sigma}_{\text{BGG}}$. Thus under the assumption that there is an $\mathsf{IND\text{-}PRE}$ secure scheme for $\mathbf{\Sigma}_{\text{BGG}}$, we obtain a scheme for $\mathbf{\Sigma}_{\text{FH-FE}}$, using Theorem 1. This construction is essentially the same as a construction in the recent work of [5], which was presented in the generic group model. Intuitively, the simulation based proof in [5] may be interpreted as a simulation based reduction from $\mathbf{\Sigma}_{\text{FH-FE}(\mathsf{IP})}$ to $\mathbf{\Sigma}_{\text{GG}}$ satisfying Definition 8.

- The second construction is for general function-hiding FE: a Δ-$\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{FH-FE}}$, based on the assumption that a Δ-secure scheme for $\mathbf{\Sigma}_{\text{OBF}}$ 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).

Function Hiding FE for Inner-Product from Generic Group Schema

Lemma 5. $\mathbf{\Sigma}_{\text{FH-FE(IP)}}$ *reduces to* $\mathbf{\Sigma}_{\text{BGG}}$.

Proof. As mentioned earlier, our construction follows that of [5]. To formally define this as a reduction, i.e., a scheme $(\mathcal{O}, \mathcal{E})^{\mathbf{\Sigma}_{\text{BGG}}}$, we need to translate the use of generic groups in that construction to fit the interface of $\mathcal{B}[\mathbf{\Sigma}_{\text{BGG}}]$. Note that unlike in the generic group model, $\mathcal{B}[\mathbf{\Sigma}_{\text{BGG}}]$ does not send any handles to the scheme's $\mathcal{O}$ algorithm. Instead, $\mathcal{O}$ will work with a concrete group. Let $\widehat{\mathbb{S}}$ denote the construction $\mathbb{S}$, but instantiated with the concrete group $\mathbb{Z}_q$, where q is the order of the (source and target) groups provided by $\mathbf{\Sigma}_{\text{BGG}}$.[7] Then, we define $\mathcal{O}$ as follows:

- **Encoding scheme $\mathcal{O}$:**
  - $\mathcal{O}_{\mathsf{setup}}$: Run $\widehat{\mathbb{S}}.\mathsf{Setup}$ and obtain $(\mathsf{MPK}, \mathsf{MSK})$, each of which is a vector of elements in $\mathbb{Z}_q$. Create an agent for each group element in MPK, and send it to $\mathcal{B}[\mathbf{\Sigma}_{\text{BGG}}]$ (which will send a handle for it to the user). (If there were to be entries in MPK which are not group elements, $\mathcal{O}$ sends them directly to the user.)
  - $\mathcal{O}_{\mathsf{auth}}$: Given an agent $P_f \in \mathcal{P}^{\text{FH-FE}}_{\mathsf{auth}}$, extract f from P_f and let $\mathsf{SK}_f = \widehat{\mathbb{S}}.\mathsf{KeyGen}(\mathsf{MSK}, f)$. For each group element in SK_f, send it to $\mathcal{B}[\mathbf{\Sigma}_{\text{BGG}}]$.
  - $\mathcal{O}_{\mathsf{user}}$: Given an agent $P_m \in \mathcal{P}^{\text{FH-FE}}_{\mathsf{user}}$, let $\mathsf{CT}_m = \widehat{\mathbb{S}}.\mathsf{Encrypt}(\mathsf{MPK}, m)$. Again, or each group element in CT_m, send it to $\mathcal{B}[\mathbf{\Sigma}_{\text{BGG}}]$.
- **Executer $\mathcal{E}$:** Given handles corresponding to a function agent SK_f and handles corresponding to a message agent CT_m, $\mathcal{E}$ invokes $\mathbb{S}.\mathsf{Decrypt}$ with these handles. During the execution, $\mathbb{S}$ will require access to the

[7]Groups of different orders can also be handled, but for simplicity, we consider the source and target groups to be of the same order.

generic group operations, and at the end will output a group element which is either the identity (in which case the predicate evaluates to true) or not (in which case it evaluates to false).[8] $\mathcal{E}$ will use access to $\mathcal{B}[\mathbf{\Sigma}_{\text{BGG}}]$ to carry out the group operations, and at the end carry out an equality check to find out whether the final handle output by $\mathbb{S}.\mathsf{Decrypt}$ encodes the identity or not.

Correctness follows from correctness of $\mathbb{S}$. To define our simulator, we use the simulator $\mathbb{S}.\mathsf{sim}$, which simulates the generic group oracle to a user. Our simulator is slightly simpler compared to that for $\mathbb{S}$: there, the simulator proactively checked if a group element for which a handle is to be simulated would be equal to a group element for which a handle was previously issued, and if so, used the handle again. This is because, in the generic group model, a single group element has only one representation. In our case, the simulator will issue serial numbers as handles (as $\mathcal{B}[\mathbf{\Sigma}_{\text{BGG}}]$ would have done), and equality checks are carried out (using information gathered from $\mathcal{B}[\mathbf{\Sigma}_{\text{FE}}]$) only to correctly respond to equality check requests made by the user to $\mathcal{B}[\mathbf{\Sigma}_{\text{BGG}}]$. In all other respects, our simulator is the same as the simulator in the generic group model. The proof that the simulation is good also follows the same argument as there. $\square$

General Construction from Obfuscation

Lemma 6. *If there exists an* Δ-IND-PRE*-secure scheme for* $\mathbf{\Sigma}_{\text{OBF}}$*, then there exists a* Δ-IND-PRE*-secure scheme for* $\mathbf{\Sigma}_{\text{FH-FE}}$*.*

Proof. Let $\Pi^* = (\mathcal{O}^*, \mathcal{E}^*)$ that be a Δ-IND-PRE-secure scheme for $\mathbf{\Sigma}_{\text{OBF}}$. Then we define a scheme $(\mathcal{O}, \mathcal{E})$ for $\mathbf{\Sigma}_{\text{FH-FE}}$ as follows.

- **Encoding scheme** $\mathcal{O}$:
  - $\mathcal{O}_{\mathsf{setup}}$: Generate $(\mathsf{VK}, \mathsf{SK})$ as the verification key and signing key for a signature scheme. Output VK as MPK.

[8]Though not the case with the construction in [5], a general algorithm in the generic group model may check for identities by comparing handles, not just at the end, but at any point during its execution. In this case, $\mathcal{E}$ should proactively check every handle it receives from $\mathcal{B}[\mathbf{\Sigma}_{\text{BGG}}]$ against all previously received handles, to see if they encode the same group element; if so, the newly received handle is replaced with the existing one.

– $\mathcal{O}_{\mathsf{auth}}$: Given an agent $P_f \in \mathcal{P}^{\text{FH-FE}}_{\mathsf{auth}}$, define an agent $P'_f \in \mathcal{P}^{\text{OBF}}_{\mathsf{user}}$, which on input x outputs $f(x)$. Let $c := \big(\mathcal{O}^*(P'_f)\big)$ and $\sigma = \mathsf{Sign}_{\mathsf{SK}}(c)$. Send (c, σ) to User.

– $\mathcal{O}_{\mathsf{user}}$: Given an agent $P_m \in \mathcal{P}^{\text{FH-FE}}_{\mathsf{user}}$, define an agent $P''_{m,\mathsf{VK}} \in \mathcal{P}^{\text{OBF}}_{\mathsf{user}}$ as follows: on input (c, σ), check if $\mathsf{Verify}_{\mathsf{VK}}(c, \sigma)$ holds, and halt otherwise; if the signature does verify, invoke $\mathcal{E}^*$ with handle c and input m, and output whatever $\mathcal{E}^*$ outputs. Let $d = \mathcal{O}^*(P''_{m,\mathsf{VK}})$). Send d to User.

- **Executer $\mathcal{E}$:** Given handle (c, σ) corresponding to a function agent and a handle d corresponding to a message agent, $\mathcal{E}$ invokes $\mathcal{E}^*$ with handle d and input (c, σ). It outputs what $\mathcal{E}^*$ outputs.

The correctness of this construction is straightforward. To argue security, consider any test $\mathsf{Test} = \mathsf{D} \circ \mathfrak{c} \circ \mathfrak{s} \in \Delta$, such that Test is hiding w.r.t. $\mathbf{\Sigma}_{\text{FH-FE}}$. Then, for any PPT adversary Adv, we need to show that $\textsc{real}\langle \mathsf{Test}(0) \mid \mathcal{O} \mid \mathsf{Adv}\rangle \approx \textsc{real}\langle \mathsf{Test}(1) \mid \mathcal{O} \mid \mathsf{Adv}\rangle$. For this we consider an intermediate hybrid variable, defined as follows. Let $\widetilde{\mathfrak{s}}(0,1)$ indicate a modified version of $\mathfrak{s}$, which when given two agents P_{m_0}, P_{m_1} in $\mathcal{P}^{\text{FH-FE}}_{\mathsf{user}}$, selects P_{m_0}, but when given two agents P_{f_0}, P_{f_1} in $\mathcal{P}^{\text{FH-FE}}_{\mathsf{auth}}$, selects P_{f_1}. Then we claim that $\textsc{real}\langle \mathsf{Test0} \mid \mathcal{O} \mid \mathsf{Adv}\rangle \approx \textsc{real}\langle \mathsf{D} \circ \mathfrak{c} \circ \widetilde{\mathfrak{s}} \mid \mathcal{O} \mid \mathsf{Adv}\rangle \approx \textsc{real}\langle \mathsf{Test1} \mid \mathcal{O} \mid \mathsf{Adv}\rangle$. For simplicity, consider D which only outputs a single pair of function agents (P_{f_0}, P_{f_1}) and a single pair of message agents (P_{m_0}, P_{m_1}). (The general case is handled using a sequence of hybrids, in a standard way.)

To show the first approximate equality, consider a test Test' and adversary Adv' which work as follows. Test' internally simulates $\mathsf{Test}(0)$ and $\mathcal{O}$ with the following differences: when $\mathcal{O}_{\mathsf{setup}}$ outputs the signing key SK, Test' forwards it to Adv'; when the two agents P_{f_0}, P_{f_1} are sent to $\mathfrak{s}$, $\mathsf{Test}'(b)$ outputs P_{f_b} to $\mathcal{B}[\mathbf{\Sigma}_{\text{OBF}}]$ (or $\mathcal{O}^*$). Adv', when it receives c from $\mathcal{O}^*$, first signs it using SK to obtain σ, and then passes on (c, σ) to an internal copy of Adv; otherwise, it lets Adv directly interact with Test'. It can be seen that $\textsc{real}\langle \mathsf{Test}'(0) \mid \mathcal{O}^* \mid \mathsf{Adv}'\rangle = \textsc{real}\langle \mathsf{Test}(0) \mid \mathcal{O} \mid \mathsf{Adv}\rangle$ and $\textsc{real}\langle \mathsf{Test}'(1) \mid \mathcal{O}^* \mid \mathsf{Adv}'\rangle = \textsc{real}\langle \mathsf{D} \circ \mathfrak{c} \circ \widetilde{\mathfrak{s}} \mid \mathcal{O} \mid \mathsf{Adv}\rangle$. Further, $\mathsf{Test}' \in \Delta$. Also, it is easy to see that if Test' is not hiding w.r.t. $\mathbf{\Sigma}_{\text{OBF}}$, then Test is not hiding w.r.t. $\mathbf{\Sigma}_{\text{FH-FE}}$ (because User's interface to $\mathbf{\Sigma}_{\text{FH-FE}}$ can be used to emulate its interface to $\mathbf{\Sigma}_{\text{OBF}}$). Thus, if Test is hiding w.r.t. $\mathbf{\Sigma}_{\text{FH-FE}}$, then

$\text{REAL}\langle \mathsf{Test}'(0) \mid \mathcal{O}^* \mid \mathsf{Adv}' \rangle \approx \text{REAL}\langle \mathsf{Test}'(1) \mid \mathcal{O}^* \mid \mathsf{Adv}' \rangle$. This establishes that $\text{REAL}\langle \mathsf{Test0} \mid \mathcal{O} \mid \mathsf{Adv} \rangle \approx \text{REAL}\langle \mathsf{D} \circ \mathfrak{c} \circ \widetilde{\mathfrak{s}} \mid \mathcal{O} \mid \mathsf{Adv} \rangle$.

To show that $\text{REAL}\langle \mathsf{D} \circ \mathfrak{c} \circ \widetilde{\mathfrak{s}} \mid \mathcal{O} \mid \mathsf{Adv} \rangle \approx \text{REAL}\langle \mathsf{Test1} \mid \mathcal{O} \mid \mathsf{Adv} \rangle$, we consider another test Test''. Now, Test'' internally simulates $\mathsf{Test}(1)$ and $\mathcal{O}$ with the following differences: when the two message agents P_{m_0}, P_{m_1} are sent to $\mathfrak{s}$, $\mathsf{Test}''(b)$ sends $(P''_{m_0,\mathsf{VK}}, P''_{m_1,\mathsf{VK}})$ to Adv'' and outputs $P''_{m_b,\mathsf{VK}}$ to $\mathcal{B}[\mathbf{\Sigma}_{\text{OBF}}]$ (or $\mathcal{O}^*$), where $P''_{m_b,\mathsf{VK}}$ was as defined in the description of $\mathcal{O}_{\mathsf{user}}$. Then,

$$\text{REAL}\langle \mathsf{Test}''(0) \mid \mathcal{O}^* \mid \mathsf{Adv} \rangle = \text{REAL}\langle \mathsf{D} \circ \mathfrak{c} \circ \widetilde{\mathfrak{s}} \mid \mathcal{O} \mid \mathsf{Adv} \rangle, \text{and}$$
$$\text{REAL}\langle \mathsf{Test}''(1) \mid \mathcal{O}^* \mid \mathsf{Adv} \rangle = \text{REAL}\langle \mathsf{Test}(1) \mid \mathcal{O} \mid \mathsf{Adv} \rangle.$$

Also, as before, $\mathsf{Test}'' \in \Delta$. If Test'' is hiding w.r.t. $\mathbf{\Sigma}_{\text{OBF}}$, then we can conclude that

$$\text{REAL}\langle \mathsf{Test}''(0) \mid \mathcal{O}^* \mid \mathsf{Adv} \rangle \approx \text{REAL}\langle \mathsf{Test}''(1) \mid \mathcal{O}^* \mid \mathsf{Adv} \rangle,$$

and hence

$$\text{REAL}\langle \mathsf{D} \circ \mathfrak{c} \circ \widetilde{\mathfrak{s}} \mid \mathcal{O} \mid \mathsf{Adv} \rangle \approx \text{REAL}\langle \mathsf{Test}(1) \mid \mathcal{O} \mid \mathsf{Adv} \rangle.$$

Thus it only remains to show that Test'' is hiding w.r.t. $\mathbf{\Sigma}_{\text{OBF}}$. Firstly, by the security of the signature scheme, for any PPT adversary User, w.h.p., it does not query $\mathbf{\Sigma}_{\text{OBF}}$ with a handle and an input (c, σ) for a c that was not produced by Test''. Now, conditioned on this event, if User distinguishes $\mathsf{Test}''(0)$ and $\mathsf{Test}''(1)$, this User can be turned into one that distinguishes between $\mathsf{Test}(0)$ and $\mathsf{Test}(1)$ when interacting with $\mathbf{\Sigma}_{\text{FH-FE}}$. Thus, since Test is hiding w.r.t. $\mathbf{\Sigma}_{\text{FH-FE}}$, it follows that Test'' is hiding w.r.t. $\mathbf{\Sigma}_{\text{OBF}}$, as was required to be shown. $\square$

3.4 Fully Homomorphic Encryption

In this section, we present a cryptographic agent schema $\mathbf{\Sigma}_{\text{FHE}}$ 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 $x_2, x_3, \ldots, x_t$ 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 $P^{\mathsf{Dec}} \in \mathcal{P}^{\text{FHE}}_{\mathsf{auth}}$ 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.[9]

Given a Δ_{det}-IND-PRE secure scheme $(\mathcal{O}, \mathcal{E})$ for $\mathbf{\Sigma}_{\text{FHE}}$, we show how to construct a semantically secure FHE scheme $\mathbb{S}_{\text{FHE}} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Eval})$. For a formal treatment of FHE, see [81].

- $\mathsf{Setup}(1^\kappa)$: Run $\mathcal{O}_{\mathsf{setup}}$ to obtain public key PK and secret key SK.

- $\mathsf{Encrypt}(x, \mathsf{PK})$: Run $\mathcal{O}_{\mathsf{user}}((0, x), \mathsf{PK})$ to obtain a ciphertext CT_x. Here 0 denotes that x is a parameter for agent P^{Msg}.

- $\mathsf{Decrypt}(\mathsf{CT}, \mathsf{SK})$: Let $D \leftarrow \mathcal{O}_{\mathsf{auth}}(1, \mathsf{SK})$, where 1 denotes that the agent is P^{Dec}. Then run a copy of $\mathcal{E}$ as follows: first feed it D and CT as messages from $\mathcal{O}$, and obtain handles h_D and h_m; then request it for a session execution with $(h_D, \perp)$ and $(h_m, \perp)$. Return the output for the agent h_D as reported by $\mathcal{E}$.

