Innovations in Computer Science 2010

Cryptographic Complexity Classes and Computational Intractability Assumptions

Hemanta K. Maji[1] Manoj Prabhakaran[1] Mike Rosulek[2]
[1]Department of Computer Science, University of Illinois, Urbana-Champaign. Partially supported by NSF grant CNS 07-47027.
[2]Department of Computer Science, University of Montana
hmaji2@uiuc.edu mmp@uiuc.edu mikero@cs.umt.edu

Abstract:
Which computational intractability assumptions are inherent to cryptography? We present a broad framework to pose and investigate this question.

We first aim to understand the "cryptographic complexity" of various tasks, independent of any computational assumptions. In our framework the cryptographic tasks are modeled as multi-party computation functionalities. We consider a universally composable secure protocol for one task given access to another task as the most natural complexity *reduction* between the two tasks. Some of these cryptographic complexity reductions are unconditional, others are unconditionally impossible, but the vast majority appear to depend on computational assumptions; it is this relationship with computational assumptions that we study.

In our detailed investigation of large classes of 2-party functionalities, we find that every reduction we are able to classify turns out to be unconditionally true or false, or else *equivalent* to the existence of one-way functions (OWF) or of semi-honest (equivalently, standalone-secure) oblivious transfer protocols (sh-OT). This leads us to conjecture that there are only a small finite number of distinct computational assumptions that are inherent among the infinite number of different cryptographic reductions in our framework.

If indeed only a few computational intractability assumptions manifest in this framework, we propose that they are of an extraordinarily fundamental nature, since the framework contains a large variety of cryptographic tasks, and was formulated without regard to any of the prevalent computational intractability assumptions.

Keywords: cryptographic complexity; computational complexity; intractability assumptions; secure multi-party computation

1 Introduction

Are the intractability assumptions employed in modern cryptography a historical accident? Historically, computational intractability assumptions are often developed along with cryptographic schemes to give security assurances to those schemes. To better understand the role of intractability in cryptography, and to place the theory in a firmer footing, specific intractability assumptions are abstracted into "general assumptions" that attempt to capture fundamental cryptographic properties. A few of these assumptions — most notably the existence of one-way functions — have turned out to be truly fundamental, in that they are necessary and sufficient for the possibility of many basic cryptographic tasks [20].

Our work continues along this line, trying to identify intractability assumptions that are intrinsic to cryptographic tasks. But rather than considering only a few familiar tasks, we propose a framework that incorporates all the different multi-party computation functionalities, which embody various ways in which *controlled access to information* manifests.[1] Our work has two motivations: first, to understand the cryptographic content of the various functionalities themselves, and second, to understand — and potentially discover — fundamental intractability assumptions intrinsic to cryptographic tasks.

Complexity of Multiparty Computation Functionalities.

The seminal work of Goldreich, Micali and Wigderson [16] on secure multi-party computation (MPC) introduced an idealization of cryptographic tasks in

[1]Indeed, cryptography could be defined as the study of controlling access to information: here access control should be understood as involving a possibly complex combination of allowing only certain information to be *learned*, and allowing information to be *influenced* only in certain ways.

terms of a trusted party, or an *ideal functionality*. An ideal functionality is an arbitrary program (possibly stateful, possibly randomized), to be executed privately by an external entity that can be trusted by all parties. This provides an extremely versatile language for capturing a great variety of kinds of controlled access to information, by simply defining various behaviors for the trusted entity — i.e., various functionalities. Further, this idealization of the cryptographic task is orthogonal to computational complexity considerations (instead, the security definition simply demands that a protocol be indistinguishable from this idealization). The later, more refined treatments, like Canetti's Universal Composition framework [7] follow the same pattern.

To study the cryptographic content of these tasks, then, is to study the complexity of these ideal functionalities. Here, it is *not* the computational complexity of the ideal functionalities (time or space required by the ideal functionality) that is of interest. Rather, we will be interested in various qualitatively different types of functionalities. To develop a formal notion of complexity, we apply *reductions* that capture the relevant cryptographic aspects. The natural notion of reduction among functionalities (without involving any computational complexity aspects) is the following. A functionality $\mathcal{F}$ is said to reduce to $\mathcal{G}$ (denoted by $\mathcal{F} \sqsubseteq \mathcal{G}$), if there exists a protocol that securely realizes $\mathcal{F}$ using access to $\mathcal{G}$. Here, the stricter the definition of secure realization used, the tighter the notion of reduction will be.[2] We shall use the strong security definition of the Universal Composition framework [7], against active (malicious) adversaries, but in a static (non-adaptive) corruption model. We write $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ to specify that security need to hold only in a probabilistic polynomial time setting.

Seeking New Worlds in Impagliazzo's Multiverse.

Impagliazzo [19] considered various complexity scenarios in terms of intractability and the cryptographic landscape in each of these worlds. In our current work, we study the cryptographic landscape given by the relative complexities of multiparty functionalities as described above.

The first of Impagliazzo's worlds (called *Algorithmica*) corresponds to the scenario P = NP. For our purposes we shall consider a slightly different formulation: we define a world called *Informatica* where P = PSPACE.[3] Prior work [34] has drawn a fairly detailed map of the landscape of 2-party functionalities in *Informatica*, delineating various complexity classes and showing that many functionalities do not unconditionally reduce to others. Further [33] shows that moving on to *Minicrypt*, a few of the relative distinctions among functionalities disappear, and more importantly, in *Cryptomania* (for our purposes, a world in which semi-honest Oblivious Transfer protocols exist) *all* the distinctions among functionalities disappear, except for ones that must remain unconditionally [9, 36].

But these worlds in Impagliazzo's multiverse may not be the only cryptographically interesting ones. In particular, one can imagine an intermediate world "strictly" between Minicrypt and Cryptomania, where the map of complexity classes of functionalities may look different from that in either world. To discover or rule out such worlds, we need a formal framework for the space of relevant computational assumptions.

A Framework for Assumptions. For any pair of (finite memory) functionalities $\mathcal{F}$ and $\mathcal{G}$, we consider *the existence of a reduction* $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ as a complexity assumption itself. Importantly, this convention defines a (potentially infinite) space of assumptions, independent of specific constructions, but directly based on cryptographic goals or functionalities. By systematically exploring this space of assumptions we hope to uncover not only new interesting assumptions that could be "black-box separated" from currently used ones, but also identify interesting cryptographic properties that give rise to them.

For some pairs of functionalities $(\mathcal{F}, \mathcal{G})$, the "assumption" $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is unconditionally true, with the security of the reduction holding even in the statistical setting (denoted by $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$). Further, some of these assumptions are unconditionally false, no matter what computational assumption is made, due to the strong demands of the UC security definition; the pairs $(\mathcal{F}, \mathcal{G})$ for which this happens have an explicit characterization [9, 36]. But most of the assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ are unresolved, and these are the subject of our study.

In initiating this exploration, we pose several questions.

[2] Studying the landscape of functionalities using a strict reduction can be compared to zooming into a map to distinguish between different elements in the map; but if one zooms in too close – uses too strict a reduction – then virtually each element appears isolated.

[3] In cryptographic language, this is a world where the adversaries may be considered to have unbounded computational resources. In this world, the only kind of security possible for efficient protocols is information theoretic security. Hence the name Informatica.

1) Maximal and Minimal Assumptions.

The first question is whether there is a "maximal" or "minimal" one among these (unresolved) assumptions. That is, among these assumptions, is there:

- (a maximal assumption) an assumption $\mathcal{F}^* \sqsubseteq_{\text{PPT}} \mathcal{G}^*$ such that it implies $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ for all such pairs $(\mathcal{F}, \mathcal{G})$?
- (a minimal assumption) an assumption $\hat{\mathcal{F}} \sqsubseteq_{\text{PPT}} \hat{\mathcal{G}}$ such that $(\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G} \wedge \mathcal{F} \not\sqsubseteq_{\text{STAT}} \mathcal{G}) \implies \hat{\mathcal{F}} \sqsubseteq_{\text{PPT}} \hat{\mathcal{G}}$?

The first of these questions was recently answered in [33], where it was shown that indeed such pairs do exist. Interestingly, the maximal assumptions in this framework are *exactly equivalent* to the familiar assumption that there exists a stand-alone secure oblivious-transfer protocol (sh-OT assumption). In other words, the sh-OT assumption is a maximal assumption in our framework; there are no additional cryptographic worlds beyond Cryptomania in our framework.

However, the question of the minimal assumption remains open. We conjecture that a minimal assumption exists and that it in fact corresponds the existence of one-way functions. Some of the results below represent support for this conjecture. In particular we show that for several interesting pairs $(\mathcal{F}, \mathcal{G})$, $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is indeed exactly equivalent to the existence of one-way functions (OWF).

2) Intermediate Assumptions.

Assuming OWF assumption is indeed the minimal assumption of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, one can ask if there are "intermediate" assumptions between OWF assumption and sh-OT assumption. Here, by an assumption being intermediate, we mean that there exists a black-box separation in the sense of [15, 21], that separates it from both OWF assumption and sh-OT assumption.

Indeed, [15] shows that the existence of a secret communication protocol is an intermediate assumption. It is easy to see that their result extends to the UC secure reduction of a secret communication functionality to a public communication functionality. In our framework we find it convenient to work with functionalities which do not communicate with the adversary when all the parties are honest (called regular functionalities in [36]). So this particular reduction appears only when we consider multi-party functionalities with three or more parties. For the case of 2-party (finite-memory, regular) functionalities, we conjecture that all assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ are in fact equivalent to either OWF assumption or sh-OT assumption (or are unconditionally true or false). This conjecture is supported by all the results derived in this work.

Moving to 3-party functionalities already provides us one such intermediate assumption. But our understanding of multi-party functionalities with 3 or more parties is very limited. (Note that our map in Figure 1 is only for 2-party functionalities.) We consider it highly likely that several other intermediate assumptions can be discovered in our framework among functionalities with more than two parties.

Our Results.

Our main results can be interpreted as evidence that the worlds Minicrypt and Cryptomania (the latter appropriately interpreted) are indeed very special. We classify a substantial number of reductions $\mathcal{F} \sqsubseteq \mathcal{G}$ (for 2-party functionalities $\mathcal{F}$ and $\mathcal{G}$) and find that, surprisingly, every one that we are able to classify is *exactly equivalent* to either the existence of one-way functions, or the existence of a semi-honest protocol for oblivious transfer. Put another way, if we defined worlds in Impagliazzo's multiverse corresponding to the various complexity classes among 2-party functionalities changing their boundaries, then we seem to obtain only two worlds (in addition to Informatica), namely Minicrypt and Cryptomania.

The new technical results here are of the form that various reductions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ imply either sh-OT assumption or OWF assumption. This is complemented by our recent results [33] showing that either sh-OT assumption or OWF assumption imply $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ for all functionalities $\mathcal{F}$ and $\mathcal{G}$ in our framework. Together, these results establish the *equivalence* between assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ and either sh-OT assumption or OWF assumption.

Our results can be summarized as follows, and should be considered as support for our main conjectures:

• **Simultaneity can be extended only in Cryptomania.** Some 2-party functionalities, like evaluating a boolean XOR, provide no information hiding, but only enforce simultaneity or input independence.[4] Such a functionality can be used to securely realize another functionality with "less" simultaneity and no information hiding even in Informatica, using a simple protocol. But we show that such a functionality can be used to securely realize another functionality with "more" simultaneity, or another functionality that hides information, only in Cryptomania (i.e., if and only if there exists a semi-honest OT protocol). As a concrete example, it is possible to securely evaluate a boolean XOR on strings of two bits given ideal

[4]Here simultaneity only refers to the input independence — Alice should not be able to choose her input based on Bob's input, and vice versa — and not to any notion of fairness.

access to a single-bit XOR functionality *if and only if* there is a semi-honest OT protocol (or equivalently, if and only if a single-bit XOR functionality can be used to realize *every* other functionality). Here, the "if" clauses follow from [33].

• **Other reductions possible in Minicrypt.** Many interesting functionalities can be securely realized against passive adversaries in Informatica. We call them "passive trivial" functionalities. Among such functionalities, those which perform some information hiding (thus excluding the functionalities like XOR considered above) can be used to realize any other passive trivial functionality in Minicrypt. (This follows from results in [33].) We show that many of those reductions are in fact *equivalent* to the OWF assumption.

A more detailed overview of our results and conjectures is presented in Section 3.

Related Works.

Until recently, most work in secure multi-party computation focused on the extremes of complexity; namely, classifying the functionalities that are trivially realizable (using only a communication channel) and those which are complete. Such classifications have been found for a wide variety of security models (i.e., reduction strengths) and subclasses of functionalities [4, 6, 9–11, 16, 22, 23, 25, 26, 29, 31, 36].

Beimel et al. [5] address a similar question as here, relating cryptographic complexity of MPC functionalities and computational complexity assumptions. But they consider computational complexity assumptions only of the form that a functionality is standalone-trivial. Restricted to a special class of SFE functionalities, they show that there is only one such assumption (other than being unconditionally true), namely the sh-OT assumption.

Recently [34] demonstrated infinitely many distinct, intermediate levels of cryptographic complexity under computationally unbounded UC security reductions (or, in our current lexicon, reductions in Informatica). On the contrary, assuming sh-OT assumption, the state of affairs is totally different — [33] show that in this setting, every (deterministic, finite) functionality is either trivial or complete. In independent work, [14] show that under sh-OT assumption, the "common random string" functionality (or in our framework of finite functionalities, the coin-tossing functionality $\mathcal{F}_{\text{COIN}}$) is complete.

Impagliazzo and Luby [20] showed that several important cryptographic primitives are equivalent to the OWF assumption. Following the approach there, we also rely on the fact that a weaker primitive called distributionally one-way functions yield OWFs. In [12], Damgård and Groth showed that if $\mathcal{F}_{\text{COM}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{COIN}}$ then the sh-OT assumption holds. This is subsumed by our results regarding exchange-like functionalities (Theorem 1), but in fact contains one of the ideas that is used in (the simpler of) our constructions.

In considering the existence of OWF, key-agreement and sh-OT as distinct computational assumptions, we rely on the black-box separation results from [15, 21]. We refer to such separations in our *conjectures*, which predict more such distinct computational assumptions coming out of our framework. However, for the formal statement of our results (which merely show that various reductions are equivalent to one of these assumptions), such a separation is not essential.

2 Technical Preliminaries

We write $[k]$ to denote the set $\{1, \ldots, k\}$. We say that a function $\mu : \mathbb{N} \to \mathbb{R}$ is *negligible* if it approaches zero faster than any inverse polynomial. That is, if for all $c > 0$, $\mu(n) < n^{-c}$ for sufficiently large n. We say that a quantity ν is *noticeable* if $\nu(n) = \Omega(n^{-c})$ for some constant $c > 0$. We say that a probability p is *overwhelming* if $1 - p(n)$ is negligible.

2.1 Security Model

We use security in the Universal Composition (UC) framework of Canetti [7]. The framework follows the paradigm introduced by Goldreich, Micali and Wigderson [16] of defining security by comparing a *real world* (in which the parties execute a protocol) to an *ideal world* (in which the task is carried out by a trusted functionality). We assume the reader has a slight familiarity with the UC framework, and now present an overview of the conventions we use in this paper, emphasizing that very few specifics of the model are critical for our results.

We write $\mathcal{F} \sqsubseteq \mathcal{G}$ if there is a protocol that securely realizes $\mathcal{F}$ in the "$\mathcal{G}$-hybrid model;" see [7] for a formal definition. Informally, a protocol is secure if for every adversary attacking the protocol (in the real world), there is a simulator interacting with $\mathcal{F}$ (in the ideal world) that achieves an indistinguishable effect in all contexts (environments). In the $\mathcal{G}$-hybrid model, the parties in the protocol can interact with any number of (asynchronous) copies of $\mathcal{G}$, and can access $\mathcal{G}$ in any "role". The parties are also given free access to a communication channel.

