New Notions of Security: Achieving Universal Composability without Trusted Setup

Manoj Prabhakaran[*]
Princeton University

Amit Sahai[†]
Princeton University

June 12, 2004

Preliminary Version

Abstract

We propose a modification to the framework of Universally Composable (UC) security [3]. Our new notion, involves comparing the protocol executions with an ideal execution involving ideal functionalities (just as in UC-security), but allowing the environment and adversary access to some super-polynomial computational power. We argue the meaningfulness of the new notion, which in particular subsumes many of the traditional notions of security.

We generalize the Universal Composition theorem of [3] to the new setting. Then under new computational assumptions, we realize secure multi-party computation (for static adversaries) without a common reference string or any other set-up assumptions, in the new framework. This is known to be impossible under the UC framework.

1 Introduction

Over the last two decades, there has been tremendous success in placing cryptography on a sound theoretical foundation, and building an amazingly successful theory out of it. The key elements in this Modern Cryptographic Theory are the definitions capturing the intuitive, yet elusive notions of security in the various cryptographic settings. The definitions of early 80's proved to be extremely successful in this regard. But with time, as the theory started addressing more and more complex concerns, further notions of security had to be introduced. One of the most important concerns theory ventured into is of complex environments where the different parties are communicating with each other concurrently in many different protocols. The original definitions turned out to be inadequate to handle this. A series of efforts in extending the original definitions culminated in the paradigm of Universally Composable (UC) Security [3], which along with modeling a general complex network of parties and providing definitions of security in that model, provided powerful tools for building protocols satisfying such definitions.

The Background: Universally Composable Security The basic underlying notion of security in the UC model and its predecessors is based on *simulation*. An "ideal" world is described, where all requisite tasks get accomplished securely, as if by magic. The goal of the protocol designer is to find a way to

[*]Email: `mp@cs.princeton.edu`.

[†]Supported by grants from the Alfred P. Sloan Foundation and the NSF ITR program. Email: `sahai@cs.princeton.edu`.

accomplish these tasks in the “real” world (where magic is hard to come by) so that no malicious adversary can take advantage of this substitution of ideal magic by real protocols. To formalize this, we say that for every malicious adversary $\mathcal{A}$ that tries to take advantage of the real world, there is an adversary $\mathcal{S}$ that can achieve *essentially the same results* in the ideal world. The “results” are reflected in the behaviour of an *environment*. In this paper we shall refer to this notion of security as *“Environmental Security.”* If a real-life protocol “Environmentally Securely realizes” a task, it ensures us that replacing the magic by reality does not open up new unforeseen threats to the system. (There may already be threats to the system even in the ideal world. But employing cryptographic primitives cannot offer a solution if the ideal system itself is badly conceived.) The ideal world adversary $\mathcal{S}$ is called a *simulator* as it simulates the real world behavior of $\mathcal{A}$, in the ideal world.

The advantage of Environmentally Secure (ES) protocols, as shown in [3], is that they are “Universally Composable,” i.e., roughly, if multiple copies of an ES-protocol are present in the system (in fact they could be copies of different protocols), then they collectively ES-realize the collection of the tasks they individually ES-realize. (Hence we shall often refer to the model in [3] as the ES/UC model, or simply ES-model or UC-model.) The importance of composability is that it makes it possible to easily reason about a system running multiple ES protocols. In particular, a protocol employing multiple ES sub-protocols can be analyzed effectively. Without this, we do not have any tool to construct secure protocols for complex tasks like Multi-party computation.

Unfortunately, the notion of Environmental Security as introduced in [3] turns out to be too strong to be achievable in standard settings. It has been shown that much of the interesting cryptographic tasks (including *e.g.* commitment, zero knowledge and secure multi-party computation) *cannot* be ES-realized when the adversary can control at least half the parties [3, 4, 6]. On the other hand, under a trusted setup assumption (of questionable applicability in many situations) – that there is public reference string chosen by a completely trusted party – it is known how to build protocols for the most ambitious of cryptographic tasks (general secure multiparty computation with dishonest majority) satisfying the Environmental Security definition. Also it is known how to achieve this when the majority of the parties are honest. In this work we seek to develop such protocols in the plain model (without trusted setup), by modifying the notion of security, while still retaining composability.

This Work: New Ideas This work seeks to modify the ES/UC model, so as to achieve strongly secure and composable protocols for important cryptographic tasks, *in the plain-model*, i.e., without the common-reference string. Our starting point is the observation that in the ideal world used by the ES/UC model, even if the adversary has unlimited computational powers, the ideal world captures the notion of security in most cases of interest. Accordingly, we generalize the ES/UC model, by providing the ideal adversary with super-polynomial computational resources. However, if composability needs to be retained, we should provide the environment also with similar computational powers, which will lead us back to the strong impossibility results of [3, 4, 6]. Thus, on the face of it, we still cannot have an attainable (in plain model) environmental security notion, unless composability is abandoned (which is undesirable, because then it is not clear how to build and prove security of protocols for complex tasks). But by introducing a novel thought-experiment into the ideal world we manage to take advantage of the new model, so that we obtain universally composable secure multiparty computation protocol for any multiparty functionality.

We introduce the new notions of environmental security in two steps, starting from the notion in [3]. First is a weakening, to get a notion called **relaxed Environmental Security** (rES), followed by a strengthening to get a framework of security called **generalized Environmental Security** (gES).

Relaxed Environmental Security: Consider the ideal world version of a commitment protocol between two parties. There is a trusted third party (ideal functionality) which has secure channels with the two parties. In the commitment phase, the functionality receives a bit from the sender, and informs the receiver that it received a bit (without telling it which bit, or anything else). Later, in the reveal phase the sender can request the functionality to reveal the bit, and it will send the bit it originally received to the receiver. The receiver will accept only a bit coming from the same copy of the functionality which accepted the commitment in the first place.

Now we observe that the computational power of an adversary is irrelevant in this ideal world. This is because the security in the ideal world is information-theoretic. Indeed, in most applications, the functionality is so defined as to capture the notion of security with no reference to the power of the adversary. It is *legitimate* for a computationally unbounded adversary to interact with the honest parties in the ideal world.

This motivates the definition of relaxed ES, which is identical to that of ES, except that now the ideal world adversary is allowed to be unbounded, or say super-polynomial (depending on how much we want to relax the notion). We argue that this is a satisfactory notion of (environmental) security for most tasks of interest.

Realizability versus Composability: Allowing *the simulator to be more powerful than the real world adversary* would help us overcome the impossibility results from [3, 4, 6]. Similar motivation is behind previous works which explored super-polynomial or quasi-polynomial simulation (*e.g.* [20]) in the context of simpler compositions. However, as mentioned above, to prove the Universal Composition theorem *we do require that the environment considered be as powerful as the ideal world adversaries*. Unfortunately, if we provide *both* the ideal adversary and the environment with the same computational power, no matter how large, the impossibility results continue to hold (see later for an explanation).

We get out of this apparent deadlock using novel techniques. We introduce ways to strengthen the relaxed Environmental Security notion resulting in new notions of security, collectively called generalized Environmental Security. It is under one such notion that we give universally composable protocols for any efficiently implementable functionality.

Generalized Environmental Security: Generalized Environmental Security (gES) is a class of security definitions, which includes the original ES notion from [3]. All these definitions imply relaxed Environmental Security. Further we shall see that some of them are realizable without any trusted setup *and* imply universal composability. The cenral notion in defining gES is that of "*Imaginary Angels*." An Imaginary Angel is essentially a super-polynomial time oracle imagined to be available to the environment and adversary in the real or ideal world. (We use the name Angel to highlight that it answers queries selectively, using its (limited) knowledge about the system, so as not to hurt the honest parties.) We get different notions of security (all under the gES framework) by employing different Imaginary Angels. The security notion obtained using an Imaginary Angel Γ will be denoted by Γ-ES. Note that if Γ is a "null-angel" (which returns $\perp$ whenever queried), the notion of Γ-ES is identical to that of ES.

At this point we can sketch the results in this work:

1. For every (say) exponential time Imaginary Angel Γ, Γ-ES implies rES. (Theorem 1.)
2. For every Imaginary Angel Γ, Γ-ES protocols are universally composable (i.e., multiple Γ-ES protocols remain Γ-ES when deployed together). (Theorem 2.)
3. There exists an Imaginary Angel Ψ such that there are Ψ-ES protocols for commitment, ZK proofs and indeed *any PPT functionality* (under new complexity assumptions). (Theorem 3.)

Roughly, the Imaginary Angel Ψ will be designed so that it will answer queries which will allow breaking the security of already corrupted parties (and thus will be of good use to the ideal world adversary in carrying out the simulation), but will be unhelpful in breaking the security of the honest parties.

We stress that an Imaginary Angel, considered available to the environment and the adversary, is only for the purpose of defining and analyzing security; the actual parties in protocols do not have access to the Imaginary Angel.

Meaningfulness of the New Notion As discussed above, usually an ideal world employing the ideal functionality captures the security requirements even when the adversary has unbounded powers (and in particular, access to the Imaginary Angel Γ). This is usually the case in most interesting applications: like e-commerce or database transactions, secure communication, and genereally various multi-party computation tasks. In such cases the notion of relaxed Environmental Security (rES) is sufficient. Then, since Γ-ES is a stronger notion of security than (i.e., implies) rES, for any Imaginary Angel Γ, guaranteeing generalized Environmental Security is meaningful and sufficient.

However there may be some situations where the extra power for the adversary is not entirely "ideal"– for instance consider playing online poker against human players in the Γ-ES-model, using (in the ideal world) an ideal functionality which interacts with the players. In the ideal world the players have access to an Imaginary Angel Γ, and they may, in principle, find that useful in finding a good strategy for the game.[1] However typically an Imaginary Angel is designed to break some specific cryptographic problem (as will be apparent with the Imaginary Angel Ψ we will use in this work) and access to it is presumably not useful in a game of poker. Thus, even in many of these situations, where it is not entirely ideal to allow unlimited power to the adversary in the ideal world, the security guarantee provided by Γ-ES-model may be considered good for all practical purposes.

It is instructive to consider what the notion of rES yields in terms of the familiar notions of security. We note that under the more traditional measures of security, in many cases rES security implies security somewhat *stronger* than that implied by ES/UC-security. For instance consider CCA2 security of encryption. Any encryption scheme which rES-realizes the commitment functionality , is in fact CCA2 secure[2] even for exponential time adversaries. But on the other hand, the traditional definition of Zero Knowledge proofs and non-malleable commitments depend on simulation; so it may not be true that a protocol which rES-realizes the zero-knowledge proof functionality is a Zero-Knowledge proof under the traditional definition. Nevertheless the Witness Indistinguishable property of that protocol does get translated to (a stronger) Witness Indistinguishable property under the traditional definition (stronger, because it holds against exponential time). Similarly, for non-malleability of commitments, an indistinguishability based definition is satisfied by a protocol which rES-realizes the commitment functionality .

Avoiding the Impossibility Results It is interesting to observe how this work manages to evade the impossibility results from [3, 4, 6] (while still retaining composability). First, let us briefly recall the result showing that under the ES/UC-framework, commitment functionality cannot be securely realized in the plain model (impossibility for other functionalities are similar in spirit). Suppose, for contradiction, there is indeed such a protocol between the sender C and receiver R. The proof proceeds by considering two "real world" situations A and B. In situation A, the adversary corrupts C and directs it to act transparently between the environment and R. The environment will run an honest commitment protocol (on behalf of C), and so the receiver will accept the commitment (and later a reveal). Since the protocol is ES/UC-secure, there exists a simulator $\mathcal{S}_A$ which can effect the same commitment and reveal in the "ideal world." In other words $\mathcal{S}_A$ can *extract* the committed bit from the protocol messages (so that it can send it to the ideal commitment functionality). Now consider situation B, where the receiver R is corrupted. The contradiction is achieved by considering an adversary $\mathcal{A}_B$ which directs R to act honestly, but sends all the messages also to an internal copy of $\mathcal{S}_A$. Now $\mathcal{S}_A$ is essentially in the same position as in situation A and can extract the committed bit, from the honest sender's commitment. However this violates the security of the protocol, leading to the contradiction.

[1] If there are no human players involved, one could use an ideal functionality which requires the players to turn in their programs *a priori*, and carry out the game according to that.

[2] This follows from the fact that the ideal encryption functionality provides unconditional secrecy, and an attack in the real world translates via the simulator into an attack in the ideal world.

We note that just allowing the adversary (real and ideal) acess to more computational resources does not by itself stop the above proof from going through. $\mathcal{A}_B$ can still run $\mathcal{S}_A$ internally and violate the protocol's security, as it has the same computational powers as $\mathcal{S}_A$. So we would like to make sure that $\mathcal{A}_B$ cannot run $\mathcal{S}_A$, presumably because $\mathcal{S}_A$ has more computational powers than $\mathcal{A}_B$. But on the other hand, for the UC theorem to hold, the environment (and hence the adversary) should be able to internally run the simulators. In other words, the composition is known to hold only when the protocol is secure in environments which can be as powerful as the simulators.[3] In our work too, we use (an extension of) the UC theorem, and need to give the environment all the power of the simulator. So it would seem that we cannot prevent $\mathcal{A}_B$ from being able to run $\mathcal{S}_A$.

However, as mentioned earlier, we manage to get out of this apparent deadlock as we allow the power of the environment/simulator to *depend on the set of corrupted parties*. The key factor is that the Imaginary Angel, to which the environment/simulator have access, will base its answers to queries *on the set of corrupted parties*. Note that above, in situations A and B, the set of corrupted parties are different. This lets us make sure that $\mathcal{A}_B$ in situation B cannot run $\mathcal{S}_A$ (which expects to be in situation B), because the Imaginary Angel behaves differently in the two situations. This prevents the proof from going through. Indeed, as our results show, the new model prevents not just the proof, but also the impossibility.

Our Assumptions The protocols we construct are proven secure in the Ψ-ES model, where Ψ is a specific Imaginary Angel related to a hash function $\mathcal{H}$ that we assume[4] to exist. While our assumptions are new and therefore not standard, we believe they are quite likely to be true. (For further discussion, see next section.) As a demonstration of the plausibility of this assumption, we show how to implement $\mathcal{H}$ and Ψ in the standard UC model with Common Reference String, *assuming only that one-way functions exist*. In particular, this also shows that our protocols give rise to UC-secure protocols in the CRS model, when the hash function is instantiated according to the construction we suggest. In this sense, our protocols are ”no worse” than CRS UC protocols.