- $\mathsf{Eval}(C, \mathsf{CT}_1, \mathsf{CT}_2, \ldots, \mathsf{CT}_n)$: Run a copy of $\mathcal{E}$ as follows: first feed it $\mathsf{CT}_1, \mathsf{CT}_2, \ldots, \mathsf{CT}_n$ as messages from $\mathcal{O}$, and obtain handles $h_1, h_2, \ldots, h_n$. Then request $\mathcal{E}$ to run a session with (h_1, f), $(h_2, \perp)$, $\ldots$, $(h_n, \perp)$. Output the ciphertext CT returned by $\mathcal{E}$.

Correctness follows easily from construction. Compactness follows from the fact that the size of string recorded for each handle by $\mathcal{E}$ is *a priori* bounded. We now show that $\mathbb{S}_{\text{FHE}}$ is semantically secure. On the contrary, suppose there

[9]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.

exists an adversary $\mathcal{A}$ who breaks the semantic security of $\mathbb{S}_{\text{FHE}}$. Consider the following $\mathsf{Test}(b)$: Upon receipt of inputs x_0, x_1 from User, Test chooses x_b and uploads it (as parameter for agent P^{Msg}). This Test is clearly hiding in the ideal world, because in the absence of a decryption agent, an adversary only obtains handles from $\mathcal{B}[\mathbf{\Sigma}_{\text{FHE}}]$. Therefore, by $\mathsf{IND\text{-}PRE}$ security of $(\mathcal{O}, \mathcal{E})$, Test is also hiding w.r.t. $\mathcal{O}$.

Now, consider an adversary Adv who runs $\mathcal{A}$ internally: first it forwards PK received from $\mathcal{O}_{\mathsf{setup}}$ to $\mathcal{A}$; then it forwards $\mathcal{A}$'s requests (x_0, x_1) to the challenger (in the semantic security game) to Test; the outputs received from $\mathcal{O}$ are forwarded to $\mathcal{A}$. Finally Adv outputs $\mathcal{A}$'s output bit. It is straightforward to see that the advantage Adv has in distinguishing interaction with $\mathsf{Test}(0)$ and $\mathsf{Test}(1)$ is exactly the advantage $\mathcal{A}$ has in the semantic security experiment.

3.5 Property Preserving Encryption

In this section, we formally define PPE and the standard notion of security for it [15].

Definition 13 (PPE scheme). *A property preserving encryption scheme for a binary property $P : \mathcal{M} \times \mathcal{M} \to \{0,1\}$ is a tuple of four* PPT *algorithms defined as follows:*

- $\mathsf{Setup}(1^\kappa)$ *takes as input the security parameter κ and outputs a secret key* SK *(and some public parameters).*
- $\mathsf{Encrypt}(m, \mathsf{SK})$ *takes as input a message $m \in \mathcal{M}$ and outputs a ciphertext* CT*.*
- $\mathsf{Decrypt}(\mathsf{CT}, \mathsf{SK})$ *takes as input a ciphertext* CT *and outputs a message $m \in \mathcal{M}$.*
- $\mathsf{Test}(\mathsf{CT}_1, \mathsf{CT}_2)$ *takes as input two ciphertexts CT_1 and CT_2 and outputs a bit b.*

We require that for all messages $m, m_1, m_2 \in \mathcal{M}$, the following two conditions hold:

- *Decryption:* $\mathsf{Pr}[\mathsf{SK} \leftarrow \mathsf{Setup}(1^\kappa); \mathsf{Decrypt}(\mathsf{Encrypt}(m, \mathsf{SK}), \mathsf{SK}) \neq m] = \mathsf{negl}(\kappa)$, *and*

- *Property testing:* $\mathsf{Pr}[\mathsf{SK} \leftarrow \mathsf{Setup}(1^\kappa); \mathsf{Test}(\mathsf{Encrypt}(m_1, \mathsf{SK}), \mathsf{Encrypt}(m_2, \mathsf{SK})) \neq P(m_1, m_2)] = \mathsf{negl}(\kappa)$,

where the probability is taken over the random choices of the four algorithms.

Security. In [15], the authors show that there exists a hierarchy of meaningful indistinguishability based security notions for PPE, which does not collapse unlike other familiar settings. At the top of the hierarchy lies *Left-or-Right* (LoR) security, a notion that is similar to full security in symmetric key functional encryption.

LoR **security.** Let $\Pi_P = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Test})$ be a PPE scheme for a binary property P. Consider an adversary $\mathcal{A}$ in the security game $\exp^{(b)}_{\mathsf{LoR},\mathcal{A}}(1^\kappa)$ described below, for $b \in \{0, 1\}$. The Setup algorithm is run to obtain a secret key SK and some public parameters. $\mathcal{A}$ is given the parameters, and access to an oracle $\mathcal{O}_b(\mathsf{SK}, \cdot, \cdot)$, such that $\mathcal{O}_b(\mathsf{SK}, m_0, m_1) = \mathsf{Encrypt}(m_b, \mathsf{SK})$. Let $Q = \{(m_1^{(0)}, m_1^{(1)}), (m_2^{(0)}, m_2^{(1)}), \ldots, (m_\ell^{(0)}, m_\ell^{(1)})\}$ denote the queries made by $\mathcal{A}$ to the oracle. At the end of the experiment, $\mathcal{A}$ produces an output bit; let this be the output of the experiment. We call $\mathcal{A}$ admissible if for every two (not necessarily distinct) pairs of messages $(m_i^{(0)}, m_i^{(1)}), (m_j^{(0)}, m_j^{(1)}) \in Q$, $P(m_i^{(0)}, m_j^{(0)}) = P(m_i^{(1)}, m_j^{(1)})$. We also refer to such messages as admissible.

Definition 14 (LoR security)**.** *The scheme* Π_P *is an* LoR *secure* PPE *scheme for a property* P *if for all* PPT *admissible adversaries* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *defined as below is negligible in the security parameter* κ*:*

$$\mathsf{Adv}_{\mathsf{LoR},\mathcal{A}}(\kappa) := \left| \Pr[\exp^{(0)}_{\mathsf{LoR},\mathcal{A}}(1^\kappa) = 1] - \Pr[\exp^{(1)}_{\mathsf{LoR},\mathcal{A}}(1^\kappa) = 1] \right|,$$

where the probability is over the random coins of the algorithms of Π_P *and that of* $\mathcal{A}$*.*

3.5.1 PPE as a schema

In this section, we present a cryptographic agent schema $\mathbf{\Sigma}_{\mathsf{PPE}}$ for property preserving encryption(PPE).

Let $P : \mathcal{M} \times \mathcal{M} \to \{0, 1\}$ be a (polynomial-time computable) binary property over the message space $\mathcal{M}$. The schema $\mathbf{\Sigma}_{\mathsf{PPE}} = (\mathcal{P}^{\mathsf{PPE}}_{\mathsf{auth}}, \emptyset)$, where $\mathcal{P}^{\mathsf{PPE}}_{\mathsf{auth}}$ has two kinds of agents, denoted by P^{Msg} and P^{Dec}, is defined as follows:

- P^{Msg} for a message $m \in \mathcal{M}$ is specified as follows: it has a message m on its parameter tape. When invoked with a command $\mathsf{compute}$ on its input tape, it reads a message m' from its communication tape, computes $P(m, m')$, outputs it and halts. When invoked with a command send, it sends its message m to the first agent in the session.

- P^{Dec} reads from its communication tape a single message and outputs it.

3.5.2 Equivalence

In this section, we show that Δ_{det}-$\mathsf{IND\text{-}PRE}$ and LoR security notions are equivalent for PPE.

Theorem 7. *A Δ_{det}-$\mathsf{IND\text{-}PRE}$ secure scheme for $\mathbf{\Sigma}_{\mathsf{PPE}}$ exists if and only if an LoR secure scheme for PPE exists.*

We first prove the only if side of the theorem. Let $(\mathcal{O}, \mathcal{E})$ be a Δ_{det}-$\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\mathsf{PPE}}$, where $\mathcal{O} = (\mathcal{O}_{\mathsf{setup}}, \mathcal{O}_{\mathsf{auth}}, \mathcal{O}_{\mathsf{user}})$. We construct a PPE scheme $\mathbb{S}_{\mathsf{PPE}}$ using $(\mathcal{O}, \mathcal{E})$ as follows.

- $\mathsf{Setup}(1^\kappa)$: Run $\mathcal{O}_{\mathsf{setup}}$ to obtain the public parameters MPK and secret key MSK.

- $\mathsf{Encrypt}(m, \mathsf{MSK})$: Output $CT_m \leftarrow \mathcal{O}_{\mathsf{auth}}((0, m), \mathsf{MSK})$ where the first bit 0 indicates that the parameter m is for the agent P^{Msg}.

- $\mathsf{Decrypt}(\mathsf{CT}, \mathsf{MSK})$: Let $D \leftarrow \mathcal{O}_{\mathsf{auth}}(1, \mathsf{MSK})$, where 1 denotes that the agent is P^{Dec}. Then run a copy of $\mathcal{E}$ as follows: first feed it D and CT as messages from $\mathcal{O}$, and obtain handles h_D and h_m; then request it for a session execution with $(h_D, \perp)$ and (h_m, send). Return the output for the agent h_D as reported by $\mathcal{E}$.

- $\mathsf{Test}(\mathsf{CT}_1, \mathsf{CT}_2)$: Run a copy of $\mathcal{E}$ as follows: first feed it CT_1 and CT_2 as messages from $\mathcal{O}$, and obtain handles h_1 and h_2. Then request $\mathcal{E}$ to run a session with $(h_1, \mathsf{compute})$ and (h_2, send). Output the answer returned by $\mathcal{E}$.

It is easy to see that the decryption and property testing properties of PPE are satisfied. In order to show that $\mathbb{S}_{\text{FE}}$ is an LoR secure PPE scheme, we consider the following $\mathsf{Test} \in \Delta_{\mathsf{det}}$. Let L be a list of pairs of messages, which is initially empty. Upon receipt of a pair (m_0, m_1) from User, Test checks if for every $(m_0', m_1') \in L$, $\mathsf{Test}(m_0, m_0') = \mathsf{Test}(m_1, m_1')$ and vice versa, and $\mathsf{Test}(m_0, m_0) = \mathsf{Test}(m_1, m_1)$. If the checks pass, Test uploads m_b and adds (m_0, m_1) to the list; otherwise this pair is ignored. (If Test receives a single message m from the User, it is treated as a pair (m, m).) Now suppose there
that the above Test is hiding w.r.t. $\mathbf{\Sigma}_{\mathsf{PPE}}$ but not w.r.t. $\mathcal{O}$. The proof is very similar to the one given for Lemma 3, so we omit it here.

Next, we prove the if side of the theorem. Let $\mathbb{S}_{\mathsf{PPE}} = (\mathsf{Setup}, \mathsf{Encrypt}, \mathsf{Decrypt}, \mathsf{Test})$ be an LoR secure PPE scheme. We construct a scheme $(\mathcal{O}, \mathcal{E})$ in the cryptographic agents framework as follows:

- $\mathcal{O}_{\mathsf{setup}}(1^\kappa)$**:** Run $\mathsf{Setup}(1^\kappa)$ to obtain $(\mathsf{MPK}, \mathsf{MSK})$.

- $\mathcal{O}_{\mathsf{auth}}((b, m); \mathsf{MSK})$**:** If $b = 0$, output $\mathsf{CT} \leftarrow \mathsf{Encrypt}(m, \mathsf{MSK})$, else output MSK itself. (Recall that 0 denotes an agent in P^{Msg}, while 1 denotes an agent in P^{Dec}.)

- $\mathcal{E}$**:** When $\mathcal{O}$ sends a ciphertext CT, forward a handle h to the User and store (h, CT). When $\mathcal{O}$ sends a key MSK, forward the handle h_{key} and store $(h_{\mathsf{key}}, \mathsf{MSK})$. When User requests a session execution with $(h_1, \mathsf{compute})$ and (h_2, send), retrieve the corresponding ciphertexts CT_1 and CT_2, and return $\mathsf{Test}(\mathsf{CT}_1, \mathsf{CT}_2)$ to the User. On the other hand, when User sends $(h_{\mathsf{key}}, \perp)$ and (h, send), retrieve the ciphertext CT corresponding to h, and return $\mathsf{Decrypt}(\mathsf{CT}, \mathsf{MSK})$ to the User.

We now show that if $(\mathcal{O}, \mathcal{E})$ is not a secure scheme then neither is $\mathbb{S}_{\mathsf{PPE}}$. That is, if there exists a $\mathsf{Test} \in \Delta_{\mathsf{det}}$ such that Test is hiding w.r.t. $\mathbf{\Sigma}_{\mathsf{PPE}}$ but not w.r.t. $\mathcal{O}$, then there exists an adversary $\mathcal{A}$ which can break the security of $\mathbb{S}_{\mathsf{PPE}}$ in the LoR security game. Recall that $\mathsf{Test}(b)$ can be represented as $\mathsf{D} \circ \mathfrak{cs}(b)$, where D is a deterministic party. It is clear that if D ever uploads a key agent, then every pair of messages (m_0, m_1) that it uploads must be such that $m_0 = m_1$ (otherwise Test would not be hiding in the ideal world). Such a Test would trivially be hiding in the real world. On the other hand, even if D never uploads a key agent, it must always upload admissible pairs of messages (see definition of LoR security) to remain hiding w.r.t. $\mathbf{\Sigma}_{\mathsf{PPE}}$. Rest of the proof is similar to Lemma 3, and hence omitted.

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 Section 3.5, and leave the others to future work on these objects.

3.6 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 3.2, an $\mathsf{IND\text{-}PRE}$-secure scheme for $\mathbf{\Sigma}_{\text{OBF}}$ 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. [17]. 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 [10] is easier to use in constructions than differing-inputs obfuscation, as exemplified by our constructions in Section 3.3.1 and Section 3.3.3.

Functional Encryption. Public-key function-hiding FE, as modeled by $\mathbf{\Sigma}_{\text{FH-FE}}$, 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 $\mathbf{\Sigma}_{\text{FH-FE}}$ 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 [4, 80], or using a relaxed simulation-based definition [82]. Δ-$\mathsf{IND\text{-}PRE}$ security implies the definitions of Boneh et al. [4, 80], but is in general incomparable with the simulation-based definition in [82]. 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 $\mathbf{\Sigma}_{\text{FE}}$) too, there are many known impossibility results, when simulation-based security definitions are used [1, 13, 11]. 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 $\mathbf{\Sigma}_{\text{FE}}$, 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

[10]or equivalently, adaptive differing-inputs obfuscation

scheme for $\mathbf{\Sigma}_{\text{FE}}$, assuming one for $\mathbf{\Sigma}_{\text{OBF}}$.

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 [83, 84, 85, 86, 87]. 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 Section 3.3.3, if we settle for Δ-$\mathsf{IND\text{-}PRE}$ security for the resulting scheme.

Chapter 4

Unbounded Simulation

A few years back, Bitansky and Canetti proposed a new definition of obfuscation, called Virtual Grey-Box (VGB) obfuscation [88]. Recently, Bitansky et al. [18] gave a surprising characterization of VGB obfuscation as being equivalent to a seemingly simpler definition of obfuscation, called *strong indistinguishability obfuscation* (SIO). Further, based on this, they showed that under a variant of a semantic-security assumption on graded encoding schemes (a.k.a. multi-linear maps) [77], any NC^1 circuit can be VGB-obfuscated.

We use VGB obfuscation as a guide to extend the agents framework to incorporate a statistical element into its security definition. One motivation for doing so is to obtain a similar security definition for *all primitives* that can be expressed as agents. These primitives include various forms of functional encryption, fully-homomorphic encryption, as well as graded encoding schemes. Going beyond the definition, we seek to prove results similar to those for VGB obfuscation in this abstract framework. Also, we seek to extend the machinery for composition in the original agents framework to our extension. In this work, we successfully carry out all elements of this program.

Firstly, we identify two new elements to incorporate into the agents framework. The first element is the notion of "statistical hiding." The main security definition in the agents framework is that of *indistinguishability preserving* (IND-PRE) security, which requires that if a "test" hides its input bit from all adversaries in an ideal world, it should hide it from all adversaries in the real world too. Originally, the adversaries in the real and ideal world were all required to be efficient. We present a new definition of s-IND-PRE (s for statistical), which requires a test to hide its input bit from all (efficient) real-world adversaries only if it hides it from all computationally unbounded[1]

[1]Even though computationally unbounded, the number of queries that an adversary can make in the ideal world is restricted to be polynomial in the security parameter.

adversaries in the ideal world.

The second definitional element we identify is a new test family, that we denote by Γ^*. It is a new family which consists of computationally unbounded, "non-interactive" tests. Here non-interactive means that the test will not receive any message from the outside (except its one bit input).

With these two new elements in place, we can define a new notion of security, namely, Γ^*-s-$\mathsf{IND\text{-}PRE}$*for any primitive* (or "schema") in the agents framework. Note that Γ^*-s-$\mathsf{IND\text{-}PRE}$ is an indistinguishability-preserving security notion, and does not involve a simulator, like in VGB obfuscation. But, our first result is that when applied to obfuscation, this definition *is equivalent to VGB obfuscation*. Further, we observe that the "semantic-security" notion for graded encoding schemes introduced by Pass et al. [77] (or more precisely, its strengthening, as used in [18]) corresponds to Γ^*-s-$\mathsf{IND\text{-}PRE}$ secure schemes for a graded encoding schema.