We consider only efficient protocols, but make a notational distinction between unconditionally (statistically) secure protocols (denoted by $\sqsubseteq_{\text{STAT}}$) and protocols whose security may depend on a computational assumption (denoted by $\sqsubseteq_{\text{PPT}}$). As is standard, we

require security against active (i.e., malicious) adversaries. All results in this work are restricted to static corruption (where the adversary has to corrupt any parties before the protocol begins).

2.2 Functionalities

Although the UC framework allows a functionality to be an arbitrary interactive program, we consider in this work only *finite* functionalities — namely, functionalities that have finite memory and use finite input and output alphabets. (See Section 6 for a brief discussion.)

Many of our results do apply to arbitrary *reactive* functionalities, which take input and give output repeatedly throughout several rounds of interaction (see Appendix B for formal definitions of this class of functionalities). However, for clarity most of our results in the main body are stated for the class of symmetric SFE functionalities, defined below:

Definition 1. *A functionality is a* secure function evaluation (SFE) *functionality if it waits for inputs $x \in X$ from Alice and $y \in Y$ from Bob (for some finite sets X and Y) and then sends $f_A(x,y)$ to Alice and $f_B(x,y)$ to Bob. If either party is corrupt, the output for that party is delivered first; then if the adversary allows, the output to the other party is also delivered.*

We say that the functionality is symmetric SFE (SSFE) *if $f_A = f_B$.*

SSFE functionalities, starting with the original *Millionaire's Problem* proposed by Yao [41], are perhaps the most well-studied class of two-party functionalities. We can completely specify an SSFE by giving its *function table*, a matrix whose (i,j) entry contains $f_A(i,j) = f_B(i,j)$. Each row corresponds to a possible input for Alice, and each column corresponds to a possible input for Bob. For instance, the boolean XOR functionality may be written as $\left[\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right]$.

Note that if neither party is corrupt, the functionality does not interact with the adversary. (Such functionalities were called regular functionalities in [36].)

Definition 2. *We say that two SFE functionalities are* isomorphic *if one can be obtained from the other by repeatedly adding/removing redundant inputs, permuting a party's inputs, relabeling a party's outputs for one of its inputs, and reversing the roles of Alice and Bob.*

By redundant input, we mean (in the case of Alice) an input x such that $f_B(x,\cdot) = f_B(x',\cdot)$ for some x', and $(x', f_A(x',y))$ uniquely determines $f_A(x,y)$ for all y.

Several specific functionalities play important roles in our results:

- $\mathcal{F}_{\text{OT}}$: Standard 1-out-of-2 oblivious transfer (OT) functionality. Alice has inputs $x_0, x_1 \in \{0,1\}$ and Bob has input $b \in \{0,1\}$. Alice receives no output, and Bob receives output x_b.
- $\mathcal{F}_{\text{COM}}$: Standard bit-commitment functionality. Alice gives input (COMMIT, x), where $x \in \{0,1\}$, and Bob receives output COMMITTED. Later, when Alice sends input REVEAL, Bob receives input (REVEAL, x), where x was the input given in Alice's first COMMIT command.
- $\mathcal{F}_{\text{COIN}}$: Standard coin-tossing functionality. After receiving input from both parties, samples a random coin $r \leftarrow \{0,1\}$ and gives it to both parties.
- $\mathcal{F}_{\text{CC}}$: A simple "cut-and-choose" functionality. It is an SSFE functionality defined as $\left[\begin{smallmatrix} 0 & 2 \\ 1 & 2 \end{smallmatrix}\right]$. In $\mathcal{F}_{\text{CC}}$, Alice provides a bit, and Bob can choose whether or not to learn it. The output also reveals Bob's choice to Alice. $\mathcal{F}_{\text{CC}}$ is the simplest non-trivial functionality that hides some information about the inputs.
- $\mathcal{F}_{\text{EXCH}}^{i \times j}$: An *exchange function*: the SSFE functionality in which Alice gives input $x \in [i]$, Bob gives input $y \in [j]$, and both parties learn (x,y). The cryptographic non-triviality of an exchange function is that it enforces inputs to be chosen independently (though we make no requirement for fairness in learning the output). The special case $\mathcal{F}_{\text{EXCH}}^{2\times 2}$ is isomorphic to the boolean-XOR SSFE functionality.

2.3 Intractability Assumptions

We consider two important computational intractability assumptions in this work:

sh-OT assumption: There exists a protocol for $\mathcal{F}_{\text{OT}}$ secure against semi-honest, PPT adversaries. It is possible to express this assumption using the definition of UC security restricted to semi-honest adversaries (in both the real and the ideal executions). However, we point out that the traditional (standalone) security definition is equivalent to the UC security definition, since the simulation required by semi-honest security does not, and need not, *extract* the inputs of the corrupt players; it simply uses the input given by the environment.

The sh-OT assumption is known to imply the existence of one-way functions [17, 20]. As such, a protocol for OT secure against semi-honest adversaries implies such a protocol secure against malicious (standalone) adversaries, using the compiler of [16].

OWF assumption: This is the standard assumption that one-way functions exist. In [20], it is shown that the OWF assumption is implied by the much weaker

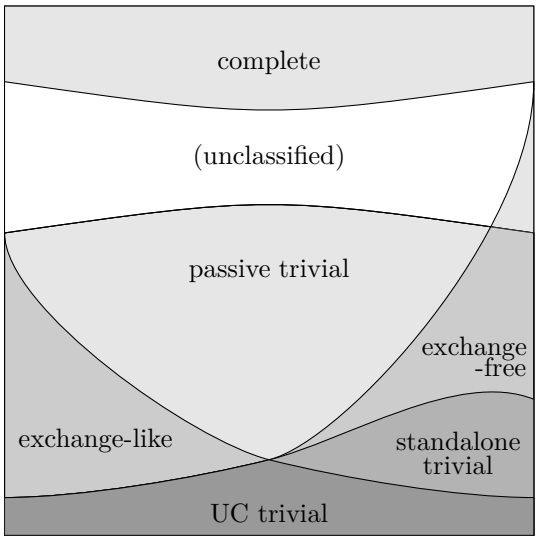

Figure 1: A map of various cryptographic complexity classes (of 2-party SSFE functionalities)

assumption that *distributionally one-way* functions exist. Thus if OWFs do not exist, then no function is distributionally one-way: for every efficient function f and polynomial p, there is an efficient algorithm that on input y samples close to uniformly (within $1/p$ statistical difference) from the set $f^{-1}(y)$.

2.4 Cryptographic Complexity Landscape Overview

In this section we provide some useful classifications of functionalities, many of which are natural "complexity classes" in our complexity-theoretic view of MPC functionalities. A visual overview is also given in Figure 1. For each class, we describe its relevance to our framework and give an overview of some of its important known properties.

The classes listed below, except the first two, are natural in our framework in the sense that they are "downward closed" with respect to $\sqsubseteq_{\text{STAT}}$. That is, if $\mathcal{G}$ is a functionality in a class and $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$ then $\mathcal{F}$ also falls in the same class. (The first one is "upward closed.") Additionally, all of these classes can be defined in terms of the $\sqsubseteq_{\text{STAT}}$ reduction, which is the most restrictive reduction in our framework.

Complete Functionalities. These are SSFE functionalities that are unconditionally "complete." That is, for all $\mathcal{G}$ in this class, and all functionalities $\mathcal{F}$, $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$. Based on the completeness of the oblivious transfer functionality [23] (which remains true with respect to the UC-secure reduction $\sqsubseteq_{\text{STAT}}$ as well [22, 24]), Kilian [25] gave a complete combinatorial characterization for these functionalities, as the evaluation of functions that contain a 2×2 minor of the form $\begin{bmatrix} a & b \\ b & b \end{bmatrix}$, where $a \neq b$. Later, Kraschewski and Müller-Quade [29] extended Kilian's characterization to the stronger $\sqsubseteq_{\text{STAT}}$ reduction.

Unclassified Functionalities. Among incomplete SSFE functionalities, we leave a set of functionalities as "unclassified." These are functionalities which are neither complete, nor passive-trivial (see next class). We know that this class is not empty: the "spiral" SSFE functionality $\begin{bmatrix} 1 & 1 & 2 \\ 4 & 0 & 2 \\ 4 & 3 & 3 \end{bmatrix}$ is known to fall into this category [4, 27, 31]. These functions do have a combinatorial characterization in terms of minors that immediately follows from the combinatorial characterizations of complete functionalities and passive-trivial functionalities (below) [4, 31]. However, we call these functionalities unclassified as very little is known about their cryptographic properties, or different sub-classes within the class.

Passive-Trivial Functionalities. These are functionalities securely realizable against a passive (a.k.a., honest-but-curious, or semi-honest) adversary in a computationally unbounded environment. For SSFE functionalities, such functions have an explicit combinatorial characterization, namely that they are evaluations of what are called "decomposable" functions:

Definition 3 (Decomposable [4, 31])**.** *An SSFE functionality $\mathcal{F} : X \times Y \to Z$ is* row decomposable *if there exists a partition $X = X_1 \cup \cdots \cup X_k$ ($X_i \neq \emptyset$), $k \geq 2$, such that the following hold for all $i \leq k$:*

- *for all $y \in Y$, $x \in X_i$, $x' \in (X \setminus X_i)$, we have $\mathcal{F}(x, y) \neq \mathcal{F}(x'y)$; and*
- *$\mathcal{F}\big|_{X_i \times Y}$ is either a constant function or column decomposable, where $\mathcal{F}\big|_{X_i \times Y}$ denotes the restriction of $\mathcal{F}$ to the domain $X_i \times Y$.*

We define being column decomposable *symmetrically with respect to X and Y. We say that $\mathcal{F}$ is simply* decomposable *if it is either constant, row decomposable, or column decomposable.*

Kushilevitz [31] and Beaver [4] independently proved that an SSFE functionality is decomposable if and only if it has a *perfectly secure* protocol. Later, this characterization was extended to the more natural case where the protocol is allowed to be only statistically secure [30, 34].

If $\mathcal{F}$ is decomposable, then a *canonical protocol* for $\mathcal{F}$ is a deterministic protocol defined inductively as follows [31]:

- If $\mathcal{F}$ is a constant function, both parties output the output value of $\mathcal{F}$, without interaction.
- If $\mathcal{F}$ is row decomposable as $X = X_1 \cup \cdots \cup X_k$, then party 1 announces the unique i such that its

input $x \in X_i$. Then both parties run a canonical protocol for $\mathcal{F}\big|_{X_i \times Y}$.
- If $\mathcal{F}$ is column decomposable as $Y = Y_1 \cup \cdots \cup Y_k$, then party 2 announces the unique i such that its input $y \in Y_i$. Then both parties run a canonical protocol for $\mathcal{F}\big|_{X \times Y_i}$.

It is a simple exercise to see that a canonical protocol is a perfectly secure protocol for $\mathcal{F}$ against passive adversaries.

An important example in this class (though not an SFE functionality) is the commitment functionality $\mathcal{F}_{\text{COM}}$. Interestingly, this class has a natural alternate definition in terms of the $\sqsubseteq_{\text{STAT}}$ reduction. If $\mathcal{F}$ is an SFE functionality, then $\mathcal{F}$ is passive-trivial if and only if $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{F}_{\text{COM}}$ [34].

Exchange-Like Functionalities. We introduce an important sub-class of passive-trivial functionalities called *exchange-like* functionalities. A simple way to define this class is the functionalities $\mathcal{F}$ such that $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{F}_{\text{EXCH}}^{i \times i}$ for some i. Intuitively, these are the functionalities that can be realized using simultaneity alone (recall that the only cryptographic non-triviality of an exchange function $\mathcal{F}_{\text{EXCH}}^{i \times i}$ is in its independence of inputs).

Among SSFE functionalities, exchange-like functionalities are exactly those that are isomorphic to $\mathcal{F}_{\text{EXCH}}^{i \times j}$ for some integers i and j. For reactive functionalities, a similar complete characterization can be made, and we do so in Appendix B.

Exchange-Free Functionalities. In contrast to exchange-like functionalities, we can define the class of *exchange-free* functionalities as those which contain no simultaneity. More formally, $\mathcal{G}$ is in this class if and only if $\mathcal{F}_{\text{EXCH}}^{2\times 2} \not\sqsubseteq_{\text{STAT}} \mathcal{G}$. All standalone-trivial functionalities (see the next class) are of this kind. But there are other functionalities too: for instance, $\begin{smallmatrix}1&1&2\\3&4&3\end{smallmatrix}$ (not standalone-trivial) and $\begin{smallmatrix}1&1&2\\4&0&2\\4&3&3\end{smallmatrix}$ (an unclassified functionality) can be shown to be exchange-free.

A useful combinatorial property of exchange-free functionalities is the following: Every exchange-free and passive-trivial functionality has a unique (up to simple renaming of the steps) decomposition (Definition 3) [34]. However, being uniquely decomposable does not necessarily mean that the SFE is exchange-free — consider the SSFE function $\begin{smallmatrix}1&1&2\\5&0&2\\4&3&3\end{smallmatrix}$.

Standalone-Trivial Functionalities. These are functionalities securely realizable against computationally unbounded adversaries in a stand-alone (i.e., isolated) environment. SSFE functionalities in this class were combinatorially characterized in [30, 34] as a strict subclass of the passive-trivial functionalities. Since $\mathcal{F}_{\text{EXCH}}^{2\times 2}$ is not in this class, every functionality in this class is exchange-free. The $\mathcal{F}_{\text{CC}}$ functionality defined earlier is an important example in this class, and is in fact the simplest non-trivial one.

Trivial Functionalities. These are the functionalities UC-securely realizable against a general adversary in a computationally unbounded environment, using protocols which only rely on (private) communication channels.[5] In the case of SSFE functionalities, these are the functionalities that have a decomposition of depth 1, or equivalently, the functionalities that are isomorphic to $\mathcal{F}_{\text{EXCH}}^{1\times k}$ for some k [9, 36]. A secure protocol for a trivial SSFE is a simple one in which one party simply sends a function of its input to the other party.

Characterizations of triviality are also known for the case of reactive functionalities [33, 36]. A deterministic, finite functionality is trivial if and only if it is both exchange-like and exchange-free. (This characterization is not true for randomized functionalities: $\mathcal{F}_{\text{COIN}}$ is both exchange-like and exchange-free).

3 Intractability Framework and Our Results

Recall that our interest is in "intractability assumptions" of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, where $\mathcal{F}$ and $\mathcal{G}$ are arbitrary deterministic finite functionalities. Among two-party functionalities, all the reductions we are able to definitively classify fall into one of four kinds (others which are only partially classified are also consistent with these four categories):

1. Reductions that are unconditionally true.
2. Reductions equivalent to the OWF assumption.
3. Reductions equivalent to the sh-OT assumption.
4. Reductions that are unconditionally false.

If $\mathcal{F}$ is non-trivial and $\mathcal{G}$ is trivial (as defined in Section 2.4), then $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is unconditionally false, no matter what computational complexity results may hold. (This is a consequence of the unconditional impossibility results in [9, 36].)