Motivations, Our Work, and Previous Work Soon after the UC framework was defined, it was observed that many important cryptographic tasks including commitment and zero knowledge, were impossible in the standard model [3, 4, 6]. Furthermore, it was recently shown that *any* model (with polynomial-time adversaries) seeking general composability in an “ideal” world / “real” world framework would suffer from the same impossibility results as the UC model [18]. Thus, if one seeks general composability in the plain setting with no setup assumptions, the definitions must be changed in some significant manner. In our Γ-ES model, where Γ is allowed to base it answers to queries on the set of corrupted parties, these impossibility results no longer hold.[5] Indeed, based on the assumptions outlined above, in Theorem 3 we show how to use the new framework to securely realize any multi-party computation with dishonest majority (for static adversaries), arguably the Holy Grail of modern cryptography, *without any set-up assumptions*. (However we do this only for the case of *static adversaries*. Extending this to adaptive adversaries is left as an open problem here.)

We stress that prior to our work, under *any* kinds of computational assumptions, in the plain model very little was known regarding composability. Essentially, all results only deal with *self-composability* of 2-party protocols, not general composability. This work started with a sequence of work on Concurrent Zero Knowledge [11, 26, 16, 22], where an arbitrary polynomial number of concurrent executions can be

[3] In particular, it can be shown that the notion of relaxed Environmental Security is not composable.

[4] We stress that our assumptions are specific computational assumptions, for which a mathematical proof or refutation could exist. We are not assuming the existence of random oracles, or any other such “mythical” object.

[5] If Γ is a fixed function (which does not depend on the set of corrupted parties), results of [3, 4, 6] will still imply impossibility of securely realizing the functionalities even if the adversary has access to Γ.

handled. For general 2-party computations, recently it was shown that in the plain model self-composition for a *bounded* number of concurrent executions can be handled [17, 21]. We stress that our result is for *general* composition of general *multi*-party computation protocols for an *unbounded* number of concurrent executions. This result was only known previously in the presence of a trusted common reference string [7].

Finally, we point out that our protocols are conceptually simpler than the corresponding ones in [7] (and of course, do not use the common reference string). We believe that the new framework will lead to considerably more efficient and intuitive protocols.

New Tools and Techniques In order to develop the multi-party computation protocol, as well as to provide a re-usable toolkit in the new model, we observe that the original Universal Composition theorem extends to our setting too. Thus, much of the convenience offered by the UC framework carries over to the Γ-ES model.

We introduce some interesting techniques on the way to developing our final protocol. We characterize the security of certain simple intermediate protocols (BCOM and BZK) in terms of non-standard functionalities that we introduce, tailor-made to suit these protocols. This is in contrast to the standard role of functionalities in the UC framework. Indeed we suggest such non-standard functionalities as a way to demonstrate some level of security and composability in natural or simple protocols, a line further explored in [24]. A somewhat similar idea appears in [5] also, in the context of secure Key-Exchange. Our non-standard functionalities are designed to capture the secrecy requirements; but the correctness requirements need to be proven separately, "stepping outside" the Γ-ES framework. We point out that this is in contrast with the treatment of correctness and secrecy requirements in the ES/UC model.

Finally, for our Ψ-ES model with Imaginary Angel Ψ, we show how to implement the Angel and related assumptions in the CRS model, assuming only one-way functions. This may be of independent interest.

Going forward with the New Model The new model of generalized Environmental Security opens up a whole range of exciting possibilities. However we point out that one needs to be careful to understand the subtleties while working in this model. Firstly, the user is required to imagine that the adversary has super-polynomial computing resources, though in reality this is not the case. When a new protocol is deployed in the system, Γ-ES-model allows it to be analyzed in the ideal world, i.e., replace all the earlier protocols by their ideal counterparts and then analyze the newly introduced protocol. However note that in the Γ-ES-model the environment and adversary have access to the Imaginary Angel Γ. Ignoring this fact may leave the new protocol open to vulnerabilities as it is deployed along with the other Γ-ES-protocols. The recommended (and the provably secure) way is to model every task as an ideal functionality and use a Γ-ES-realization to carry it out.

Note that in the Γ-ES-model, Γ is a single Imaginary Angel that defines the security model. If a protocol is shown secure in the Γ'-ES-model for another Imaginary Angel Γ', it may not compose with a Γ-ES-protocol. We point out that this is usually not a big problem because the specific nature of the Imaginary Angel will be used only for basic primitives and all other functionalities are built on top of it. For instance, in this work we use an Imaginary Angel Ψ to realize a basic commitment functionality, which the other protocols build on. However it is the case that computational assumptions will typically need to be made relative to the Imaginary Angel. So it is desirable to have a standard Imaginary Angel model (or at most a few), relative to which the usual assumptions (one-way functions, trap-door permutations etc.) are well studied.

Though candidates for our current assumptions may be instantiated by using some popular cryptographic hash function used in practice, the assumptions we make about them are non-standard. The main problem left open by this work is to use more standard and better studied assumptions. Indeed, it will be interesting to come up with entirely new constructions and Imaginary Angels, for which the corresponding assumptions are better understood. Another possibility is to use *"complexity leveraging"* techniques to reduce some of the assumptions to more standard ones. See Section 2.3 for a discussion.

2 Preliminaries

Notation For two distributions $\mathcal{X}$ and $\mathcal{Y}$ with security parameter k, we write $\mathcal{X} \approx \mathcal{Y}$ to mean that $\mathcal{X}$ and $\mathcal{Y}$ are indistinguishable by probabilistic polynomial (in k) size circuits. We denote the distribution $f(\mathcal{X})$ by the set notation $\{f(x)|x \leftarrow \mathcal{X}\}$.

2.1 Relaxed Environmental Security

The model for the defintion of relaxed Environmental Security (rES) is the same as the ES/UC model in [3] (see below for details on the model), except we do not require that the ideal world adversaries be PPT. The definition of security below allows the ideal world adversary to be super-polynomial. The other entities are implicitly assumed to be PPT (though the definition of security does not inherently require so). The REAL, IDEAL and HYBRID executions are defined exactly as in [3][6]. Also, the distributions of the environment's output ($\text{REAL}_{\pi,\mathcal{A},\mathcal{Z}}$ and $\text{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}$ below) are defined as in [3].

Definition 1 *A protocol π is said to rES-realize the functionality $\mathcal{F}$ against the class $\mathcal{C}$ of adversaries relaxed to the class $\mathcal{C}^*$ of ideal adversaries if $\forall \mathcal{A} \in \mathcal{C}$, $\exists \mathcal{S} \in \mathcal{C}^*$ such that $\forall \mathcal{Z}$, $\text{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}} \approx \text{REAL}_{\pi,\mathcal{A},\mathcal{Z}}$.*

2.2 Generalized Environmental Security: The Γ-ES Model

The Γ-ES model is the same as the ES/UC model in [3], except that the adversary and the environment are given access to an "Imaginary Angel" Γ. This is the case in the real, ideal and hybrid executions, as defined in the ES/UC model (see below). We stress, however, that **all protocols and honest parties** *are still polynomial-time, without any Imaginary Angels*. The Imaginary Angel is merely a means of defining and analyzing security. We allow the Imaginary Angel to base it answers on the set of corrupted parties. An Imaginary Angel Γ takes in a query q and returns an answer $\Gamma(q, \mathfrak{X})$, where $\mathfrak{X}$ is the set of corrupted parties at the time the query is made. We point out that there is a single Imaginary Angel Γ throughout the Γ-ES model.

Real, Ideal and Hybrid executions with an Imaginary Angel We define REAL^Γ execution (with parties $P_1, \ldots, P_n$ running protocol π, an adversary $\mathcal{A}^\Gamma$, and an environment $\mathcal{Z}^\Gamma$) just like the REAL execution in [3] except that now the adversary $\mathcal{A}^\Gamma$ and environment $\mathcal{Z}^\Gamma$ have access to the Imaginary Angel Γ, which they may query any number of times. Analogous to $\text{REAL}_{\pi,\mathcal{A},\mathcal{Z}}$ in [3], we define $\text{REAL}^\Gamma_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}^\Gamma}$ as the distribution ensemble (one distribution for each choice of security parameter and input to the parties) on the output produced by $\mathcal{Z}^\Gamma$ on interacting with the parties running protocol π and the adversary $\mathcal{A}^\Gamma$.

Similarly the IDEAL^Γ execution is defined exactly like the IDEAL execution in [3], except that the environment $\mathcal{Z}^\Gamma$ and the ideal-execution adversary $\mathcal{S}^\Gamma$ have access to the Imaginary Angel Γ. Analogous to $\text{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}}$, we define $\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma}$ as the distribution ensemble of the output of $\mathcal{Z}^\Gamma$ on interacting with the "dummy" parties, the ideal functionality $\mathcal{F}$ and the ideal adversary (simulator) $\mathcal{S}^\Gamma$.

Finally, the HYB^Γ execution is defined as the hybrid execution in [3], except that the environment $\mathcal{Z}^\Gamma$ and the hybrid-execution adversary $\mathcal{H}^\Gamma$ have access to the Imaginary Angel Γ. Analogous to $\text{HYB}^{\mathcal{F}}_{\pi,\mathcal{H},\mathcal{Z}}$,

[6]Figures 1., 2. and 3. in [3]

$\text{HYB}^{\Gamma,\mathcal{F}}_{\pi,\mathcal{H}^\Gamma,\mathcal{Z}^\Gamma}$ denotes the distribution ensemble on the output of $\mathcal{Z}^\Gamma$ on interacting with the parties running protocol π in the $\mathcal{F}$-hybrid model (with multiple copies of $\mathcal{F}$) and the hybrid-execution adversary $\mathcal{H}^\Gamma$. [7]

Note that above, if Γ is a polynomial time computable function (in particular if it is a trivial function which returns $\perp$ on all input), then the modified model is *identical* to the original ES/UC model.

Definition 2 *A protocol π is said to Γ-ES-realize the functionality $\mathcal{F}$ against the class $\mathcal{C}$ of adversaries. if $\forall \mathcal{A}^\Gamma \in \mathcal{C}$, $\exists \mathcal{S}^\Gamma \in \mathcal{C}$ such that $\forall \mathcal{Z}^\Gamma$, $\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma} \approx \text{REAL}^\Gamma_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}^\Gamma}$.*

The first thing we point out about this definition (which depends on the particular Imaginary Angel Γ) is that it *implies* relaxed Environmental Security.

Theorem 1 *Let $\mathcal{C}$ be a class of adversaries. Let $\mathcal{C}' \supseteq \mathcal{C}$ be a class of adversaries with access to some Imaginary Angel Γ and $\mathcal{C}^*$ the class of adversaries obtained from adversaries in $\mathcal{C}'$ by replacing oracle access to Γ by a (super-polynomial time) machine which implements Γ. Then, if a protocol π Γ-ES-realizes the functionality $\mathcal{F}$ against $\mathcal{C}'$ then π rES-realizes the functionality $\mathcal{F}$ against $\mathcal{C}$ relaxed to $\mathcal{C}^*$.*

Proof: This is a simple consequence of the definitions. If π Γ-ES-realizes the functionality $\mathcal{F}$ against $\mathcal{C}'$, then $\forall \mathcal{A}^\Gamma \in \mathcal{C}'$, $\exists \mathcal{S}^\Gamma \in \mathcal{C}'$ such that $\forall \mathcal{Z}^\Gamma$, $\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma} \approx \text{REAL}^\Gamma_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}^\Gamma}$. In particular $\forall \mathcal{A} \in \mathcal{C}$, $\exists \mathcal{S}^\Gamma \in \mathcal{C}'$ such that $\forall \mathcal{Z}$, $\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma} \approx \text{REAL}^\Gamma_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}^\Gamma}$ (by simply restricting the universally quantified adversaries to $\mathcal{C} \subseteq \mathcal{C}'$ and environments to those not accessing Γ). Now for each $\mathcal{S}^\Gamma \in \mathcal{C}'$ there is an $\mathcal{S}^* \in \mathcal{C}^*$ which replaces the oracle calls to Γ by (a super-polynomial time) computation. Further $ideal_{\mathcal{F},\mathcal{S}^*,\mathcal{Z}}$ is identical to $\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma}$. Clearly $\text{REAL}^\Gamma_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}^\Gamma}$is identical to $\text{REAL}_{\pi,\mathcal{A},\mathcal{Z}}$. Thus we get that $\forall \mathcal{A} \in \mathcal{C}$, $\exists \mathcal{S}^* \in \mathcal{C}^*$ such that $\forall \mathcal{Z}$, $\text{IDEAL}_{\mathcal{F},\mathcal{S},\mathcal{Z}} \approx \text{REAL}_{\pi,\mathcal{A},\mathcal{Z}}$. Thus, by definition, π rES-realizes the functionality $\mathcal{F}$ against $\mathcal{C}$ relaxed to $\mathcal{C}^*$. $\square$

The following is a restatement of the UC theorem in [3], where we replace the REAL and HYBRID executions by REAL^Γ and HYB^Γ executions respectively. The theorem holds for adaptive adversaries as well. The proof (as well as an extension to the *specialized simulator* case and a simple generalization of the setting) appears in Appendix B.

Theorem 2 (Extended Universal Composition Theorem) *Let $\mathcal{C}$ be a class of adversaries with access to the Imaginary Angel Γ, and $\mathcal{F}$ be an ideal functionality. Let π be an n-party protocol in the $\mathcal{F}$-hybrid model and let ρ be an n-party protocol that Γ-ES-realizes $\mathcal{F}$ against adversaries of class $\mathcal{C}$. Then, $\forall \mathcal{A}^\Gamma \in \mathcal{C}$, $\exists$ (a hybrid-model adversary) $\mathcal{H}^\Gamma \in \mathcal{C}$ such that $\forall \mathcal{Z}^\Gamma$ we have:*

$$\text{REAL}^\Gamma_{\pi^\rho,\mathcal{A}^\Gamma,\mathcal{Z}^\Gamma} \approx \text{HYB}^{\Gamma,\mathcal{F}}_{\pi,\mathcal{H}^\Gamma,\mathcal{Z}^\Gamma}$$

All the parties are assumed to have unique IDs, but possibly chosen adversarially. Like in previous works on Universally Composable multi-party computation, we work in the authenticated channels model.

2.3 The Hash Function, the Imaginary Angel and the Assumptions

In this work, we use hash functions with concrete assumptions. Below we sketch the assumptions we use in this work.

We assume a hash function $\mathcal{H} : \{0,1\}^k \to \{0,1\}^\ell$, with the following properties: The k-bit input to $\mathcal{H}$ is considered to be an element $(\mu, r, x, b) \in \mathfrak{I} \times \{0,1\}^{k_1} \times \{0,1\}^{k_2} \times \{0,1\}$, where $\mathfrak{I}$ is the set of IDs used for the parties, and k_1, k_2, ℓ are all polynomially related to k. Then,

[7]By abuse of notation, sometimes we will use $\text{REAL}^\Gamma_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}^\Gamma}$, $\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma}$and $\text{HYB}^{\Gamma,\mathcal{F}}_{\pi,\mathcal{H}^\Gamma,\mathcal{Z}^\Gamma}$ to denote (the distribution of) the entire view of the environment $\mathcal{Z}^\Gamma$, instead of (the distribution of) just its output.