Next, we recover the main result of [18] using a simpler proof – by showing that SIO is also equivalent to Γ^*-s-$\mathsf{IND\text{-}PRE}$ secure obfuscation. Our proof is simpler because Γ^*-s-$\mathsf{IND\text{-}PRE}$ secure obfuscation serves as the "right" intermediate notion between VGB obfuscation and SIO.

But perhaps more significantly, our version of this theorem is not a result about obfuscation, but a significantly more general result about the framework itself. We show that *for all primitives that can be expressed as agents*, Γ^*-s-$\mathsf{IND\text{-}PRE}$ security is equivalent to a simulation-based security notion as well as to a restricted indistinguishability definition (which, for the case of obfuscation, reduces to SIO).

The final component in our extension of the agents framework is a composition theorem. Given that our new security definition involves a computationally unbounded adversary in the ideal world, the original composition theorem in Section 2.2 breaks down. However, we present a new information-theoretic variant of the notion of reduction between schema, to reestablish a composition theorem. Specifically, we show that a *statistical reduction* from a schema $\mathbf{\Sigma}$ to another schema $\mathbf{\Sigma}^*$ can be combined with a Γ^*-s-$\mathsf{IND\text{-}PRE}$ secure scheme for $\mathbf{\Sigma}^*$, to obtain a Γ^*-s-$\mathsf{IND\text{-}PRE}$ secure scheme for $\mathbf{\Sigma}$.

An illustrative application of this composition theorem is to recover another result of [18] regarding the existence of VGB obfuscation for all NC^1 circuits.

Indeed, once cast in our framework, this result is natural and immediate: [76] gave (using a different terminology) a reduction from obfuscation of NC^1 circuits to graded encoding schemas, and [77, 18] put forth the assumption that there exists an Γ^*-s-IND-PRE secure scheme for the graded encoding schema. Under this assumption, our composition theorem immediately yields the result that VGB obfuscation exists for all NC^1 circuits.

4.1 Technical Overview

We outline the definitional aspects first, and then present a high-level sketch of the proof of our main theorem (IND-CON $\Leftrightarrow$ Γ^*-s-IND-PRE $\Leftrightarrow$ Γ^*-s-SIM), and the composition theorem.

4.1.1 Security Definitions

We extend the indistinguishability preservation notion naturally to consider *statistical hiding* in the ideal world. In s-IND-PRE security, a test in Γ is required to be hiding in the real world only if it is statistically hiding in the ideal world — i.e., hiding against computationally unbounded adversaries (who are still limited to making polynomial number of accesses to the agents uploaded by the test). Further, we introduce a sharper *quantitative notion* of s-IND-PRE security, which makes explicit the (polynomial) gap permitted between the extent of ideal world hiding and real world hiding.[2]

We also introduce a new test family denoted by Γ^*, which consists of computationally unbounded tests, which do not accept any messages from the adversary. Alternately, a test in Γ^* can be considered as sampling a collection of agents to upload, and a string of bits to communicate to the adversary (taking only a challenge bit as input in the experiments).

Combined, the above two elements fully define Γ^*-s-IND-PRE. Next, we

[2]In IND-PRE security it is only required that a negligible distinguishing probability in the ideal world translates to a negligible distinguishing probability in the real world. The security notion here is tighter in that it requires indistinguishability to be preserved up to a polynomial loss, even if the original distinguishing probability in the ideal world is not negligible.

turn our attention to giving two security definitions which are not of the indistinguishability-preserving genre. Firstly, s-SIM is a statistical simulation based security notion, which, on the face of it, is a stronger definition than s-IND-PRE. In s-SIM security, it is required that for every real world adversary Adv, there is an ideal world simulator $\mathcal{S}$, which has a similar distinguishing probability as Adv has in the real world experiment. To be a strong security guarantee, we require that the simulator cannot depend on the test (but it can depend on Adv). We instantiate s-SIM security against the test-family Γ^*. This generalizes the notion of VGB security for obfuscation.[3]

The other security definition we introduce, called IND-CON (for indistinguishability of concentrated distributions) generalizes the notion of SIO introduced by [18] for obfuscation, to all schemas. Here indistinguishability is required only against tests which upload agents from two distributions which are not only indistinguishable in the ideal world, but in fact "concentrated" — with high probability, the outcome of any query strategy[4] is already determined.

4.1.2 Equivalence of Security Notions

It is easy to see that Γ^*-s-SIM $\Rightarrow$ Γ^*-s-IND-PRE $\Rightarrow$ IND-CON.[5] Our main result is a proof that the reverse implications hold as well, and hence the three notions are identical.

Our proof could be seen as a simplification and significant generalization of

[3]Γ^*-s-SIM security for obfuscation is easily seen to imply VGB security, but unlike in VGB obfuscation, it allows obfuscation of multiple programs, and allows auxiliary information to be given to the adversary. However, the equivalence results we prove imply that even this apparently stronger notion is equivalent to SIO security.

[4]As opposed to the case of obfuscation, for general schemas, a query can typically depend on previous queries. For example, in a graded encoding schema, it may be the case that a "zero-test" can be performed only after a sequence of operations on encodings provided by the test. A query-strategy is a polynomially deep (but exponentially large) tree which fully specifies a (deterministic) choice of ideal world queries based on the outcomes of the previous queries, and potentially using the agents generated by those queries.

[5]In this chain, we may insert a weaker version of s-SIM, which allows the simulator to depend on the test as well as the adversary (but not on the challenge bit given to the test), between Γ^*-s-SIM and Γ^*-s-IND-PRE security. Since all these notions turn out to be the same, in this paper we avoid defining the weaker simulation. However, for more general test families, or without the requirement of statistical security, this notion of a simulation could be of independent interest.

the proof in [18] that SIO implies VGB obfuscation. We briefly overview the proof of [18] before explaining our version. There it is shown how to construct a computationally unbounded simulator which receives access to a single circuit computing a binary function, makes only polynomially many queries to the circuit, and learns a sufficiently accurate approximation of the circuit so that it can simulate it to the given adversary, provided that the obfuscation scheme is SIO secure. The simulator iteratively narrows down the set of possibilities for the circuit it is given access to, by making carefully chosen queries. Firstly, the simulator narrows down the possibilities to a set of circuits R such that a uniform distribution over R is a concentrated distribution (this is called the *concentration step* of the proof). However, the adversary may behave differently on certain circuits within this set; the computationally unbounded simulator can identify this subset D[6] To determine if the circuit is from D using a small number of queries, the simulator relies on SIO security: since the adversary can distinguish the obfuscation of each of the circuits in D from the obfuscation of a random circuit in R (with distinguishing advantage of the same sign), it follows that it can distinguish the obfuscation of a random circuit in D from a random circuit in R. Hence, by SIO security, it must be the case that the uniform distribution over D is not concentrated around the same majority outcome as R is (and possibly, not concentrated at all). This is exploited to argue that a small set of queries can be found to check if the circuit is in D or not (this is called the *majority-separation step*). If after making these queries, the simulator determines that the circuit is not in D, it can obfuscate a random cricuit from R and present it to the adversary. On the other hand, if it is in D, this allows the simulator to make significant progress, because as D is not concentrated, it must be a significantly small fraction of R. The simulator iterates the concentration and majority-separation steps alternately until it determines that the circuit is not in D. It can be argued that the number of iterations (and the number of queries within each iteration) is logarithmic in the size of the space of circuits being obfuscated.

In our proofs, the simulation is required only in showing that Γ^*-s-IND-PRE security implies Γ^*-s-SIM security. Here, the simulator can rely on the

[6]More precisely there are two parts of D, corresponding to positive and negative distinguishing advantage. For simplicity, here we assume that only one such part is non-empty.

"stronger" s-IND-PRE security guarantee, and obtain a "separating query" more directly, without relying on R being concentrated: indeed, if D is distinguishable from R in the real world, then s-IND-PRE security guarantees that there is a (small-depth) query strategy that separates the two. Performing this query strategy either allows D to be significantly shrunk, or allows R to be significantly shrunk (since otherwise, it will not be a sufficiently separating query strategy). If R shrinks, then D is redefined with respect to the new R (and may become as large as the new R). Iterating this procedure makes D empty, with the number of iterations being logarithmic in the size of the space of agents.

Roughly, the above argument corresponds to the majority-separation step in the proof of [18]. An analogue of the concentration step appears in the proof that IND-CON security implies Γ^*-s-IND-PRE security, described below.

A potentially difficult part in proving IND-PRE security in general is that it requires one to show that *every* ideal-hiding test is real-hiding, and it is not clear which tests are ideal-hiding. Our proof can in fact be viewed as a characterization of tests in Γ^* that are statistically ideal-hiding. A test in Γ^* can be identified with a pair of distributions $\mathcal{D}_0$ and $\mathcal{D}_1$, corresponding to the collection of agents (and auxiliary information) it generates when the challenge bit is 0 and 1 respectively. For a test to be ideal hiding, the outcome of any (polynomial depth) query-strategy must have essentially the same distribution for both $\mathcal{D}_0$ and $\mathcal{D}_1$, but the distributions are not concentrated (which requires the outcome of any query strategy to be essentially deterministic). We give a simple combinatorial lemma which shows that there is an efficient query strategy that breaks down any distribution $\mathcal{D}$ into concentrated distributions (plus a negligible mass on an unconcentrated distribution). The query strategy reveals which constituent concentrated distribution a collection of agents come from. Hence, if $\mathcal{D}_0$ and $\mathcal{D}_1$ are ideal-hiding, then both of them should have essentially the same distribution over concentrated distributions. Now, for each concentrated distribution, IND-CON security guarantees that the two distributions are real-hiding too.

4.1.3 Simplification and Generalization

We highlight two contributions of our result, given the prior work of [18]. Technically, it simplifies the proof by changing a nested iterative construction (used in the simulator), into two separate constructions, each with a simple iterative procedure. At a more conceptual level, apparently technical aspects in the proof of [18] – namely, the concentration step and the majority-separation step – are reflected in two separate concrete concepts (namely, $\mathsf{IND\text{-}CON} \Rightarrow \Gamma^*\text{-}s\text{-}\mathsf{IND\text{-}PRE}$ and $\Gamma^*\text{-}s\text{-}\mathsf{IND\text{-}PRE} \Rightarrow \Gamma^*\text{-}s\text{-}\mathsf{SIM}$).

But more importantly our result also ties these results to the new framework of cryptographic agents. While the development of the notions of VGB obfuscation and SIO were important contributions to our understanding of obfuscation, our result shows that their equivalence has more to do with certain structural properties of the security definition (captured in $\Gamma^*\text{-}s\text{-}\mathsf{IND\text{-}PRE}$ security) rather than obfuscation itself. Indeed, we show that the same security definition, applied to the graded encoding schema captures the independently developed notion of "semantic-security" for graded encoding [77].[7] More broadly, $\Gamma^*\text{-}s\text{-}\mathsf{IND\text{-}PRE}$ security can be used to model über assumptions for a variety of cryptographic encoding schemes (e.g., groups, groups with bi-linear pairings etc.). Our result shows that *in all these cases*, there is an equivalent simulation based security notion as well as a low-level security notion for concentrated distributions.

4.1.4 Composition Theorem

The composition theorem in Section 2.2 breaks down in the case of $s\text{-}\mathsf{IND\text{-}PRE}$ security, since it involves an ideal-world adversary who is computationally unbounded. However, if the reduction is a *statistical reduction* – i.e., $\mathbf{\Sigma}$ can be information-theoretically securely constructed based on $\mathbf{\Sigma}^*$– then we show that the composition theorem holds. Further, the composition theorem holds even if we restrict to the test family Γ^*.

A consequence of this composition theorem is that we can readily obtain the

[7]The original notion in [77] essentially corresponds to $s\text{-}\mathsf{IND\text{-}PRE}$ security for a test family which requires the tests to be efficient. Without this requirement, the security notion is termed strong-sampler semantic-security.

result that, if a strong-sampler semantically-secure graded encoding scheme exists, then there exists a VGB obfuscation scheme for NC^1 circuits. We point out that in obtaining this result, we do not rely on the $\mathsf{IND\text{-}CON}$ security definition at all. While [18] crucially used the notion of SIO for obtaining this result, the notion of Γ^*-s-$\mathsf{IND\text{-}PRE}$ is sufficient: the proof relies on the fact that Γ^*-s-$\mathsf{IND\text{-}PRE}$ is equivalent to VGB security for obfuscation and to strong-sampler semantic security for graded encoding schemes and on the composition theorem for Γ^*-s-$\mathsf{IND\text{-}PRE}$ (as well as the existence of a statistical reduction from obfuscation for NC^1 to graded encoding schemes).

4.2 Definitions

Recall the formal definition of cryptographic agents from Section 2.1. While the way various entities interact in the real and ideal worlds remains the same, we now allow adversaries to be unbounded in the ideal world—but restricted to making a polynomially bounded number of queries. This is captured by the following statistical hiding definition.

Definition 15 (Statistical ideal world hiding). *A* Test *is* η-s-hiding w.r.t. a schema $\mathbf{\Sigma}$ *if, for all unbounded users* User *who make at most* η *queries,*

$$|\Pr[\textsc{ideal}\langle\mathsf{Test}(0) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle = 1] - \Pr[\textsc{ideal}\langle\mathsf{Test}(1) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle = 1]| \leq \frac{1}{\eta},$$

where $\textsc{ideal}\langle\mathsf{Test}(b) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle$ is the output of User in an execution of the ideal system, when Test gets b as input.

The real world hiding definition is now parameterized by adversary's running time.

Definition 16 (Real world hiding.). *A* Test *is η-hiding w.r.t. $\mathcal{O}$ if for all adversaries* Adv *who run for at most η time,*

$$|\Pr[\textsc{real}\langle\mathsf{Test}(0) \mid \mathcal{O} \mid \mathsf{Adv}\rangle = 1] - \Pr[\textsc{real}\langle\mathsf{Test}(1) \mid \mathcal{O} \mid \mathsf{Adv}\rangle = 1]| \leq \frac{1}{\eta},$$

where $\textsc{real}\langle\mathsf{Test}(b) \mid \mathcal{O} \mid \mathsf{Adv}\rangle$ is the output of Adv in an execution of the real system, when Test gets b as input.

Γ^* **test family.** This family consists of computationally unbounded tests which do not accept any messages from the user/adversary. W.l.o.g, such a test is fully characterized by a distribution over $\vec{P} \in \{0,1\}^* \times \mathcal{P}^*_{\mathsf{test}}$.[8]

The first part of $\vec{P}$, which we denote as $\vec{P}_0 \in \{0,1\}^*$, is a message from test to the user/adversary; the remaining components of the vector $\vec{P}$ denote a (possibly empty) collection of agents from $\mathcal{P}_{\mathsf{test}}$.

For $\vec{P} \in \{0,1\}^\ell \times \mathcal{P}^i_{\mathsf{test}}$, we write $\mathcal{O}(\vec{P})$ to denote a random encoding of $\vec{P}$ which consists of $(\vec{P}_0, \mathcal{O}(\vec{P}_1), \cdots, \mathcal{O}(\vec{P}_i))$ (as well as the public-key MPK if $\mathcal{O}$ involves a set-up). We write $\mathsf{Adv}(\mathcal{O}(\vec{P}))$ to denote the random variable corresponding to the bit output by Adv when given a random sample of $\mathcal{O}(\vec{P})$.

Definition 17 (s-$\mathsf{IND\text{-}PRE}$ security)**.** *An admissible cryptographic agent scheme* $\Pi = (\mathcal{O}, \mathcal{E})$ *is said to be a* p-Γ^*-s-$\mathsf{IND\text{-}PRE}$-secure scheme *for a schema* $\mathbf{\Sigma}$ *if for all* $\mathsf{Test} \in \Gamma^*$ *and every polynomial* η*, if* Test *is* $p(\eta)$*-*s*-hiding w.r.t.* $\mathbf{\Sigma}$*, then it is* η*-hiding w.r.t.* $\mathcal{O}$*.*

If $\Pi = (\mathcal{O}, \mathcal{E})$ *is* p-Γ^*-s-$\mathsf{IND\text{-}PRE}$*-secure for some polynomial* p*, then we simply refer to it as* Γ^*-s-$\mathsf{IND\text{-}PRE}$-secure scheme*.*

We also define a simulation-based security notion in our agents framework.

Definition 18 (Simulation-based security)**.** *An admissible cryptographic agent scheme* $\Pi = (\mathcal{O}, \mathcal{E})$ *is said to be a* p-Γ^*-s-SIM-secure scheme *for a schema* $\mathbf{\Sigma}$ *if for all polynomials* ℓ, η *and any adversary* Adv *which runs in time at most* $\ell(\kappa)$*, there exists a computationally unbounded simulator* $\mathcal{S}$ *that makes at most* $p(\eta(\kappa), \ell(\kappa))$ *queries, such that for all* $\mathsf{Test} \in \Gamma^*$*,*

$$|\Pr[\text{IDEAL}\langle \mathsf{Test} \mid \mathbf{\Sigma} \mid \mathcal{S}\rangle = 1] - \Pr[\text{REAL}\langle \mathsf{Test} \mid \mathcal{O} \mid \mathsf{Adv}\rangle = 1]| \leq \frac{1}{\eta(\kappa)}.$$

A cryptographic agent scheme $\Pi = (\mathcal{O}, \mathcal{E})$ *is said to be a* Γ^*-s-SIM-secure scheme *if it is a* p-Γ^*-s-SIM*-secure scheme for some (bivariate) polynomial* p*.*

We remark that one can consider a weaker notion of simulation where $\mathcal{S}$ can depend on Test. As we shall see, for Γ^*, this weaker notion is no different from the notion defined above.