Recent results in a companion paper [33] show that all other assumptions (i.e., those that are not known to be unconditionally false) of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ are *implied by* the sh-OT assumption. Thus the remaining question is to determine which "standard" intractability assumption is *necessary* for $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$. Note that the space of functionalities we consider is infinite, and *a priori* one might expect a large number

[5] Recall that in our model functionalities — including communication channels — interact only with the parties. A channel with an eavesdropper is modeled as a 3-party functionality.

of, if not infinitely many, "distinct" assumptions (distinct in the sense of [21]). Indeed, such a phenomenon is not without precedence: in [15] for instance, an infinite hierarchy of complexity assumptions related to cryptographic protocols is derived. However, the assumptions in such an infinite hierarchy tend to have quantitative differences, and may not relate to conceptual differences. Indeed, the hierarchy of assumptions in [15] is based on the number of rounds in a protocol that is assumed to exist. On the other hand, the above assumptions (the OWF assumption and the sh-OT assumption) are conceptually different. Indeed, our framework formalizes a way to avoid such quantitative distinctions (since our notion of reduction ignores such differences).

In the following *conjectures* we refer to two assumptions being *distinct*. As mentioned above, this can be formalized as in [15, 21], by considering oracles given which one assumption holds but not the other. Since none of our results need this concept of distinct assumptions, we do not present the formalization here, but instead refer the readers to [15, 21].

Conjecture 1 (Quantization of Computational Assumptions.)**.** *For every m, there are only finitely many distinct assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, where $\mathcal{F}$ and $\mathcal{G}$ are finite m-party functionalities functionalities.*

We believe that there are several such new assumptions to be discovered. But we conjecture that we need to look beyond the more familiar case of deterministic 2-party functionalities to discover them. For instance, assumptions appear in our framework for 3-party functionalities that we believe do not appear for 2-party functionalities (see below), and we anticipate that more assumptions exist corresponding to complicated security requirements among many parties.

Conjecture 2. *Assumptions of the form $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, where $\mathcal{F}$ and $\mathcal{G}$ are deterministic finite two-party functionalities fall into one of the four classes listed above.*

We present significant progress towards affirming the above conjecture. Note that we are looking to prove that assumptions $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ are *equivalent* to the OWF assumption or the sh-OT assumption, unless they can be shown to be unconditionally true or false. One part of showing the equivalence is to show that the OWF assumption or the sh-OT assumption is sufficient for such a reduction, presumably by giving explicit protocols. This was carried out in [33]:

Proposition 1 (Based on results in [33].)**.** *The assumption $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is:*

- *unconditionally true (i.e., $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$) if $\mathcal{G}$ is complete or if $\mathcal{F}$ is trivial;*
- *unconditionally false, if $\mathcal{G}$ is trivial and $\mathcal{F}$ is non-trivial;*
- *equivalent to the* sh-OT *assumption if $\mathcal{G}$ is complete and $\mathcal{F}$ is passive-trivial;*
- *implied by the* sh-OT *assumption if $\mathcal{G}$ is non-trivial;*
- *implied by the* OWF *assumption if $\mathcal{G}$ is not exchange-like and $\mathcal{F}$ is passive-trivial.*

In this work, the focus is on showing the converse, that either the OWF assumption or sh-OT assumption is necessary for $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$.

Reductions equivalent to the sh-OT assumption

We show a large class of functionalities to be equivalent to the sh-OT assumption:

Theorem 1. *Let $\mathcal{G}$ be a non-trivial exchange-like functionality. Then for all $\mathcal{F}$, either $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$ or $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* sh-OT *assumption.*

Further, we characterize precisely when $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$, when $\mathcal{G}$ is exchange-like. Recall that the non-triviality of exchange-like functionalities is due solely to their simultaneity (independence) of inputs. If $\mathcal{F}$ is exchange-like and has a higher "bandwidth" of simultaneity (i.e., its dimensions are larger than $\mathcal{G}$'s), or if $\mathcal{F}$ is not exchange-like (i.e., it performs some hiding of the parties' inputs), then $\mathcal{F} \not\sqsubseteq_{\text{STAT}} \mathcal{G}$. Intuitively, simultaneity cannot be amplified or used for information hiding, except in Cryptomania.

Theorem 1 extends to the case of *reactive* exchange-like functionalities as well. Analysis of arbitrary reactive functionalities was introduced in [33], and we build on the analysis there, to characterize exchange-like reactive functionalities.

Reductions equivalent to the OWF assumption

Given Proposition 1 and Theorem 1, Conjecture 2 would be settled for the class of passive-trivial functionalities if the following conjecture is true:

Conjecture 3. *For any two functionalities $\mathcal{F}$ and $\mathcal{G}$, either $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$, or $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ implies the* OWF *assumption.*

That is, we would like to prove that the OWF assumption is indeed minimal for conditional reductions. While this may sound obvious, proving such a conjecture turns out to be difficult. Essentially, assuming that the OWF assumption does not hold, one

must show attacks on any protocol purportedly carrying out such a reduction. We obtain the following results, to partially confirm the above conjecture.

Theorem 2. *For every non-exchange-like, passive-trivial SFE functionality $\mathcal{G}$, there are infinitely many standalone-trivial SFE functionalities $\mathcal{F}$ such that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* OWF *assumption.*

Theorem 3. *Let $\mathcal{F}$ and $\mathcal{G}$ be passive-trivial but not UC-trivial. If $\mathcal{F}$ is not standalone-trivial and $\mathcal{G}$ is standalone-trivial, then $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* OWF *assumption.*

3.1 Beyond 2-Party Functionalities

The class of two-party functionalities that we considered is quite rich, but still omits some very important and familiar kinds of cryptographic tasks. In particular, the classical task of secret communication, is modeled as a functionality with three parties, Alice, Bob and Eve, and is not captured by any of the two-party functionalities.[6] Let us denote the 3-party private channel functionality (in which the third party learns only that the channel was invoked, when the first party sends a message to the second party) by $\mathcal{F}_{\text{PVT}}$. In contrast, let $\mathcal{F}_{\text{PUB}}$ be the a 3-party public-channel functionality, in which the message from the first party is received by both the other parties (though if the first party is corrupt, it is allowed to send different messages to the two others). The assumption in question is $\mathcal{F}_{\text{PVT}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{PUB}}$.

A key-agreement protocol, as considered in [15, 21], yields such a reduction. This assumption, corresponding to secrecy against third-party eavesdroppers, seems to be of a different flavor than any assumption arising out of cryptographic complexity of 2-party functionalities (wherein there is no external adversary).

Conjecture 4. *For any pair of two-party functionalities $\mathcal{F}, \mathcal{G}$, the assumption $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ can be black-box separated (à la [15, 21]) from the assumption $\mathcal{F}_{\text{PVT}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{PUB}}$.*

A first step to studying the assumptions arising of 3-party functionalities would be to understand various cryptographic complexity classes (based on statistical reductions). However, even the class of trivial SFE functions is not well-understood in this case. [37, Appendix. B] includes a few "trivial" 3-party functionalities, which are securely realizable using protocols that involve more than a single message (as with protocols for trivial 2-party functionalities).

Given the variety of cryptographic functionalities that exist in the 3-party scenario, we conjecture that there is at least one "undiscovered assumption" corresponding to a reduction among 3-party functionalities.

Conjecture 5. *There exist 3-party functionalities $\mathcal{F}$, $\mathcal{G}$, such that the assumption $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ can be black-box separated (a la [15, 21]) from* OWF *assumption,* sh-OT *assumption and $\mathcal{F}_{\text{PVT}} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{PUB}}$.*

4 Reductions Equivalent to the sh-OT Assumption

In this section we introduce a natural subclass of SFE functionalities and completely characterize $\sqsubseteq_{\text{PPT}}$ reductions involving this class. In Appendix B we greatly generalize the definition and results to *reactive* functionalities which need not give identical output to Alice and Bob; however, the results for SFE functionalities capture the major intuitions.

4.1 Exchange-Like Functionalities

Definition 4. *Let $\mathcal{F}$ be an SFE functionality. We say that $\mathcal{F}$ is* exchange-like *if $\mathcal{F} = \mathcal{F}_{\text{EXCH}}^{i \times j}$ for some i, j.*

The cryptographic non-triviality of an exchange-like functionality is in its independence of inputs. Exchange-like functionalities ultimately hide nothing about the parties' inputs, but enforce that the two parties' inputs are chosen independently.

Lemma 1 ([33])**.** *If $\mathcal{F}$ is not exchange-like, then either $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{STAT}} \mathcal{F}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq_{\text{STAT}} \mathcal{F}$.*

Proof sketch. The proof is a simple combinatorial characterization, and we sketch the main ideas here. Kraschewski and Müller-Quade [29] show (strengthening a result of Kilian [25]) that if $\mathcal{F}$ contains a 2×2 minor of the form $\left[\begin{smallmatrix} a & b \\ b & b \end{smallmatrix}\right]$, where $a \neq b$ (called a *generalized-*OR *minor*, then $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{STAT}} \mathcal{F}$. Otherwise, if $\mathcal{F}$ contains no generalized-OR minor, but contains a minor of the form $\left[\begin{smallmatrix} a & b \\ c & b \end{smallmatrix}\right]$ or $\left[\begin{smallmatrix} a & c \\ b & b \end{smallmatrix}\right]$, where a, b, c are distinct (called a *generalized-*CC *minor*), then it is straight-forward to see that the protocol in which both parties simply restrict their inputs to that 2×2 minor of $\mathcal{F}$ is an unconditionally UC-secure protocol for $\mathcal{F}_{\text{CC}}$ (see [33]). Note that *in general*, restricting inputs to a minor of an SFE is not a secure protocol, since malicious parties may send different inputs to $\mathcal{F}$.

[6] Recall that our functionalities do not communicate with the adversary when all parties are honest. This convention requires modeling corrupt parties explicitly, in the protocol and in the functionality. Hence secret communication corresponds to a 3-party functionality.

Thus, if $\mathcal{F}_{\text{OT}} \not\sqsubseteq_{\text{STAT}} \mathcal{F}$ and $\mathcal{F}_{\text{CC}} \not\sqsubseteq_{\text{STAT}} \mathcal{F}$, then $\mathcal{F}$ cannot have any generalized-CC or generalized-OR minors. Every 2×2 minor of $\mathcal{F}$ must have be of one of the forms $\left[\begin{smallmatrix} a & a \\ b & b \end{smallmatrix}\right]$, $\left[\begin{smallmatrix} a & b \\ a & b \end{smallmatrix}\right]$, or $\left[\begin{smallmatrix} a & b \\ c & d \end{smallmatrix}\right]$, where a, b, c, d are all distinct. It is easy to see that $\mathcal{F}$ is therefore an exchange function (possibly with duplicate inputs).

4.2 Reductions Involving Exchange-Like Functionalities

Our main classification involving exchange-like functionalities is the following:

Theorem 1 (restated). *If $\mathcal{G}$ is exchange-like and non-trivial, then either $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$, or $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* sh-OT *assumption.*

Proof. From [33], we have that $\mathcal{G}$ is $\sqsubseteq_{\text{PPT}}$-complete under the sh-OT assumption, since it is non-trivial. Thus $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ under the sh-OT assumption.

For the other direction, we break the proof into two parts, depending on the status of $\mathcal{F}$. These are carried out in the following two lemmas.

Lemma 2. *If $\mathcal{F}$ is not exchange-like, and $\mathcal{G}$ is exchange-like and non-trivial, then $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ implies the* sh-OT *assumption.*

Proof. Given that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$, we directly construct a passive secure protocol for $\mathcal{F}_{\text{OT}}$. From Lemma 1, we have that either $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{STAT}} \mathcal{F}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq_{\text{STAT}} \mathcal{F}$. Thus either $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{PPT}} \mathcal{G}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq_{\text{PPT}} \mathcal{G}$ by the universal composition theorem.

In the first case, $\mathcal{F}_{\text{OT}}$ has the property that any UC-secure protocol for $\mathcal{F}_{\text{OT}}$ (even in a hybrid world) is also itself a semi-honest-secure protocol [36]. $\mathcal{G}$ also has a semi-honest-secure protocol (namely, its canonical protocol since it is decomposable). Composing these two protocols yields a semi-honest (plain) protocol for $\mathcal{F}_{\text{OT}}$ and we are done.

In the other case, suppose π is the secure protocol for $\mathcal{F}_{\text{CC}}$ in the $\mathcal{G}$-hybrid world. Recall that $\mathcal{F}_{\text{CC}}$ has a function table $\left[\begin{smallmatrix} 0 & 2 \\ 1 & 2 \end{smallmatrix}\right]$, which we interpret as Alice sending a bit (top row or bottom row), Bob choosing whether or not to receive it (left column or right column), and Alice learning Bob's choice (whether or not the output was 2). We directly use π to construct a semi-honest $\mathcal{F}_{\text{OT}}$ protocol as follows, with Alice acting as the OT sender (with inputs x_0, x_1) and Bob the receiver (with input b):

- The parties instantiate two parallel instances of π, with Alice acting as the sender in both. Since there is no access to an external $\mathcal{G}$, Bob will simulate Alice's interface with instances of $\mathcal{G}$— that is, Alice will send her $\mathcal{G}$-inputs directly to Bob, and he will give simulated responses on behalf of these simulated instances of $\mathcal{G}$. Alice uses x_0 and x_1 as her respective inputs to the two instances of protocol π, and runs the protocol honestly.
- In protocol instance $(1-b)$, Bob carries out the simulation of $\mathcal{G}$-instances and the π protocol completely honestly. He runs the π protocol on the input that does not reveal Alice's input (i.e., he chooses the "right column" input to $\mathcal{F}_{\text{CC}}$).
- In protocol instance b, Bob honestly runs the UC simulator for π, treating Alice as the adversary (including simulating Alice's interface with $\mathcal{G}$-instances). At some point, the simulator extracts Alice's bit x_b which would normally be sent to $\mathcal{F}_{\text{CC}}$. Bob continues running the simulator as if $\mathcal{F}_{\text{CC}}$ responded with output 2. When the interaction completes, Bob outputs x_b.

By the UC security of π, Alice's view is computationally independent of b (i.e., she cannot distinguish an interaction with π's simulator from an interaction in which the receiver and $\mathcal{G}$ are honest). By the soundness of the simulator, we also see that Bob correctly learns x_b; we must argue that he has no advantage guessing x_{1-b}. If all $\mathcal{G}$-instances were external to the $(1-b)$ interaction (instead of Bob simulating them), then the security of π would imply that Bob has no advantage in guessing x_{1-b}, since the protocol's output is 2. Being an exchange function, however, $\mathcal{G}$ has the property that Bob always learns all of Alice's inputs anyway. Thus Alice can send her $\mathcal{G}$-inputs directly to Bob, without any affect on the security of the protocol. This is exactly what happens in the $(1-b)$ interaction.

We note that the technique of running two protocol instances, one honestly and the other using the simulator, to obtain a semi-honest OT protocol was also used by Damgård and Groth [12], though in the context of a protocol for $\mathcal{F}_{\text{COM}}$ instead of $\mathcal{F}_{\text{CC}}$, and considering a common random string instead of an arbitrary exchange-like $\mathcal{G}$. This theme also occurs in our constructions of semi-honest OT protocols throughout this section, although we greatly generalize the technique and the corresponding analysis.

For the case where $\mathcal{F}$ is also exchange-like, it is no longer true that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to sh-OT assumption for all $\mathcal{F}$ and $\mathcal{G}$. For some $\mathcal{F}$ and $\mathcal{G}$, it can be easily seen that $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$. We completely characterize when each case holds.