A1 (Collisions and Indistinguishability): For every $\mu \in \mathcal{I}$ and $r \in \{0,1\}^{k_1}$, there is a distribution $\mathcal{D}_r^\mu$ over $\{(x,y,z)|\mathcal{H}(\mu,r,x,0) = \mathcal{H}(\mu,r,y,1) = z\} \neq \phi$, such that

$$\{(x,z)|(x,y,z) \leftarrow \mathcal{D}_r^\mu\} \approx \{(x,z)|x \leftarrow \{0,1\}^{k_2}, z = \mathcal{H}(\mu,r,x,0)\}$$
$$\{(y,z)|(x,y,z) \leftarrow \mathcal{D}_r^\mu\} \approx \{(y,z)|y \leftarrow \{0,1\}^{k_2}, z = \mathcal{H}(\mu,r,y,1)\}$$

Further, even if the distinguisher is given sampling access to the set of distributions $\{\mathcal{D}_{r'}^{\mu'}|\mu' \in \mathcal{I}, r' \in \{0,1\}^{k_1}\}$, these distributions still remain indistinguishable.

A2 (Difficult to find collisions with same prefix): For all PPT circuits M and every id $\mu \in \mathcal{I}$, for a random $r \leftarrow \{0,1\}^{k_1}$, probability that $M(r)$ outputs (x,y) such that $\mathcal{H}(\mu,r,x,0) = \mathcal{H}(\mu,r,y,1)$ is negligible. This remains true even when M is given sampling access to the set of distributions $\{\mathcal{D}_{r'}^{\mu'}|\mu' \neq \mu, r' \in \{0,1\}^{k_1}\}$.

The first assumption simply states that there are collisions in the hash function, which are indistinguishable from a random hash of 0 or 1. Note that this assumption implies that for every $\mu \in \mathcal{I}$ and every $r \in \{0,1\}^{k_1}$ $\mathcal{H}(\mu,r,\{0,1\}^{k_2},0)$ and $\mathcal{H}(\mu,r,\{0,1\}^{k_2},1)$ are indistinguishable (because they are indistinguishable from $\{z|(x,y,z) \leftarrow \mathcal{D}_r^\mu\}$).

We make one more cryptographic assumption for our constructions:

A3 There exists a family of trapdoor permutations $\mathcal{T}$ over $\{0,1\}^n$, which remains secure even if the adversary has sampling access to $\mathcal{D}_r^\mu$ for all μ and r.

We use the notation $(f, f^{-1}) \leftarrow \mathcal{T}$ to specify generating a permutation $f : \{0,1\}^n \rightarrow \{0,1\}^n$ and its inverse (trapdoor) f^{-1}. We let $B(\cdot)$ denote a hardcore predicate associated with this permutation, which retains its security even if the adversary has sampling access to $\mathcal{D}_r^\mu$ for all μ and r, (for instance, it is easy to see that the Goldreich-Levin bit [14] continues to be a hardcore predicate as required, under Assumption A3). We will also need a perfectly binding (non-interactive) commitment scheme C, whose hiding property (in a stand-alone setting) holds against PPT adversaries with access to the distributions $\mathcal{D}_r^\mu$ for all μ and r. C can be readily constructed from $\mathcal{T}$ and B.

Plausibility of Our Assumptions. Our set of assumptions on our hash function essentially give it the nature of a kind of non-malleable commitment (NMC). We make several observations here. NMC in the standard model is something that has been known to exist for over a decade [10], and recently even constant-round NMC has been realized in the standard model [1]. Further work on realizing simple NMC under standard complexity assumptions remains an important and exciting research area, partly because, tantalizingly, NMC is essentially something we know most functions satisfy (in the sense that a random oracle realizes it immediately), and it is something we expect any "sufficiently unstructured" hash function (such as something like SHA) to satisfy; indeed we know that just one-way functions suffice to implement NMC in the CRS model [9, 8]. We make these observations to highlight two points: First, assuming that some hash function has NMC-like properties is not at all unreasonable. Second, since NMC is already known to exist, but known NMC protocols do not (and indeed *cannot*) yield the results we want, what we are doing is not just trivial given NMC – *i.e.* we are not making an assumption which "obviously" implies the goal we want to achieve.

As further evidence of the plausibility of our assumptions, we show that our hash functions, our assumptions, and the Imaginary Angel below can be realized in the CRS model *assuming only that one-way functions exist*; see Section 7. The fact that only one-way functions are needed to realize our assumptions in the CRS model gives further evidence that Assumption A3 is valid, since it is about *trapdoor* primitives, as opposed to merely *one-way* primitives.

Complexity Leveraging to Reduce Assumptions By choosing parameters appropriately, at least one of our assumptions can be reduced to a more standard one. Specifically, Assumption A3, which assumes trapdoor permutations secure against adversaries with sampling access to $\mathcal{D}_r^{\mu}$ can be replaced with an assumption of trapdoor permutations secure against super-polynomial adversaries.

Consider choosing the domain of the trapdoor permutation $\{0,1\}^n$ such that the input size of the hash function $k = n^{\epsilon}$, for some constant $0 < \epsilon < 1$. Then we can safely replace Assumption A3 by the assumption that the trapdoor permutations are secure against circuits of size $2^{n^{\epsilon}}$. This implies Assumption A3 (given Assumptions A1 and A2) because a circuit of size $2^k = 2^{n^{\epsilon}}$ can represent the distributions $\mathcal{D}_r^{\mu}$ for all (μ, r). Note that this is only to change the assumption to a more standard one (trapdoor permutations secure against sub-exponential circuits), and has no effect on the model. In particular, we are not changing the power of the real or ideal adversaries.

The Imaginary Angel Ψ Suppose $\mathfrak{X}$ is the set of corrupted parties. (Since we are dealing with static adversaries, this is a fixed set). On query (μ, r) the Imaginary Angel Ψ checks if $\mu \in \mathfrak{X}$, i.e., if the party with ID μ is corrupted or not. If it is, Ψ draws a sample from $\mathcal{D}_r^{\mu}$ described above and returns it; else it returns $\perp$. The results in this work are in the Ψ-ES-model.

2.4 Conventions

We point out a few conventions we follow in this work. All parties and functionalities referred to in the Γ-ES-model are (uniform or non-uniform) probabilistic polynomial time machines. Adversaries and environments are *non-uniform* PPT machines. The functionalities do not have access to any information about the system other than what the honest parties would have– in particular, a functionality would not know the set of corrupted parties. (In [7] such functionalities are referred to as "well-formed.")

When we say a protocol Γ-ES-realizes a functionality against static adversaries, we require that it be a *non-trivial* protocol (as defined in [7]): i.e., if the real world adversary corrupts no parties and forwards all messages promptly, the ideal world adversary (simulating the real-world execution with the protocol) is required do the same.

The following restrictions of the class of adversaries are standard. A *static* adversary can corrupt the parties only at the onset of computation. A *semi-honest* (or passive) adversary has read-only access to the internal state of the corrupted parties, but cannot modify the program run by the parties.

The following notation is also standard: if Π is a protocol in the $\mathcal{F}$-hybrid model (with Imaginary Angel Γ) and π is a protocol which securely realizes $\mathcal{F}$ (with respect to Γ) in $\mathcal{F}'$-hybrid model, then the protocol Π^{π} is a protocol in the $\mathcal{F}'$-hybrid model obtained from Π by replacing interaction with $\mathcal{F}$ by interaction with programs implementing the protocol π.

3 Secure Multi-Party Computation in the Ψ-ES Model

In this section we present our main result: for any multi-party computation (MPC) functionality $\mathcal{F}$, a protocol which Ψ-ES-realizes $\mathcal{F}$ against *static* adversaries. The overall structure of our Secure multi-party computation protocol follows that in [7], which in turn follows [15, 13]. But we differ from [7] in a very crucial manner: we introduce basic tools and protocols which allow us to achieve security (in the Ψ-ES model), *without a Common Random String*.

3.1 One-to-many Commit-and-prove

Following [7], first we construct a protocol which Ψ-ES-realizes $\mathcal{F}_{\mathrm{CP}}^{1:\mathrm{M}}$ against static adversaries, where $\mathcal{F}_{\mathrm{CP}}^{1:\mathrm{M}}$ is the "One-to-Many Commit-and-Prove" functionality shown in Figure 8 (see Section 6).

Lemma 1 *Under assumptions A1, A2 and A3, there is a protocol* OM-CP *which Ψ-ES-realizes $\mathcal{F}_{\mathrm{CP}}^{1:M}$ against static adversaries.*

Lemma 1 contains the central contribution of this work. In Sections 4 and 5 we build tools for proving it, and in Section 6 we give the proof.

3.2 MPC from Commit-and-prove

Given Lemma 1, the rest of the construction closely follows that in [7]. First we begin with a protocol which Ψ-ES-realizes $\mathcal{F}$ against static *semi-honest* adversaries. A semi-honest adversary is one which does not alter the behaviour of the parties it corrupts (see [7] for more details). Then using Lemma 1 we construct a *protocol compiler* which can take a protocol secure against semi-honest (static) adversaries and generates a protocol secure against general (static) adversaries, thereby completing the proof. These two steps are further elaborated below. For full details we refer the reader to [7].

MPC for Semi-Honest Parties In general, all the proofs for the semi-honest case from [7] are information-theoretic, and immediately imply their Ψ-ES analogs. First, we observe that the Oblivious Transfer functionality (denoted by $\mathcal{F}_{\mathrm{OT}}$) is realized by the same protocol as in [15, 13, 7]. The proof as given in [7] that the protocol securely realizes $\mathcal{F}_{\mathrm{OT}}$ with respect to semi-honest static adversaries carries over directly to the Ψ-ES model, under Assumption A3.

This allows us to work in the $\mathcal{F}_{\mathrm{OT}}$-hybrid model. Again, the protocols for semi-honest parties, in the $\mathcal{F}_{\mathrm{OT}}$-hybrid model carry over exactly as they are given in [7]. As observed there, there is no assumption on the computational power of the adversary and environment in the proof of security (under the $\mathcal{F}_{\mathrm{OT}}$-hybrid model). Thus, using the secure realization of $\mathcal{F}_{\mathrm{OT}}$ with respect to Ψ above, we get a secure multi-party computation protocol for semi-honest parties in the Ψ-ES model.

From the above, we conclude following:

Lemma 2 (Following [7]): *Under Assumption A3, for any multi-party functionality $\mathcal{F}$, there exists a protocol which Ψ-ES-realizes $\mathcal{F}$ against* semi-honest *static adversaries.*

Protocol Compiler As mentioned above, to complete the construction, we need to show how to convert the above protocol for semi-honest parties into one secure against malicious parties. We note that the compiler given in [7] under the $\mathcal{F}_{\mathrm{CP}}^{1:\mathrm{M}}$-hybrid model works in the Ψ-ES model as well. The proof in [7] that this compiler works in the $\mathcal{F}_{\mathrm{CP}}^{1:\mathrm{M}}$-hybrid model is information-theoretic, and holds for all classes of adversaries and environments; hence it is easily verified that the proof carries over to the Ψ-ES model.

Lemma 3 (Following [7]): *There exists a protocol compiler* Comp *which takes a multi-party protocol Π, and outputs a protocol* $\mathsf{Comp}(\Pi)$ *in the $\mathcal{F}_{\mathrm{CP}}^{1:M}$-hybrid model such that, for every protocol Π and static adversary $\mathcal{A}^{\Psi}$, there exists a* semi-honest *static adversary ${\mathcal{A}'}^{\Psi}$ such that for every environment $\mathcal{Z}^{\Psi}$,*

$$\mathrm{REAL}^{\Psi}_{\Pi,{\mathcal{A}'}^{\Psi},\mathcal{Z}^{\Psi}} \equiv \mathrm{HYB}^{\Psi,\mathcal{F}_{\mathrm{CP}}^{1:M}}_{\mathsf{Comp}(\Pi),\mathcal{A}^{\Psi},\mathcal{Z}^{\Psi}}$$

Our main theorem readily follows.

Theorem 3 *Under assumptions A1, A2 and A3, there is a protocol which Ψ-ES-realizes any multi-party functionality*[8] *against static adversaries.*

PROOF (SKETCH). Consider any multi-party functionality $\mathcal{F}$. By Lemma 2 there is a protocol Π which Ψ-ES-realizes $\mathcal{F}$ against semi-honest static adversaries. Applying Lemma 3, we obtain a protocol $\Pi' = \mathsf{Comp}(\Pi)$ in the $\mathcal{F}_{\text{CP}}^{1:\text{M}}$-hybrid model against (possibly malicious) static adversaries. Finally using Lemma 1 and the composition theorem, we get that $\Pi'^{\text{OM-CP}}$ Ψ-ES-realizes $\mathcal{F}$ against (possibly malicious) static adversaries. $\Box$

The rest of the paper (till Section 7) is devoted to proving Lemma 1.

4 Basic Building Blocks

In this section, we build the basic functionalities we need to achieve the result of secure (static) multi-party computation in the Ψ-ES model. Because we are not availing of any common reference string, our path is a bit more complicated than it would be otherwise. We introduce a new modeling and proof technique based on intermediate non-standard functionalities. In some cases, to establish our results, we need to "step outside" the Ψ-ES framework, because our intermediate functionalities do not fully capture the security properties we need from our protocols for their later application. This section develops all the tools we'll need to realize the *commitment functionality* in the Ψ-ES model, which we'll do in the next section.

A note about session-ID's. In the UC framework, and similarly in our Ψ-ES framework, every functionality should be instantiated with a unique *session-ID* in order to distinguish it from other instantiations. This is an important part of the modeling, but it can be distracting in (often already complicated) protocols and functionality specifications. For sake of ease of reading, we omit session-ID's from our description, but they are implicit[9]. When we need to specify the session-ID, we use the notation $\mathcal{F}[sid]$ to denote a copy of the functionality $\mathcal{F}$ invoked with session-ID sid. Messages to and from $\mathcal{F}[sid]$ are tagged with sid. Every protocol invocation also has an associated unique session-ID sid. Again, the parties would have agreed on sid before the protocol starts. All messages in the protocol are tagged by sid, and when a party receives a message tagged by sid it is forwarded to the program running the protocol with that session-ID. Agreeing upon the session-ID is not part of the functionality or protocol: while describing the protocol or functionality we leave out specifying how the session-IDs are agreed upon.

But when an ideal functionality $\mathcal{F}$ is invoked from *within* a protocol π (or another functionality, as the case may be) that we describe, as part of the desciption of π we need to specify how the session-ID of that invocation of $\mathcal{F}$ is decided. In general, one can simply pass on the same session-ID as of π, *annotated* with a unique identifier that lets us refer to the functionality being called. For instance, if a protocol π with session-ID sid, uses n copies of the functionality $\mathcal{F}$, it will instantiate those copies as $\mathcal{F}[(sid_1)]$, $\mathcal{F}[(sid_2)]$, $\ldots$ $\mathcal{F}[(sid_n)]$, where sid_i could simply be sid concatenated with i. In Section 4.2 we illustrate this convention by explicitly incorporating this in the specification of the protocol BCOM* (Figure 2).