[8] In our results, we can typically assume an upperbound on the number of bits communicated by the test, since there will be a bound on the running time of an adversary that it interacts with.

4.2.1 Concentrated distributions

Recall that in the ideal world, User can make queries — i.e., requests to run sessions — to $\mathcal{B}[\mathbf{\Sigma}]$ and obtain the outcome of the session (and handles for the updated configurations of the agents involved in the session). User can carry this out repeatedly, and adaptively. The following definition captures this procedure (for a deterministic User).

Definition 19 (Query Strategy)**.** *A d-query-strategy is a tree of depth at most d where each internal node u is labeled with a query Q_v and each outgoing edge from u is labeled with a different possible outcome of Q_u. The* execution *of a query strategy on a collection of agents $\vec{P}$ is a path in this tree starting from the root node, such that an edge from node u, labeled with an answer* ans*, is present in the path if and only if $\vec{P}(Q_u) = \mathsf{ans}$. The outcome of the entire execution, denoted by $\vec{P}(Q)$ is the (concatenated) outcomes of all the queries in the path.*

Let $\mathcal{P}$ denote the family of agents. Also, for a query strategy Q, and $\vec{P} \in \{0,1\}^\ell \times \bigcup_{i=0}^{\ell} \mathcal{P}^i$, let $\vec{P}(Q)$ denote the outcome of executing the query strategy Q on $\vec{P}$. We use the convention that the first query in Q is an empty query and its answer is the auxiliary information $\vec{P}_0 \in \{0,1\}^*$.

Definition 20 (Concentrated distributions)**.** *A distribution ensemble $\mathcal{D}$ over $\mathcal{P}^\ell$ is said to be η-concentrated if for all κ there exists a function A (called an answer function) which maps query strategies to answers, such that for all depth $\eta(\kappa)$ query strategy Q,*

$$\Pr_{\vec{P} \leftarrow \mathcal{D}(\kappa)}[\vec{P}(Q) \neq A(Q)] \leq \frac{1}{\eta(\kappa)}.$$

A pair of distribution ensembles $(\mathcal{D}_0, \mathcal{D}_1)$ is said to be η-concentrated if they are both η-concentrated with the same answer function.

Definition 21 (Indistinguishability of concentrated distributions.)**.** *An admissible scheme $\Pi = (\mathcal{O}, \mathcal{E})$ is q-$\mathsf{IND\text{-}CON}$ secure for $\mathbf{\Sigma} = (\mathcal{P}_{\mathsf{auth}}, \mathcal{P}_{\mathsf{user}})$ if for every polynomial η, and any pair of distribution ensembles $(\mathcal{D}_0, \mathcal{D}_1)$ over $\bigcup_{i=0}^{\eta(\kappa)} \mathcal{P}_{\mathsf{test}}^{\eta(\kappa)}$ which are $q(\eta)$-concentrated, we have that for any* PPT *adversary*

Adv *with running time at most* $\eta(\kappa)$*,*

$$\left|\mathsf{Pr}_{\vec{P}\leftarrow\mathcal{D}_0(\kappa)}[\mathsf{Adv}(\mathcal{O}(\vec{P}))=1]-\mathsf{Pr}_{\vec{P}\leftarrow\mathcal{D}_1(\kappa)}[\mathsf{Adv}(\mathcal{O}(\vec{P}))]=1\right|\leq\frac{1}{\eta(\kappa)}.$$

A scheme $\Pi=(\mathcal{O},\mathcal{E})$ *is* IND-CON *secure if it is* q-IND-CON *secure for some polynomial* q*.*

A probability lemma. The following is a simple lemma which can be used to relate two distributions with a small statistical difference to a single common distribution; further, the lemma allows the common distribution to avoid a subset S of the sample space, provided the given distributions have low mass on it. Below, $\Delta(\cdot,\cdot)$ denotes the statistical difference between two distributions.

Lemma 7. *For any two probability distributions* $\mathcal{A}_0$*,* $\mathcal{A}_1$ *over the same sample space, and any subset* S *of the sample space, there exists* $\epsilon\leq\Delta(\mathcal{A}_0,\mathcal{A}_1)+\min\{\Pr_{a\leftarrow\mathcal{A}_0}[a\in S],\Pr_{a\leftarrow\mathcal{A}_1}[a\in S]\}$*, a distribution* $\mathcal{A}_{\overline{S}}$ *over* $\overline{S}$*, and two distributions* $\mathcal{A}'_0,\mathcal{A}'_1$ *such that for each* $b\in\{0,1\}$*,* $\mathcal{A}_b$ *is equal to the distribution of* a *in the following experiment:*

$$\alpha\sim\text{Bernoulli}(\epsilon);\ \textit{if }\alpha=0,a\leftarrow\mathcal{A}_{\overline{S}},\ \textit{else }a\leftarrow\mathcal{A}'_b.$$

4.3 Equivalence of Definitions

In this section we prove our main results (Theorem 8 and Theorem 9).

Theorem 8 (Equivalence of IND-CON and IND-PRE)**.** *A cryptographic agent scheme* $\Pi=(\mathcal{O},\mathcal{E})$ *is a* Γ^*-s-IND-PRE*-secure scheme for a schema* $\mathbf{\Sigma}$ *if and only if it is* IND-CON *secure for* $\mathbf{\Sigma}$*.*

To prove Theorem 8, or more specifically, that IND-CON $\Rightarrow$ Γ^*-s-IND-PRE, we rely on the following lemma, which gives a query strategy that can be used to narrow down a distribution over agents to a concentrated distribution (except with negligible probability over the choice of the agents). As sketched in Section 4.1.2, this lemma gives a characterization of hiding tests in terms of concentrated distributions and is at the heart of proving Theorem 8.

Below, for a distribution $\mathcal{D}$ over agent vectors and a query strategy Q, $\mathcal{D}|_{Q\to\mathsf{ans}}$ denotes the distribution obtained by restricting $\mathcal{D}$ to the subset $\{\vec{P}|\vec{P}(Q) = \mathsf{ans}\}$.

Lemma 8. *Let $\mathcal{P}_{\mathsf{test}}$ be a set of agents with polynomially long representation. Then, for any polynomial ρ, there exists a polynomial π such that for any polynomial η, any function $\varepsilon > 0$, and any distribution $\mathcal{D}$ over $\mathcal{R}^\eta = \{0,1\}^\eta \times \mathcal{P}_{\mathsf{test}}^\eta$, there is a $\pi(\eta \cdot \log \frac{1}{\varepsilon})$-query strategy Q such that*

$$\Pr_{\vec{P}\leftarrow\mathcal{D}}[\mathcal{D}|_{Q\to\vec{P}(Q)} \text{ not } \rho(\eta)\text{-concentrated}] \leq \varepsilon.$$

Proof. The query strategy can be defined as repeatedly, conditioned on the previous queries and answers, identifying and carrying out a query strategy whose answer is not concentrated (i.e., no one answer has probability more than $1 - \rho(\eta)$) until the remaining distribution is $\rho(\eta)$-concentrated, or the budget on the number of queries (depth of the strategy) has been exhausted. We shall show that this leads to the mass in unconcentrated leaves of the query strategy tree to be at most ε.

More formally, consider a tree in which each node v is associated with a subset $R_v \subseteq \mathcal{R}^\eta$ and (unless it is a leaf node) with a query strategy Q_v of depth at most $\sigma := \rho(\eta)$. The set at the root is the entire set $\mathcal{R}^\eta$. For $R \subseteq \mathcal{R}^\eta$, let $\mathcal{D}|_R$ denote the distribution $\mathcal{D}$ restricted to the set R. A node v is a leaf node either if the distribution $\mathcal{D}|_{R_v}$ is σ-concentrated or if v is at a depth σ. For every internal node v, Q_v is a query strategy of depth at most σ such that for all ans, $\Pr_{\vec{P}\leftarrow\mathcal{D}|_{R_v}}[\vec{P}(Q_v) = \mathsf{ans}] \leq 1 - \frac{1}{\sigma}$. Note that such a Q_v exists since $\mathcal{D}|_{R_v}$ is not σ-concentrated (v being an internal node). For each possible answer ans to Q_v, v has a child v_{ans} such that $R_{v_{\mathsf{ans}}} = \{\vec{P} \in R_v \mid \vec{P}(Q_v) = \mathsf{ans}\}$.

Let L_ℓ be the set of all nodes at depth ℓ. Note that for each $v \in L_\ell$, $|R_v| \geq 1$, whereas $\sum_{v\in L_\ell} |R_v| \leq |\mathcal{R}^\eta|$. Therefore, $|L_\ell| \leq |\mathcal{R}^\eta|$. On the other hand, note that if u is a child of v in this tree, then $\Pr_{\vec{P}\leftarrow\mathcal{D}}[\vec{P} \in R_u \mid \vec{P} \in R_v] \leq 1 - \frac{1}{\sigma}$. Thus for all $v \in L_\ell$, $\Pr_{\vec{P}\leftarrow\mathcal{D}}[\vec{P} \in R_v] \leq (1 - \frac{1}{\sigma})^\ell$. Hence, $\Pr_{\vec{P}\leftarrow\mathcal{D}}[\vec{P} \in \bigcup_{v\in L_\ell} R_v] \leq (1 - \frac{1}{\sigma})^\ell \cdot |\mathcal{R}^\eta|$.

We can choose $\ell = \Omega(\sigma \cdot \log(|\mathcal{R}^\eta|/\varepsilon))$ so that $\Pr_{\vec{P}\leftarrow\mathcal{D}}[\vec{P} \in \bigcup_{v\in L_\ell} R_v] \leq \varepsilon$. Finally, note that $|\mathcal{R}^\eta| = \zeta^\eta$ for some polynomial ζ (determined by the size of $\mathcal{P}_{\mathsf{test}}$) and $\sigma = \mathrm{poly}(\eta)$, so that ℓ is polynomial in $\eta \cdot \log \frac{1}{\varepsilon}$. □

We prove the two directions of Theorem 8 separately. Intuitively, $\mathsf{IND\text{-}CON}$ security is a "weaker" notion, and hence the first direction below is easier to see. The second direction relies on Lemma 8.

$\underline{\Gamma^*\text{-}s\text{-}\mathsf{IND\text{-}PRE} \Rightarrow \mathsf{IND\text{-}CON}}$: Suppose that for some polynomial q, $\Pi = (\mathcal{O}, \mathcal{E})$ is a q-Γ^*-s-$\mathsf{IND\text{-}PRE}$ secure scheme for a schema $\mathbf{\Sigma}$. We shall show that Π is q-$\mathsf{IND\text{-}CON}$ secure for $\mathbf{\Sigma}$.

Let η be a polynomial, and $(\mathcal{D}_0, \mathcal{D}_1)$ be a pair of distribution ensembles which are $q(\eta)$-concentrated. Let A denote the answer function that maps depth $q(\eta)$ query strategies to answers, so that for any such query strategy Q, for both $b \in \{0,1\}$, we have $\Pr_{\vec{P} \leftarrow \mathcal{D}_b}[\vec{P}(Q) \neq A(Q)] \leq \frac{1}{q(\eta)}$.

Consider the test Test which on input $b \in \{0,1\}$, uploads a sample from the distribution $\mathcal{D}_b$. Observe that $\mathsf{Test} \in \Gamma^*$. Consider any unbounded ideal-world user User that makes at most $q(\eta)$ queries. For each setting of the random-tape of User, its behavior can be identified with a query strategy of depth at most $q(\eta)$. For any such strategy Q, irrespective of the bit b, with probability at least $1 - 1/q(\eta)$ User receives the answer $A(Q)$. Thus, for any User which makes at most $q(\eta)$ queries $|\Pr[\textsc{ideal}\langle \mathsf{Test}(0) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle = 1] - \Pr[\textsc{ideal}\langle \mathsf{Test}(1) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle = 1]| \leq 1/q(\eta)$. That is, Test is $q(\eta)$-s-hiding w.r.t. $\mathbf{\Sigma}$.

Then, since Π is a q-Γ^*-s-$\mathsf{IND\text{-}PRE}$ secure scheme for $\mathbf{\Sigma}$, we have that Test is η-hiding w.r.t. $\mathcal{O}$. That is, for any adversary Adv with running time at most η, $|\Pr[\textsc{real}\langle \mathsf{Test}(0) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle = 1] - \Pr[\textsc{real}\langle \mathsf{Test}(1) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle = 1]| \leq 1/\eta$. But $\Pr[\textsc{real}\langle \mathsf{Test}(b) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle = 1]$ is simply $\Pr_{\vec{P} \leftarrow \mathcal{D}_b}[\mathsf{Adv}(\mathcal{O}(\vec{P})) = 1]$.

Hence, by the definition of $\mathsf{IND\text{-}CON}$ security, Π is q-$\mathsf{IND\text{-}CON}$ secure for $\mathbf{\Sigma}$.

$\underline{\mathsf{IND\text{-}CON} \Rightarrow \Gamma^*\text{-}s\text{-}\mathsf{IND\text{-}PRE}}$: Suppose Π is an $\mathsf{IND\text{-}CON}$ secure scheme for $\mathbf{\Sigma}$. Then, there is a polynomial q such that it is q-$\mathsf{IND\text{-}CON}$ secure. We shall show that Π is p-Γ^*-s-$\mathsf{IND\text{-}PRE}$ secure, for some polynomial p.

Let Test be an arbitrary test in Γ^*, that is η^*-hiding w.r.t. $\mathbf{\Sigma}$. We shall show that Test is η-hiding w.r.t. Π, where $\eta^* = p(\eta)$ (for a polynomial p to be determined).

We consider the space $\mathcal{R}^\eta$ of all possible agents vector produced by tests, i.e., $\mathcal{R}^\eta = \{0,1\}^\eta \times \mathcal{P}^\eta_{\mathsf{test}}$.[9] Let $\mathcal{D}_0$ and $\mathcal{D}_1$ be the distributions over $\mathcal{R}^\eta$, produced by Test on input $b = 0$ and $b = 1$ respectively. Now, we apply Lemma 8 to the distribution $\mathcal{D}_0$, with η as above, $\rho(\eta) := 2q(\eta/2)$, and (say) $\varepsilon = 2^{-\eta}$. Let Q be the query startegy guaranteed by the lemma. Also, let $\mu = \rho(\eta)/2$.

Let $B = \{\mathsf{ans} \mid \mathcal{D}_0|_{Q\to\mathsf{ans}}$ is not 2μ-concentrated$\}$. Also let $C = \{\mathsf{ans} \mid \mathcal{D}_0|_{Q\to\mathsf{ans}}$ is 2μ-concentrated around some answer function A, but $\mathcal{D}_1|_{Q\to\mathsf{ans}}$ is not μ-concentrated around $A\}$. Then, for $\mathsf{ans} \notin B\cup C$, the pair of distributions $(\mathcal{D}_0|_{Q\to\mathsf{ans}}, \mathcal{D}_1|_{Q\to\mathsf{ans}})$ is μ-concentrated.

We argue, relying on the fact that Test is η^*-hiding, that the mass of $B \cup C$ under $\mathcal{D}_0$ is $O(\mu/\eta^*)$. Firstly, mass of B under $\mathcal{D}_0$ is bounded by Lemma 8 to at most ε. Next, for each $\mathsf{ans} \in C$, let A_{ans} be the answer function that $\mathcal{D}_0|_{Q\to\mathsf{ans}}$ is 2μ-concentrated around. Since $\mathcal{D}_1|_{Q\to\mathsf{ans}}$ is not μ-concentrated around A, there is some query strategy Q_{ans} with depth at most μ, such that $\Pr_{\vec{P}\leftarrow\mathcal{D}_1|_{Q\to\mathsf{ans}}}[\vec{P}(Q_{\mathsf{ans}}) \neq A_{\mathsf{ans}}(Q_{\mathsf{ans}})] > 1 - 1/\mu$. But since Q' has depth less than 2μ, $\Pr_{\vec{P}\leftarrow\mathcal{D}_0|_{Q\to\mathsf{ans}}}[\vec{P}(Q_{\mathsf{ans}}) \neq A_{\mathsf{ans}}(Q_{\mathsf{ans}})] \leq 1 - 1/(2\mu)$. Hence, there is a query strategy Q' (obtained by extending each leaf of Q with answer $\mathsf{ans} \in C$ with the query strategy Q_{ans}) of depth at most $\pi(\eta^2) + \mu$ (which we shall arrange to be less than η^*), such that

$$\begin{aligned}
\Pr_{\vec{P}\leftarrow\mathcal{D}_0}[\vec{P}(Q') = \mathsf{ans}||A_{\mathsf{ans}}(Q_{\mathsf{ans}}) \text{ for } \mathsf{ans} \in C] &\geq \Pr_{\vec{P}\leftarrow\mathcal{D}_0}[\vec{P}(Q) \in C] \cdot (1 - \frac{1}{2\mu}) \\
\Pr_{\vec{P}\leftarrow\mathcal{D}_1}[\vec{P}(Q') = \mathsf{ans}||A_{\mathsf{ans}}(Q_{\mathsf{ans}}) \text{ for } \mathsf{ans} \in C] &< \Pr_{\vec{P}\leftarrow\mathcal{D}_1}[\vec{P}(Q) \in C] \cdot (1 - \frac{1}{\mu}) \\
&\leq (\Pr_{\vec{P}\leftarrow\mathcal{D}_0}[\vec{P}(Q) \in C] + 1/\eta^*)\cdot \\
&\qquad (1 - \frac{1}{\mu})
\end{aligned}$$

The difference between these two probabilities is more than $\Pr_{\vec{P}\leftarrow\mathcal{D}_0}[\vec{P}(Q) \in C] \cdot \frac{1}{2\mu} - \frac{1}{\eta^*}$. But as the depth of Q' is less than η^*, and Test is η^*-hiding, this difference is upperbounded by $\frac{1}{\eta^*}$. Hence $\Pr_{\vec{P}\leftarrow\mathcal{D}_0}[\vec{P}(Q) \in C] \leq \frac{4\mu}{\eta^*}$.