Theorem 4. *Let $\mathcal{F}$ and $\mathcal{G}$ be exchange-like, so without loss of generality, $\mathcal{F} = \mathcal{F}_{\text{EXCH}}^{i \times j}$ and $\mathcal{G} = \mathcal{F}_{\text{EXCH}}^{i' \times j'}$. Then if $i \leq i'$ and $j \leq j'$, or if $i \leq j'$ and $j \leq i'$,*

then $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$. *Otherwise,* $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ *is implies the* sh-OT *assumption.*

Put differently, if an $i \times j$ rectangle can fit inside an $i' \times j'$ rectangle (in either of its two axis-aligned orientations), then $\mathcal{F}_{\text{EXCH}}^{i \times j} \sqsubseteq_{\text{STAT}} \mathcal{F}_{\text{EXCH}}^{i' \times j'}$; otherwise, $\mathcal{F}_{\text{EXCH}}^{i \times j} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{EXCH}}^{i' \times j'}$ is equivalent to the sh-OT assumption. Since i and j represent the "bandwidth" (in the two directions) of an exchange-like functionality, a simple way to interpret this theorem is that *increasing the bandwidth of an exchange-like functionality is equivalent to the* sh-OT *assumption,* while decreasing the bandwidth is cryptographically trivial.

Main Ideas.

The protocol that demonstrates $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$ is elementary. To perform an $i \times j$ exchange using $\mathcal{G}$, simply send inputs directly to $\mathcal{G}$ (with Alice and Bob exchanged if necessary). Each party aborts if the other party provided an input to the $i' \times j'$ exchange which was out of bounds for an $i \times j$ exchange. The security of this protocol is straight-forward.

We first sketch the main ideas behind proving the other direction As an illustrative, example suppose that $\mathcal{F} = \mathcal{F}_{\text{EXCH}}^{i \times i}$ and $\mathcal{G} = \mathcal{F}_{\text{EXCH}}^{(i-1) \times (i-1)}$, and that we have a protocol π demonstrating $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$. The role of the simulator for π is to first extract the input of a corrupt party, send it to $\mathcal{F}$ in the ideal world, and then continues to simulate π consistently given the output from $\mathcal{F}$.

Again for the benefit of this simplified overview, suppose that the simulator for a *passively* corrupt Alice always extracts during round r_A.[7] Since the simulator does not contact $\mathcal{G}$ throughout the first $r_A - 1$ rounds of the simulation, Alice's view is independent of Bob's input during these rounds. If Bob's input is random (uniform in $[i]$), then after round r_A, Alice still cannot guess Bob's input with probability greater than $\zeta = (i-1)/i$, since there are only $i - 1$ possible responses from the simulated $\mathcal{G}$ that the simulator can give to complete the simulation of round r_A. By the soundness of the simulation, an honest Alice cannot predict Bob's input with probability greater than $\zeta + \text{negl}(k)$ after r_A rounds of an *honest* interaction with Bob. Similarly, if the simulator for a passively corrupt Bob always extracts during round r_B, then an honest Bob cannot predict Alice's random input with probability greater than $\zeta + \text{negl}(k)$ after r_B rounds of an honest interaction with Alice.

By symmetry, suppose that $r_A \leq r_B$. Then we can obtain a semi-honest protocol for a weak variant of OT as follows. As in the proof of Lemma 2, the parties run two instances of π, with Alice providing random inputs to both instances. In one instance Bob runs the simulator, and in the other instance Bob runs the protocol and simulates $\mathcal{G}$ honestly. Both instances are truncated after r_A rounds. Using the same reasoning as before, Alice cannot distinguish which interaction was honest and which used the simulator; Bob learns one of Alice's inputs since the simulator extracts in round r_A, but cannot predict Alice's other input with probability better than $(i-1)/i$, since $r_A \leq r_B$. This slight uncertainty can be easily amplified to obtain a full-fledged protocol for oblivious transfer (see Appendix C for information on amplifying weak OT protocols).

The main proof is more involved in several ways. First, r_A and r_B need not be fixed rounds, but may be random variables. In this case, the parties must essentially guess $\min\{r_A, r_B\}$. Still, we can obtain a weak OT protocol in which Bob has noticeable uncertainty about one of Alice's inputs, and which is therefore amenable to amplification. Second, the case where the dimensions of $\mathcal{F}$ and $\mathcal{G}$ are incomparable requires a much more careful analysis.

Weak Oblivious Transfer & Amplification.

As mentioned above, we will construct a protocol for a weak variant of OT which can then be amplified into full-fledged OT protocol. Similar to the definition of (p, q)-OT used in [13], we define the following weak OT security:

Definition 5 (q-weak-OT)**.** *A q-weak-OT is a protocol between a sender and receiver that satisfies the following conditions:*

- *The sender has inputs $(x_0, x_1) \in \mathbb{Z}_N^2$. The receiver has input $b \in \{0, 1\}$ and correctly receives output x_b with overwhelming probability.*
- *A passively corrupt sender has only negligible advantage in guessing the bit b.*
- *No passively corrupt receiver can guess x_{1-b} with probability noticeably greater than q, when the sender's inputs are random.*

Thus, $\frac{1}{N}$-weak-OT corresponds to the standard definition of OT, when the sender's input domain is $\mathbb{Z}_N$. Note that this definition considers only simple indistinguishability-based security properties of (weak) OT protocols, while the formal definition of the sh-OT assumption demands an *efficiently simulatable* OT protocol for semi-honest adversaries. How-

[7] If a round *begins* with a call to the external functionality $\mathcal{G}$, then the round concludes when the parties receive their output from this external functionality. Extracting *during* round r means that the simulator extracts after seeing the adversary's input to the external functionality, and before delivering the corresponding output.

ever, if an OT protocol is $1/N$-weak as above, then it is an efficient and sound simulation to simulate the honest protocol with *any* input for the honest party that is consistent with the required output. Thus this indistinguishability-based definition is equivalent to the more succinct simulation-based definition.

In Appendix C, we show that any $(1 - \frac{1}{\text{poly}})$-weak OT protocol can be amplified into a full-fledged OT protocol.

The Construction

As an introduction to how we deal with a simulator that may extract in an unpredictable round, we show the following simpler result:

Lemma 3. *Let $\mathcal{F}_{\text{COIN}}$ be the ideal coin-tossing functionality. Then $\mathcal{F}_{\text{EXCH}}^{2\times 2} \sqsubseteq \mathcal{F}_{\text{COIN}}$ is equivalent to the* sh-OT *assumption.*

Proof. Let π be a secure protocol for $\mathcal{F}_{\text{EXCH}}^{2\times 2}$ in the $\mathcal{F}_{\text{COIN}}$-hybrid model. We will transform π to obtain a protocol for $\mathcal{F}_{\text{OT}}$ secure against semi-honest adversaries.

Let s_B be the random variable denoting the round in which the simulator extracts from a passively corrupt Alice and sends her input to $\mathcal{F}_{\text{EXCH}}^{2\times 2}$. Suppose Bob's input is chosen at random. Then fix any passive adversarial strategy for Alice which outputs a guess of Bob's input at each step of the protocol, and define t_A as the random variable denoting the round when this guess is correct with probability at least $\zeta = 3/4$ (where the probability is over the randomness independent of Alice's view), *when interacting with the simulator*. By the soundness of the simulation, Alice's view is completely independent of Bob's input through the first s_B rounds. Thus $t_A \geq s_B + 1$, and in particular, $E[t_A] \geq E[s_B] + 1$.

Now consider running this passive adversarial strategy for Alice against an honest Bob in the actual protocol execution, instead of against the simulator. We define u_A to be the random variable denoting the first round in which Alice's guess of Bob's random input is correct with probability at least ζ. By the security of π, the two interactions must be indistinguishable to this Alice strategy, thus $|E[u_A] - E[t_A]| < \epsilon/\zeta = \epsilon'$, where ϵ is the negligible simulation error of the protocol. Thus $E[u_A] \geq E[s_B] + 1 - \epsilon'$.

Similarly we can define u_B and s_A with the roles of Alice and Bob reversed, and conclude that $E[u_B] \geq E[s_A] + 1 - \epsilon'$. Then, either $E[u_A] \geq E[s_A] + 1 - \epsilon'$, or $E[u_B] \geq E[s_B] + 1 - \epsilon'$; otherwise we would get that $E[u_A] < E[s_A] + 1 - \epsilon' \leq E[u_B] < E[s_B] + 1 - \epsilon'$, contradicting a previous conclusion.

By symmetry, we assume that $E[u_B] \geq E[s_B] + 1 - \epsilon'$. In other words, in an interaction with an honest Alice, the simulator will, on average, extract Alice's input earlier than any passive Bob could guess Alice's input with probability at least ζ.

Protocol for a weak variant of $\mathcal{F}_{\text{OT}}$. Alice has randomly chosen inputs $x_0, x_1 \in \{0,1\}$, and Bob has input $b \in \{0,1\}$. Protocol π is given as a UC-secure protocol for $\mathcal{F}_{\text{EXCH}}^{2\times 2}$ in the $\mathcal{F}_{\text{COIN}}$-hybrid model.

1. Alice honestly runs two instances of the protocol π with Bob, using inputs x_0 and x_1, respectively.
2. Bob picks a random $r \in [r(\kappa)]$, where $r(\kappa)$ is a polynomial bound on the number of rounds in π and κ is the global security parameter.
3. In the bth instance of π, Bob runs the simulator for π against Alice (including simulating her interface with instances of $\mathcal{F}_{\text{COIN}}$), and halts the interaction after the rth round of π.
4. In the $(1-b)$ instance of π, Bob runs the π protocol honestly with Alice on a fixed input (say, 0), and also honestly simulates all instances of $\mathcal{F}_{\text{COIN}}$ for Alice. Bob halts the interaction after the rth round of π.
5. If the simulator has extracted x_b, then Bob outputs it. Otherwise, he asks Alice for (x_0, x_1), and she sends it to him.

Figure 2: Weak oblivious transfer protocol, assuming $\mathcal{F}_{\text{EXCH}}^{2\times 2} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{COIN}}$.

Now consider the protocol given in Figure 2. First, since Alice cannot distinguish a simulated instance of π from an honest execution of π, Alice has no advantage in predicting Bob's bit b. Thus the protocol gives complete privacy for Bob.

A passively corrupt Bob in this weak-OT protocol can guess Alice's input x_{1-b} correctly with probability at most

$$\begin{aligned}
&\Pr[s_B \leq r < u_B]\zeta + (1 - \Pr[s_B \leq r < u_B]) \\
&\quad = 1 - (1-\zeta)\Pr[s_B \leq r < u_B] \\
&\quad \leq 1 - (1-\zeta)E[u_B - s_B]/r(\kappa) \\
&\quad \leq 1 - (1-\zeta)(1-\epsilon')/r(\kappa)
\end{aligned}$$

by the definition of u_B. Or, in other words, Bob's guess must be incorrect with probability at least $(1-\zeta)(1-\epsilon')/r(\kappa)$, which is an inverse polynomial in the security parameter. In Appendix C, we show how a weak $\mathcal{F}_{\text{OT}}$ protocol with this security property can be amplified to give a full-fledged (semi-honest) $\mathcal{F}_{\text{OT}}$ protocol.

We now prove Theorem 4 for the following special case:

Lemma 4. *Let $i \geq 3$. Then $\mathcal{F}_{\text{EXCH}}^{i\times i} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$ implies the* sh-OT *assumption.*

This special case illustrates most of the non-trivial ideas used to show Theorem 4, although slightly more involved arguments are required for exchange functions of arbitrary sizes. The full details of all the cases are given in Appendix A.

Proof. The main difference between this proof and the one of Lemma 3 is that we must now consider information about the parties' inputs that can be exchanged via access to ideal functionality $\mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$.

Note that in the proof of the Lemma 3, we would obtain a suitable weak OT protocol (i.e., amenable to amplification) even if ζ is defined to be any constant (in fact, even if ζ is at most $1 - \frac{1}{\text{poly}}$ in the security parameter).

As before, we let s_B be the round during which the simulator extracts from a passively corrupt Alice. Clearly Alice's view is independent of Bob's input throughout the first $s_B - 1$ rounds of interaction. If s_B is not a round in which the parties access $\mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$, then Alice's view is independent of Bob's input through s_B rounds as well. Otherwise, if s_B is a round in which Alice sends an input to her interface of $\mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$, then the simulator can send the extracted input to $\mathcal{F}_{\text{EXCH}}^{i\times i}$, receive the output (i.e., Bob's input), and then complete the round by simulating the response of the simulated $\mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$ functionality to Alice. This response from $\mathcal{F}_{\text{EXCH}}^{(i-1)\times(i-1)}$ is an element of $[i-1]$ and is the *only* part of Alice's view that can depend on Bob's input. Thus, Alice cannot guess Bob's input with probability greater than $(i-1)/i$ after s_B rounds (see Lemma 6).

If we define ζ to be any constant greater than $(i-1)/i$, and define t_B as the first point at which Alice can guess Bob's input with probability at least ζ, then we have $t_B \geq s_A + 1$. Similarly, we can reverse the roles of Alice and Bob and obtain $t_A \geq s_B + 1$. The rest of the proof is identical to that of Lemma 3, with a different (but still constant) value of ζ.

4.3 Extension to Reactive Functionalities

In Appendix B, we extend all of the results in this section to a large class of *reactive* functionalities — namely, those functionalities which can be modeled as a finite state machine. We note that this class of functionalities also includes those which do not give identical outputs to the two parties.

New techniques for understanding and using arbitrary reactive functionalities for cryptographic purposes were introduced in [33]. Our definition and analysis of exchange-like reactive functionalities here closely follows many of the same approaches as [33]. For completeness, we include all of the relevant details in Appendix B.

Intuitively, a reactive functionality is exchange-like if it never hides information. Note that there are two fundamentally different ways in which a reactive functionality can hide information: it can hide information in a single round of interaction (as a non-exchange-like SFE hides information), or it can hide information in its internal memory. Formalizing this second requirement is somewhat involved, and requires an automata-theoretic analysis of reactive functionalities.

If a reactive functionality is exchange-like, then we show that it is equivalent (under $\sqsubseteq_{\text{STAT}}$ reductions) to a "bundle" of exchange functions of different sizes. Such a bundle is simply a functionality in which one party publicly selects one of the component exchange functions and then the two parties evaluate it. As such, Theorem 4, our result about amplifying the "bandwidth" of an exchange-like functionality, carries over naturally to the case of reactive functionalities.

For the other case, when $\mathcal{F}$ is non-exchange-like and $\mathcal{G}$ is exchange-like, we first observe that $\mathcal{F}$ can fail to be exchange-like for only two reasons. If $\mathcal{F}$ hides information in a single round, then $\mathcal{F}$ can easily be used to obtain a non-exchange-like SFE function and so the proof of Lemma 2 goes through essentially unaltered. On the other hand, if $\mathcal{F}$ hides information between rounds, we show that this hiding can be exploited to show $\mathcal{F}_{\text{COM}} \sqsubseteq_{\text{STAT}} \mathcal{F}$. Intuitively, $\mathcal{F}_{\text{COM}}$ is the canonical example of a functionality that hides information between rounds (between the commit phase and reveal phase). Then, by using a similar technique as in the proof of Lemma 2, we can easily construct a semi-honest protocol for OT. The parties run two instances of the *commit* phase of the protocol for $\mathcal{F}_{\text{COM}}$. The receiver plays honestly in one instance, and uses the simulator to extract in the other instance. In fact, this is essentially the semi-honest OT protocol from Damgård and Groth [12], except that we consider it in the context of an arbitrary exchange-like ideal functionality, instead of a common random string.

Thus, Theorem 1 applies for all deterministic, finite, *reactive* functionalities.

5 Reductions Equivalent to the OWF Assumption

Our results in this section build on the technique in [34] that was used to derive the following separation in cryptographic complexity.

Lemma 5 ([34])**.** *Let $\mathcal{F}$ and $\mathcal{G}$ be SSFE functional-*

ities. If $\mathcal{F}$ has unique decomposition depth n and $\mathcal{G}$ has decomposition depth $m < n$, then $\mathcal{F} \not\sqsubseteq_{\text{STAT}} \mathcal{G}$.