4.1 Basic Commitment Protocol

In Figure 1(a) we give a protocol BCOM for commitment, in the $\mathcal{F}_{\text{ENC}}$-hybrid. $\mathcal{F}_{\text{ENC}}$ is the encryption functionality, which receives a message from a party and delivers it to the destination party, publishing the length of the message to the adversary.

[8] see Section 2.4.

[9] Because there is no "joint state" represented by a CRS, we are in the lucky and relatively simple situation of only having to associate a *single* session-ID to each functionality (as opposed to a session-ID and a *sub-session-ID*). So almost all of the "complications" of dealing with multiple session-ID's that arise in [7] do not arise for us. This is one reason we feel comfortable omitting them from the protocol description, to avoid clutter.

Protocol BCOM

The parties are a sender or committer C, and a receiver R. The security parameter is k, and k_1, k_2 are polynomial in k. The sender C gets as input a bit b, which it wants to commit to.

COMMIT PHASE:

1. R picks $r \leftarrow \{0,1\}^{k_1}$ and sends it to C.
2. C chooses $r' \leftarrow \{0,1\}^{k_2}$ and computes $c = \mathcal{H}(\mu_R, r, r', b)$. C requests $\mathcal{F}_{\text{ENC}}$ to send c to R.
3. R receives c from $\mathcal{F}_{\text{ENC}}$ and accepts the commitment.

REVEAL PHASE:

1. C requests $\mathcal{F}_{\text{ENC}}$ to send (b, r') to R, which the receiver R receives.
2. R checks if $\mathcal{H}(\mu_R, r, r', b) = c$. If so it accepts b as revealed.

(a) The Basic Commitment Protocol (BCOM)

Functionality $\mathcal{F}_{\widetilde{\text{COM}}}$

The parties are sender C and receiver R, with an adversary $\mathcal{S}^{\Psi}$. The security parameter is k, and k_1, k_2 are polynomial in k.

COMMIT PHASE:

1. $\mathcal{F}_{\widetilde{\text{COM}}}$ picks $r \leftarrow \{0,1\}^{k_1}$ and sends it to C.
2. $\mathcal{F}_{\widetilde{\text{COM}}}$ receives c from C.
3. $\mathcal{F}_{\widetilde{\text{COM}}}$ sends the message COMMIT to R

REVEAL PHASE:

1. $\mathcal{F}_{\widetilde{\text{COM}}}$ receives (b, r') from C
2. $\mathcal{F}_{\widetilde{\text{COM}}}$ checks if $\mathcal{H}(\mu_R, r, r', b) = c$. If so it sends the message (REVEAL, b) to R and the adversary $\mathcal{S}^{\Psi}$.

(b) A functionality realized by the protocol BCOM

Figure 1: The Basic Commitment Protocol and a Functionality it realizes.

We will use protocol BCOM as a component in later protocols. Thus we would like to show some sort of composable security for this protocol. But note that this protocol cannot be a Ψ-ES secure commitment protocol (in particular, it does not provide a way for a simulator to *extract* the values committed to by a corrupted sender). So we introduce a novel technique to formalize and analyze the security of this protocol.

Lemma 4 *Protocol* BCOM *Ψ-ES-realizes $\mathcal{F}_{\widetilde{\text{COM}}}$ shown in Figure 1(b) against static adversaries, in the $\mathcal{F}_{\text{ENC}}$-hybrid model.*

PROOF (SKETCH). For every PPT adversary $\mathcal{A}^{\Psi}$ we demonstrate a PPT simulator $\mathcal{S}^{\Psi}$ such that no PPT environment $\mathcal{Z}^{\Psi}$ can distinguish between interacting with the parties and $\mathcal{A}^{\Psi}$ in the REAL$^{\Psi}$ world, and interacting with the parties and $\mathcal{S}^{\Psi}$ in the IDEAL$^{\Psi}$ world.

$\mathcal{S}^{\Psi}$ internally runs $\mathcal{A}^{\Psi}$ (which expects to work in the $\mathcal{F}_{\text{ENC}}$-hybrid with the parties running the BCOM protocol), and works as an interface between $\mathcal{A}^{\Psi}$ and the parties. When $\mathcal{A}^{\Psi}$ starts the BCOM protocol, $\mathcal{S}^{\Psi}$

initiates a session with the IDEAL functionality. If $\mathcal{A}^\Psi$ corrupts both parties, $\mathcal{S}^\Psi$ allows it to directly interact with them. Below we consider the other three possible cases.

Both C, R honest If $\mathcal{A}^\Psi$ corrupts neither of the two parties C and R, then all it sees are the random string r from R to C, and the message from $\mathcal{F}_{\text{ENC}}$ giving the length of the commit and reveal messages from C. So by knowing the parameters k_1, k_2, ℓ (which we assume it does) $\mathcal{S}^\Psi$ can perfectly simulate the protocol to $\mathcal{A}^\Psi$. (Encryption is used in the protocol specifically to take care of the situation where the adversary corrupts neither parties.)

R honest, C corrupted Suppose the adversary corrupts only the sender C. Note that there is very little difference between what C sees in the real and ideal executions. In fact $\mathcal{S}^\Psi$ simply forwards the messages from $\mathcal{A}^\Psi$ (meant for $\mathcal{F}_{\text{ENC}}$, to be delivered to R) to $\mathcal{F}_{\widetilde{\text{COM}}}$ (as if it was indeed sent to R via $\mathcal{F}_{\text{ENC}}$), and reports to $\mathcal{A}^\Psi$ the message from $\mathcal{F}_{\widetilde{\text{COM}}}$ (as if it came from R). It is easily verified that this is a good simulation.

C honest, R corrupted Finally, suppose that the adversary corrupts the receiver alone. When $\mathcal{A}^\Psi$ sends out the first message r in the protocol, $\mathcal{S}^\Psi$ sends a query (μ_R, r) to the Imaginary Angel Ψ and (since R is corrupted), receives $(x, y, z) \leftarrow \mathcal{D}_r^{\mu_R}$, where $\mathcal{D}_r^{\mu_R}$ is the distribution over $\{(x, y, z)|\mathcal{H}(\mu_R, r, x, 0) = \mathcal{H}(\mu_R, r, y, 1) = z\}$ as specified in the assumption on $\mathcal{H}$. Then, when $\mathcal{F}_{\widetilde{\text{COM}}}$ gives the COMMIT message, $\mathcal{S}^\Psi$ sends z to $\mathcal{A}^\Psi$ as a message from the REAL sender. Later if $\mathcal{F}_{\widetilde{\text{COM}}}$ gives the message $(\text{REVEAL}, 0)$, then $\mathcal{S}^\Psi$ sends $(0, x)$ to $\mathcal{A}^\Psi$, and if $\mathcal{F}_{\widetilde{\text{COM}}}$ gives the message $(\text{REVEAL}, 1)$, then $\mathcal{S}^\Psi$ sends $(1, y)$ to $\mathcal{A}^\Psi$. Under the assumption A1 on $\mathcal{D}_r^\mu$, we have that $\mathcal{Z}^\Psi$ cannot distinguish between the real execution and the simulation. $\square$

A priori the functionality $\mathcal{F}_{\widetilde{\text{COM}}}$ does not offer any guarantee that the commitment is binding on a corrupt sender. The following lemma formulates the binding property outside the Ψ-ES-framework (i.e., we do not give a functionality reflecting the binding property).

Lemma 5 *Consider a copy of $\mathcal{F}_{\widetilde{\text{COM}}}$ interacting with a corrupt sender C and an honest receiver R, in a system with environment $\mathcal{Z}^\Psi$ and multiple other copies of the same or other functionalities as well as one or more protocols and adversary $\mathcal{A}^\Psi$. Then, after finishing the commit phase, there is a fixed bit b^* (determined by the entire system state), such that C can make $\mathcal{F}_{\widetilde{\text{COM}}}$ accept a reveal to $1 - b^*$ with only negligible probability.*

Proof: We define the *value* of the commitment b^* as follows: consider the entire system at the end of the commitment phase. Let p_0 be the probability of the sender (legally) revealing this commitment as 0, and the probability p_1 of the sender revealing it as 1. Let $b^* = 0$ if $p_0 \geq p_1$; else let $b^* = 1$. We say that the binding is broken if the sender manages to reveal the commitment to $1 - b^*$. We shall demonstrate a (non-uniform) PPT machine M^Ψ which accepts $r \leftarrow \{0,1\}^k$ and outputs (x, y) such that $\mathcal{H}(r, x, 0) = \mathcal{H}(r, y, 1)$, with a probability polynomially related to the probability of the sender breaking the binding.

M^Ψ simulates the system internally, starting at the point the session is initiated (which is given to it non-uniformly). Recall that in this session the (corrupted) sender is to interact with $\mathcal{F}_{\widetilde{\text{COM}}}$, which chooses a random string $r \leftarrow \{0,1\}^k$ and sends it to the sender. But instead, M^Ψ will accept r as an input and send that as the first message to the sender. Then the sender may respond with a string c. At this point M^Ψ makes two copies of the system, and runs them with independent randomness. If the sender eventually reveals the commitment as $(x, 0)$ in the first run and as $(y, 1)$ in the second run, then M^Ψ outputs (x, y) and *succeeds*. Else it fails and terminates.

Let σ denote the state of the system at the point M^Ψ makes a copy of the system. Define random variable p_0^σ (respectively, p_1^σ) as the probability that starting from the state σ, the sender successfully reveals the commitment as 0 (respectively, 1). Then,

$$\begin{aligned}\mathbf{Pr}\left[\text{Sender reveals } 1-b^*\right] &\leq \mathbf{E}_\sigma[\min\{p_0^\sigma, p_1^\sigma\}] \\ \leq \mathbf{E}_\sigma[\sqrt{p_0^\sigma p_1^\sigma}] \leq \sqrt{\mathbf{E}_\sigma[p_0^\sigma p_1^\sigma]} &= \sqrt{\mathbf{Pr}\left[M^\Psi \text{ succeeds}\right]}\end{aligned}$$

because after forking two copies of the system, M^Ψ succeeds (i.e., it manages to output (x, y) such that $\mathcal{H}(r, x, 0) = \mathcal{H}(r, y, 1)$) when in the first run the event with probability p_0^σ occurs and in the second the event with probability p_1^σ. Here $\mathbf{E}_\sigma[f(\sigma)]$ stands for $\sum_\sigma f(\sigma)\,\mathbf{Pr}\left[\sigma\right]$. Note that the randomness involved in determining $\mathbf{Pr}\left[\sigma\right]$ includes all the randomness used by M^Ψ to simulate the system up to the point σ, and the input r that it receives.

Since the running time of M^Ψ is linearly related to the running time of the entire system, M^Ψ is a PPT machine. Hence the probability that M^Ψ succeeds is negligible by assumption A2 (even though M^Ψ has access to Ψ, as long as R is not corrupted). So the probability that the sender reveals $1-b^*$ is also negligible. $\square$

4.2 Multi-bit Commitment with Selective Reveal.

We define a multi-bit version of the functionality $\mathcal{F}_{\widetilde{\text{COM}}}$, called $\mathcal{F}^*_{\widetilde{\text{COM}}}$.

$\mathcal{F}^*_{\widetilde{\text{COM}}}$ is equivalent to n parallel sessions of $\mathcal{F}_{\widetilde{\text{COM}}}$ (where n is specified by the sender C). $\mathcal{F}^*_{\widetilde{\text{COM}}}$ internally runs n copies of the program for $\mathcal{F}_{\widetilde{\text{COM}}}$ and interacts with the sender C according to them. In the commit phase, it sends COMMIT if all n parallel copies of $\mathcal{F}_{\widetilde{\text{COM}}}$ output COMMIT (as the message for R). In the reveal phase, the sender can choose to run the reveal phase of any subset $\{i_1, \ldots, i_t\} \subseteq [n]$ of parallel sessions. Then if in reveal phase of each of the t chosen sessions, $\mathcal{F}_{\widetilde{\text{COM}}}$ of that session outputs the reveal message (REVEAL, b_j) (intended for the receiver), then $\mathcal{F}^*_{\widetilde{\text{COM}}}$ sends the message $(\text{REVEAL}, (i_1, b_1), \ldots, (i_t, b_t))$ to R.

Recall our convention regarding session-IDs: when a protocol is started from within another protocol, the latter should specify how the session ID of the sub-protocol is generated and agreed upon. We make explicit our conventions regarding how this is done, by specifying the details for the functionality $\mathcal{F}^*_{\widetilde{\text{COM}}}$ and a protocol BCOM* for it, in Figure 2. Similar schemes can be employed for all other protocols in this work.

Let BCOM$^{*\text{BCOM}}$ denote the protocol in the $\mathcal{F}_{\text{ENC}}$-hybrid model which is obtained from BCOM* by replacing $\mathcal{F}_{\widetilde{\text{COM}}}$ invocations by the protocol BCOM. Then it is easy to show the following lemmas. The first follows from the fact that Lemma 5 holds in a general setting, and from a union bound. The second follows from the composition theorem (Theorem 2). We omit the proofs.

Lemma 6 *In a setting as in Lemma 5, after finishing the commit phase with $\mathcal{F}^*_{\widetilde{\text{COM}}}$, there is a fixed string in $\{0,1\}^n$ (where n is the number of bits as specified by C at the beginning of the protocol) such that C can make $\mathcal{F}^*_{\widetilde{\text{COM}}}$ accept a (selective) reveal inconsistent with that string with only negligible probability.*

Lemma 7 *There is a protocol which Ψ-ES-realizes $\mathcal{F}^*_{\widetilde{\text{COM}}}$ against static adversaries, in the $\mathcal{F}_{\text{ENC}}$-hybrid model.*

4.3 Basic Zero Knowledge Proof

Consider a proto-typical 3-round Zero Knowledge Proof protocol (a Σ-protocol) for proving membership in an NP-complete language (like 3-colorability or Hamiltonicity), in which the prover uses the basic commitment functionality $\mathcal{F}_{\widetilde{\text{COM}}}$ from above, to carry out the commitments (first round) and the reveals (last round).

Functionality $\mathcal{F}^*_{\widetilde{\text{COM}}}$

The parties are sender C and receiver R, with an adversary $\mathcal{S}^\Psi$. Let the session ID (already agreed up on by C and R, on instantiating this copy of $\mathcal{F}^*_{\widetilde{\text{COM}}}$) be sid. Let the input to C at the commit phase be $(b_1, \ldots, b_n)$, the bits to which it wishes to commit, and that at the reveal phase be $\{i_1, \ldots, i_t\} \subseteq [n]$, the set of commitments which it wishes to reveal.