Now, we view the test, on each input b, as sampling its agents vector $\vec{P}$ by first sampling the answer $\vec{P}(Q)$, and then sampling $\vec{P}$ conditioned on this

[9]Note that we truncate the auxiliary information to $\eta(\kappa)$ bits, and the number of agents uploaded by the test to $\eta(\kappa)$. This is because, to show that Test is η-hiding w.r.t. Π, it is enough to consider adversaries who read at most η bits of the messages from Test.

answer. $\vec{P}(Q)$ itself is sampled from the distribution $\mathcal{A}_b = \{\vec{P}(Q)\}_{\vec{P}\leftarrow\mathcal{D}_b}$. Now, we invoke Lemma 7 on the distributions $\mathcal{A}_0$ and $\mathcal{A}_1$ with the set $S = B \cup C$. This results in $\epsilon = O(\frac{\mu}{\eta^*})$, given the above bound (and since $\Delta(\mathcal{A}_0, \mathcal{A}_1) \leq 1/\eta^*$). Thus, the test, with probability $1-\epsilon$ samples $\mathsf{ans} \notin B \cup C$ (from a distribution independent of b) and then samples $\vec{P} \leftarrow \mathcal{D}_b|_{Q\to\mathsf{ans}}$. (With the remaining ϵ probability, it samples $\vec{P}$ depending on b as appropriate.) Recall that, for $\mathsf{ans} \notin B \cup C$, we have that $(\mathcal{D}_0|_{Q\to\mathsf{ans}}, \mathcal{D}_1|_{Q\to\mathsf{ans}})$ is μ-concentrated, where $\mu = q(\eta/2)$. Hence we can apply the q-$\mathsf{IND\text{-}CON}$ security to conclude that no adversary can distnguish between $b = 0$ and $b = 1$ in the real experiment with advantage more than $\epsilon + (1-\epsilon)\eta/2$. We shall set $\epsilon < \eta/2$ so that this advantage is less than η, as we need to prove. This requirement and the above bounds on η^* can be satisfied by setting, say, $\eta^* = \pi(\eta^2) + q(\eta)$. Thus, we choose p such that $p(\eta) > \pi(\eta^2) + q(\eta)$.

Theorem 9 (Equivalence of $\mathsf{IND\text{-}PRE}$ and SIM)**.** *A cryptographic agent scheme $\Pi = (\mathcal{O}, \mathcal{E})$ is a Γ^*-s-$\mathsf{IND\text{-}PRE}$-secure scheme for a schema $\mathbf{\Sigma}$ if and only if it is Γ^*-s-SIM-secure for the same schema.*

Proof. Intuitively, Γ^*-s-$\mathsf{IND\text{-}PRE}$ security is "weaker" than Γ^*-s-SIM security, and hence the first direction below is easier to see.

$\underline{\Gamma^*\text{-}s\text{-}\mathsf{SIM} \Rightarrow \Gamma^*\text{-}s\text{-}\mathsf{IND\text{-}PRE}:}$

Suppose $\mathbf{\Sigma} = (\mathcal{O}, \mathcal{E})$ is a p-Γ^*-s-SIM secure scheme for $\mathbf{\Sigma}$, for some (bivariate) polynomial p. We shall show that $\mathbf{\Sigma}$ is a q-Γ^*-s-$\mathsf{IND\text{-}PRE}$ schema for a polynomial q to be determined.

For a $\mathsf{Test} \in \Gamma^*$ and η, suppose there exists a PPT adversary Adv which runs in at most η time but can distinguish between Test with bit 0 and 1 with probability at least $1/\eta$. That is,

$$\begin{aligned}&|\Pr[\textsc{real}\langle\mathsf{Test}(0) \mid \mathcal{O} \mid \mathsf{Adv}\rangle = 1]\\ &\qquad\qquad - \Pr[\textsc{real}\langle\mathsf{Test}(1) \mid \mathcal{O} \mid \mathsf{Adv}\rangle = 1]| > 1/\eta. \quad (4.1)\end{aligned}$$

We need to show that there is an ideal world user User, which makes at most $q(\eta)$ queries and achieves a distinguishing advantage of at least $1/q(\eta)$.

Since Π is p-Γ^*-s-SIM secure, given Adv which runs in time at most η, there exists an unbounded simulator $\mathcal{S}$ making at most $p(3\eta, \eta)$ queries, such that for all tests (and in particular, for Test) and $b \in \{0,1\}$:

$$|\Pr[\text{IDEAL}\langle \mathsf{Test}(b) \mid \mathbf{\Sigma} \mid \mathcal{S}\rangle = 1] - \Pr[\text{REAL}\langle \mathsf{Test}(b) \mid \mathcal{O} \mid \mathsf{Adv}\rangle = 1]| \leq \frac{1}{3\eta}. \quad (4.2)$$

And therefore,

$$|\Pr[\text{IDEAL}\langle \mathsf{Test}(0) \mid \mathbf{\Sigma} \mid \mathcal{S}\rangle = 1] - \Pr[\text{IDEAL}\langle \mathsf{Test}(1) \mid \mathbf{\Sigma} \mid \mathcal{S}\rangle = 1]| > \frac{1}{\eta} - \frac{2}{4\eta} = \frac{1}{2\eta}.$$

We set q such that $q(\eta) \geq p(3\eta, \eta)$ and $\frac{1}{3\eta} \geq \frac{1}{q(\eta)}$. For instance, we can set $q(x) = p(3x, x) + 3x$.

Note that in the above proof, we could allow $\mathcal{S}$ to depend on Test, and therefore, even the weaker notion of simulation mentioned after Definition 18 implies IND-PRE security.

Γ^*-s-SIM $\Rightarrow$ Γ^*-s-IND-PRE: Suppose $\Pi = (\mathcal{O}, \mathcal{E})$ is q-Γ^*-s-IND-PRE secure for a schema $\mathbf{\Sigma}$. Fix a PPT adversary Adv whose running time is upperbounded by a polynomial ℓ, and a polynomial η. We shall construct a simulator $\mathcal{S}$ for Adv in the ideal world, which makes at most $p(\eta, \ell)$ queries and suffers a simulation error of at most $1/\eta$. W.l.o.g, we may assume that $\eta \geq \ell$ (if not, we set η to be ℓ below).

In the ideal world, when a test $\mathsf{Test} \in \Gamma^*$ uploads $\vec{P}^* \in \{0,1\}^* \times \mathcal{P}^*_{\mathsf{test}}$, $\mathcal{S}$ attempts to learn a sufficiently accurate approximation $\vec{P}^\dagger$ using a polynomial depth query strategy, and then faithfully simulate $\mathcal{O}(\vec{P}^\dagger)$ to Adv. Note that since Adv's running time is upperbounded by the polynomial ℓ, w.l.o.g, the simulator considers $\vec{P}$ to be in $\{0,1\}^\ell \times \mathcal{P}^{\ell'}_{\mathsf{test}}$, where ℓ' is the lesser of ℓ and the actual number of agents uploaded by Test.

$\mathcal{S}$ defines $R_i \subseteq \{0,1\}^\ell \times \mathcal{P}^{\ell'}_{\mathsf{test}}$ and $D_i \subseteq R_i$ inductively as follows, for integers $i \geq 0$, up till $i = i^*$ such that $D_{i^*} = \emptyset$. It then samples $\vec{P}^\dagger \leftarrow R_{i^*}$, and completes the simulation. We shall argue that this is a good simulation and that it can be carried out with at most $p(\ell)$ queries.

Below, we write $\mathsf{Adv}(\mathcal{O}(R_i))$ to denote the random variable corresponding to the output of Adv when a random $\vec{P} \leftarrow R_i$ is encoded using $\mathcal{O}$ and given to Adv; also, recall that $\mathsf{Adv}(\mathcal{O}(\vec{P}))$ denotes the similar random variable when the fixed agent vector $\vec{P}$ is encoded and given to Adv.

1. Firstly, for each i, we define D_i^* in terms of R_i, as follows. D_i^* is the larger of the two sets $D_{i,0}^*$ and $D_{i,1}^*$, where $D_{i,b}^*$ is given by

   $$\left\{ \vec{P} \in D_i^* \mid (-1)^b(\Pr[\mathsf{Adv}(\mathcal{O}(\vec{P})) = 1] - \Pr[\mathsf{Adv}(\mathcal{O}(R_i)) = 1]) > \frac{1}{\eta}. \right\}$$

   We shall maintain the invariant that $D_i \subseteq D_i^*$, for all $i \geq 0$.

2. $R_0 = \{0,1\}^\ell \times \mathcal{P}_{\mathsf{test}}^{\ell'}$, and $D_0 = D_0^*$.

3. If $D_i \neq \emptyset$, we define R_{i+1} and D_{i+1} as follows. Consider the test $\mathsf{Test}_i \in \Gamma^*$, which on input $b = 0$ uploads $\vec{P} \leftarrow D_i$, and on input $b = 1$, uploads $\vec{P} \leftarrow R_i$.[10] Since D_i is not empty, and $D_i \subseteq D_{i,\beta}^*$ for some $\beta \in \{0,1\}$, we have

   $$\begin{aligned} &|\Pr[\textsc{real}\langle \mathsf{Test}_i(0) \mid \mathcal{O} \mid \mathsf{Adv}\rangle = 1] - \Pr[\textsc{real}\langle \mathsf{Test}_i(1) \mid \mathcal{O} \mid \mathsf{Adv}\rangle = 1]| \\ &\quad = (-1)^\beta \frac{1}{|D_i|} \sum_{\vec{P} \in D_i} (\Pr[\mathsf{Adv}(\mathcal{O}(\vec{P})) = 1] - \Pr[\mathsf{Adv}(\mathcal{O}(R_i)) = 1]) > \frac{1}{\eta} \end{aligned}$$

   since for each $\vec{P} \in D_i \subseteq D_i^*$, we have $(-1)^\beta(\Pr[\mathsf{Adv}(\mathcal{O}(\vec{P})) = 1] - \Pr[\mathsf{Adv}(\mathcal{O}(R_i)) = 1]) > \frac{1}{\eta}$.

   That is, Test_i is not η-hiding (against Adv, which runs for less than $\ell \leq \eta$ time). Since the scheme $\Pi = (\mathcal{O}, \mathcal{E})$ is Γ^*-s-IND-PRE-secure, there must exist an ideal world adversary, or equivalently, a query strategy Q_i of depth at most $q(\eta)$ which has advantage of more than $\sigma := 1/q(\eta)$ in distinguishing $\mathsf{Test}_i(0)$ and $\mathsf{Test}_i(1)$.

   $\mathcal{S}$ executes the query strategy Q_i to obtain an answer ans_i. It defines $R_i' = \{\vec{P} \in R_i \mid Q_i(\vec{P}) = \mathsf{ans}_i\}$, and $D_i' = \{\vec{P} \in D_i \mid Q_i(\vec{P}) = \mathsf{ans}_i\}$. If $|R_i'| \leq (1-\sigma)|R_i|$, then set $R_{i+1} = R_i'$ and $D_{i+1} = D_{i+1}^*$. Otherwise, set $R_{i+1} = R_i$, $D_{i+1} = D_i'$.

Note that if $|R_i'| > (1-\sigma)|R_i|$ then $|D_i'| \leq (1-\sigma)|D_i|$, because otherwise Q_i cannot distinguish Test_i with advantage σ (as, for $b = 0$ and $b = 1$, it

[10]Note that Test_i may be computationally inefficient. This is the only reason we are not able to prove analogous results for a test-family that is like Γ^* but restricted to PPT tests.

receives an answer other than ans_i with probability less than σ). Therefore, we make progress in each iteration: either $|R_{i+1}| \leq (1-\sigma)|R_i|$ (in which case $|D_{i+1}| \leq |R_{i+1}|$), or $|R_{i+1}| = |R_i|$ and $|D_{i+1}| \leq (1-\sigma)|D_i|$. Hence, for $i^* \leq \log^2_{1-\sigma} |R_0|$ we have $D_{i^*} = \emptyset$.

The total number of queries made by the simulator above is bounded by $q(\eta) \cdot \log^2_{1-\sigma} |R_0|$. Note that $\log_2 |R_0| \leq \ell + n_{\mathbf{\Sigma}} \cdot \ell$, where $n_{\mathbf{\Sigma}}$ is a (polynomial) upperbound on the number of bits required to represent an agent in the schema $\mathbf{\Sigma}$. Also, $\frac{1}{\log_2(1-\sigma)} = O(q(\eta))$, so that $\log^2_{1-\sigma} |R_0| = O((n_{\mathbf{\Sigma}} \cdot \ell \cdot q(\eta))^2)$. Hence, we can set $p(\eta, \ell)$ to be this polynomial. $\square$

4.3.1 Extensions: Limited Agent-Space and Resettable Tests

Firstly, in the above results we can use a test-family which is a subset of Γ^* as follows. Note that the tests in Γ^* may upload any number of agents and send messages of any length (i.e., we considered $\vec{P} \in \{0,1\}^* \times \mathcal{P}^*_{\mathsf{test}}$). But our proofs go through unchanged if we restrict to a subset of Γ^* which uses an arbitrary subset of $\vec{P} \in \{0,1\}^* \times \mathcal{P}^*_{\mathsf{test}}$. (In this case, $\mathsf{IND\text{-}CON}$ is suitably modified to use the same subset of agent vectors.) In particular, we may restrict to the test-family $\Gamma^*_1 \subseteq \Gamma^*$ which uploads a single agent.

Secondly, we consider the possibility of using a test-family that is larger than Γ^*. Above, the restriction to Γ^* was crucial in allowing the construction of a composite query strategy by grafting a query strategy onto the leaves of another query strategy. However, if the test allowed itself to be treated as an agent – i.e., allowing a User to access Test from any state in its history – then the above equivalences would carry over. Thus, we may define a test-familty Γ_{reset} consisting of tests which are allowed to accept messages from the user/adversary and react to them, but also allows the user/adversary to reset it to the beginning (without changing its random tape). Then the above proofs extend to show that $\mathsf{IND\text{-}CON} \Leftrightarrow \Gamma_{\mathsf{reset}}\text{-}s\text{-}\mathsf{IND\text{-}PRE} \Leftrightarrow \Gamma_{\mathsf{reset}}\text{-}s\text{-}\mathsf{SIM}$, for all schemas. Note that tests in Γ^* are effectively resettable and hence $\Gamma_{\mathsf{reset}} \supseteq \Gamma^*$. We defer a formal definition of Γ_{reset} to the final version.

4.4 Reductions and Compositions

We define a new *information-theoretic* notion of reduction between schemata which would allow for composition of Γ^*-s-IND-PRE secure schemes. When compared to Definition 8, the main difference is that we require the hybrid world to be secure against *unbounded* adversaries (who make a polynomial number of queries). Another difference is that the correctness requirement is w.r.t. all tests, not just polynomially bounded ones.

Definition 22 (Statistical Reduction)**.** *We say that a (hybrid) cryptographic agent scheme* $\Pi = (\mathcal{O}, \mathcal{E})$ statistically reduces $\mathbf{\Sigma}$ *to* $\mathbf{\Sigma}^*$ *with respect to* $\widetilde{\Gamma}$*, if there exists an* unbounded $\mathcal{S}$ *such that for all* unbounded User *(both of whom make a polynomial number of queries),*

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

If there exists a scheme that reduces $\mathbf{\Sigma}$ *to* $\mathbf{\Sigma}^*$*, then we say* $\mathbf{\Sigma}$ *reduces to* $\mathbf{\Sigma}^*$*. (Note that correctness is required for all* Test*, and not just in* $\widetilde{\Gamma}$*.)*

Proofs of the following two theorems closely follow the corresponding ones in Section 2.2.

Theorem 10 (Composition)**.** *For any two schemata,* $\mathbf{\Sigma}$ *and* $\mathbf{\Sigma}^*$*, if* $(\mathcal{O}, \mathcal{E})$ *reduces* $\mathbf{\Sigma}$ *to* $\mathbf{\Sigma}^*$ *with respect to* Γ^* *and* $(\mathcal{O}^*, \mathcal{E}^*)$ *is a* Γ^*-s-IND-PRE *secure scheme for* $\mathbf{\Sigma}^*$*, then* $(\mathcal{O} \circ \mathcal{O}^*, \mathcal{E}^* \circ \mathcal{E})$ *is a* Γ^*-s-IND-PRE *secure scheme for* $\mathbf{\Sigma}$.