In [34], Lemma 5 is proven by attacking any purported protocol π for $\mathcal{F}$ in the $\mathcal{G}$-hybrid world.

First, they show (for plain protocols, not in any hybrid world) that for every adversary $\mathcal{A}$ that attacks the canonical protocol for $\mathcal{F}$, there is a corresponding adversary $\mathcal{A}'$ that attacks π, achieving the same effect in all environments. (Indeed, any functionality whose decomposition depth is at least 2 has a simple attack against its canonical protocol that violates security in the UC sense.) Intuitively, the protocol π must reveal information in the same order as the canonical protocol. More formally, at every point during the canonical protocol (say, a partial transcript t), there is a corresponding "frontier" in π — a maximal set of partial transcripts of π. If two inputs both induce transcript t in the canonical protocol (recall that it is a deterministic protocol), then they also induce statistically indistinguishable distributions on partial transcripts at the frontier. But if the two inputs do not both induce transcript t in the canonical protocol, then at the frontier they induce distributions on partial transcripts that have statistical distance almost 1. Then the adversary $\mathcal{A}'$ runs the protocol π honestly, except for occasionally "swapping" its effective input at one of these frontiers. The properties of the frontiers assure that such a swap will only negligibly affect the outcome of the interaction.

Next, to attack a protocol π in the $\mathcal{G}$-hybrid world, they imagine a plain protocol $\widehat{\pi}$ which is π composed with the canonical protocol for $\mathcal{G}$. The plain protocol $\widehat{\pi}$ has frontiers for each step of the canonical protocol (equivalently, step of the decomposition). In our setting, there are more frontiers in $\widehat{\pi}$ than there are rounds in the canonical protocol for $\mathcal{G}$, so not all the frontiers can be contained entirely within the $\mathcal{G}$-subprotocols. Thus an adversary attacking π can behave honestly in all interactions with the ideal $\mathcal{G}$, and still encounter a frontier at which to "swap" its effective input (i.e., outside of the $\mathcal{G}$-subprotocols in $\widehat{\pi}$). Indeed, there is an attack against $\mathcal{F}$ in which an adversary need only encounter one such frontier, so the protocol π is not secure.

Leveraging one-way functions.

While these frontier-based attacks from [34] are formulated for computationally unbounded adversaries, we show below that they can in fact be carried out under the assumption that one-way functions *do not exist*. In other words, that if a reduction exists between particular functions, then the OWF assumption is true.

These frontier-based attacks require unbounded computation because computing the frontier involves computing global statistical properties about the protocol — namely, the probability that the protocol assigns to various partial transcripts on different inputs. The attacks are otherwise efficient, so given access to an oracle that can compute these probabilities, the attack can be easily effected. In fact, these quantities need not be computed exactly for the attacks to violate security. Thus we will describe how to compute the appropriate quantities given that OWFs do not exist. We use the guarantee of no distributionally one-way functions (Section 2.3; [20]). We define a function related to the given protocol, and use the ability to approximately sample its preimage distribution to obtain a good estimate of the desired probabilities.

Theorem 2 (restated). *For every non-exchange-like, passive-trivial SFE functionality $\mathcal{G}$, there are infinitely many standalone-trivial SFE functionalities $\mathcal{F}$ such that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* OWF *assumption.*

Proof. First, if $\mathcal{G}$ is non-exchange-like, then $\mathcal{F}_{\text{COM}} \sqsubseteq_{\text{PPT}} \mathcal{G}$ under the OWF assumption, by the protocol construction in [33]. Then, $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{F}_{\text{COM}}$ since $\mathcal{F}$ is passive-trivial [34]. The non-trivial direction is to show that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ implies OWF assumption for infinitely many such $\mathcal{F}$.

Let k be the round complexity of a semi-honest protocol for $\mathcal{G}$, since $\mathcal{G}$ is passive-trivial. Then let $\mathcal{F}$ be any standalone-trivial functionality whose decomposition depth is strictly greater than k. In the complete characterization of standalone-triviality [34], there are SFE functionalities with arbitrarily high decomposition depth, so there are infinite number of such $\mathcal{F}$ satisfying this condition.

Now $\mathcal{F}$ and $\mathcal{G}$ satisfy the conditions of Lemma 5, so it suffices to show that the frontier-based attack from [34] can be carried out under the assumption that (distributionally) one-way functions do not exist. As described above, the attack against a protocol π for $\mathcal{F}$ in the $\mathcal{G}$-hybrid model is based on frontiers in the protocol. For a partial transcript u and inputs x for Alice and y for Bob, the probability that the protocol generates u as the prefix of its transcript can be expressed as $\alpha(u, x)\beta(u, y)$, where each of the two terms depends on only one party's input (see, for example, [4]).

The frontiers used in the attack are then all defined in terms of the following quantity:

$$\eta(u, x_0, x_1) = \frac{\alpha(u, x_0) - \alpha(u, x_1)}{\alpha(u, x_0) + \alpha(u, x_1)}$$

or the symmetric quantity with respect to the roles of Alice & Bob. Intuitively, $\eta(u, x_0, x_1)$ measures how correlated the transcript u is to Alice's input being x_0 versus x_1. In fact, the entire frontier-based attack can be carried out in polynomial time given an oracle that answers questions of the form "Is $\eta(u, x_0, x_1) \geq 1 - \nu(k)$?", where ν is a certain negligible function in the security parameter. If instead the oracle can answer questions of this form where $\nu(k) = 1/k^c$ for a chosen constant c, then the adversary's attack may fail with at most an extra 1/poly factor. All the attacks from [34] demonstrate that the real and ideal worlds can be distinguished with constant bias, so they can indeed tolerate this additional 1/poly slack factor. Thus it suffices to show how to implement such an oracle.

We compute $\eta(u, x_0, x_1)$ as follows: First, consider the function $f(x, r_A, y, r_B, i) = (\tau, x)$, where τ is the first i bits of the transcript produced by the protocol when executed honestly on inputs (x, y), where r_A and r_B are the random tapes of Alice and Bob, respectively. We use the guarantee of no distributionally one-way functions to sample from $f^{-1}(u, x_0)$ and $f^{-1}(u, x_1)$. If both preimages are empty, then the protocol never generates u as a partial transcript on inputs x_0 or x_1. If only one is empty, then $\eta(u, x_0, x_1) = 1$.

Otherwise, assume u is indeed a possible partial transcript for both x_0 and x_1 (i.e., the protocol assigns positive probability to u when Alice has inputs x_0 or x_1). Our previous sampling of f^{-1} has yielded an input y^* such that u is a possible partial transcript when executing π on inputs (x_0, y^*). Thus u is also a possible partial transcript on inputs (x_1, y^*). Now define:

$$g(x, r_A, y, r_B, i) = \begin{cases} (\tau, y) & \text{if } x \in \{x_0, x_1\} \\ \bot & \text{otherwise} \end{cases}$$

We now sample n times from $g^{-1}(u, y^*)$. Let n_i be the number of times the sampled preimage included x_i as the first component. Then $(n_0 - n_1)/n$ is an estimate of $\eta(u, x_0, x_1)$. By setting n to be a sufficiently large polynomial in the security parameter, we can ensure that the estimate is within an additive factor $1/k^c$ of the actual value, with high probability.

Theorem 3 (restated). *Let $\mathcal{F}$ and $\mathcal{G}$ be passive-trivial but not UC-trivial. If $\mathcal{F}$ is not standalone-trivial and $\mathcal{G}$ is standalone-trivial, then $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ is equivalent to the* OWF *assumption.*

Proof. The fact that $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ under the OWF assumption is by the same argument as in the previous proof.

For the other direction, suppose π is a secure protocol for $\mathcal{F}$ in the $\mathcal{G}$-hybrid world. Standalone secure protocols *for SFE functionalities* are closed under composition. Thus we have a standalone-secure protocol π' for $\mathcal{F}$ without any trusted party.

Being passive-trivial, $\mathcal{F}$ is surely decomposable, and we consider two cases. When $\mathcal{F}$ is uniquely decomposable, then [34] showed that in the *unbounded* setting, for every adversary $\mathcal{A}$ attacking the canonical protocol, there is an adversary $\mathcal{A}'$ attacking π' such that no environment can distinguish between the two interactions. When $\mathcal{F}$ is uniquely decomposable but not standalone-trivial, there is a simple attack against the canonical protocol for $\mathcal{F}$ that violates standalone security with constant probability. Thus translating this attack into an efficient (assuming that OWF assumption is false) attack on π' using the techniques described in the previous proof, we see that π' is not standalone-secure; a contradiction.

On the other hand, if $\mathcal{F}$ is not uniquely decomposable, then $\mathcal{F}_{\text{XOR}} \sqsubseteq_{\text{STAT}} \mathcal{F}$ via a simple protocol. As such, by composing several protocols, we obtain a standalone-secure protocol π for $\mathcal{F}_{\text{XOR}}$. Consider an interaction using π in which the honest party chooses an input at random. In [32], a frontier-based attack on $\mathcal{F}_{\text{XOR}}$ is described (for unbounded adversaries), which we outline below. We show here that the attack can be carried out assuming that the OWF assumption is false, to bias the honest party's output towards 0 by a noticeable amount:

At each partial transcript u, compute an estimate of $|\eta(u, 0, 1)|$ (which measures the transcript's bias towards Alice's input 0 or 1, defined in the previous proof) At the beginning of the protocol, the value of this function is 0, and at the end of the protocol, it is 1 with overwhelming probability since the protocol results in Bob correctly learning Alice's input.

Similarly, define $\eta'(u, 0, 1)$ as a transcript's bias towards Bob's input. By symmetry, with probability at least 1/2, the partial transcript achieves $|\eta(u, 0, 1)| > 1/2$ before it achieves $|\eta'(u, 0, 1)| > 1/2$. Thus an attack for Bob is to discover via the sampling procedure described above the first point at which $|\eta(u, 0, 1)| > 1/2$ but $|\eta'(u, 0, 1)| \leq 1/2$. At that point, Bob switches his input to match Alice's, in order to bias the output towards 0. Bob reaches such a point with probability at least 1/2, Since $|\eta'(u, 0, 1)| \leq 1/2$, the correctness of the protocol implies that Bob's output will be 0 with overwhelming probability. Thus this attack successfully biases the output towards 0 with bias 1/4 minus some inverse polynomial in the security parameter.

6 Expanding the Framework

The framework that we have presented is quite general thanks to the general nature of functionalities. However, there are several dimensions in which this framework can be extended, leading to possibly further computational complexity assumptions to appear. We mention a few such extensions below.

Using Infinite Domain/Memory Functionalities.

In this paper, we have confined ourselves to "finite" functionalities (which are finite state machines with finite input/output alphabets). One could instead consider the more general class, with infinite alphabets and/or infinite state space. On the positive side, this allows modeling "object-oriented" cryptography which involves such concepts as encryption and signatures (as opposed to "service-oriented" cryptography which is more natural in the setting of multi-party computation). Note that currently, the sh-OT assumption and the assumption that a key-agreement protocol exists, do not specify the number of rounds of these protocols. Indeed a 2-round key-agreement protocol *is* a public-key encryption scheme and a 2-round sh-OT protocol is a computational version of the so-called dual-mode encryption scheme.[8]

On the other hand, this considerably complicates things: for instance, Proposition 1 does not hold any more. [33] points out an (infinite domain) functionality $\mathcal{G}$ such that $\mathcal{G}$ is not trivial, but say $\mathcal{F}_{\text{OT}} \sqsubseteq_{\text{PPT}} \mathcal{G}$ does not hold under the sh-OT assumption. It is likely that several intermediate assumptions, like in the hierarchy of assumptions presented in [15] will manifest in this framework. A similar complication was encountered in [18]. We leave it is an interesting challenge to extend the framework to include infinite functionalities, but without sacrificing the economy and conceptual clarity of the set of complexity assumptions produced by the framework.

Using Alternate Reductions.

If we use a different notion of reduction in place of $\sqsubseteq_{\text{PPT}}$, the assumption $\mathcal{F} \sqsubseteq_{\text{PPT}} \mathcal{G}$ will change its meaning. While it is not clear if any reasonable notion of reduction will give a *larger set of assumptions* when all pairs $(\mathcal{F}, \mathcal{G})$ are considered, it certainly is true that for specific pairs $(\mathcal{F}, \mathcal{G})$ the assumption becomes different. The variants one could consider weaker notions of security like security against passive (semi-honest) or standalone adversaries, or stronger notions of security like simultaneous security against passive and active adversaries, or against adaptive adversaries. One could also consider tighter non-standard reduction notions derived by imposing constraints on the protocols carrying out the reductions, like constant round complexity, for instance. Another example is to restrict to protocols which use a given functionality in only one direction (with say, Alice and Bob in the protocol playing fixed roles, say sender and receiver respectively, when interacting with the given functionality).

It will indeed be interesting if a computational complexity assumption manifests when using some other (meaningful) notions of reduction, but not the one we use.

More Functionalities.

In this work, we restricted ourselves to a large, but restricted class of functionalities. In particular, our functionalities are not "fair": they allow the adversary to learn the output and then decide whether to deliver the output or not, to the other party. Another class we considered only briefly is that of randomized functionalities. These classes of functionalities are understood only in bits and pieces. A systematic study of their cryptographic complexities remains an open problem.

Special Pairs.

Finally, we mention that when considering some of the above extensions, it might be interesting to consider (only) special pairs of functionalities, where the reduction may have extra meaning. For instance, when restricting protocols to use a given 2-party functionality $\mathcal{F}$ in only one direction, it is particularly interesting to consider reducing the functionality $\mathcal{F}^{-1}$ which reverses the roles of its parties. Another interesting question is of "parallel repetition" which considers reducing $\mathcal{F}^t$, a synchronous repetition of t copies of $\mathcal{F}$, to the functionality $\mathcal{F}$ (wherein the protocol is allowed to use multiple asynchronous copies of $\mathcal{F}$). For instance, $\mathcal{F}_{\text{COIN}}^t$ reduces to $\mathcal{F}_{\text{COIN}}$ unconditionally, but by Theorem 1 we know that a parallel repetition of $\mathcal{F}_{\text{EXCH}}^{2,2}$ reduces to $\mathcal{F}_{\text{EXCH}}^{2,2}$ if and only if the sh-OT assumption holds.

[8] Here, a dual-mode encryption scheme is a public-key encryption scheme, in which there is an alternate mode to generate a public-key (which remains indistinguishable from a key generated in the normal mode) such that the semantic security of the encryption is retained even given the randomness used in key generation. Dual mode encryption introduced in [28, 35] is stronger in that the alternate mode is required to result in a public-key such that a ciphertext produced using that key is statistically independent of the message.

References

[1] *Proc. 20th STOC.* ACM, 1988.
[2] *Proc. 21st STOC.* ACM, 1989.
[3] *Proc. 30th FOCS.* IEEE, 1989.

[4] D. Beaver. Perfect privacy for two-party protocols. In J. Feigenbaum and M. Merritt, editors, *Proceedings of DIMACS Workshop on Distributed Computing and Cryptography*, volume 2, pages 65–77. American Mathematical Society, 1989.

[5] A. Beimel, T. Malkin, and S. Micali. The all-or-nothing nature of two-party secure computation. In M. J. Wiener, editor, *CRYPTO*, volume 1666 of *Lecture Notes in Computer Science*, pages 80–97. Springer, 1999.

[6] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proc.* 20*th STOC* [1], pages 1–10.