COMMIT PHASE:

1. $\mathcal{F}^*_{\widetilde{\text{COM}}}$ receives the number n from C.
2. $\mathcal{F}^*_{\widetilde{\text{COM}}}$ internally starts n copies of the program for $\mathcal{F}_{\widetilde{\text{COM}}}$ namely $\mathcal{F}_{\widetilde{\text{COM}}}[(sid_1)], \ldots, \mathcal{F}_{\widetilde{\text{COM}}}[(sid_n)]$, with C as the sender and using the ID of R as the receiver-ID.
3. $\mathcal{F}^*_{\widetilde{\text{COM}}}$ lets the t parallel sessions of $\mathcal{F}_{\widetilde{\text{COM}}}$ interact with C in the commit phase.
4. If all the n copies of $\mathcal{F}_{\widetilde{\text{COM}}}$ produce the COMMIT message, then $\mathcal{F}^*_{\widetilde{\text{COM}}}$ sends the message COMMIT to R and $\mathcal{S}^\Psi$.

REVEAL PHASE:

1. $\mathcal{F}^*_{\widetilde{\text{COM}}}$ receives the set $\{i_1, \ldots, i_t\}$ from C.
2. $\mathcal{F}^*_{\widetilde{\text{COM}}}$ lets the t parallel sessions $\mathcal{F}_{\widetilde{\text{COM}}}[(sid_{i_1})], \ldots, \mathcal{F}_{\widetilde{\text{COM}}}[(sid_{i_t})]$ interact with C in the reveal phase.
3. On receiving message (REVEAL, b_i) from the session of $\mathcal{F}_{\widetilde{\text{COM}}}$ with session-ID (sid_i), for all $i \in \{i_1, \ldots, i_t\}$, $\mathcal{F}^*_{\widetilde{\text{COM}}}$ sends $(\text{REVEAL}, (i_1, b_{i_1}), \ldots, (i_t, b_{i_t}))$ to R and $\mathcal{S}^\Psi$.

(a) Multi-bit Commitment With Selective Reveal Functionality

Protocol BCOM*

The parties are sender C and receiver R. Let the session ID (already agreed up on by C and R, on instantiating this copy of BCOM*) be sid. Let the input to C at the commit phase be $(b_1, \ldots, b_n)$, the bits to which it wishes to commit, and that at the reveal phase be $\{i_1, \ldots, i_t\} \subseteq [n]$, the set of commitments which it wishes to reveal.

COMMIT PHASE:

1. C sends the number n to R.
2. C and R initiate n parallel sessions of $\mathcal{F}_{\widetilde{\text{COM}}}$ with session-IDs $(sid_1), \ldots, (sid_n)$.
3. C interacts (in parallel) with the n sessions of $\mathcal{F}_{\widetilde{\text{COM}}}$ to commit to the bits $b_1, \ldots, b_n$.
4. On receiving COMMIT message from all the n sessions, R accepts the commitment.

REVEAL PHASE:

1. C sends the set $\{i_1, \ldots, i_t\}$ to R.
2. C interacts in parallel with the sessions of $\mathcal{F}_{\widetilde{\text{COM}}}$ with session-IDs $(sid_{i_1}), \ldots, (sid_{i_t})$ to reveal to the bits $b_{i_1}, \ldots, b_{i_t}$.
3. On receiving message (REVEAL, b_i) from the session with session-ID (sid_i), for all $i \in \{i_1, \ldots, i_t\}$, R accepts $(i_1, b_{i_1}), \ldots, (i_t, b_{i_t})$ as the revealed information.

(b) Protocol BCOM* in the $\mathcal{F}_{\widetilde{\text{COM}}}$-hybrid model

Figure 2: The Basic Multi-bit Commitment with Selective-reveal Functionality $\mathcal{F}^*_{\widetilde{\text{COM}}}$ and the protocol BCOM*.

Let us denote this protocol by BZK. Then, like we defined $\mathcal{F}_{\widetilde{\text{COM}}}$ from BCOM, we can define a basic Zero Knowledge Proof functionality $\mathcal{F}_{\widetilde{\text{ZK}}}$ from BZK. The description of the functionality is simple: $\mathcal{F}_{\widetilde{\text{ZK}}}$ interacts with the prover according to the protocol BZK, playing the verifiers role. If the prover completes the proof according to the protocol, $\mathcal{F}_{\widetilde{\text{ZK}}}$ sends a message PROVEN to the verifier.

Figure 3(a) gives the protocol BZK, and Figure 3(b) gives the functionality $\mathcal{F}_{\widetilde{\text{ZK}}}$, for proving 3-colorability. Note that both BZK and $\mathcal{F}_{\widetilde{\text{ZK}}}$ are defined in the $\mathcal{F}^*_{\widetilde{\text{COM}}}$-hybrid model. (See section B.1.)

Protocol BZK

The parties are prover P and verifier V. The common input is a graph $G([n], E)$. In addition, the prover P gets a witness $\sigma : [n] \to \{1,2,3\}$. The size of the graph n is polynomial in the security parameter k. P verifies that σ is a valid coloring of G. (Else it aborts the protocol.)

1. Repeat the following n^3 times in parallel:
   (a) P chooses π, a random permutation of $\{1,2,3\}$. Let $b_i = \pi(\sigma(i))$.
   (b) P interacts with $\mathcal{F}^*_{\widetilde{\text{COM}}}$ to commit n (2-bit) numbers $(b_1, \ldots, b_n)$ to V.
   (c) V picks a random edge $(i,j) \leftarrow E$ and sends it to P.
   (d) P sends $(\text{REVEAL}, \{i,j\})$ to $\mathcal{F}^*_{\widetilde{\text{COM}}}$, and interacts with it in the reveal phase to reveal $(i,b_i),(j,b_j))$. V receives $(\text{REVEAL}, (i,b_i),(j,b_j))$ from $\mathcal{F}^*_{\widetilde{\text{COM}}}$.
2. V checks if in all n^3 runs $b_i, b_j \in \{1,2,3\}$ and $b_i \neq b_j$. If so it accepts the proof.

(a) Basic ZK Proof Protocol (BZK) in the $\mathcal{F}^*_{\widetilde{\text{COM}}}$-hybrid model

Functionality $\mathcal{F}_{\widetilde{\text{ZK}}}$

The parties are prover P and verifier V, with an adversary $\mathcal{S}^\Psi$. The common input is a graph $G([n], E)$. The size of the graph n is polynomial in the security parameter k.

1. Repeat the following n^3 times in parallel:
   (a) $\mathcal{F}_{\widetilde{\text{ZK}}}$ interacts with $\mathcal{F}^*_{\widetilde{\text{COM}}}$ playing the part of the receiver (using V's ID) with P as sender.
   (b) When $\mathcal{F}^*_{\widetilde{\text{COM}}}$ returns the COMMIT message $\mathcal{F}_{\widetilde{\text{ZK}}}$ picks a random edge $(i,j) \leftarrow E$ and sends it to P.
   (c) $\mathcal{F}_{\widetilde{\text{ZK}}}$ continues interacting with P through $\mathcal{F}^*_{\widetilde{\text{COM}}}$, until $\mathcal{F}^*_{\widetilde{\text{COM}}}$ returns a message $(\text{REVEAL}, (i,b_i),(j,b_j))$.
2. $\mathcal{F}_{\widetilde{\text{ZK}}}$ checks if in all n^3 runs $b_i \neq b_j$. If so it sends the message ACCEPTABLE to V and $\mathcal{S}^\Psi$.

(b) A functionality realized by the BZK protocol

Figure 3: The Basic ZK Proof Protocol and a Functionality it realizes.

Lemma 8 *Protocol* BZK *Ψ-ES-realizes $\mathcal{F}_{\widetilde{\text{ZK}}}$ against static adversaries (where* BZK *and $\mathcal{F}_{\widetilde{\text{ZK}}}$ are both in the $\mathcal{F}^*_{\widetilde{\text{COM}}}$-hybrid model).*

Proof: We need to show that for every PPT adversary $\mathcal{A}^\Psi$ there is a PPT simulator $\mathcal{S}^\Psi$ such that no PPT environment $\mathcal{Z}^\Psi$ can distinguish between interacting with the parties and $\mathcal{A}^\Psi$ in the HYB^Ψ world, and interacting with the parties and $\mathcal{S}^{\Psi}$ in the IDEAL$^{\Psi}$ world.

$\mathcal{S}^{\Psi}$ internally runs $\mathcal{A}^{\Psi}$ (which expects to work in the $\mathcal{F}_{\text{ENC}}$-hybrid with two parties P, V running a BZK protocol), and works as an interface between $\mathcal{A}^{\Psi}$ and the parties. When $\mathcal{A}^{\Psi}$ starts the BZK protocol, $\mathcal{S}^{\Psi}$ initiates a session with the IDEAL functionality. If $\mathcal{A}^{\Psi}$ corrupts both parties $\mathcal{S}^{\Psi}$ allows it to directly interact with them. Below we consider the other three possible cases (the first two will be handled the same way as in the proof of security of BCOM).

Both P, V honest: Note that in the protocol BZK the only message sent by V is a random edge of G, where as messages sent by P are all to the functionality $\mathcal{F}^*_{\widetilde{\text{COM}}}$ (whose contents are hidden from $\mathcal{A}^{\Psi}$). Apart from this all that $\mathcal{A}^{\Psi}$ sees are the COMMIT and REVEAL messages sent out by $\mathcal{F}^*_{\widetilde{\text{COM}}}$, the latter being to pairs $b, b' \leftarrow \{1, 2, 3\}, b \neq b'$. Clearly $\mathcal{S}^{\Psi}$ can perfectly simulate these messages.

V honest, P corrupted: In this case, $\mathcal{S}^{\Psi}$ simply forwards the messages from $\mathcal{A}^{\Psi}$ to $\mathcal{F}^*_{\widetilde{\text{COM}}}$, and reports to $\mathcal{A}^{\Psi}$ the messages from $\mathcal{F}_{\widetilde{\text{ZK}}}$ as if it is from V.

P honest, V corrupted: Finally, suppose that the adversary corrupts the verifier. Now $\mathcal{S}^{\Psi}$ sends COMMIT messages to the corrupted V as if coming from $\mathcal{F}^*_{\widetilde{\text{COM}}}$. If V responds with an $(i, j) \in E$, $\mathcal{S}^{\Psi}$ picks $b, b' \leftarrow \{1, 2, 3\}, b \neq b'$ and sends them in a REVEAL message. Clearly this is a perfect simulation of an interaction with the honest prover P and $\mathcal{F}^*_{\widetilde{\text{COM}}}$. □

The functionality $\mathcal{F}_{\widetilde{\text{ZK}}}$ does not make any guarantees of soundness, *a priori*. But as with $\mathcal{F}_{\widetilde{\text{COM}}}$, we shall demonstrate this property outside the UC/ES-framework.

Lemma 9 *Consider a corrupt prover P interacting with a copy of $\mathcal{F}_{\widetilde{\text{ZK}}}$ and an honest verifier V, in a system with environment $\mathcal{Z}^{\Psi}$ and multiple other copies of the same or other functionalities as well as one or more protocols (which can all be w.l.o.g considered part of the environment) and adversary $\mathcal{A}^{\Psi}$. Then $\mathcal{F}_{\widetilde{\text{ZK}}}$ accepts the proof to a false statement with negligible probability.*

Proof: In each of the n^3 repetitions of the commitment in the interaction of $\mathcal{F}_{\widetilde{\text{ZK}}}$ with P, consider the point at which $\mathcal{F}^*_{\widetilde{\text{COM}}}$ sends the COMMIT message to $\mathcal{F}_{\widetilde{\text{ZK}}}$. Then, by the binding property of $\mathcal{F}^*_{\widetilde{\text{COM}}}$ (Lemma 6), we know that there exist values $(b_1^*, \ldots, b_n^*)$ such that the probability that $\mathcal{F}^*_{\widetilde{\text{COM}}}$ will send a reveal message with (i, b_i) for $b_i \neq b_i^*$ is negligible.

For convenience, we define the following events: Unsound is the event that $\mathcal{F}_{\widetilde{\text{ZK}}}$ accepts the proof of an incorrect statement. AllBadGraphs is the event that in all the n^3 sessions, bits $(b_1^*, \ldots, b_n^*)$ give invalid colorings; BadCom is the event that $\mathcal{F}^*_{\widetilde{\text{COM}}}$ will send a reveal message with (i, b_i) where $b_i \neq b_i^*$. AllGoodEdges is the event that in each of the n^3 sessions, for the edge (i, j) selected by $\mathcal{F}_{\widetilde{\text{ZK}}}$, $b_i^* \neq b_j^*$. Then

$$\begin{aligned}
\mathbf{Pr}[\text{Unsound}] &\leq \mathbf{Pr}[\text{AllBadGraphs} \wedge \mathcal{F}_{\widetilde{\text{ZK}}} \text{ sends 3-COLORABLE}] \\
&\leq \mathbf{Pr}[\text{AllBadGraphs} \wedge (\text{AllGoodEdges} \vee \text{BadCom})] \\
&\leq \mathbf{Pr}[(\text{AllBadGraphs} \wedge \text{AllGoodEdges}) \vee \text{BadCom}] \\
&\leq \mathbf{Pr}[\text{AllBadGraphs} \wedge \text{AllGoodEdges}] + \mathbf{Pr}[\text{BadCom}] \\
&\leq \mathbf{Pr}[\text{AllGoodEdges} | \text{AllBadGraphs}] + \mathbf{Pr}[\text{BadCom}]
\end{aligned}$$

Now conditional on AllBadGraphs, in each session, $\mathcal{F}_{\widetilde{\text{ZK}}}$ will query an edge without valid colorings (as given by $b_1^*, \ldots, b_n^*$) with probability at least $\frac{1}{|E|} \geq \frac{1}{n^2}$. That is, with probability at most $1 - \frac{1}{n^2}$ it will query an edge with a valid coloring (a good edge). So,

$$\mathbf{Pr}[\text{AllGoodEdges} | \text{AllBadGraphs}] \leq (1 - 1/n^2)^{n^3} = 2^{-\Omega(n)}$$

Since $\mathbf{Pr}[\text{BadCom}]$ is also negligible, we conclude that $\mathbf{Pr}[\text{Unsound}]$ is negligible. □

5 Commitment

The basic protocols and non-standard functionalities given in the previous section now allow us to achieve the "fully" ideal Ψ-ES commitment functionality $\mathcal{F}_{\text{COM}}$ given below. Since for the sake of simplicity in describing our protocols we allowed the session IDs to be implicit, we do the same in specifying the functionality.

Functionality $\mathcal{F}_{\text{COM}}$

The parties are a sender C and a receiver R, with adversary $\mathcal{S}^{\Psi}$.