Proof. 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 PPT User, we have

$$\begin{aligned}
\text{REAL}\langle \mathsf{Test} \mid \mathcal{O}' \mid \mathcal{E}' \circ \mathsf{User}\rangle &= \text{REAL}\langle \mathsf{Test}' \mid \mathcal{O}^* \mid \mathcal{E}^* \circ \mathsf{User}'\rangle \\
&\overset{(a)}{\approx} \text{IDEAL}\langle \mathsf{Test}' \mid \mathbf{\Sigma}^* \mid \mathsf{User}'\rangle \\
&= \text{IDEAL}\langle \mathsf{Test} \circ \mathcal{O} \mid \mathbf{\Sigma}^* \mid \mathcal{E} \circ \mathsf{User}\rangle \\
&\overset{(b)}{\approx} \text{IDEAL}\langle \mathsf{Test} \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle
\end{aligned}$$

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

It remains to prove that for all $\mathsf{Test} \in \Gamma^*$, if Test is hiding w.r.t. $\mathbf{\Sigma}$ then Test is hiding w.r.t. $\mathcal{O}'$. Note that if $\mathsf{Test} \in \Gamma^*$, then $\mathsf{Test}' \in \Gamma^*$ too.

Firstly, we argue that Test is hiding w.r.t. $\mathbf{\Sigma} \Rightarrow \mathsf{Test}'$ is hiding w.r.t. $\mathbf{\Sigma}^*$. Suppose Test' is not hiding w.r.t. $\mathbf{\Sigma}^*$. This implies that there is some (unbounded) User such that

$$\text{IDEAL}\langle \mathsf{Test}'(0) \mid \mathbf{\Sigma}^* \mid \mathsf{User}\rangle \not\approx \text{IDEAL}\langle \mathsf{Test}'(1) \mid \mathbf{\Sigma}^* \mid \mathsf{User}\rangle.$$

But, by the statistical security of the reduction $(\mathcal{O}, \mathcal{E})$ of $\mathbf{\Sigma}$ to $\mathbf{\Sigma}^*$,

$$\text{IDEAL}\langle \mathsf{Test}'(b) \mid \mathbf{\Sigma}^* \mid \mathsf{User}\rangle \approx \text{IDEAL}\langle \mathsf{Test}(b) \mid \mathbf{\Sigma} \mid \mathcal{S} \circ \mathsf{User}\rangle,$$

for $b = \{0, 1\}$. Then,

$$\text{IDEAL}\langle \mathsf{Test}(0) \mid \mathbf{\Sigma} \mid \mathcal{S} \circ \mathsf{User}\rangle \not\approx \text{IDEAL}\langle \mathsf{Test}(1) \mid \mathbf{\Sigma} \mid \mathcal{S} \circ \mathsf{User}\rangle,$$

showing that Test is not hiding w.r.t. $\mathbf{\Sigma}$. Thus we have,

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

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

Theorem 11 (Transitivity of Reduction)**.** *For any three schemata, $\mathbf{\Sigma}_1, \mathbf{\Sigma}_2, \mathbf{\Sigma}_3$, if $\mathbf{\Sigma}_1$ statistically reduces to $\mathbf{\Sigma}_2$ and $\mathbf{\Sigma}_2$ statistically reduces to $\mathbf{\Sigma}_3$, then $\mathbf{\Sigma}_1$ statistically reduces to $\mathbf{\Sigma}_3$.*

Proof. If $\Pi_1 = (\mathcal{O}_1, \mathcal{E}_1)$ and $\Pi_2 = (\mathcal{O}_2, \mathcal{E}_2)$ are schemes that carry out the statistical reduction of $\mathbf{\Sigma}_1$ to $\mathbf{\Sigma}_2$ and that of $\mathbf{\Sigma}_2$ to $\mathbf{\Sigma}_3$, respectively, we claim that the scheme $\Pi = (\mathcal{O}_1 \circ \mathcal{O}_2, \mathcal{E}_2 \circ \mathcal{E}_1)$ is a statistical reduction of $\mathbf{\Sigma}_1$ to $\mathbf{\Sigma}_3$. 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 $\mathcal{S}_2 \circ \mathcal{S}_1$. $\square$

4.5 Applications

In this section we briefly summarize how the above results can be instantiated to rederive the main results of [18]. We first define schemata for graded encoding schemes and obfuscation.

Graded Encoding Schema Following "set-based" graded encoding [18, 14, 76, 77], we define the graded encoding schema $\mathbf{\Sigma}_{GE} = (\emptyset, \mathcal{P}^{GE}_{\mathsf{user}})$, where $\mathcal{P}^{GE}_{\mathsf{user}}$ contains a single type of agent. The schema is specified by a ring $\mathcal{R}(+, \times)$ and a subset $\mathfrak{S}$ of $2^{[k]}$ for a level $k \in \mathbb{N}$. An agent $P^{\mathsf{Msg}} \in \mathcal{P}^{GE}_{\mathsf{user}}$ is specified by a pair (x, S) where $x \in \mathcal{R}$ and $S \in \mathfrak{S}$. Initially it also makes a copy of $(x_1, S_1) = (x, S)$ and stores (x_1, S_1) on its work-tape. When invoked without an input, it just sends (x_1, S_1) to the first program in the session. When invoked with an input *Oper* on its input tape, it first parses *Oper* as follows:

- *Oper* $= +$ (resp. $-$): It reads a message (x_2, S_2) from its incoming communication tape. If $S_1 = S_2$, it updates its work-tape with $(x_1 + x_2, S_1)$ (resp. $(x_1 - x_2, S_1)$); otherwise, it writes $\perp$ on its outgoing communication tape and enters a blocking state.

- *Oper* $= \times$: It reads a message (x_2, S_2) from its incoming communication tape. If $S_2 \in \mathfrak{S}$ and $S_1 \cap S_2 = \emptyset$, it updates its work-tape with $(x_1 \times x_2, S_1 \cup S_2)$; otherwise, it writes $\perp$ on its outgoing communication tape and enters a blocking state.

- *Oper* $=$ $\mathsf{Zero\text{-}Test}$: It first checks whether S_1 is the universe set $[k]$. If not, it writes $\perp$ on its outgoing communication tape and enters a blocking state. Otherwise, if $x = 0$ it writes 1; otherwise, 0.

Now, we state the following propositions which easily follow from the definitions. Below we refer to the test-family Γ^*_1 from Section 4.3.1.

Proposition 1. *For a function family $\mathcal{F}$, a Γ_1^*-s-SIM secure scheme for $\mathbf{\Sigma}_{\text{OBF}(\mathcal{F})}$ is a VGB obfuscation scheme for $\mathcal{F}$, and vice-versa.*

With the modification to IND-CON also to distributions over a single agent (circuit), we have the following proposition.

Proposition 2. *For a function family $\mathcal{F}$, an* IND-CON *secure scheme for $\mathbf{\Sigma}_{\text{OBF}(\mathcal{F})}$ is an SIO scheme for $\mathcal{F}$ and vice versa.*

The next proposition refers to the graded encoding schema.

Proposition 3. *A graded encoding scheme is a strong-sampler semantically secure graded encoding scheme if and only if it is a Γ^*-s-$\mathsf{IND\text{-}PRE}$ secure scheme for the schema $\mathbf{\Sigma}_{GE}$.*

Finally, following is an adaptation of the result in [76].

Proposition 4. *For any function family $\mathcal{F} \in \mathsf{NC}^1$, there exists a statistical reduction from $\mathbf{\Sigma}_{\text{OBF}(\mathcal{F})}$ to $\mathbf{\Sigma}_{GE}$.*

From the above propositions, and the theorems in the previous sections, we obtain the following corollaries. In particular, the first corollary follows from Theorem 8, Theorem 9 (as extended in Section 4.3.1), Proposition 1 and Proposition 2.

Corollary 12 ([18])**.** *An obfuscation scheme is a VGB obfuscation for a function family $\mathcal{F}$ if and only if it is an SIO for $\mathcal{F}$.*

Combining Proposition 4 with the composition theorem (Theorem 1), Theorem 9, Proposition 3, and Proposition 1, we obtain the following corollary.

Corollary 13 ([18])**.** *If there exists a strong-sampler sematically-secure graded encoding scheme, then there exists a VGB obfuscation scheme for any function family $\mathcal{F} \in \mathsf{NC}^1$.*

Chapter 5

Adversarial Objects

The cryptographic agents model is well suited to study a wide variety of interesting cryptographic tools like functional encryption, obfuscation, fully homomorphic encryption, etc. By instantiating the framework with different test families, and giving different capabilities to ideal world adversaries, one can get various levels of security for these tools, establish novel connections between them, and understand how complex they are with respect to each other. Despite the generality and power of the model, it does not capture attacks involving objects maliciously generated by an adversary, because there is no way a test can operate on them. Hence, non-malleable properties of fundamental primitives like public-key encryption, signatures, etc. cannot be studied in the current form of the framework.

Let us illustrate the problem with the example of public-key encryption (pke). A natural schema $\mathbf{\Sigma}_{\mathsf{pke}}$ for pke is straightforward: an agent $P_m \in \mathcal{P}_{\mathsf{user}}$ simply sends m to the first agent in the session; $\mathcal{P}_{\mathsf{auth}}$ has only one agent who reads a message $\tilde{m}$ from its incoming communication tape and writes the same on its output tape. One can easily show that semantic security for pke is equivalent to Δ_{det}-$\mathsf{IND\text{-}PRE}$ security for $\mathbf{\Sigma}_{\mathsf{pke}}$. Semantic security, however, provides only minimal guarantees; in particular, it does not protect against *malleability* attacks. Hence, a more appropriate requirement is cca (chosen ciphertext attack) security. Here, adversary is additionally given access to a decryption oracle, who can decrypt any ciphertext except the challenge. If no adversary is able to take advantage of this extra facility, then we know that it is not possible to create ciphertexts of related messages from a ciphertext of a given message.

Now, if we want to capture malleability attacks on public-key encryption in our framework, we would like a test who can receive ciphertexts from the adversary and check if they are related to the ciphertexts it sent. Our model certainly allows tests and adversaries to send arbitrary messages to each other, but unfortunately, tests do not have a direct way to decrypt ciphertexts. Specifically, the secret key is sitting with $\mathcal{O}_{\mathsf{auth}}$, and it can be released only if a test uploads the key agent. Even if it does so, the secret key goes to the adversary.

As a result, we need to extend the framework so that tests can run sessions of their own on the objects transferred by the adversary and the ones they create. Further, it should be possible to upload an object and get private access to it, i.e., unless test explicitly asks an object to be transferred, it won't be. With these changes in the framework, we can design a test such that indistinguishability preservation w.r.t. to it would imply cca security. Whenever this test receives an object from the adversary, it will run a session on that object with the secret key (which is never transferred), and transfer the output. When adversary sends two messages, one is chosen based on the bit b, and uploaded to create an object, which is then transferred. From now on, test would decrypt any object for the adversary as long as it is not the one it sent.

There is one problem with this approach though. The test described above operates on the objects received from the adversary, but there are no objects in the ideal world! One could derive a special test from the one above whose behavior is similar, except that it works with handles instead of objects. But in the ideal world, handles an adversary transfers may not have any meaning for the special test. Furthermore, working with a test and another derived from it could be cumbersome, and would require changing the way we define $\mathsf{IND\text{-}PRE}$ security. It would be nice to define a test which is oblivious to the mechanics of real/ideal world.

We thus allow adversaries to transfer objects to tests in much the same way as tests transfer to them. So in the ideal world, an adversary could ask the schema to transfer an agent it has created, and test would then get access to it via a local handle. Further, we require that in the real world, a test is run with its own $(\mathcal{O}, \mathcal{E})$ so that when an adversary transfers an object, $\mathcal{E}$ provides a handle to test. Hence, one can define a single test whose behavior is meaningful in both real and ideal worlds. Also note that both these worlds are very symmetric in terms of how various entities are placed, and how they communicate with each other.

The new framework we have defined seems adequate to study security in the presence of maliciously generated objects, but at least in the case of public-key encryption, we have only looked at ciphertext objects. Public and secret keys could be corrupt too, but we can't take that into account because they are not treated as objects in the current formulation. Specifically, the set-up process is run exactly once by $\mathcal{O}$ and in an honest manner. Achieving security in this restricted setting was already quite challenging for sophisticated primitives like functional encryption. But for a simpler object like public-key encryption we could ask for more security, and it would be quite interesting if we can achieve it.

We thus modify our framework further and remove all structural requirements from $\mathcal{O}$. Secret and public key objects can be created and exchanged like any other object. Both tests and adversaries can create such objects and if they like, they can transfer them to the other entity. There is no need to have separate agent families for tests and users; we can achieve the same effect by designing tests who generate a secret key object but don't transfer it to users. The description of the framework becomes much simpler as a result.

5.1 The New Model

Ideal World. The ideal system for a schema $\mathbf{\Sigma}$ consists of two parties Test and User and a fixed third party $\mathcal{B}[\mathbf{\Sigma}]$ (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'.

$\mathcal{B}[\mathbf{\Sigma}]$ maintains two lists of handles $\mathsf{List}_{\mathsf{Test}}$ and $\mathsf{List}_{\mathsf{User}}$, which contain the set of handles belonging to Test and User respectively. At the beginning of an execution, both the lists are empty, and new handles are always generated by a deterministic procedure.

While Test and User can arbitrarily talk to each other, the interaction with $\mathcal{B}[\mathbf{\Sigma}]$ can be summarized as follows:

- **Instantiating agents.** Let $\mathbf{\Sigma} = (\mathcal{P}, \mathcal{P}^\dagger)$ where we associate $\mathcal{P}$ with honest parties and $\mathcal{P}^\dagger$ with corrupt ones. Test and User can, at any point, choose an agent from $\mathcal{P}$ and send it to $\mathcal{B}[\boldsymbol{\Sigma}]$. More precisely, they can send a string str to $\mathcal{B}[\boldsymbol{\Sigma}]$, and $\mathcal{B}[\boldsymbol{\Sigma}]$ will instantiate an agent with the given string (along with its own security parameter) as the contents of the parameter tape, and all other tapes being empty. However, if User is corrupt, it can ask $\mathcal{B}[\boldsymbol{\Sigma}]$ to instantiate agents from $\mathcal{P}^\dagger$ too. In any case, whenever an agent is instantiated, $\mathcal{B}[\boldsymbol{\Sigma}]$ adds (h, str) to the list of the party who uploaded the string, where h is a new handle, and sends h to it.

- **Request for Session Execution.** At any point in time, Test or 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 $(i_1, \ldots, i_t)$ [1]. $\mathcal{B}[\boldsymbol{\Sigma}]$ executes the session to obtain a collection of outputs and updated configurations of agents. It generates new handles $\tilde{h}_1, \ldots, \tilde{h}_u$ corresponding to the updated configurations, adds them to the list of the party who requested for session execution, and returns the new handles along with the output of the session to it. (If an agent halts in a session, no new handle is given out for that agent).

- **Transferring agents.** The two modes of interacting with $\mathcal{B}[\boldsymbol{\Sigma}]$ described above can be considered *local*: when Test (resp. User) uploads an agent or requests for a session execution, only Test (resp. User) hears back from $\mathcal{B}[\boldsymbol{\Sigma}]$. The other party does not get any information in this process (unless they directly communicate with each other). In particular, the agents one party has created (either directly or via session executions) are not accessible to the other party. The new mode of interaction, *transferring agents*, described below fills exactly this gap.

  At any point in time, a party, say Test, can request $\mathcal{B}[\boldsymbol{\Sigma}]$ to provide User with access to an agent in its list by sending the handle h corresponding to that agent, along with a special command TRANSFER. Upon receiving this request, $\mathcal{B}[\boldsymbol{\Sigma}]$ adds a new handle h' to $\mathsf{List}_{\mathsf{User}}$ (with the same

[1]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 party can access a configuration of an agent any number of times, by using the same handle.

configuration as h), and sends h' to User. Analogously, we can describe what happens when User issues the TRANSFER command.

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

Definition 23. *We say that* Test *is* hiding w.r.t. $\mathbf{\Sigma}$ *if* $\forall$ *PPT party* User*,*

$$\text{IDEAL}\langle\mathsf{Test}(0) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle \approx \text{IDEAL}\langle\mathsf{Test}(1) \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle.$$

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

Discussion. The ideal world is very *symmetric* with respect to Test and User: both Test and User interact with $\mathcal{B}[\mathbf{\Sigma}]$ in the same way. (This is in contrast to the original formulation of Cryptographic Agents wherein only User can receive outputs from $\mathcal{B}[\mathbf{\Sigma}]$.) One crucial difference, however, is that while a corrupt User can upload agents from both $\mathcal{P}$ and $\mathcal{P}^\dagger$, Test can only upload agents from the former.

Real World. A *cryptographic scheme* (or simply scheme) Π is a stateless (possibly randomized) algorithm Impl with a repository Repo. The repository is a table where each entry consists of a unique handle and a cryptographic object (represented, for instance, as a binary string). At the start of an execution, Repo is empty. Whenever the scheme gets an input $((h_1, \dots h_t), (i_1, \dots, i_\ell))$ (either $t = 0$ for creating an object, or $t = \ell$ for a session execution), objects corresponding to $h_1, \dots, h_t$ are retrieved from Repo and given to Impl (along with $i_1, \dots, i_\ell$). Impl is executed to obtain two kinds of results: objects and outputs. The objects are added to Repo, with a new handle for each, and the new handles, along with the outputs, are returned.