[7] R. Canetti. Universally composable security: A new paradigm for cryptographic protocols. Electronic Colloquium on Computational Complexity (ECCC) TR01-016, 2001. Previous version "A unified framework for analyzing security of protocols" availabe at the ECCC archive TR01-016. Extended abstract in FOCS 2001.

[8] R. Canetti, editor. *Theory of Cryptography, Fifth Theory of Cryptography Conference, TCC 2008, New York, USA, March 19-21, 2008*, volume 4948 of *Lecture Notes in Computer Science*. Springer, 2008.

[9] R. Canetti, E. Kushilevitz, and Y. Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In E. Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*. Springer, 2003.

[10] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols. In *Proc.* 20*th STOC* [1], pages 11–19.

[11] B. Chor and E. Kushilevitz. A zero-one law for boolean privacy (extended abstract). In *STOC* [2], pages 62–72.

[12] I. Damgård and J. Groth. Non-interactive and reusable non-malleable commitment schemes. In *Proc.* 35*th STOC*, pages 426–437. ACM, 2003.

[13] I. Damgård, J. Kilian, and L. Salvail. On the (im)possibility of basing oblivious transfer and bit commitment on weakened security assumptions. In J. Stern, editor, *EUROCRYPT*, volume 1592 of *Lecture Notes in Computer Science*, pages 56–73. Springer, 1999.

[14] I. Damgård, J. B. Nielsen, and C. Orlandi. On the necessary and sufficient assumptions for uc computation. Cryptology ePrint Archive: Report 2009/247, 2009.

[15] Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The relationship between public key encryption and oblivious transfer. In *FOCS*, pages 325–335, 2000.

[16] O. Goldreich, S. Micali, and A. Wigderson. How to play ANY mental game. In ACM, editor, *Proc.* 19*th STOC*, pages 218–229. ACM, 1987. See [**?** , Chap. 7] for more details.

[17] I. Haitner. Semi-honest to malicious oblivious transfer - the black-box way. In Canetti [8], pages 412–426.

[18] D. Harnik, M. Naor, O. Reingold, and A. Rosen. Completeness in two-party secure computation: A computational view. *J. Cryptology*, 19(4):521–552, 2006.

[19] R. Impagliazzo. A personal view of average-case complexity. In *Structure in Complexity Theory Conference*, pages 134–147, 1995.

[20] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography (extended abstract). In *Proc.* 30*th FOCS* [3], pages 230–235.

[21] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *STOC* [2], pages 44–61.

[22] Y. Ishai, M. Prabhakaran, and A. Sahai. Founding cryptography on oblivious transfer - efficiently. In Wagner [40], pages 572–591.

[23] J. Kilian. Founding cryptography on oblivious transfer. In *STOC* [1], pages 20–31.

[24] J. Kilian. *Uses of Randomness in Algorithms and Protocols.* PhD thesis, Department of Electrical Engineering and Computer Science, Mas sachusetts Institute of Technology, 1989.

[25] J. Kilian. A general completeness theorem for two-party games. In *STOC*, pages 553–560. ACM, 1991.

[26] J. Kilian. More general completeness theorems for secure two-party computation. In *Proc.* 32*th STOC*, pages 316–324. ACM, 2000.

[27] J. Kilian, E. Kushilevitz, S. Micali, and R. Ostrovsky. Reducibility and completeness in private computations. *SIAM J. Comput.*, 29(4):1189–1208, 2000.

[28] G. Kol and M. Naor. Cryptography and game theory: Designing protocols for exchanging information. In Canetti [8], pages 320–339.

[29] D. Kraschewski and J. Müller-Quade. Completeness theorems with constructive proofs for symmetric, asymmetric and general 2-party-functions, 2008. Unpublished Manuscript, 2008. http://iks.ira.uka.de/eiss/completeness.

[30] R. Künzler, J. Müller-Quade, and D. Raub. Secure computability of functions in the it setting with dishonest majority and applications to long-term security, 2009.

[31] E. Kushilevitz. Privacy and communication complexity. In *FOCS* [3], pages 416–421.

[32] M. Mahmoody-Ghidary, H. K. Maji, P. Ouppaphan, M. Prabhakaran, and M. Rosulek. Exploring the limits of random oracles and trusted coins using frontier analysis of protocols. Manuscript, 2009.

[33] H. Maji, M. Prabhakaran, and M. Rosulek. A zero-one law for deterministic 2-party secure computation. Manuscript. Results from preliminary version appear in [39], 2009.

[34] H. K. Maji, M. Prabhakaran, and M. Rosulek. Complexity of multi-party computation problems: The case of 2-party symmetric secure function evaluation. In Reingold [38], pages 256–273.

[35] C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In Wagner [40], pages 554–571.

[36] M. Prabhakaran and M. Rosulek. Cryptographic complexity of multi-party computation problems: Classifications and separations. In Wagner [40], pages 262–279.

[37] M. Prabhakaran and M. Rosulek. Cryptographic complexity of multi-party computation problems: Classifications and separations. *Electronic Colloquium on Computational Complexity (ECCC)*, 15(50), 2008.

[38] O. Reingold, editor. *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, volume 5444 of *Lecture Notes in Computer Science*. Springer, 2009.

[39] M. Rosulek. *The Structure of Secure Multi-Party Computation*. PhD thesis, Department of Computer Science, University of Illinois at Urbana-Champaign, 2009.

[40] D. Wagner, editor. *Advances in Cryptology - CRYPTO 2008: 28th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2008, Proceedings*, volume 5157 of *Lecture Notes in Computer Science*. Springer, 2008.

[41] A. C. Yao. Protocols for secure computation. In *Proc. 23rd FOCS*, pages 160–164. IEEE, 1982.

A Complete Proof of Theorem 1

We first establish a convenient technical lemma:

Lemma 6. *For each $j \in [i]$, let $\mathcal{D}_j$ be a probability distribution over the elements $\{m_1, \ldots, m_{i-1}\}$. Now consider the following experiment: Choose $j \in [i]$ uniformly at random, and then output a sample according to $\mathcal{D}_j$.*

The probability of correctly predicting j given only the output of this procedure is at most $(i-1)/i$.

Proof. Let $p_{u,v}$ be the probability of sampling message m_v when using $\mathcal{D}_u$. So, we have:

$$\sum_{v=1}^{i-1} p_{u,v} = 1 \text{ for all } u \in [i]$$

Let $q_{v,u}$ be the probability of outputting u after seeing message v. So, we have:

$$\sum_{u=1}^{i} q_{v,u} = 1 \text{ for all } v \in [i-1]$$

The probability of being correct is:

$$\zeta = \frac{\sum_{u=1}^{i}\sum_{v=1}^{i-1} p_{u,v}q_{v,u}}{i}$$

This is maximized if $q_{v,u} = \alpha_u p_{u,v}$. Therefore, $\zeta \leq \frac{\sum_{u=1}^{i}\alpha_v \sum_{v=1}^{i-1} p_{u,v}^2}{i} \leq \frac{\sum_{u=1}^{i-1}\alpha_u}{i} = \frac{i-1}{i}$

Finally we prove our general result.

Lemma 7. *Let i, j, i', j' be such that ($i > i'$ or $j > j'$) and ($i > j'$ or $j > i'$). Then $\mathcal{F}_{\text{EXCH}}^{i\times j} \sqsubseteq_{\text{PPT}} \mathcal{F}_{\text{EXCH}}^{i'\times j'}$ implies the* sh-OT *assumption.*

Proof. The proof is very similar to that of Lemma 4 Note that in the proofs of Lemma 4 and Lemma 3, we would obtain a suitable weak OT protocol (i.e., amenable to amplification) even if ζ is $1 - \frac{1}{\text{poly}}$ in the security parameter, and one of $\{E[t_B - s_B], E[t_A - s_A]\}$ is at least $\frac{1}{\text{poly}}$ in the security parameter.

Case 1: ($\max\{i,j\} > \max\{i',j'\}$): Suppose by symmetry that $i \geq j$ and $i > i' \geq j'$, and that Bob feeds input from $[i]$ into the ideal functionality. We define ζ to be any constant greater than $\frac{i-1}{i}$, and define s_B and t_A as earlier. Similar to the argument in , we get that $t_A \geq s_B + 1$ (because $i - 1 \geq i' \geq j'$). It is always the case that $t_B \geq s_A$. So, we get the condition that $t_A \geq s_B + 1$ and $t_B \geq s_A$.

In general we can say that:

$$(t_A \geq s_B \text{ and } t_B \geq s_A + 1), \text{ or}$$
$$(t_B \geq s_A \text{ and } t_A \geq s_B + 1)$$

These conditions imply that:

$$E[u_A] \geq E[s_A] + \left(\frac{1}{2} - \epsilon'\right), \text{ or}$$
$$E[u_B] \geq E[s_B] + \left(\frac{1}{2} - \epsilon'\right)$$

Observe that in our weak OT construction, all we needed was that one of $\{E[u_A - s_A], E[u_B - s_B]\}$ is at least $\frac{1}{\text{poly}}$ in the security parameter. So the construction in Lemma 4 yields a suitable weak-OT protocol.

Case 2: ($\min\{i',j'\} < i, j \leq \max\{i',j'\}$): Observe that even if for some polynomial $\lambda(\cdot)$ we have:

$$\left(E[t_A] \geq E[s_B] \text{ and } E[t_B] \geq E[s_A] + \frac{1}{\lambda(\kappa)}\right), \text{ or}$$
$$\left(E[t_B] \geq E[s_A] \text{ and } E[t_A] \geq E[s_B] + \frac{1}{\lambda(\kappa)}\right)$$

we can use the approach mentioned above to get the weak OT protocol. So, we only need to consider the remaining case, when $E[t_B] \in \left[E[s_A], E[s_A] + \frac{1}{\lambda(\kappa)}\right)$ and $E[t_A] \in \left[E[s_B], E[s_B] + \frac{1}{\lambda(\kappa)}\right)$, where $\lambda(\cdot)$ is a suitably chosen large polynomial.

In this case, we will prove that:

1. One of $\{\Pr(t_B \geq s_B + 1), \Pr(t_A \geq s_A + 1)\}$ is at least $\frac{1}{5}$
2. Both $|\Pr(u_A = r) - \Pr(t_A = r)|$ and $|\Pr(u_B = r) - \Pr(t_B = r)|$ are at most $\frac{1}{\rho(\kappa)}$ for any polynomial ρ

These will imply that we obtain a suitable weak-OT protocol in this case as well.

Now, we show that the above mentioned properties hold. If $E[t_B] \in \left[E[s_A], E[s_A] + \frac{1}{\lambda(\kappa)}\right)$ and $E[t_A] \in \left[E[s_B], E[s_B] + \frac{1}{\lambda(\kappa)}\right)$, then with probability $\geq 1 - \frac{2}{\lambda(\kappa)n}$, where n is the maximum number of rounds in the protocol, we will have the event that $t_B = s_A$ and $t_A = s_B$. Consider the set of rounds S where $t_A = s_B$ is possible. Similarly define T to be the set of rounds where $t_B = s_A$ is possible. Without loss of generality, we can assume that Alice uses i' side of $\mathcal{F}_{\text{EXCH}}^{i' \times j'}$ only in even rounds and the j' side of the $\mathcal{F}_{\text{EXCH}}^{i' \times j'}$ only in odd rounds. So, we conclude that the sets S and T are disjoint.

Let $x_S(r)$ be the probability of the event $t_A = s_B \leq r$. Similarly define $x_T(r)$ as the probability of the event $t_B = s_A \leq r$. At the extremes, $x_S(0) = x_T(0) = 0$ and $x_S(n) = x_T(n) = 1 - \frac{2}{\lambda(\kappa)n}$. So look at the smallest r such that $\max\{x_S(r), x_T(r)\} \geq \frac{1}{2}\left(1 - \frac{2}{\lambda(\kappa)n}\right)$. Observe that at any given round r only $x_S(r)$ or $x_T(r)$ changes. By symmetry, assume that $x_S(r)$ reaches the threshold first. Then since $y_S(r)$ could not have changed at this round, we get that $y_S(r) \leq \frac{1}{2}\left(1 - \frac{2}{\lambda(\kappa)n}\right)$. Then for sufficiently large values of κ, we see that with probability at least

$$\left(\frac{1}{2} - \frac{1}{\lambda(\kappa)n}\right)^2 \geq \frac{1}{4} - \frac{2}{\lambda(\kappa)n} \geq \frac{1}{5},$$

the event $s_B \leq t_B - 1$ is true.

Now, all we need to show is that $\Pr(u_B = r)$ and $\Pr(t_B = r)$ are within $1/\text{poly}(\kappa)$. We pick a suitable polynomial ρ. We run an honest execution of the protocol against a simulator for Alice. We can estimate $\Pr(t_B = r)$ within $\frac{1}{\rho}$ additive error in polynomial time. Similarly, we run an honest execution of the protocol against honest Alice. We can estimate $\Pr(u_B = r)$ within $\frac{1}{\rho}$ additive error in polynomial time.

If $|\Pr(t_B = r) - \Pr(u_B = r)| > \frac{3}{\rho}$, then we can create a polynomial time distinguisher which distinguishes between the real and ideal world. So, for every round r, $|\Pr(t_B = r) - \Pr(u_B = r)| \leq \frac{3}{\rho}$.

Given the guarantee that, for all r, $|\Pr(t_B = r) - \Pr(u_B = r)| \leq \frac{3}{\rho}$ and $\Pr(s_B \leq t_B - 1) \geq \frac{1}{5}$, the construction given earlier gives us a weak OT.

B Reactive Exchange-Like Functionalities

In this section we extend the results of Section 4 to a large class of *reactive* functionalities. Namely, the following:

Definition 6. *A* deterministic finite functionality (DFF) *is a functionality with a finite set of internal states Q, whose behavior is as follows:*

1. *Set the internal state q to the distinguished start state $q_0 \in Q$.*
2. *Wait for inputs $x \in X$ from Alice and $y \in Y$ from Bob, where X, Y are finite input sets.*
3. *If $\delta(q, x, y)$ is defined, then send "delayed outputs" $f_A(q, x, y)$ to Alice and $f_B(q, x, y)$ to Bob, where δ, f_A, f_B are deterministic functions.*
4. *Set $q \leftarrow \delta(q, x, y)$ and repeat from step 2.*

To reason about the behavior of reactive functionalities, we follow [33, 39] and develop a way of saying that one input x "achieves the same effect" as another input x', in the context of a reactive functionality. Intuitively, this happens when every behavior that can be induced by sending x at a certain point can also be induced by sending x' instead, and thereafter appropriately translating subsequent inputs and outputs. We can define this formally in terms of the UC security definition:

Definition 7 (Dominating Inputs)**.** *Let $\mathcal{F}$ be a DFF, and let $x, x' \in X$ be inputs for Alice. We say that x* dominates *x' in the first round of $\mathcal{F}$, and write $x \geq_A x'$, if there is a secure protocol for $\mathcal{F}$ in the $\mathcal{F}$-hybrid setting, where the protocol for Bob is to run the dummy protocol (as Bob), and the protocol for Alice has the property that whenever the environment provides input x' for Alice in the first round, the protocol instead sends x to the functionality in the first round.*

We define domination for Bob inputs analogously, with the roles of Alice and Bob reversed. Note that the definition requires that any behavior of $\mathcal{F}$ that is possible when Alice uses x' as her first input can also be induced *in an online fashion* by using x as her first input (and subsequently translating inputs/outputs according to some strategy). Domination is reflexive and transitive.