COMMIT PHASE:

$$\begin{aligned} C \rightarrow \mathcal{F}_{\text{COM}} &: \quad b \\ \mathcal{F}_{\text{COM}} \rightarrow R &: \quad \text{COMMIT} \end{aligned}$$

REVEAL PHASE:

$$\begin{aligned} C \rightarrow \mathcal{F}_{\text{COM}} &: \quad \text{REVEAL} \\ \mathcal{F}_{\text{COM}} \rightarrow R, \mathcal{S}^{\Psi} &: \quad (\text{REVEAL}, b) \end{aligned}$$

Figure 4: The Commitment Functionality

Let C be a perfectly binding commitment scheme. Let $\mathcal{T}_k$ be a family of trapdoor-permutations (f, f^{-1}) on $\{0,1\}^k$, which can be efficiently sampled (but f is a one-way-permutation by itself). B stands for a hardcore predicate for the family of trapdoor permutations used. These primitives are assumed to be secure against adversaries with access to Ψ (see Section 2.3). The protocol is based on the commit-with-extract protocol from [2].

Protocol COM

The parties are a sender C and a receiver R. k is the security parameter.

COMMIT PHASE:

1. R draws $r_R \leftarrow \{0,1\}^k$ and sends $c = \mathsf{C}(r_R; r)$ where r is also drawn at random.
2. C draws $(f, f^{-1}) \leftarrow \mathcal{T}_k$ and sends f to R. C interacts with $\mathcal{F}_{\widetilde{\text{ZK}}}$ to prove to R that $(\exists r', g : (f, g) \leftarrow \mathcal{T}_k(r'))$. R receives the message PROVEN from $\mathcal{F}_{\widetilde{\text{ZK}}}$.
3. C draws $r_C \leftarrow \{0,1\}^k$ and sends it to R.
4. R sends r_R to C.
5. R interacts with $\mathcal{F}_{\widetilde{\text{ZK}}}$ to prove to C that $(\exists r' : c = \mathsf{C}(r_R; r'))$ C receives the message PROVEN from $\mathcal{F}_{\widetilde{\text{ZK}}}$.
6. Let b be the bit C wants to commit to. C compute $b' = B(f^{-1}(r_R \oplus r_C)) \oplus b$ and sends b' to R. R accepts the commitment.

REVEAL PHASE:

1. C sends the bit b to R.
2. C interacts with $\mathcal{F}_{\widetilde{\text{ZK}}}$ to prove to R that $(\exists t : f(t) = r_R \oplus r_C \bigwedge b' = B(t) \oplus b)$.
3. Up on receiving the message PROVEN from $\mathcal{F}_{\widetilde{\text{ZK}}}$, R accepts b as the revealed bit.

Figure 5: Protocol COM which Ψ-ES-realizes $\mathcal{F}_{\text{COM}}$ against static adversaries

Theorem 4 *Protocol* COM *Ψ-ES-realizes $\mathcal{F}_{\text{COM}}$ against static adversaries, in the $\mathcal{F}_{\widetilde{\text{ZK}}}$-hybrid model.*

PROOF (SKETCH). Given an adversary $\mathcal{A}^{\Psi}$, we need to construct a simulator $\mathcal{S}^{\Psi}$ such that for all environments $\mathcal{Z}^{\Psi}$, we have $\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^{\Psi},\mathcal{Z}^{\Psi}} \approx \text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^{\Psi},\mathcal{Z}^{\Psi}}$. As is usual, $\mathcal{S}^{\Psi}$ internally simulates $\mathcal{A}^{\Psi}$. If both the sender C and receiver R running the protocol COM are corrupted, $\mathcal{S}^{\Psi}$ lets $\mathcal{A}^{\Psi}$ interact with them directly. We briefly discuss the other three cases below.

When both C and R are honest, $\mathcal{S}^{\Psi}$ simulates the protocol exactly until the step where the bit b is used (Step 7). At this step, it sends out a random bit as b'. In the reveal phase $\mathcal{S}^{\Psi}$ can easily simulate a proof from $\mathcal{F}_{\widetilde{\text{ZK}}}$ to open it either way. The hiding property of the hard-core bit B can be used to show that the simulation is indistinguishable from an actual execution. When R is corrupt and C is honest, the same simulator works, for the same reasons. However the reduction to the security of B becomes slightly more involved in this case.

When R is honest and C corrupt, $\mathcal{S}^{\Psi}$ should be able to *extract* the committed bit. The idea here is that $\mathcal{S}^{\Psi}$ (playing the part of R in the protocol) will cheat in the proof using the simulated $\mathcal{F}_{\widetilde{\text{ZK}}}$ in Step 5 (reveal phase of the coin-flipping part), and have $r_R \oplus r_C$ match a random string r such that it knows $B(r_R \oplus r_C)$. This will allow it to extract the bit b. Soundness of $\mathcal{F}_{\widetilde{\text{ZK}}}$ (Lemma 9) ensures that C cannot feasibly open to a bit other than b. Also it ensures that f is indeed a permutation, which along with the hiding property of the commitment C ensures that the simulation is indistinguishable from an actual execution.

The full proof is somewhat tedious. See Appendix C □

Corollary 5 *Under assumptions A1, A2 and A3, there is a protocol which Ψ-ES-realizes $\mathcal{F}_{\text{COM}}$ against static adversaries.*

PROOF (SKETCH). Employing the composition theorem (Theorem 2) to compose protocols in Theorem 4, and Lemmas 4 and 8, we conclude that there is a protocol in the $\mathcal{F}_{\text{ENC}}$-hybrid model which Ψ-ES-realizes $\mathcal{F}_{\text{COM}}$ against static adversaries. So to complete the proof we need to specify how to Ψ-ES-realize $\mathcal{F}_{\text{ENC}}$ against static adversaries. For this we use the same protocol as in [3], namely a CCA2-secure encryption with the receiving party generating the public-key/secret-key pair afresh for each session. But since we are working in the Ψ-ES model, we need the CCA2-secure encryption scheme to remain secure even when the adversary has access to Ψ. This can be accomplished based on assumption A3, by using any CCA2-secure encryption based on trapdoor permutations, for instance the one from [25]. □

6 One-to-Many Commit-and-Prove

In this section we outline the proof of Lemma 1, which completes the proof of our main theorem- Theorem 3. As in [7], we use two other functionalities, namely Zero-Knowledge ($\mathcal{F}_{\text{ZK}}$) and Authenticated Broadcast ($\mathcal{F}_{\text{BC}}$).

Ψ-ES **Zero Knowledge** Canetti and Fischlin [4] show how to UC-securely realize $\mathcal{F}_{\text{ZK}}$ in the $\mathcal{F}_{\text{COM}}$-hybrid model, in an information-theoretic sense: that is, without any computational assumptions, or restrictions on the class of adversaries. It is easy to show that the same protocol Ψ-ES-realizes $\mathcal{F}_{\text{ZK}}$ against static adversaries, in the $\mathcal{F}_{\text{COM}}$-hybrid model. Since we have already shown how to Ψ-ES-realize $\mathcal{F}_{\text{COM}}$ against static adversaries (Theorem 4), from the composition theorem, Theorem 2, it follows that there is a protocol which Ψ-ES-realizes $\mathcal{F}_{\text{ZK}}$ against static adversaries.

Functionality $\mathcal{F}_{\text{ZK}}$

$\mathcal{F}_{\text{ZK}}$ proceeds as follows, running with a prover P, a verifier V and an adversary $\mathcal{S}^\Psi$, and parameterized with a relation R:

- Upon receiving (PROVE, x, w) from P, do: if $(x, w) \in R$, then send (PROVEN, x) to V and $\mathcal{S}^\Psi$ and halt. Otherwise, halt.

Figure 6: $\mathcal{F}_{\text{ZK}}$ functionality

Functionality $\mathcal{F}_{\text{BC}}$

$\mathcal{F}_{\text{BC}}$ proceeds as follows, running with parties $P_1, \dots, P_n$ and an adversary $\mathcal{S}^\Psi$:

- Upon receiving a message $(\mathcal{P}, x)$ from P_i, where $\mathcal{P}$ is a set of parties, send $(P_i, \mathcal{P}, x)$ to all parties in $\mathcal{P}$ and to $\mathcal{S}^\Psi$, and halt.

Figure 7: The ideal broadcast functionality

Authenticated Broadcast The functionality $\mathcal{F}_{\text{BC}}$ ensures that all the parties to which a message is addressed receive the same message (if they do receive the message). Following [7], we use the protocol from [12]. The protocol in [12] securely realizes $\mathcal{F}_{\text{BC}}$ in an information-theoretic manner: it does not require any computational restrictions on the class of adversaries. Thus, in particular, this protocol Ψ-ES-realizes $\mathcal{F}_{\text{BC}}$ against static adversaries.

Functionality $\mathcal{F}_{\text{CP}}^{\mathbf{1:M}}$

The parties are a sender C and a set of possible receivers $P_1, \dots, P_n$, with an adversary $\mathcal{S}^\Psi$. The functionality is parameterized by a relation R. The security parameter is k.
COMMIT PHASE

- Upon receiving a message $(\text{COMMIT}, \mathcal{P}, w)$ from C where $\mathcal{P}$ is a set of parties and $w \in \{0,1\}^k$, append the value w to the list $\overline{w}$, record $\mathcal{P}$, and send the message $(\text{RECEIPT}, C, \mathcal{P})$ to all parties $P \in \mathcal{P}$ and to $\mathcal{S}^\Psi$. (Initially, the list $\overline{w}$ is empty). But, if a COMMIT message has already been received with a different set of parties $\mathcal{P}' \neq \mathcal{P}$ ignore this message.

PROVE PHASE

- Upon receiving a message (PROVE, x) from C, where $x \in \{0,1\}^{\text{poly}(k)}$, compute $R(x, \overline{w})$. If $R(x, \overline{w}) = 1$, then send the message (PROVEN, x) to all parties $P_i \in \mathcal{P}$ and to $\mathcal{S}^\Psi$. Otherwise, ignore the message.

Figure 8: The One-to-many commit-and-prove functionality

The proof of Lemma 1 easily follows from the following lemma and the observations above, using the composition theorem.

Lemma 10 *There is a protocol which Ψ-ES-realizes $\mathcal{F}_{\text{CP}}^{1:M}$ against static adversaries in the ($\mathcal{F}_{\text{BC}}$,$\mathcal{F}_{\text{ZK}}$)-hybrid model (under Assumption A3).*

PROOF (SKETCH).
A protocol (in the $(\mathcal{F}_{\text{BC}}, \mathcal{F}_{\text{ZK}})$-hybrid model) is shown in Figure 9.

To commit to a value w, the sender C computes a commitment c to w under a perfectly binding commitment C obtained from the trapdoor-permutation of Assumption A3 (which remains hiding even to adver-

Protocol OM-CP

The parties are a sender C and a set of possible receivers $P_1, \ldots, P_n$, with an adversary $\mathcal{S}^{\Psi}$. The protocol is parameterized by a relation R. The security parameter is k.

COMMIT PHASE

1. On input $(\text{COMMIT}, \mathcal{P}, w)$, the sender C computes $c = \mathsf{C}(w; r)$ using a randomly chosen r. It adds r to a list $\overline{r}$ (initially empty).
2. C broadcasts the message $(\mathcal{P}, c)$ by sending it to $\mathcal{F}_{\text{BC}}$.
3. For each $P_j \in \mathcal{P}$, party C sends the message $(\text{PROVE}, c, (w, r))$ to a copy of $\mathcal{F}_{\text{ZK}}$ invoked with P_j as the verifier, and parametrized by the relation $R' = \{(c, (w, r)) \mid c = \mathsf{C}(w; r)\}$.
4. On receiving $(C, \mathcal{P}, c)$ from $\mathcal{F}_{\text{BC}}$ and (PROVEN, c) from $\mathcal{F}_{\text{ZK}}$, (after verifying that both values of c are the same) each $P_j \in \mathcal{P}$ broadcasts the message ACCEPTABLE.
5. On receiving the message ACCEPTABLE from all parties in $\mathcal{P}$, party P_j adds c to a list $\overline{c}$ (initially $\overline{c}$ is empty; C also maintains this list).

PROVE PHASE

1. On input (PROVE, x), the C broadcasts the message $(\mathcal{P}, x)$ by sending it to $\mathcal{F}_{\text{BC}}$.
2. For each $P_j \in \mathcal{P}$, party C sends the message $(\text{PROVE}, (x, \overline{c}), (\overline{w}, \overline{r}))$ to a copy of $\mathcal{F}_{\text{ZK}}$ invoked with P_j as the verifier and parametrized by the relation $R'' = \{((x, \overline{c}), (\overline{w}, \overline{r})) \mid R(x, \overline{w}) \bigwedge \overline{c}_j = \mathsf{C}(\overline{w}_j; \overline{r}_j) \text{ for all } j\}$
3. On receiving $(\mathcal{P}, x)$ from $\mathcal{F}_{\text{BC}}$ and $(\text{PROVEN}, (x, \overline{c}))$ from $\mathcal{F}_{\text{ZK}}$, (after verifying that both x are the same, and $\overline{c}$ and $\mathcal{P}$ match the locally stored values) each $P_j \in \mathcal{P}$ broadcasts the message TRUE (x).
4. On receiving the message TRUE (x) from all parties in $\mathcal{P}$, party P_j accepts x.

Figure 9: A protocol which Ψ-ES-realizes $\mathcal{F}_{\text{CP}}^{\text{1:M}}$ against static adversaries in the $(\mathcal{F}_{\text{BC}}, \mathcal{F}_{\text{ZK}})$-hybrid model

saries with sampling access to $\mathcal{D}_r^{\mu}$ for all μ and r). Then it broadcasts c and proves to each party separately, using the $\mathcal{F}_{\text{ZK}}$ functionality, that c is indeed a valid commitment. Each party on receiving this proof broadcasts this fact. If all parties accept the respective proofs and announce it, they all proceed to accept the commitment by adding c to a list $\overline{c}$. Later, to prove $R(x, \overline{w})$, where x is an input and $\overline{w}$ is the list of all commitments made so far, the C proofs the statement (formulated in terms of x and $\overline{c}$) to each party separately using the $\mathcal{F}_{\text{ZK}}$ functionality. As before, on accepting the proof, each party broadcasts this fact. Finally they all accept the proof if all parties complete this broadcast step. It easily follows from the security of the commitment scheme C that this protocol Ψ-ES-realizes $\mathcal{F}_{\text{CP}}^{\text{1:M}}$ against static adversaries. $\square$

7 A CRS implementation of $\mathcal{H}$ and Ψ

We show how to use the common reference string (CRS) model to implement a hash function $\mathcal{H}$ such that assumptions we need do hold, and in the IDEAL world, the simulator $\mathcal{S}$ can implement the Imaginary Angel Ψ by itself. Thus the entire construction we have given here can be used to obtain an alternate implementation of the commitment functionality $\mathcal{F}_{\text{COM}}$ in the standard UC paradigm (i.e., no Imaginary Angels to define the IDEAL world) in the CRS model. Our construction uses ideas from [8] and [7].