The real world for a schema $\mathbf{\Sigma} = (\mathcal{P}, \mathcal{P}^\dagger)$ consists of two parties Test and User, each with their own copy of a scheme $\Pi = (\mathsf{Impl}, \mathsf{Repo})$, say $\Pi_{\mathsf{Test}} = (\mathsf{Impl}_{\mathsf{Test}}, \mathsf{Repo}_{\mathsf{Test}})$ and $\Pi_{\mathsf{User}} = (\mathsf{Impl}_{\mathsf{User}}, \mathsf{Repo}_{\mathsf{User}})$. Like the ideal world, Test and User can arbitrarily talk to each other, but exchange of objects takes place in a different way—by connecting $\mathsf{Repo}_{\mathsf{Test}}$ and $\mathsf{Repo}_{\mathsf{User}}$

with each other. If Test is interested in sending an object to User, it uploads the handle corresponding to that object along with a command TRANSFER. $\mathsf{Repo}_{\mathsf{Test}}$ retrieves the object and sends it to $\mathsf{Repo}_{\mathsf{User}}$, who stores it together with a new handle, and returns that handle to User. Analogously, we can describe how objects can be transferred from User to Test.

Note that we do not allow Test direct access to the cryptographic objects stored in its repository. In particular, it cannot send a handle to $\mathsf{Repo}_{\mathsf{Test}}$, and get the object corresponding to it in return. Also observe that if User is corrupt, which we denote by Adv, it may not run the scheme it is supposed to. It can run any arbitrary algorithm and send any object of its choice to $\mathsf{Repo}_{\mathsf{Test}}$.

We define the random variable $\textsc{real}\langle \mathsf{Test}(b) \mid \Pi \mid \mathsf{Adv}\rangle$ 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, $\textsc{Time}\langle \mathsf{Test} \mid \Pi \mid \mathsf{Adv}\rangle$ is the maximum number of steps taken by Test (with a random input), Π and Adv in total.

Definition 24. *We say that* Test *is* hiding w.r.t. Π *if* $\forall$ *PPT party* Adv*,*

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

When Π is understood, we may simply say that Test is real-hiding.

$\mathsf{IND\text{-}PRE}$ Security. We are ready to present the security definition of a cryptographic agent scheme $\Pi = (\mathsf{Impl}, \mathsf{Repo})$ implementing a schema $\mathbf{\Sigma}$. Below, the *honest real-world user*, corresponding to an ideal-world user User, is defined as the composite program $\Pi_{\mathsf{User}} \circ \mathsf{User}$. Let Γ_{ppt} denote the family of all PPT Test.

Definition 25. *A cryptographic agent scheme* $\Pi = (\mathsf{Impl}, \mathsf{Repo})$ *is said to be a* Γ-$\mathsf{IND\text{-}PRE}$-secure scheme *for a schema* $\mathbf{\Sigma} = (\mathcal{P}, \mathcal{P}^{\dagger})$ *if the following conditions hold.*

- *Correctness.* $\forall$ PPT User *and* $\forall$ $\mathsf{Test} \in \Gamma_{\mathsf{ppt}}$, $\textsc{ideal}\langle \mathsf{Test} \mid \mathbf{\Sigma} \mid \mathsf{User}\rangle \approx \textsc{real}\langle \mathsf{Test} \mid \Pi \mid \Pi_{\mathsf{User}} \circ \mathsf{User}\rangle$. *If equality holds,* Π *is said to have perfect correctness.*

- *Efficiency. There exists a polynomial* poly *such that,* $\forall$ PPT User*,* $\forall$

$\mathsf{Test} \in \Gamma_{\mathsf{ppt}}$,

$$\textsc{Time}\langle \mathsf{Test} \mid \Pi \mid \Pi_{\mathsf{User}} \circ \mathsf{User} \rangle \leq \mathrm{poly}(\textsc{Time}\langle \mathsf{Test} \mid \mathbf{\Sigma} \mid \mathsf{User} \rangle, \kappa).$$

- *Indistinguishability Preservation.* $\forall$ $\mathsf{Test} \in \Gamma$,

$$\mathsf{Test} \textit{ is hiding w.r.t. } \mathbf{\Sigma} \Rightarrow \mathsf{Test} \textit{ is hiding w.r.t. } \Pi.$$

When Γ *is the family* Γ_{ppt}*, we simply say that* Π *is an* IND-PRE*-secure scheme for* $\mathbf{\Sigma}$.

Note that the correctness and efficiency requirements are w.r.t. all PPT Test.

5.2 Public-key encryption

Public-key encryption is perhaps the most basic application of cryptography. We treat every component of an encryption scheme—secret-key, public-key, and ciphertexts—as objects that can be maliciously generated and distributed. Besides the usual commands like Encrypt:m and Decrypt that correspond to encryption and decryption, we have many others that capture the functionality we expect an encryption scheme to provide, but rarely describe it explicitly. For instance, commands of the form "is__?" check the type of an agent, and those of the form "Comp__" compare two agents to see if they are the *same*. We have a separate "PubGen" command to enable a clearer separation between secret-key and public-key. Invoking this command on a secret-key agent, transforms it into a public-key agent.

We now formally define a schema $\mathbf{\Sigma}_{\mathsf{pke}} = (\mathcal{P}_{\mathsf{pke}}, \mathcal{P}^{\dagger}_{\mathsf{pke}})$ for public-key encryption. The agent family $\mathcal{P}_{\mathsf{pke}}$ captures the functionality available to honest users, and $\mathcal{P}^{\dagger}_{\mathsf{pke}}$ provides some additional features for corrupt users. We would ideally want $\mathcal{P}^{\dagger}_{\mathsf{pke}}$ to be empty for best possible security, but it may be too strong a requirement to be satisfied.

An agent in the family $\mathcal{P}_{\mathsf{pke}}$ is initialized with $(\mathsf{Sec\text{-}Key}, \mathsf{tag})$ on the parameter tape by giving the command Init, for a $\mathsf{tag} \leftarrow_R \{0,1\}^{\kappa}$ chosen by $\mathcal{B}$. It behaves on various inputs as follows.

- "PubGen": If the parameter tape has $(\mathsf{Sec\text{-}Key}, \mathsf{tag})$, then set it to $(\mathsf{Pub\text{-}Key}, \mathsf{tag})$.

- "$\mathsf{Encrypt}$:m": If the parameter tape has $(\mathsf{Pub\text{-}Key}, \mathsf{tag})$, then sample $\mathsf{id\text{-}tag} \leftarrow_R \{0,1\}^\kappa$ and change it to $(\mathsf{Cptxt}, m, \mathsf{tag}, \mathsf{id\text{-}tag})$.

- "$\mathsf{Decrypt}$": If the parameter tape has $(\mathsf{Cptxt}, m, \mathsf{tag}, \mathsf{id\text{-}tag})$, then send tag to the other agent in the session. Else, if the parameter tape has $(\mathsf{Sec\text{-}Key}, \mathsf{tag})$ and the incoming communication tape has (tag^*, m) such that $\mathsf{tag} = \mathsf{tag}^*$, then write m on the output tape, else write $\perp$.

- "$\mathsf{IsSK?}$": If the parameter tape has $(\mathsf{Sec\text{-}Key}, \mathsf{tag})$, then write true on the output tape, otherwise write false.

- "$\mathsf{IsPK?}$": If the parameter tape has $(\mathsf{Pub\text{-}Key}, \mathsf{tag})$, then write true on the output tape, otherwise write false.

- "$\mathsf{IsCT?}$": If the parameter tape has $(\mathsf{Cptxt}, m, \mathsf{tag}, \mathsf{id\text{-}tag})$, then write true on the output tape, otherwise write false.

- "CompSK": If the parameter tape has $(\mathsf{Sec\text{-}Key}, \mathsf{tag})$, then check if the first agent in the session. If yes, then send $(\mathsf{Sec\text{-}Key}, \mathsf{tag})$ to the second agent. Else, if the incoming communication tape has tag^* such that $\mathsf{tag} = \mathsf{tag}^*$, then write true to the output tape, else write false.

- "CompPK": If the parameter tape has $(\mathsf{Pub\text{-}Key}, \mathsf{tag})$, then check if the first agent in the session. If yes, then send $(\mathsf{Pub\text{-}Key}, \mathsf{tag})$ to the second agent. Else, if the incoming communication tape has tag^* such that $\mathsf{tag} = \mathsf{tag}^*$, then write true to the output tape, else write false.

- "CompCT": If the parameter tape has $(\mathsf{Cptxt}, m, \mathsf{tag}, \mathsf{id\text{-}tag})$, then check if the first agent in the session. If yes, then send $(\mathsf{Cptxt}, \mathsf{tag}, \mathsf{id\text{-}tag})$ to the second agent. Else, if the incoming communication tape has $(\mathsf{tag}^*, \mathsf{id\text{-}tag}^*)$ such that $\mathsf{tag} = \mathsf{tag}^*$ and $\mathsf{id\text{-}tag} = \mathsf{id\text{-}tag}^*$, then write true to the output tape, else write false.

We leave open the problem of designing an $\mathsf{IND\text{-}PRE}$ secure scheme for $\mathbf{\Sigma}_{\mathsf{pke}}$ for a suitable test family. Even a Δ-$\mathsf{IND\text{-}PRE}$ scheme would imply strong properties like cca-security [89, 90] and key-anonymity [91], among others.

Chapter 6

Conclusion

In this work, we provided a general unifying framework to model various cryptographic primitives and their security notions, along with powerful reduction and composition theorems. Our framework easily captures seemingly disparate objects such as obfuscation, functional encryption, fully homomorphic encryption, property preserving encryption as well as idealized models such as the generic group model and the random oracle model.

Given that various cryptographic primitives can all be treated as objects of the same kind (schema), it is natural to compare them with each other. We have shown that obfuscation is complete (under standard computational assumptions), but completely leave open the question of *characterizing* complete schemata. We also raise the question of characterizing *trivial* schemata—those which can be reduced to the null schema, as well as characterizing *realizable* schemata—those which have (say) Γ_{ppt}-$\mathsf{IND\text{-}PRE}$-secure schemes.

We presented a hierarchy of security notions $\{\Delta^*\text{-}\mathsf{IND\text{-}PRE}, \Delta_{\mathsf{det}}\text{-}\mathsf{IND\text{-}PRE}\}$ $\leq \Delta\text{-}\mathsf{IND\text{-}PRE} \leq \mathsf{IND\text{-}PRE} \leq \mathsf{SIM}$ defined using various test families (or, in the case of SIM, as a reduction to the null-schema), but the relationships between these for any given schema are not fully understood. We leave it as an open problem to provide separations between these various notions of security for various schemata. For the case of functional encryption we provide a separation of Δ_{det}-$\mathsf{IND\text{-}PRE}$ from $\mathsf{IND\text{-}PRE}$. For obfuscation we conjecture that all the above notions are different from each other for some function family.

We gave reduction and composition theorems for Γ_{ppt} and Δ (which extends to Δ^* as well), and use them for showing the completeness of obfuscation and designing a number of schemes. Given the importance of such theorems in simplifying the design and analysis of schemes, it is interesting to explore alternate notions of reduction that give more flexibility, i.e., place less restrictions on the designer. There has been a recent boom of constructions based on different flavors of obfuscation (indistinguishability obfuscation in particular). Can the new notion of reduction help us view these constructions in the agents framework, and help drive further research?

We generalized the equivalence of unbounded simulation and strong indistinguishability for obfuscation to any arbitrary schema in our framework, by first giving analogous definitions SIM and IND-CON, respectively, and then showing that they are both equivalent to IND-PRE. What does this equivalence tell us about the security of other schemata like functional encryption, fully-homomorphic encryption, etc? We know several impossibility results for functional encryption when the simulator is bounded in the ideal world. But the case of unbounded power has not received much attention. Indeed, it is plausible that SIM secure FE schemes exist for several interesting function families. Our equivalence provides a potentially easier way to resolve this question, since one can work with the more approachable indistinguishability based definition instead.

Finally, we saw how extending our agents framework can help one cleanly capture and study a wide range of attacks. We gave a schema for public-key encryption, and leave open the problem of designing a secure scheme for it, which would at once imply many desirable properties.

We believe that with time the agents framework—owing to its generality, flexibility, and power—would emerge as the *right* one to study the security of various cryptographic primitives, both classic and modern.

References

[1] D. Boneh, A. Sahai, and B. Waters, "Functional encryption: Definitions and challenges," in *TCC 2011*, ser. LNCS, Y. Ishai, Ed., vol. 6597. Springer, Heidelberg, Mar. 2011, pp. 253–273.

[2] A. Sahai and B. R. Waters, "Fuzzy identity-based encryption," in *EUROCRYPT 2005*, ser. LNCS, R. Cramer, Ed., vol. 3494. Springer, Heidelberg, May 2005, pp. 457–473.

[3] E. Shen, E. Shi, and B. Waters, "Predicate privacy in encryption systems," in *TCC 2009*, ser. LNCS, O. Reingold, Ed., vol. 5444. Springer, Heidelberg, Mar. 2009, pp. 457–473.

[4] D. Boneh, A. Raghunathan, and G. Segev, "Function-private identity-based encryption: Hiding the function in functional encryption," in *CRYPTO 2013, Part II*, ser. LNCS, R. Canetti and J. A. Garay, Eds., vol. 8043. Springer, Heidelberg, Aug. 2013, pp. 461–478.

[5] S. Agrawal, S. Agrawal, S. Badrinarayanan, A. Kumarasubramanian, M. Prabhakaran, and A. Sahai, "On the practical security of inner product functional encryption," in *PKC 2015*, ser. LNCS, J. Katz, Ed., vol. 9020. Springer, Heidelberg, Mar. / Apr. 2015, pp. 777–798.

[6] A. Sahai and H. Seyalioglu, "Worry-free encryption: functional encryption with public keys," in *ACM CCS 10*, E. Al-Shaer, A. D. Keromytis, and V. Shmatikov, Eds. ACM Press, Oct. 2010, pp. 463–472.

[7] S. Gorbunov, V. Vaikuntanathan, and H. Wee, "Functional encryption with bounded collusions via multi-party computation," in *CRYPTO 2012*, ser. LNCS, R. Safavi-Naini and R. Canetti, Eds., vol. 7417. Springer, Heidelberg, Aug. 2012, pp. 162–179.

[8] S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich, "Reusable garbled circuits and succinct functional encryption," in *45th ACM STOC*, D. Boneh, T. Roughgarden, and J. Feigenbaum, Eds. ACM Press, June 2013, pp. 555–564.

[9] D. Boneh and M. K. Franklin, “Identity-based encryption from the Weil pairing,” in *CRYPTO 2001*, ser. LNCS, J. Kilian, Ed., vol. 2139. Springer, Heidelberg, Aug. 2001, pp. 213–229.

[10] A. O’Neill, “Definitional issues in functional encryption,” Cryptology ePrint Archive, Report 2010/556, 2010, http://eprint.iacr.org/2010/556.

[11] S. Agrawal, S. Gorbunov, V. Vaikuntanathan, and H. Wee, “Functional encryption: New perspectives and lower bounds,” in *CRYPTO 2013, Part II*, ser. LNCS, R. Canetti and J. A. Garay, Eds., vol. 8043. Springer, Heidelberg, Aug. 2013, pp. 500–518.

[12] C. Matt and U. Maurer, “A definitional framework for functional encryption,” Cryptology ePrint Archive, Report 2013/559, 2013, http://eprint.iacr.org/2013/559.

[13] M. Bellare and A. O’Neill, “Semantically-secure functional encryption: Possibility results, impossibility results and the quest for a general definition,” in *CANS 13*, ser. LNCS, M. Abdalla, C. Nita-Rotaru, and R. Dahab, Eds., vol. 8257. Springer, Heidelberg, Nov. 2013, pp. 218–234.

[14] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters, “Candidate indistinguishability obfuscation and functional encryption for all circuits,” in *54th FOCS*. IEEE Computer Society Press, Oct. 2013, pp. 40–49.

[15] O. Pandey and Y. Rouselakis, “Property preserving symmetric encryption,” in *EUROCRYPT 2012*, ser. LNCS, D. Pointcheval and T. Johansson, Eds., vol. 7237. Springer, Heidelberg, Apr. 2012, pp. 375–391.

[16] S. Hada, “Zero-knowledge and code obfuscation,” in *ASIACRYPT 2000*, ser. LNCS, T. Okamoto, Ed., vol. 1976. Springer, Heidelberg, Dec. 2000, pp. 443–457.

[17] B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang, “On the (im)possibility of obfuscating programs,” in *CRYPTO 2001*, ser. LNCS, J. Kilian, Ed., vol. 2139. Springer, Heidelberg, Aug. 2001, pp. 1–18.

[18] N. Bitansky, R. Canetti, Y. T. Kalai, and O. Paneth, “On virtual grey box obfuscation for general circuits,” in *CRYPTO 2014, Part II*, ser. LNCS, J. A. Garay and R. Gennaro, Eds., vol. 8617. Springer, Heidelberg, Aug. 2014, pp. 108–125.

[19] S. Micali, R. Pass, and A. Rosen, “Input-indistinguishable computation,” in *47th FOCS*. IEEE Computer Society Press, Oct. 2006, pp. 367–378.

[20] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *41st ACM STOC*, M. Mitzenmacher, Ed. ACM Press, May / June 2009, pp. 169–178.