Definition 8 (Exchange States)**.** *Let $\mathcal{F}$ be a DFF, and let q be one of its states. We define $\mathcal{F}[q]$ as the functionality obtained by modifying $\mathcal{F}$ so that its start state is q.*

We say that q is an exchange state *if:*

- *The input/output behavior of $\mathcal{F}$ at state q — $(f_A(q,\cdot,\cdot), f_B(q,\cdot,\cdot))$ — is (isomorphic to) an exchange function; and*
- *For all Alice inputs $x, x' \in X$ such that $f_B(q,x,\cdot) \equiv f_B(q,x',\cdot)$, there exists an Alice input $x^* \in X$ such that $x^* \geq_A x$ and $x^* \geq_A x'$ in $\mathcal{F}[q]$; and*
- *For all Bob inputs $y, y' \in Y$ such that $f_A(q,\cdot,y) \equiv f_A(q,\cdot,y')$, there exists a Bob input $y^* \in Y$ such that $y^* \geq_B y$ and $y^* \geq_B y'$ in $\mathcal{F}[q]$.*

Suppose q is an exchange state. Then we can define $x \overset{q}{\sim} x'$ if $f_B(q,x,\cdot) \equiv f_B(q,x',\cdot)$. The relation $\overset{q}{\sim}$ induces equivalence classes over X. When q is an exchange state, then within each such equivalence class, there exists at least one input x^* which dominates all other members of its class. For each equivalence class, we arbitrarily pick a single such input x^* and call it a *master* input for state q. Similarly we define master inputs for Bob by exchanging the roles of Alice and Bob.

Definition 9. *Let $\mathcal{F}$ be a DFF, We say that a transition is* safe *if it leaves an exchange state q on inputs (x,y), where x and y are both master inputs for state q.*

B.1 Exchange-Like Definition

Our generalization of exchange-like functionalities is in terms of these automata-theoretic properties.

Definition 10 (Exchange-like)**.** *We say that a DFF $\mathcal{F}$ is* exchange-like *if no non-exchange state in $\mathcal{F}$ is reachable via a sequence of safe transitions from $\mathcal{F}$'s start state.*

We justify the use of the term "exchange-like" in the following lemma. Namely, exchange-like functionalities are equivalent to a collection of several (non-reactive) exchange functions.

Lemma 8. *Let $\langle \mathcal{F}_1, \ldots, \mathcal{F}_n \rangle$ be a DFF which in the first round accepts input $k \in [n]$ from Alice, outputs k to Bob, and then simulates $\mathcal{F}_k$.*

If a DFF $\mathcal{G}$ is exchange-like, then $\mathcal{G}$ is equivalent (under $\sqsubseteq_{\text{STAT}}$ reductions) to some $\langle \mathcal{F}_{\text{EXCH}}^{i_1 \times j_1}, \ldots, \mathcal{F}_{\text{EXCH}}^{i_n \times j_n} \rangle$.

Proof sketch. Given $\mathcal{G}$, we first define a related functionality $R(\mathcal{G})$ which is simply $\mathcal{G}$ with all non-safe transitions deleted. Then using an argument almost identical to [33], we have that $\mathcal{G} \sqsubseteq R(\mathcal{G}) \sqsubseteq \mathcal{G}$ (see the following lemmas). Intuitively, for these functionalities, the parties can be made to use only "master" inputs without loss of generality.

Now for each reachable state q in $R(\mathcal{G})$, its input/output function at that state is an $i_q \times j_q$ exchange function. Let $\mathcal{F} = \langle \mathcal{F}_{\text{EXCH}}^{i_q \times j_q} \mid q \in Q \rangle$; then the protocol for $R(\mathcal{G})$ using access to $\mathcal{F}$ is for both parties to do the following: Maintain the current state q, and in each round, instantiate a new instance of $\mathcal{F}$ and ensure that Alice sends input q to $\mathcal{F}$ (Bob aborts otherwise). Then use $\mathcal{F}$ again to perform the input/output function of $\mathcal{G}$. The output of $\mathcal{F}$ uniquely determines both party's inputs, and thus the next value of $\mathcal{G}$'s internal state q, so we repeat.

To show that $\mathcal{F} \sqsubseteq R(\mathcal{G})$, for each reachable state q in $R(\mathcal{G})$, let $(x_1, y_1), \ldots, (x_n, y_n)$ be a sequence of inputs that leaves $R(\mathcal{G})$ in state q. The protocol for $\mathcal{F}$ is to have Alice first send her input q to Bob. Then both parties send the corresponding input sequence to $R(\mathcal{G})$ to place it in state q. Either party can determine from its view whether the other party has input the correct sequence. If this is not the case, then the parties abort. Otherwise, they send their next round inputs to $R(\mathcal{G})$ directly and use the output as their own output (after normalizing the inputs/outputs to $[i_q] \times [j_q]$).

To complete the proof sketched above, we now describe the construction of a "normalized" version $R(\mathcal{F})$ of an exchange-like functionality $\mathcal{F}$. We first define an intermediate functionality:

Definition 11. *We define $r(\mathcal{F})$ to be the functionality which runs $\mathcal{F}$, except that in the first round only, it allows only* safe *transitions to be taken (i.e., transitions on master inputs only). $r(\mathcal{F})$ can be written as a copy of $\mathcal{F}$ plus a new start state. The new start state of $r(\mathcal{F})$ duplicates all the safe transitions of $\mathcal{F}$'s start state.*

Observation 5. *If a safe transition was just taken in $\mathcal{F}$, then Alice (resp. Bob) can uniquely determine Bob's (resp. Alice's) input in the previous round and the current state of $\mathcal{F}$, given only the previous state of $\mathcal{F}$ and Alice's (resp. Bob's) input and output in the previous round.*

Proof. We will show that Alice has no uncertainty about which master input Bob used, thus no uncertainty about the resulting state of $\mathcal{F}$. If a safe transition was just taken from q, then q was an exchange state and its associated SFE $(f_A(q,\cdot,\cdot), f_B(q,\cdot,\cdot))$ is isomorphic to an exchange function. Note that our definition of dominating inputs subsumes the definition of redundant inputs in the context of function isomorphism.

Thus if y, y' are distinct master inputs for Bob, then $f_A(q,\cdot,y) \not\equiv f_A(q,\cdot,y')$. As such, for any master input

x for Alice, $f_A(q, x, y) \neq f_A(q, x, y')$. Alice has no uncertainty about which master input Bob used. This argument is symmetric for Bob as well.

Lemma 9. *If the start state of $\mathcal{F}$ is an exchange state, then $r(\mathcal{F}) \sqsubseteq \mathcal{F} \sqsubseteq r(\mathcal{F})$. Furthermore, if q is reachable from the start state of $\mathcal{F}$ via a safe transition, then $\mathcal{F}[q] \sqsubseteq \mathcal{F}$.*

Proof. The protocol for $r(\mathcal{F}) \sqsubseteq \mathcal{F}$ is the dummy protocol, since $r(\mathcal{F})$ implements simply a subset of the behavior of $\mathcal{F}$. Simulation is trivial unless in the first round, the corrupt party (say, Alice) sends an input x to $\mathcal{F}$ which is not a master input for q_0. The simulator must send the corresponding master input x^* (from the $\overset{q_0}{\sim}$ equivalence class of x) in the ideal world, and then it uses the translation protocol guaranteed by the definition of $x^* \geq_A x$ to provide a consistent view to Alice and induce correct outputs for Bob.

Similarly, the protocol for $\mathcal{F} \sqsubseteq r(\mathcal{F})$ is simply the dual of the above protocol. On input x in the first round, Alice sends x^* to $r(\mathcal{F})$, where x^* is the master input from the $\overset{q_0}{\sim}$-equivalence class of x. Thereafter, Alice runs the protocol guaranteed by the fact that $x^* \geq_A x$. Bob's protocol is analogous. Simulation is a trivial dummy simulation, since any valid sequence of inputs to $r(\mathcal{F})$ in the real world also produces the same outcome in the $\mathcal{F}$-ideal world ($r(\mathcal{F})$ implements a subset of the behavior of $\mathcal{F}$).

Note that in $r(\mathcal{F})$, the added start state has no incoming transitions; thus $(r(\mathcal{F}))[q] = \mathcal{F}[q]$ if q is a state in $\mathcal{F}$. So to show $\mathcal{F}[q] \sqsubseteq \mathcal{F}$, it suffices to show that $(r(\mathcal{F}))[q] \sqsubseteq r(\mathcal{F})$. Suppose q is reachable in $\mathcal{F}$ from the start state via safe transition on master inputs x^*, y^*. The protocol for $\mathcal{F}[q]$ is for Alice and Bob to send x^* and y^* to $r(\mathcal{F})$, respectively, as a "preamble". Each party can determine with certainty, given their input and output in this preamble, whether $r(\mathcal{F})$ is in state q (since only safe transitions can be taken from the start state of $r(\mathcal{F})$). If the functionality is not in q, then the parties abort. Otherwise, the functionality is $r(\mathcal{F})$ in state q as desired, so the parties thereafter run the dummy protocol. Simulation is trivial – the simulator aborts if the corrupt party does not send its specified input (x^* or y^*) in the preamble; otherwise it runs a dummy simulation.

The claim used in the proof of Lemma 8 is the following:

Lemma 10. *Let $R(\mathcal{F})$ be $\mathcal{F}$ with all non-safe transitions removed. Then $\mathcal{F} \sqsubseteq R(\mathcal{F}) \sqsubseteq \mathcal{F}$.*

Proof. First we show that $\mathcal{F} \sqsubseteq R(\mathcal{F})$. We prove a stronger claim; namely that if q is safely reachable (i.e., reachable from the start state by a sequence of safe transitions) in $\mathcal{F}$, then $\mathcal{F}[q] \sqsubseteq (R(\mathcal{F}))[q]$. To prove this stronger claim, we construct a family of protocols $\hat{\pi}_q$, for every such q.

First, let π_q denote the protocol guaranteed by $\mathcal{F}[q] \sqsubseteq r(\mathcal{F}[q])$ (Lemma 9). Then the protocol $\hat{\pi}_q$ is as follows:

1. Run π_q to interact with the functionality.
2. After the first round, we will have sent an input to the functionality and received an output. Assuming that the functionality was $(R(\mathcal{F}))[q]$, use the first round's input/output to determine the next state q' (Observation 5)
3. Continue running π_q, but hereafter, instead of letting it interact directly with the functionality, we recursively instantiate $\hat{\pi}_{q'}$. We let our π_q instance interface with $\hat{\pi}_{q'}$, which we let interact directly with the functionality.

The protocol is recursive, and after k rounds, must maintain a stack depth of size k. We prove by induction on k that $\hat{\pi}_q$ is a secure protocol for $\mathcal{F}[q]$ using $(R(\mathcal{F}))[q]$, against environments that run the protocol for $k \geq 0$ steps. The claim is trivially true for $k = 0$.

Note that simulation is trivial if either party is corrupt. Such an adversary is running the protocol interacting with $(R(\mathcal{F}))[q]$, which is a subset of the functionality $\mathcal{F}[q]$. Thus the simulator is a dummy simulator. It suffices to show that the output of the protocol is correct (indistinguishable from the ideal interaction) when both parties are honest.

In the first round, both parties are running π_q, interacting with $(R(\mathcal{F}))[q]$. Although π_q is designed to interact with $r(\mathcal{F}[q])$, the behavior of both these functionalities is identical in the first round (including the next-state function). Thus the first round of outputs is correct, by the security of π_q. For the same reason, step 2 of $\hat{\pi}_q$ correctly identifies the next state q' of $(R(\mathcal{F}))[q]$. Clearly $(R(\mathcal{F}))[q][q'] = (R(\mathcal{F}))[q']$, so after step 1 of the protocol, the functionality is identical to a fresh instantiation of $(R(\mathcal{F}))[q']$. At the same time, we also instantiate a fresh instance of $\hat{\pi}_{q'}$ to interact with this functionality. By the inductive hypothesis, hereafter π_q is interacting with an interface that is indistinguishable from an ideal interaction with $\mathcal{F}[q']$. However, an external functionality which behaves like $R(\mathcal{F})[q]$ in the first round, then after transitioning to state q' behaves like $\mathcal{F}[q']$, is simply the functionality $r(\mathcal{F}[q])$. In other words, the entire protocol $\hat{\pi}_q$ is indistinguishable from running π_q on $r(\mathcal{F}[q])$. By definition of π_q, this is indistinguishable from an ideal interaction with $\mathcal{F}[q]$ itself.

The protocol for $R(\mathcal{F}) \sqsubseteq \mathcal{F}$ is the dual of the above protocol. The protocol is the dummy proto-

col, and the simulator recursively uses the protocols π_q as above.

B.2 Main Classification

We now prove the main result regarding exchange-like functionalities, that is:

Theorem 6. *Let $\mathcal{F}$ and $\mathcal{G}$ be DFFs. If $\mathcal{G}$ is exchange-like and non trivial, then either $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$, or $\mathcal{F} \sqsubseteq \mathcal{G}$ is equivalent to the SHOT assumption.*

We split the proof into two parts, depending on whether $\mathcal{F}$ itself is also exchange-like.

When $\mathcal{F}$ is exchange-like

For the case where both $\mathcal{F}$ and $\mathcal{G}$ are exchange-like, we prove a simple combinatorial characterization for when $\mathcal{F} \sqsubseteq \mathcal{G}$, which generalizes our result for SFE functionalities. Namely, $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$ if and only if every exchange contained in $\mathcal{F}$ can "fit inside" an exchange contained in $\mathcal{G}$. Otherwise, the existence of a secure protocol is equivalent to the sh-OT assumption. More formally:

Lemma 11. *Let S and T be finite subsets of $\mathbb{Z}^2$. Say that $S \leq T$ if for every $(i,j) \in S$, there exists $(i',j') \in T$ such that $i \leq i' \wedge j \leq j'$ or $i \leq j' \wedge j \leq i'$.*

Let $\mathcal{F} = \langle \mathcal{F}_{\text{EXCH}}^{i \times j} \mid (i,j) \in S \rangle$ and $\mathcal{G} = \langle \mathcal{F}_{\text{EXCH}}^{i \times j} \mid (i,j) \in T \rangle$. Then if $S \leq T$, then $\mathcal{F} \sqsubseteq_{\text{STAT}} \mathcal{G}$, and if $S \not\leq T$, then $\mathcal{F} \sqsubseteq \mathcal{G}$ is equivalent to the sh-OT assumption.

The proof is a straight-forward generalization of our characterization for SFE. Since every exchange-like DFF is equivalent to such a collection of exchanges, this establishes the main result for DFFs as well.

When $\mathcal{F}$ is not exchange-like.

We describe how any non-exchange-like functionality can be used to unconditionally realize $\mathcal{F}_{\text{COM}}$. The argument here is reproduced directly from [33, 39] with minimal modification (corresponding to our definition of exchange states which is a relaxation of the definition of "simple states" there).

In [33, 39], the following property about dominating inputs is given:

Lemma 12 ([33, 39])**.** *Let $\mathcal{F}$ be a DFF. Then there is an environment $\mathcal{Z}_0$ with the following properties:*

- *$\mathcal{Z}_0$ sends a constant number of inputs to $\mathcal{F}$,*
- *$\mathcal{Z}_0$ chooses inputs for both parties at random,*
- *$\mathcal{Z}_0$ always outputs 1 when interacting with two parties running the dummy protocol on an instance of $\mathcal{F}$,*
- *For every $x, x' \in X$, if $x \not\geq_A x'$, then $\mathcal{Z}_0$ has a constant probability of outputting 0 when interacting with an Alice protocol that sends x instead of x' in the first round.*

Using this fact, we obtain the following claim, used in the proof of main theorem:

Lemma 13. *If a non-exchange state in $\mathcal{F}$ is reachable via a sequence of safe transitions from $\mathcal{F}$'s start state, then either $\mathcal{F}_{\text{COM}} \sqsubseteq \mathcal{F}$ or $\mathcal{F}_{\text{OT}} \sqsubseteq \mathcal{F}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq \mathcal{F}$.*

Proof. Without loss of generality (by Lemma 9) we assume that the start state of $\mathcal{F}$ is a non-exchange state.