The CRS We assume a secure digital signature scheme [27]. Generate a pair of signing and verifying keys (SK, VK). The CRS consists of the verification key VK. The algorithms of signing and verifying be sign and verify respectively.

The hash function $\mathcal{H}$ Recall that $\mathcal{H}$ takes four inputs: (μ, r, x, b). We define $\mathcal{H}$ step by step:

- Let $G([n], E)$ be the graph describing an instance of the HAMILTONICITY problem, obtained by reducing the statement $(\exists\sigma : \mathsf{verify}(VK, \mu, \sigma) = 1)$ (where the inputs to verify are the key, the message and the signature in that order). That is, a Hamiltonian cycle in G can be converted to a signature σ of μ verifiable under VK, and vice-versa.

- Let PRG be a pseudorandom generator. Following [19], let C_r be a bit commitment scheme defined as follows $\mathsf{C}_r(0) = \mathsf{PRG}(s)$ and $\mathsf{C}_r(1) = \mathsf{PRG}(s) \oplus r$, for a random seed s to the PRG. Note that if PRG's output is much longer than its input (say $\mathsf{PRG} : \{0,1\}^{k_1/4} \to \{0,1\}^{k_1}$) then for all but negligible fraction of choices of r, C_r is a perfectly binding scheme (because the set $\{r : r = \mathsf{PRG}(s_1) \oplus \mathsf{PRG}(s_2)\}$ has at most $2^{k_1/2}$ elements).

- Consider a second (equivocable) commitment scheme Q_r as follows. Pick a random permutation of the vertices of G, π. Let $A_{\pi(G)}$ be the adjacency matrix of $\pi(G)$, the graph G with vertices permuted according to π. Then $Q_r(0) = M^{(0)}$ where $M^{(0)}$ is a commitment to $A_{\pi(G)}$: $M^{(0)}_{ij} = \mathsf{C}_r(A_{\pi(G)_{ij}}; s_{ij})$. To compute $Q_r(1)$, pick a random cycle on n vertices. Define matrix $M^{(1)}$ as follows: if an edge (i, j) is in the cycle $M^{(1)}_{ij} = \mathsf{C}_r(1)$, else set $M^{(1)}_{ij}$ to be random string from $\{0,1\}^{k_1}$. Then $Q_r(1) = M^{(1)}$. Let $k_2 = poly(k_1)$ be an upperbound on the total number of random bits it takes to compute $M^{(b)}$.

- Let $\mathcal{H}(\mu, r, x, b) = M^{(b)}$ where x is used as the random tape in computing $M^{(b)}$.

Implementing the Imaginary Angel Ψ In the CRS model, the simulator can choose (SK, VK) and set VK as the CRS. The Imaginary Angel can be implemented using the secret key SK. On input (μ, r), Ψ checks if $\mu \in \mathfrak{X}$, the set of corrupted IDs (which the simulator can keep track of). If not it just outputs $\perp$. If μ is corrupted Ψ proceeds as follows: use SK to produce a signature $\sigma = \mathsf{sign}(SK, \mu)$. Use σ to find a Hamiltonian cycle in G. It computes $z = Q_r(0)$ using a random tape x. Now it goes on to construct a random tape y: consider the cycle obtained by applying π (used in computing $Q_r(0)$) to the Hamiltonian cycle in G. Pretend that this cycle was directly chosen as a random cycle over $[n]$- i.e., add it to the tape y. For (i, j) not part of this cycle, pretend that z_{ij} was chosen directly as a random string from $\{0,1\}^{k_1}$, and add them to the tape y. For (i, j) in the cycle, retain the part of x corresponding to the randomness s_{ij} used in computing $z_{ij} = \mathsf{C}_r(1; s_{ij})$. Output (x, y, z).

Now we argue that $\mathcal{H}$ and Ψ satisfy assumptions A1, A2 and A3. Distribution $\mathcal{D}^{\mu}_r$ is defined as the output distribution of $\Psi(\mu, r)$ conditioned on $\mu \in \mathfrak{X}$.

A1 (x, z) in the output of $\Psi(\mu, r)$ is indeed identical to $\{(x, z) | x \leftarrow \{0,1\}^{k_2}, z = \mathcal{H}(\mu, r, x, 0)\}$. The difference between (y, z) and $\{(y, z) | y \leftarrow \{0,1\}^{k_2}, z = \mathcal{H}(\mu, r, y, 1)\}$ is that the entries z_{ij} for (i, j) not in the cycle specified in y are not random, but rather the output of PRG on random inputs, possibly xor-ed with the string r. Indistinguishability assumed in A1 follows from the security of PRG.

A2 Sampling access to $\mathcal{D}^{\mu'}_r$ is equivalent to oracle access to $\sigma = \mathsf{sign}(SK, \mu')$ (on each query, a fresh signature is given). Also, finding a collision is equivalent to finding σ assuming that the commitment scheme C_r is perfectly binding, which is the case for all but a negligible fraction of r. But the security of the signature scheme guarantees that (for a randomly generated (SK, VK)) it is infeasible for a PPT circuit M to produce a valid signature $\mathsf{sign}(SK, \mu)$ even if it has seen multiple signatures on other messages $\mu' \neq \mu$. Thus (except with negligible probability over the generation of CRS), A2 holds.

A3 This follows from the standard assumption of trapdoor permutations, because security of the trapdoor permutation is required to hold only when (f, f^{-1}) is independent of (SK, VK).

Acknowledgments

We would like to thank Boaz Barak, Ran Canetti, Oded Goldreich, Yuval Ishai, Yehuda Lindell, Silvio Micali, Rafael Pass and Alon Rosen for useful discussions. We also thank Jonathan Katz for suggesting a modification to the protocol in [2] on which our protocol COM (Figure 5) is based, thereby allowing us to simplify the proof of Theorem 4 considerably.

References

[1] Boaz Barak. Constant-Round Coin-Tossing with a Man in the Middle or Realizing the Shared Random String Model. FOCS 2002: 345-355

[2] Boaz Barak and Yehuda Lindell. Strict Polynomial-time in Simulation and Extraction. Electronic Colloquium on Computational Complexity (ECCC)(026): (2002)

[3] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. Electronic Colloquium on Computational Complexity (ECCC) (016): (2001) (Preliminary version in *IEEE Symposium on Foundations of Computer Science*, pages 136–145, 2001.)

[4] Ran Canetti and Marc Fischlin. Universally composable commitments. In *CRYPTO*, pages 19–40, 2001.

[5] Ran Canetti and Hugo Krawczyk. Universally Composable Notions of Key Exchange and Secure Channels. EUROCRYPT 2002: 337-351.

[6] R. Canetti, E. Kushilevitz, and Y. Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In *EUROCRYPT*, pages 68–86, 2003.

[7] R. Canetti, Y. Lindell, R. Ostrovsky, and A. Sahai. Universally composable two-party and multi-party secure computation. In *ACM Symposium on Theory of Computing*, pages 494–503, 2002.

[8] Ivan Damgard and Jens Groth. Non-interactive and reusable non-malleable commitment schemes. STOC 2003: 426-437

[9] Giovanni Di Crescenzo, Yuval Ishai and Rafail Ostrovsky. Non-Interactive and Non-Malleable Commitment. STOC 1998: 141-150

[10] Danny Dolev, Cynthia Dwork and Moni Naor. Nonmalleable Cryptography. SIAM J. Comput. 30(2): 391-437 (2000)

[11] Cynthia Dwork, Moni Naor and Amit Sahai. Concurrent Zero-Knowledge. STOC 1998. 409-418

[12] S. Goldwasser and Y. Lindell. Secure Computation without Agreement. DISC 2002: 17-32

[13] O. Goldreich. *Secure Multi-Party Computation*. Manuscript. Preliminary version, 1998. Available from `http://www.wisdom.weizmann.ac.il/~oded/pp.html`.

[14] O. Goldreich and L. Levin. A Hard Predicate for All One-Way Functions. In 21*st STOC*, pages 25–32, 1989.

[15] O. Goldreich, S. Micali and A. Wigderson. How to Play any Mental Game – A Completeness Theorem for Protocols with Honest Majority. In *19th STOC*, pages 218–229, 1987. For details see [13].

[16] Joe Kilian and Erez Petrank. Concurrent and resettable zero-knowledge in poly-logarithmic rounds. STOC 2001: 560-569.

[17] Yehuda Lindell. Bounded-concurrent secure two-party computation without setup assumptions. STOC 2003. 683-692.

[18] Yehuda Lindell. General composition and universal composability in secure multi-party computation. In *IEEE Symposium on Foundations of Computer Science*, pages 394-403, 2003.

[19] Moni Naor. Bit Commitment using Pseudorandom Generators. *Journal of Cryptology*, 4(2):151–158, 1991.

[20] Rafael Pass. Simulation in Quasi-Polynomial Time, and Its Application to Protocol Composition. EUROCRYPT 2003: 160-176.

[21] Rafael Pass and Alon Rosen. Bounded-Concurrent Secure Two-Party Computation in a Constant Number of Rounds. FOCS 2003: 404-413.

[22] Manoj Prabhakaran, Alon Rosen and Amit Sahai. Concurrent Zero Knowledge with Logarithmic Round-Complexity. FOCS 2002: 366-375

[23] Manoj Prabhakaran and Amit Sahai. Confronting the Composition Conundrum: Universal Composability without Set-up Assumptions. At the Cryptology ePrint Archive `http://eprint.iacr.org/`. 2004.

[24] Manoj Prabhakaran and Amit Sahai. Revisiting Concurrency: Monitored Functionalities and Client-Server Computation. Manuscript under preparation.

[25] Amit Sahai. Non-malleable Non-interactive Zero Knowledge and Adaptive Chosen Ciphertext Security. FOCS 1999: 543-553

[26] Ransom Richardson and Joe Kilian. On the Concurrent Composition of Zero-Knowledge Proofs. EUROCRYPT 1999: 415-431

[27] John Rompel. One-Way Functions are Necessary and Sufficient for Secure Signatures. STOC 1990: 387-394

A Sufficient Assumptions

We sketch how the assumptions in Section 2.3 suffice for all our constructions.

We assume a perfectly binding (non-interactive) commitment scheme C, whose hiding property holds against PPT adversaries with access to the distributions $\mathcal{D}_r^\mu$ for all $\mu \in \mathfrak{I}$ and $r \in \{0,1\}^{k_1}$. We use this at two points: in the Ψ-ES commitment protocol COM, and the Ψ-ES many-to-one commitment-and-prove protocol OM-CP. In both these places it is possible to replace this with the following (in the $\mathcal{F}_{\widetilde{\text{ZK}}}$-hybrid model): the sender picks a trapdoor permutation $(f, f^{-1}) \leftarrow \mathcal{T}$, and sends f to the receiver. Then it uses the functionality $\mathcal{F}_{\widetilde{\text{ZK}}}$ to prove to the receiver that f was indeed generated by $\mathcal{T}$ (all we need is that f be a permutation). The commitment is then done using the hard core bit of f as in [14]. Then from Assumption A3 this is a hiding commitment scheme as well. It is easy to see that C can be replaced by this scheme in both COM and OM-CP protocols. Thus we restrict ourselves to the assumptions given in Section 2.3.

Since we employed the $\mathcal{F}_{\text{ENC}}$-hybrid model (in the protocol BCOM), we also assume that there is a protocol which Ψ-ES-realizes $\mathcal{F}_{\text{ENC}}$ against static adversaries. But again, as noted in the proof of Corollary 5, such a protocol indeed exists based on assumption A3.

B The Specialized Simulator UC Theorem with Imaginary Angels

In this section we extend the statement of the composition theorem, Theorem 2 to "specialized simulator" UC [18], and prove the extended version. As noted in [18], a stronger notion of security is achieved in the specialized simulator setting by allowing the output of the environment (in REAL^Γ, IDEAL^Γ or HYB^Γ executions) to be an arbitrarily long string. We shall also adopt this. (But it will be clear that the composition theorem holds for even the weaker notion of security where the environment is restricted to outputting a single bit.) Note that the composition theorem holds in particular with a "null Angel" for the Imaginary Angel Γ.

Claim 1 *The following are equivalent:*

1. $\forall \mathcal{A}^\Gamma \in \mathcal{C}\ \exists s = s(k)\ \forall \mathcal{Z}^\Gamma\ \exists \mathcal{S}^\Gamma$, $|\mathcal{S}^\Gamma| \leq s$ *such that* $\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma} \approx \text{REAL}^\Gamma_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}^\Gamma}$
2. $\exists s = s(k)\ \forall \mathcal{Z}^\Gamma\ \exists \mathcal{S}^\Gamma$, $|\mathcal{S}^\Gamma| \leq s$ *such that* $\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma} \approx \text{REAL}^\Gamma_{\pi,\tilde{\mathcal{A}}_\mathcal{C},\mathcal{Z}^\Gamma}$

Here $s = s(k)$ is restricted to polynomials, and so is $|\mathcal{A}^\Gamma|$ for $\mathcal{A}^\Gamma \in \mathcal{C}$.

If these (equivalent) conditions are satisfied we say π securely realizes *the functionality $\mathcal{F}$*under specialized simulator rUC *with respect to the Imaginary Angel Γ against the class $\mathcal{C}$ of adversaries.*

Proof: The proof mimics the proof in [3].[10] Since $\tilde{\mathcal{A}}_\mathcal{C} \in \mathcal{C}$, the first statement implies the second. To show that the second statement implies the first, assume that the second statement holds. Then given $\mathcal{A}^\Gamma \in \mathcal{C}$ and $\mathcal{Z}'^\Gamma$ we show how to construct a simulator $\mathcal{S}'^\Gamma$ as required by the first statement.