[21] B. Lynn, M. Prabhakaran, and A. Sahai, "Positive results and techniques for obfuscation," in *EUROCRYPT 2004*, ser. LNCS, C. Cachin and J. Camenisch, Eds., vol. 3027. Springer, Heidelberg, May 2004, pp. 20–39.

[22] S. Goldwasser and G. N. Rothblum, "On best-possible obfuscation," in *TCC 2007*, ser. LNCS, S. P. Vadhan, Ed., vol. 4392. Springer, Heidelberg, Feb. 2007, pp. 194–213.

[23] D. Hofheinz, J. Malone-Lee, and M. Stam, "Obfuscation for cryptographic purposes," in *TCC 2007*, ser. LNCS, S. P. Vadhan, Ed., vol. 4392. Springer, Heidelberg, Feb. 2007, pp. 214–232.

[24] R. Canetti, "Towards realizing random oracles: Hash functions that hide all partial information," in *CRYPTO'97*, ser. LNCS, B. S. Kaliski Jr., Ed., vol. 1294. Springer, Heidelberg, Aug. 1997, pp. 455–469.

[25] R. Canetti, D. Micciancio, and O. Reingold, "Perfectly one-way probabilistic hash functions (preliminary version)," in *30th ACM STOC*. ACM Press, May 1998, pp. 131–140.

[26] H. Wee, "On obfuscating point functions," in *37th ACM STOC*, H. N. Gabow and R. Fagin, Eds. ACM Press, May 2005, pp. 523–532.

[27] R. Canetti, G. N. Rothblum, and M. Varia, "Obfuscation of hyperplane membership," in *TCC 2010*, ser. LNCS, D. Micciancio, Ed., vol. 5978. Springer, Heidelberg, Feb. 2010, pp. 72–89.

[28] S. Hohenberger, G. N. Rothblum, a. shelat, and V. Vaikuntanathan, "Securely obfuscating re-encryption," in *TCC 2007*, ser. LNCS, S. P. Vadhan, Ed., vol. 4392. Springer, Heidelberg, Feb. 2007, pp. 233–252.

[29] R. Canetti and R. R. Dakdouk, "Obfuscating point functions with multibit output," in *EUROCRYPT 2008*, ser. LNCS, N. P. Smart, Ed., vol. 4965. Springer, Heidelberg, Apr. 2008, pp. 489–508.

[30] S. Garg, C. Gentry, and S. Halevi, "Candidate multilinear maps from ideal lattices," in *EUROCRYPT 2013*, ser. LNCS, T. Johansson and P. Q. Nguyen, Eds., vol. 7881. Springer, Heidelberg, May 2013, pp. 1–17.

[31] Z. Brakerski and G. N. Rothblum, "Obfuscating conjunctions," in *CRYPTO 2013, Part II*, ser. LNCS, R. Canetti and J. A. Garay, Eds., vol. 8043. Springer, Heidelberg, Aug. 2013, pp. 416–434.

[32] Z. Brakerski and G. N. Rothblum, "Black-box obfuscation for d-CNFs," in *ITCS 2014*, M. Naor, Ed. ACM, Jan. 2014, pp. 235–250.

[33] Z. Brakerski and G. N. Rothblum, "Virtual black-box obfuscation for all circuits via generic graded encoding," in *TCC 2014*, ser. LNCS, Y. Lindell, Ed., vol. 8349. Springer, Heidelberg, Feb. 2014, pp. 1–25.

[34] 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/2013/500.

[35] P. Ananth, D. Boneh, S. Garg, A. Sahai, and M. Zhandry, "Differing-inputs obfuscation and applications," Cryptology ePrint Archive, Report 2013/689, 2013, http://eprint.iacr.org/2013/689.

[36] A. Sahai and B. Waters, "How to use indistinguishability obfuscation: deniable encryption, and more," in *46th ACM STOC*, D. B. Shmoys, Ed. ACM Press, May / June 2014, pp. 475–484.

[37] E. Boyle, K.-M. Chung, and R. Pass, "On extractability obfuscation," in *TCC 2014*, ser. LNCS, Y. Lindell, Ed., vol. 8349. Springer, Heidelberg, Feb. 2014, pp. 52–73.

[38] 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/2013/641.

[39] S. Garg, C. Gentry, S. Halevi, and D. Wichs, "On the implausibility of differing-inputs obfuscation and extractable witness encryption with auxiliary input," in *CRYPTO 2014, Part I*, ser. LNCS, J. A. Garay and R. Gennaro, Eds., vol. 8616. Springer, Heidelberg, Aug. 2014, pp. 518–535.

[40] N. Bitansky and V. Vaikuntanathan, "Indistinguishability obfuscation from functional encryption," Cryptology ePrint Archive, Report 2015/163, 2015, http://eprint.iacr.org/2015/163.

[41] P. Ananth and A. Jain, "Indistinguishability obfuscation from compact functional encryption," in *CRYPTO 2015, Part I*, ser. LNCS, R. Gennaro and M. J. B. Robshaw, Eds., vol. 9215. Springer, Heidelberg, Aug. 2015, pp. 308–326.

[42] P. Ananth, A. Jain, and A. Sahai, "Achieving compactness generically: Indistinguishability obfuscation from non-compact functional encryption," Cryptology ePrint Archive, Report 2015/730, 2015, http://eprint.iacr.org/2015/730.

[43] A. Shamir, "Identity-based cryptosystems and signature schemes," in *CRYPTO'84*, ser. LNCS, G. R. Blakley and D. Chaum, Eds., vol. 196. Springer, Heidelberg, Aug. 1984, pp. 47–53.

[44] C. Cocks, "An identity based encryption scheme based on quadratic residues," in *8th IMA International Conference on Cryptography and Coding*, ser. LNCS, B. Honary, Ed., vol. 2260. Springer, Heidelberg, Dec. 2001, pp. 360–363.

[45] X. Boyen and B. Waters, "Anonymous hierarchical identity-based encryption (without random oracles)," in *CRYPTO 2006*, ser. LNCS, C. Dwork, Ed., vol. 4117. Springer, Heidelberg, Aug. 2006, pp. 290–307.

[46] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," in *40th ACM STOC*, R. E. Ladner and C. Dwork, Eds. ACM Press, May 2008, pp. 197–206.

[47] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert, "Bonsai trees, or how to delegate a lattice basis," in *EUROCRYPT 2010*, ser. LNCS, H. Gilbert, Ed., vol. 6110. Springer, Heidelberg, May 2010, pp. 523–552.

[48] S. Agrawal, D. Boneh, and X. Boyen, "Efficient lattice (H)IBE in the standard model," in *EUROCRYPT 2010*, ser. LNCS, H. Gilbert, Ed., vol. 6110. Springer, Heidelberg, May 2010, pp. 553–572.

[49] S. Agrawal, D. Boneh, and X. Boyen, "Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE," in *CRYPTO 2010*, ser. LNCS, T. Rabin, Ed., vol. 6223. Springer, Heidelberg, Aug. 2010, pp. 98–115.

[50] D. Boneh and B. Waters, "Conjunctive, subset, and range queries on encrypted data," in *TCC 2007*, ser. LNCS, S. P. Vadhan, Ed., vol. 4392. Springer, Heidelberg, Feb. 2007, pp. 535–554.

[51] V. Goyal, O. Pandey, A. Sahai, and B. Waters, "Attribute-based encryption for fine-grained access control of encrypted data," in *ACM CCS 06*, A. Juels, R. N. Wright, and S. Vimercati, Eds. ACM Press, Oct. / Nov. 2006, available as Cryptology ePrint Archive Report 2006/309. pp. 89–98.

[52] J. Bethencourt, A. Sahai, and B. Waters, "Ciphertext-policy attribute-based encryption," in *2007 IEEE Symposium on Security and Privacy*. IEEE Computer Society Press, May 2007, pp. 321–334.

[53] A. B. Lewko, T. Okamoto, A. Sahai, K. Takashima, and B. Waters, "Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption," in *EUROCRYPT 2010*, ser. LNCS, H. Gilbert, Ed., vol. 6110. Springer, Heidelberg, May 2010, pp. 62–91.

[54] J. Katz, A. Sahai, and B. Waters, "Predicate encryption supporting disjunctions, polynomial equations, and inner products," in *EUROCRYPT 2008*, ser. LNCS, N. P. Smart, Ed., vol. 4965. Springer, Heidelberg, Apr. 2008, pp. 146–162.

[55] S. Agrawal, D. M. Freeman, and V. Vaikuntanathan, "Functional encryption for inner product predicates from learning with errors," in *ASIACRYPT 2011*, ser. LNCS, D. H. Lee and X. Wang, Eds., vol. 7073. Springer, Heidelberg, Dec. 2011, pp. 21–40.

[56] B. Waters, "Functional encryption for regular languages," in *CRYPTO 2012*, ser. LNCS, R. Safavi-Naini and R. Canetti, Eds., vol. 7417. Springer, Heidelberg, Aug. 2012, pp. 218–235.

[57] S. Gorbunov, V. Vaikuntanathan, and H. Wee, "Attribute-based encryption for circuits," in *45th ACM STOC*, D. Boneh, T. Roughgarden, and J. Feigenbaum, Eds. ACM Press, June 2013, pp. 545–554.

[58] S. Garg, C. Gentry, S. Halevi, A. Sahai, and B. Waters, "Attribute-based encryption for circuits from multilinear maps," in *CRYPTO 2013, Part II*, ser. LNCS, R. Canetti and J. A. Garay, Eds., vol. 8043. Springer, Heidelberg, Aug. 2013, pp. 479–499.

[59] S. Goldwasser, Y. T. Kalai, R. A. Popa, V. Vaikuntanathan, and N. Zeldovich, "How to run turing machines on encrypted data," in *CRYPTO 2013, Part II*, ser. LNCS, R. Canetti and J. A. Garay, Eds., vol. 8043. Springer, Heidelberg, Aug. 2013, pp. 536–553.

[60] M. Barbosa and P. Farshim, "On the semantic security of functional encryption schemes," in *PKC 2013*, ser. LNCS, K. Kurosawa and G. Hanaoka, Eds., vol. 7778. Springer, Heidelberg, Feb. / Mar. 2013, pp. 143–161.

[61] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, "Fully homomorphic encryption over the integers," in *EUROCRYPT 2010*, ser. LNCS, H. Gilbert, Ed., vol. 6110. Springer, Heidelberg, May 2010, pp. 24–43.

[62] Z. Brakerski and V. Vaikuntanathan, "Fully homomorphic encryption from ring-LWE and security for key dependent messages," in *CRYPTO 2011*, ser. LNCS, P. Rogaway, Ed., vol. 6841. Springer, Heidelberg, Aug. 2011, pp. 505–524.

[63] Z. Brakerski and V. Vaikuntanathan, "Efficient fully homomorphic encryption from (standard) LWE," in *52nd FOCS*, R. Ostrovsky, Ed. IEEE Computer Society Press, Oct. 2011, pp. 97–106.

[64] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, "(Leveled) fully homomorphic encryption without bootstrapping," in *ITCS 2012*, S. Goldwasser, Ed. ACM, Jan. 2012, pp. 309–325.

[65] Z. Brakerski, "Fully homomorphic encryption without modulus switching from classical GapSVP," in *CRYPTO 2012*, ser. LNCS, R. Safavi-Naini and R. Canetti, Eds., vol. 7417. Springer, Heidelberg, Aug. 2012, pp. 868–886.

[66] Z. Brakerski and V. Vaikuntanathan, "Lattice-based FHE as secure as PKE," in *ITCS 2014*, M. Naor, Ed. ACM, Jan. 2014, pp. 1–12.

[67] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," in *CRYPTO 2013, Part I*, ser. LNCS, R. Canetti and J. A. Garay, Eds., vol. 8042. Springer, Heidelberg, Aug. 2013, pp. 75–92.

[68] 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 *14th IMA International Conference on Cryptography and Coding*, ser. LNCS, M. Stam, Ed., vol. 8308. Springer, Heidelberg, Dec. 2013, pp. 65–84.

[69] S. Chatterjee and M. P. L. Das, "Property preserving symmetric encryption revisited," Cryptology ePrint Archive, Report 2013/830, 2013, http://eprint.iacr.org/2013/830.

[70] M. Bellare and P. Rogaway, "Random oracles are practical: A paradigm for designing efficient protocols," in *ACM CCS 93*, V. Ashby, Ed. ACM Press, Nov. 1993, pp. 62–73.

[71] V. Shoup, "Lower bounds for discrete logarithms and related problems," in *EUROCRYPT'97*, ser. LNCS, W. Fumy, Ed., vol. 1233. Springer, Heidelberg, May 1997, pp. 256–266.

[72] U. M. Maurer, "Abstract models of computation in cryptography (invited paper)," in *10th IMA International Conference on Cryptography and Coding*, ser. LNCS, N. P. Smart, Ed., vol. 3796. Springer, Heidelberg, Dec. 2005, pp. 1–12.

[73] A. Boldyreva, D. Cash, M. Fischlin, and B. Warinschi, "Foundations of non-malleable hash and one-way functions," in *ASIACRYPT 2009*, ser. LNCS, M. Matsui, Ed., vol. 5912. Springer, Heidelberg, Dec. 2009, pp. 524–541.

[74] M. Bellare, V. T. Hoang, and S. Keelveedhi, "Instantiating random oracles via UCEs," in *CRYPTO 2013, Part II*, ser. LNCS, R. Canetti and J. A. Garay, Eds., vol. 8043. Springer, Heidelberg, Aug. 2013, pp. 398–415.

[75] D. Boneh and A. Silverberg, “Applications of multilinear forms to cryptography,” Cryptology ePrint Archive, Report 2002/080, 2002, http://eprint.iacr.org/2002/080.

[76] B. Barak, S. Garg, Y. T. Kalai, O. Paneth, and A. Sahai, “Protecting obfuscation against algebraic attacks,” in *EUROCRYPT 2014*, ser. LNCS, P. Q. Nguyen and E. Oswald, Eds., vol. 8441. Springer, Heidelberg, May 2014, pp. 221–238.

[77] R. Pass, K. Seth, and S. Telang, “Indistinguishability obfuscation from semantically-secure multilinear encodings,” in *CRYPTO 2014, Part I*, ser. LNCS, J. A. Garay and R. Gennaro, Eds., vol. 8616. Springer, Heidelberg, Aug. 2014, pp. 500–517.

[78] 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*, ser. LNCS, P. Q. Nguyen and E. Oswald, Eds., vol. 8441. Springer, Heidelberg, May 2014, pp. 578–602.

[79] E. Boyle, S. Goldwasser, and I. Ivan, “Functional signatures and pseudorandom functions,” in *PKC 2014*, ser. LNCS, H. Krawczyk, Ed., vol. 8383. Springer, Heidelberg, Mar. 2014, pp. 501–519.

[80] D. Boneh, A. Raghunathan, and G. Segev, “Function-private subspace-membership encryption and its applications,” in *ASIACRYPT 2013, Part I*, ser. LNCS, K. Sako and P. Sarkar, Eds., vol. 8269. Springer, Heidelberg, Dec. 2013, pp. 255–275.

[81] C. Gentry, “A fully homomorphic encryption scheme,” Ph.D. dissertation, Stanford University, 2009, crypto.stanford.edu/craig.

[82] A. De Caro and V. Iovino, “On the power of rewinding simulators in functional encryption,” Cryptology ePrint Archive, Report 2013/752, 2013, http://eprint.iacr.org/2013/752.

[83] R. Canetti, O. Goldreich, and S. Halevi, “The random oracle methodology, revisited (preliminary version),” in *30th ACM STOC*. ACM Press, May 1998, pp. 209–218.

[84] M. Fischlin, “A note on security proofs in the generic model,” in *ASIACRYPT 2000*, ser. LNCS, T. Okamoto, Ed., vol. 1976. Springer, Heidelberg, Dec. 2000, pp. 458–469.

[85] J. B. Nielsen, “Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case,” in *CRYPTO 2002*, ser. LNCS, M. Yung, Ed., vol. 2442. Springer, Heidelberg, Aug. 2002, pp. 111–126.

[86] A. W. Dent, "Adapting the weaknesses of the random oracle model to the generic group model," in *ASIACRYPT 2002*, ser. LNCS, Y. Zheng, Ed., vol. 2501. Springer, Heidelberg, Dec. 2002, pp. 100–109.

[87] M. Bellare, A. Boldyreva, and A. Palacio, "An uninstantiable random-oracle-model scheme for a hybrid-encryption problem," in *EUROCRYPT 2004*, ser. LNCS, C. Cachin and J. Camenisch, Eds., vol. 3027. Springer, Heidelberg, May 2004, pp. 171–188.

[88] N. Bitansky and R. Canetti, "On strong simulation and composable point obfuscation," in *CRYPTO 2010*, ser. LNCS, T. Rabin, Ed., vol. 6223. Springer, Heidelberg, Aug. 2010, pp. 520–537.

[89] M. Naor and M. Yung, "Public-key cryptosystems provably secure against chosen ciphertext attacks," in *22nd ACM STOC*. ACM Press, May 1990, pp. 427–437.

[90] D. Dolev, C. Dwork, and M. Naor, "Non-malleable cryptography (extended abstract)," in *23rd ACM STOC*. ACM Press, May 1991, pp. 542–552.

[91] M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval, "Key-privacy in public-key encryption," in *ASIACRYPT 2001*, ser. LNCS, C. Boyd, Ed., vol. 2248. Springer, Heidelberg, Dec. 2001, pp. 566–582.