First, suppose the start state q_0 of $\mathcal{F}$ is a non-exchange state because its input/output behavior in the first round is not an exchange function. Then in the $\mathcal{F}$-hybrid setting we can easily securely realize the SFE functionality $\mathcal{G} = (f_A(q_0, \cdot, \cdot), f_B(q_0, \cdot, \cdot))$, by the simple dummy protocol. Even though $\mathcal{F}$ may keep in its memory arbitrary information about the first-round inputs, the information can never be accessed since honest parties never send inputs to $\mathcal{F}$ after its first round, and $\mathcal{F}$ waits for inputs from both parties before giving any output. Thus $\mathcal{G} \sqsubseteq \mathcal{F}$. By Lemma 1, we have that either $\mathcal{F}_{\text{OT}} \sqsubseteq \mathcal{F}$ or $\mathcal{F}_{\text{CC}} \sqsubseteq \mathcal{F}$ in this case.

Otherwise, assume that the input/output behavior in the first round is an exchange function SFE, and that q_0 is a non-exchange state for one of the other reasons in the definition of exchange states. The two cases are symmetric, and we present the case where Alice can commit to Bob. Suppose there are Alice inputs $x_0^*, x_1^* \in X$ such that $f_B(q_0, x_0^*, \cdot) \equiv f_B(q_0, x_1^*, \cdot)$, but for all $x \in X$, either $x \not\geq_A x_0^*$ or $x \not\geq_A x_1^*$. Intuitively, this means that $\mathcal{F}$ *binds* Alice to her choice between inputs x_0^* and x_1^* — there are behaviors of $\mathcal{F}$ possible when her first input is x_b^*, which are not possible when her first input is x_{1-b}^*. We formalize this intuition by using the first input round of $\mathcal{F}$ to let Alice commit a bit to Bob.

Recall the "universal" environment $\mathcal{Z}_0$ from Lemma 12, and suppose it runs for m rounds and has a distinguishing probability $p > 0$. Our protocol for $\mathcal{F}_{\text{COM}}$ is to instantiate $N = 2\lceil \log_{1-p} 0.5 \rceil \kappa = \Theta(\kappa)$ independent instances of $\mathcal{F}$, where κ is the security parameter. We will write $\mathcal{F}_i$ to refer to the ith instance of $\mathcal{F}$. The protocol is as follows:

1. (Commit phase, on Alice input (COMMIT, b), where $b \in \{0,1\}$) Alice sends x_b^* to each $\mathcal{F}_i$. For each i, Bob sends a random $y_{i1} \in Y$ to $\mathcal{F}_i$ and waits for output $f_B(q_0, y_{i1}, x_0^*) = f_B(q_0, y_{i1}, x_1^*)$. If he receives a different input, he aborts. Otherwise, he outputs COMMITTED.
2. (Reveal phase, on Alice input REVEAL) Alice sends b to Bob. For each i, Alice sends her input/output view of $\mathcal{F}_i$ to Bob (x_b^* and the first-round response from $\mathcal{F}_i$). If any of these reported views involve Alice sending something other than

x_b^* to $\mathcal{F}_i$, then Bob aborts. Otherwise, Bob sets $x_{i1} = x_b^*$ for all i.

3. For $j = 2$ to m:
   (a) Bob sends Alice a randomly chosen $x_{ij} \in X$. Alice sends x_{ij} to $\mathcal{F}_i$.
   (b) Bob sends a randomly chosen input $y_{ij} \in Y$ to $\mathcal{F}_i$.
   (c) For each i, Alice reports to Bob her output from $\mathcal{F}_i$ in this round.
4. If for any i, Alice's reported view or Bob's outputs from $\mathcal{F}_i$ does not match the (deterministic) behavior of $\mathcal{F}$ on input sequence $(x_{i1}, y_{i1}), (x_{i2}, y_{i2}), \ldots$, then Bob aborts. Otherwise, he outputs (REVEAL, b).

When Bob is corrupt, the simulation is to do the following for each i: When Bob sends y_{i1} to $\mathcal{F}$ in the commit phase, simulate $\mathcal{F}_i$'s response as $f_B(q_0, x_0^*, y_{i1}) = f_B(q_0, x_1^*, y_{i1})$. In the reveal phase, to open to a bit b, simulate that Alice sent Bob x_b^* and the view that is consistent with that input: $f_A(q_0, x_b^*, y_{i1})$. Maintain the corresponding state q_i of $\mathcal{F}_i$ after seeing inputs (x_b^*, y_{i1}). Then when Bob sends x_{ij} to Alice and y_{ij} to $\mathcal{F}_i$, simulate that $\mathcal{F}_i$ gave the correct output to Bob and that Alice reported back the correct output from $\mathcal{F}_i$ that is consistent with $\mathcal{F}$ receiving inputs x_{ij}, y_{ij} in state q_i. Each time, also update the state q_i according to those inputs. It is clear that the simulation is perfect.

When Alice is corrupt, the simulation is as follows: The simulator faithfully simulates each instance of $\mathcal{F}$ and the behavior of an honest Bob. If at any point, the simulated Bob aborts, then the simulation aborts. Suppose Alice sends $\tilde{x}_{i1}$ to each $\mathcal{F}_i$ in the commit phase, and that the simulation has not aborted at the end of the commit phase. If the majority of $\tilde{x}_{i1}$ values satisfy $\tilde{x}_{i1} \geq_A x_0^*$, then the simulator sends $(\text{COMMIT}, 0)$ to $\mathcal{F}_{\text{COM}}$; otherwise it sends $(\text{COMMIT}, 1)$. Note that by the properties of $\mathcal{F}$, each $\tilde{x}_{i1}$ cannot dominate both x_0^* and x_1^*. Let b be the bit that the simulator sent to $\mathcal{F}_{\text{COM}}$.

If the simulated Bob ever outputs (REVEAL, b), then the simulator sends REVEAL to $\mathcal{F}_{\text{COM}}$. The simulation is perfect except for the case where the simulated Bob outputs $(\text{REVEAL}, 1-b)$ (in this case, the real world interaction ends with Bob outputting $(\text{REVEAL}, 1-b)$, while the ideal world interaction aborts). We show that this event happens with negligible probability, and thus our overall simulation is statistically sound.

Suppose Alice sends $b' = 1 - b$ at the beginning of the reveal phase. Say that an instance $\mathcal{F}_i$ is *bad* if $\tilde{x}_{i1} \not\geq_A x_{1-b}^*$. Note that at least half of the instances of $\mathcal{F}_i$ are bad. When an instance $\mathcal{F}_i$ is bad, $\mathcal{Z}_0$ can distinguish with probability at least p between the cases of $\mathcal{F}$ receiving first input $\tilde{x}_{i1}$ and x_{1-b}^* from Alice. However, in each instance of $\mathcal{F}_i$, Bob is sending random inputs to Alice (who sent $\tilde{x}_{i1}$ as the first input to $\mathcal{F}_i$), sending random inputs himself to $\mathcal{F}_i$, obtaining his own output and Alice's reported output from $\mathcal{F}_i$ in an on-line fashion, and comparing the result to the known behavior of $\mathcal{F}$ (when x_{1-b}^* is the first input of Alice). This is exactly what $\mathcal{Z}_0$ does in the definition of $\tilde{x}_{i1} \geq_A x_{1-b}^*$, so Bob will detect an error with probability p in each bad instance. In the real world, Bob would accept in this reveal phase with probability at most $(1-p)^{-N/2} \leq 2^{-\kappa}$, which is negligible as desired.

Lemma 14. *If $\mathcal{G}$ is exchange-like and non-trivial, and $\mathcal{F}$ is not exchange-like, then $\mathcal{F} \sqsubseteq \mathcal{G}$ is equivalent to the* sh-OT *assumption.*

Proof. From [33], we have that $\mathcal{G}$ is $\sqsubseteq$-complete given the sh-OT assumption, and thus $\mathcal{F} \sqsubseteq \mathcal{G}$. The main challenge is proving that $\mathcal{F} \sqsubseteq \mathcal{G}$ implies the sh-OT assumption.

Since $\mathcal{F}$ is not exchange-like, then either $\mathcal{F}_{\text{OT}} \sqsubseteq \mathcal{F}$, $\mathcal{F}_{\text{CC}} \sqsubseteq \mathcal{F}$, or $\mathcal{F}_{\text{COM}} \sqsubseteq \mathcal{F}$. Thus, by the universal composition theorem we assume that we have a secure protocol for either $\mathcal{F}_{\text{CC}}$, $\mathcal{F}_{\text{OT}}$, or $\mathcal{F}_{\text{COM}}$ using $\mathcal{G}$. The cases involving $\mathcal{F}_{\text{CC}}$ and $\mathcal{F}_{\text{OT}}$ have been addressed already in the proof of the characterization for non-reactive exchange-like functionalities.

Thus we describe the case where $\mathcal{F}_{\text{COM}} \sqsubseteq \mathcal{G}$ via protocol π. It is similar to the proof of the case involving $\mathcal{F}_{\text{CC}}$ for non-reactive functionalities. Without loss of generality, we assume that $\mathcal{G}$ is simply a collection of exchange functions, as in Lemma 8. The semi-honest protocol for $\mathcal{F}_{\text{OT}}$ is as follows, with Alice the sender (having inputs x_0, x_1) and Bob the receiver (having input b):

- The parties instantiate two parallel instances of π, with Alice acting as the sender. Since there is no access to an external $\mathcal{G}$, Bob will simulate Alice's interface with instances of $\mathcal{G}$— that is, Alice will send her $\mathcal{G}$ inputs directly to Bob, and he will give simulated responses from instances of $\mathcal{G}$. Alice commits to bit x_0 in the first instance, and x_1 in the second instance. Alice honestly runs π and halts after the commitment phase finishes.
- In protocol instance $(1-b)$, Bob carries out the simulation of $\mathcal{G}$-instances and the π protocol completely honestly.
- In protocol instance b, Bob honestly runs the UC simulator for π, treating Alice as the adversary (including simulating Alice's interface with $\mathcal{G}$-instances). At the end of the commitment phase, the simulator extracts Alice's bit x_b, which Bob

outputs.

By the UC security of π, Alice's view is computationally independent of b (i.e., she cannot distinguish an interaction with π's simulator from an interaction in which the receiver and $\mathcal{G}$ are honest). Bob correctly learns x_b, and we must argue that he has no advantage guessing x_{1-b}. If all $\mathcal{G}$-instances were external to the $(1-b)$ interaction as ideal functionalities, then the security of π would imply that Bob has no advantage in guessing x_{1-b} after the commitment phase. Being a collection of exchange functions, $\mathcal{G}$ has the property that Bob always learns all of Alice's inputs. Thus Alice can send her $\mathcal{G}$-inputs directly to Bob, without loss of generality. This is exactly what happens in the $(1-b)$ interaction.

C Oblivious Transfer Amplification

We first establish the following convenient technical lemma:

Lemma 15 (Noisy Channel Bounds)**.** *Consider a noisy channel $\mathcal{C}$, which either forwards an input element $x \in \mathbb{Z}_N$ unchanged with probability q, or replaces it with a uniformly chosen element from $\mathbb{Z}_N \setminus \{x\}$.*

Suppose a string $s = s_1 \dots s_k \in \mathbb{Z}_N^k$ is passed through $\mathcal{C}$, and $t = t_1 \dots t_k$ is the result. Then the probability that $\sum_{i=1}^k t_i = \sum_{i=1}^k s_i$ is at most

$$\frac{1}{N} + \exp\left(-\frac{1}{N} - \frac{(1-q)k}{(N-1)}\right).$$

Proof. Without loss of generality, suppose that $\sum_{i=1}^k s_i = 0$. Consider the following polynomial:

$$f(x) = \left(q + \frac{1-q}{N-1}x + \dots \frac{1-q}{N-1}x^{N-1}\right)^k$$

Observe that the probability that $\sum_{i=1}^k t_i = 0$ is given by the following expression:

$$\sum_{\lambda \in \mathbb{Z}} [x^{\lambda N}] f(x) = \frac{\sum_{i=0}^{n-1} f(\omega^i)}{N},$$

where $1, \omega, \dots, \omega^{N-1}$ are distinct roots of $z^N = 1$. We can evaluate the expression in the following manner:

$$\begin{aligned}
\frac{1}{N}\sum_{i=0}^{N-1} f(\omega^i) &= \frac{1}{N}\sum_{i=0}^{N-1}\left[\frac{Nq-1}{N-1} + \frac{1-q}{N-1}\sum_{j=0}^{N-1}\omega^{ij}\right]^k \\
&= \frac{1}{N} + \frac{(N-1)\left(\frac{Nq-1}{N-1}\right)^k}{N} \\
&= \frac{1}{N} + \left(1 - \frac{1}{N}\right)\left(1 - \frac{1-q}{N-1}\right)^k \\
&\leq \frac{1}{N} + \exp\left(-\frac{1}{N} - \frac{(1-q)k}{(N-1)}\right)
\end{aligned}$$

We can amplify a q-weak-OT using an algorithm taken from [13].

Definition 12 (R-Reduce)**.** *R-Reduce$(k, \mathcal{W})$ is defined as the following protocol, where $\mathcal{W}$ is a weak-OT.*

1. *Let $(x_0, x_1) \in \mathbb{Z}_N^2$ be the input of the sender; and $b \in \{0,1\}$ be the input of the receiver.*
2. *The sender generates random $(x_{0i}, x_{1i}) \in \mathbb{Z}_N^2$, for $i \in [k]$. Let $r_0 = \sum_{i=1}^k x_{0i}$ and $r_1 = \sum_{i=1}^k x_{1i}$. The sender sends $z_0 = x_0 + r_0$ and $z_1 = x_1 + r_1$ to the receiver*
3. *Both parties execute $\mathcal{W}$, k times with input $(x_{0i}, x_{1i}) \in \mathbb{Z}_N^2$ for the sender and input b for the receiver.*
4. *The receiver outputs $x_b = z_b - (\sum_{i=1}^k x_{b,i})$.*

Lemma 16. *If $\mathcal{W}$ is a q-weak-OT, then R-Reduce$(k, \mathcal{W})$ is a $(\frac{1}{N} + \nu(q,k))$-weak-OT, where:*

$$\nu(q,k) \leq \exp\left(-\frac{1}{N} - \frac{(1-q)k}{(N-1)}\right)$$

Proof. We consider the probability that the receiver can successfully guess x_{1-b}. Let $s = s_1 \dots s_k \in \mathbb{Z}_N^k$ be chosen uniformly at random.Suppose we are given a string $t_1 \dots t_k \in \mathbb{Z}_N^k$ which has the property that $t_i = s_i$ with probability q. Observe that if t_i is wrong, it adds an error $s_i - t_i$ which is uniformly random over $\mathbb{Z}_N$. So, in general with probability q it either adds 0 error; or adds a random error from the set $\mathbb{Z}_N \setminus \{0\}$ with probability $(1-q)/(N-1)$. Then, using Lemma 15, the probability that $\sum_{i=1}^k s_i = \sum_{i=1}^k t_i$ is at most:

$$\frac{1}{N} + \exp\left(-\frac{1}{N} - \frac{(1-q)k}{(N-1)}\right)$$

Thus, if $q \leq 1 - \frac{1}{\text{poly}(\kappa)}$, then R-Reduce$(\kappa/(1-q), \mathcal{W})$ is a full-fledged 1-out-of-2 OT protocol.