Consider an environment $\mathcal{Z}^\Gamma$ which internally simulates $\mathcal{Z}'^\Gamma$ and $\mathcal{A}^\Gamma$. Whenever $\mathcal{A}^\Gamma$ tries to interact with the parties (by trying to view the messages sent by the parties, by delivering a message to a party, or by corrupting a party) $\mathcal{Z}^\Gamma$ directs $\tilde{\mathcal{A}}_\mathcal{C}$ to do this and forwards to $\mathcal{A}^\Gamma$ any information it gets by doing so. In addition to this, $\mathcal{A}^\Gamma$ may query Γ, but this can also be simulated by $\mathcal{Z}^\Gamma$ as it also has access to Γ. Note that

[10]But we choose to present it as a direct proof instead of a proof by contradiction.

the set of parties corrupted by $\tilde{A}_{\mathcal{C}}$ is the same as the set that the simulated $\mathcal{A}^\Gamma$ expects to be corrupted, so the answers from Γ are perfectly simulated. Thus,

$$\text{REAL}^{\Gamma}_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}'^\Gamma} = \text{REAL}^{\Gamma}_{\pi,\tilde{A}_{\mathcal{C}},\mathcal{Z}^\Gamma} \tag{1}$$

By assumption $\exists \mathcal{S}^\Gamma$ such that

$$\text{REAL}^{\Gamma}_{\pi,\tilde{A}_{\mathcal{C}},\mathcal{Z}^\Gamma} \approx \text{IDEAL}^{\Gamma}_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma} \tag{2}$$

We build $\mathcal{S}'^\Gamma$ exactly as in [3]: $\mathcal{S}'^\Gamma$ internally simulates $\mathcal{A}^\Gamma$ and $\mathcal{S}^\Gamma$, and (a) acts transparently between $\mathcal{A}^\Gamma$ (simulated internally) and $\mathcal{Z}'^\Gamma$, (b) when $\mathcal{A}^\Gamma$ tries to interact with the parties, engages $\mathcal{A}^\Gamma$ with $\mathcal{S}^\Gamma$ (also simulated internally) and (c) acts transparently between $\mathcal{S}^\Gamma$ and the functionality $\mathcal{F}$, and also between $\mathcal{S}^\Gamma$ and the (dummy) parties. In addition it acts transparently between $\mathcal{A}^\Gamma$ and the Imaginary Angel Γ, as well as between $\mathcal{S}^\Gamma$ and Ψ. Then it is easy to see that

$$\text{IDEAL}^{\Gamma}_{\mathcal{F},\mathcal{S}^\Gamma,\mathcal{Z}^\Gamma} = \text{IDEAL}^{\Gamma}_{\mathcal{F},\mathcal{S}'^\Gamma,\mathcal{Z}'^\Gamma} \tag{3}$$

From Equations (1)(2) and (3) we get $\text{REAL}^{\Gamma}_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}'^\Gamma} \approx \text{IDEAL}^{\Gamma}_{\mathcal{F},\mathcal{S}'^\Gamma,\mathcal{Z}'^\Gamma}$. Noting that $|\mathcal{S}'^\Gamma|$ is essentially $|\mathcal{A}^\Gamma| + |\mathcal{S}^\Gamma|$ where $|\mathcal{A}^\Gamma|$ and $|\mathcal{S}^\Gamma|$ are polynomial in k completes the proof. □

The following is a restatement of the theorem in [3], but with two differences: first, we replace the REAL and HYBRID executions by REAL^Γ and HYB^Γ executions respectively. Secondly, we change the order of the quantifiers (from $\forall\mathcal{A}\exists\mathcal{S}\forall\mathcal{Z}$ to $\forall\mathcal{A}\forall\mathcal{Z}\exists\mathcal{S}$) as mentioned above.

Theorem 6 *Let $\mathcal{C}$ be a class of real-world adversaries and $\mathcal{F}$ be an ideal functionality. Let π be an n-party protocol in the $\mathcal{F}$-hybrid model and let ρ be an n-party protocol that Γ-ES-realizes $\mathcal{F}$ against adversaries of class $\mathcal{C}$ under specialized-simulator UC. Then for any real-world adversary $\forall\mathcal{A}^\Gamma \in \mathcal{C}\exists$ a polynomial $h = h(k)$ such that $\forall\mathcal{Z}^\Gamma\exists$ a hybrid-model adversary $\mathcal{H}^\Gamma \in \mathcal{C}$ such that we have:*

$$\text{REAL}^{\Gamma}_{\pi^\rho,\mathcal{A}^\Gamma,\mathcal{Z}^\Gamma} \approx \text{HYB}^{\Gamma,\mathcal{F}}_{\pi,\mathcal{H}^\Gamma,\mathcal{Z}^\Gamma}$$

Further if ρ securely realizes $\mathcal{F}$ not under specialized-simulator UC, then $\mathcal{H}^\Gamma$ does not depend on $\mathcal{Z}^\Gamma$.

Much of the difference in the proof is due to the extension to the specialized simulator setting. Note that in the proof of Claim 1 we showed that for every $\mathcal{A}^\Gamma \in \mathcal{C}$ and $\mathcal{Z}'^\Gamma$, there exists $\mathcal{Z}^\Gamma$ such that $\text{REAL}^{\Gamma}_{\pi,\tilde{A}_{\mathcal{C}},\mathcal{Z}^\Gamma}$ and $\text{REAL}^{\Gamma}_{\pi,\mathcal{A}^\Gamma,\mathcal{Z}'^\Gamma}$ are identical. So to prove Theorem 2, it suffices to show that for any $\mathcal{Z}^\Gamma$ there exists $\mathcal{H}^\Gamma \in \mathcal{C}$ such that

$$\text{REAL}^{\Gamma}_{\pi^\rho,\tilde{A}_{\mathcal{C}},\mathcal{Z}^\Gamma} \approx \text{HYB}^{\Gamma,\mathcal{F}}_{\pi,\mathcal{H}^\Gamma,\mathcal{Z}^\Gamma}$$

Proof: Given an environment $\mathcal{Z}^\Gamma$, we want to construct a hybrid-world $\mathcal{H}^\Gamma$ and argue that it satisfies

$$\text{REAL}^{\Gamma}_{\pi^\rho,\tilde{A}_{\mathcal{C}},\mathcal{Z}^\Gamma} \approx \text{HYB}^{\Gamma,\mathcal{F}}_{\pi,\mathcal{H}^\Gamma,\mathcal{Z}^\Gamma}$$

Suppose m is an upper bound on the number of invocations of the protocol ρ within π. First, for $\ell = 1,\ldots,m$, we construct environments $\mathcal{Z}^\Gamma_\ell$ for testing protocol ρ (as a stand-alone protocol) and corresponding simulators $\mathcal{S}^\Gamma_\ell$ (guaranteed to exist by the hypothesis that ρ securely realizes $\mathcal{F}$). $\mathcal{H}^\Gamma$ is constructed from the m simulators $\mathcal{S}^\Gamma_1,\ldots,\mathcal{S}^\Gamma_m$ and $\mathcal{H}^\Gamma$ will essentially be $\sum_{\ell=1}^{m}|\mathcal{S}^\Gamma_\ell| \le ms$, where $s = s(k)$ is the bound on

Adversary $\mathcal{H}_\ell^\Gamma$

Given a security parameter k, adversary $\mathcal{H}_\ell^\Gamma$ proceeds as follows, interacting with parties $P_1, \ldots, P_n$ running protocol π^ρ in the $\mathcal{F}^{(\ell+1)}$-hybrid model (i.e., at most $\ell+1$ copies of $\mathcal{F}$) with an environment $\mathcal{Z}^\Gamma$ and Imaginary Angel Γ.
Suppose $m = m(k)$ is an upperbound on the number of session IDs for copies of ρ and $\mathcal{F}$ together. The $\ell+1$ copies of $\mathcal{F}$, denoted by $\mathcal{F}_{(1)}, \ldots, \mathcal{F}_{(\ell+1)}$, are associated with the first $\ell+1$ session IDs. Internally, $\mathcal{H}_\ell^\Gamma$ will run the ℓ simulators $\mathcal{S}_1^\Gamma, \ldots, \mathcal{S}_\ell^\Gamma$ and $m-\ell$ copies of $\tilde{\mathcal{A}}_\mathcal{C}$. Let $\mathcal{X}_j^\Gamma$ denote $\mathcal{S}_j^\Gamma$ for $j \le \ell$ and the $j-\ell$-th copy of $\tilde{\mathcal{A}}_\mathcal{C}$ for $j > \ell$.

1. If activated with an instruction from $\mathcal{Z}^\Gamma$ to report the new messages sent by the parties, proceed as follows:
   (a) Collect the messages sent by the parties $P_1, \ldots, P_n$
   (b) Activate each (running) $\mathcal{X}_j^\Gamma$ with an instruction to report new messages sent by the parties in the copy of ρ with session ID j (for $j \le \ell$ the copy of ρ is simulated by $\mathcal{S}_j^\Gamma$).
   (c) Combine the messages gathered from the above two steps and report them back to $\mathcal{Z}^\Gamma$ as the messages sent by parties running π^ρ.
2. If activated with an instruction to deliver a π-message to some party P_i then deliver that message as instructed. If the instruction is to deliver a message with one of the m session IDs of ρ, say the j-th session ID, then forward that instruction to $\mathcal{X}_j^\Gamma$ and activate it.
3. If activated with an instruction to corrupt a party P_i, proceed as follows:
   (a) Corrupt the party P_i (if allowed by class $\mathcal{C}$) and obtain P_i's state regarding the execution of π.
   (b) Activate each (running) $\mathcal{X}_j^\Gamma$ with the same instruction, and collect the internal states of P_i regarding the copies of ρ as reported by the $\mathcal{X}_j^\Gamma$'s (for $j \le \ell$ this is information simulated by $\mathcal{S}_j^\Gamma$).
   (c) Combine the states obtained in the above two steps to obtain a (simulated) state of P_i with respect to protocol π^ρ, and report it back to $\mathcal{Z}^\Gamma$.
4. If activated with an input from a copy $\mathcal{F}_{(j)}$ of $\mathcal{F}$ for $j \le \ell$, or a ρ-message with session ID $j > \ell$ from some party, forward that message to $\mathcal{X}_j^\Gamma$ and activate it.
5. If $\mathcal{X}_j^\Gamma$ tries to deliver a message to some party or to $\mathcal{F}_{(j)}$ (only for $j \le \ell$), deliver the message to the party or $\mathcal{F}_{(j)}$ respectively. If $\mathcal{X}_j^\Gamma$ ($j \le \ell$) tries to make a query to the Imaginary Angel Γ, make the query and return the answer to $\mathcal{X}_j^\Gamma$. Note that by step 3(a) and 3(b), at any time the set of corrupted parties (on which Γ's answers may depend) is the same for all simulated $\mathcal{X}_j^\Gamma$ ($j \le \ell$) and $\mathcal{H}_\ell^\Gamma$, so the Imaginary Angel access of $\mathcal{X}_j^\Gamma$ is perfectly simulated by $\mathcal{H}_\ell^\Gamma$.

Figure 10: The adversaries $\mathcal{H}_\ell^\Gamma$ for π^ρ in the $\mathcal{F}^{(\ell+1)}$-hybrid model.

the simulator size coming from the security of ρ. Our construction will be such that the following relations

Environment $\mathcal{Z}_\ell^\Gamma$

Given a security parameter k and input z, environment $\mathcal{Z}_\ell^\Gamma$ proceeds as follows, interacting with parties $P_1, \ldots, P_n$ running protocol ρ and with an adversary $\mathcal{A}^\Gamma$.

1. If activated for the first time, initialize a variable s to hold the global state of the following system: there are n parties $P_1', \ldots, P_n'$ running protocol π^ρ in the $\mathcal{F}^{(\ell)}$-hybrid model, and adversary $\mathcal{H}_{\ell-1}^\Gamma$. Initialize s so that $\mathcal{Z}^\Gamma$ gets z as its input, and gets activated.

   If not the first activation, the update the variable s as follows:

   (a) If P_i has output a new value y, then update the state of the simulated party P_i' by including a message y received from $\mathcal{F}_{(\ell)}$

   (b) If $\mathcal{A}^\Gamma$ has a new output, forward it to $\mathcal{H}_{\ell-1}^\Gamma$ as output of $\mathcal{X}_\ell^\Gamma$ (as defined in Figure 10).

2. Simulate execution of the system starting in state s until one of the following events occurs in the simulation:

   (a) Party P_i' sends a message x to $\mathcal{F}_{(\ell)}$. In this case, save the current state of simulation in s, and activate P_i with input x.

   (b) $\mathcal{X}_\ell^\Gamma$ is activated with input v. In this case save the state in s and activate $\mathcal{A}^\Gamma$ with input v.

   (c) Environment $\mathcal{Z}^\Gamma$ halts. In this case output whatever $\mathcal{Z}^\Gamma$ outputs and halt.

Figure 11: The environments $\mathcal{Z}_\ell^\Gamma$ for a single copy of ρ.

hold:

$$\text{REAL}^\Gamma_{\pi^\rho,\tilde{\mathcal{A}}_\mathcal{C},\mathcal{Z}^\Gamma} = \text{REAL}^\Gamma_{\rho,\tilde{\mathcal{A}}_\mathcal{C},\mathcal{Z}_1^\Gamma} \tag{4}$$

$$\text{REAL}^\Gamma_{\rho,\tilde{\mathcal{A}}_\mathcal{C},\mathcal{Z}_\ell^\Gamma} \approx \text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}_\ell^\Gamma,\mathcal{Z}_\ell^\Gamma} \quad \text{for } \ell = 1, \ldots, m \tag{5}$$

$$\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}_\ell^\Gamma,\mathcal{Z}_\ell^\Gamma} = \text{REAL}^\Gamma_{\rho,\tilde{\mathcal{A}}_\mathcal{C},\mathcal{Z}_{\ell+1}^\Gamma} \quad \text{for } \ell = 1, \ldots, m-1 \tag{6}$$

$$\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}_m^\Gamma,\mathcal{Z}_m^\Gamma} = \text{HYB}^{\Gamma,\mathcal{F}}_{\pi,\mathcal{H}^\Gamma,\mathcal{Z}^\Gamma} \tag{7}$$

Thus for all polynomials $m = m(k)$, we will have $\text{REAL}^\Gamma_{\pi^\rho,\tilde{\mathcal{A}}_\mathcal{C},\mathcal{Z}^\Gamma} \approx \text{HYB}^{\Gamma,\mathcal{F}}_{\pi,\mathcal{H}^\Gamma,\mathcal{Z}^\Gamma}$ as required. The rest of the proof describes $\mathcal{Z}_\ell^\Gamma$ and $\mathcal{S}_\ell^\Gamma$ and argues why the above relations hold.

Equation (4) is established by setting up $\mathcal{Z}_1^\Gamma$ to internally simulate $\mathcal{Z}^\Gamma$, π and all but one invocation of ρ by π. Given $\mathcal{Z}_\ell^\Gamma$, from the fact that ρ securely realizes $\mathcal{F}$, $\mathcal{S}_\ell^\Gamma$ is obtained as that simulator for which Equation (5) holds. Given $\mathcal{Z}_\ell^\Gamma$ and $\mathcal{S}_\ell^\Gamma$, $\mathcal{Z}_{\ell+1}^\Gamma$ is constructed so that Equation (6) will hold: for this $\mathcal{Z}_{\ell+1}^\Gamma$ will internally simulate the IDEAL^Γ system consisting of $\mathcal{F}$, $\mathcal{S}_\ell^\Gamma$ and $\mathcal{Z}_\ell^\Gamma$, but leaving out one invocation of ρ that is simulated by $\mathcal{Z}_\ell^\Gamma$. Thus $\mathcal{Z}_{\ell+1}^\Gamma$ simulates one more copy of $\mathcal{F}$ and one less invocation of ρ than $\mathcal{Z}_\ell^\Gamma$. By the time we get to $\mathcal{Z}_m^\Gamma$, it simulates $m-1$ copies of $\mathcal{F}$ internally, along with $\mathcal{S}_1^\Gamma, \ldots, \mathcal{S}_{m-1}^\Gamma$. Thus $\text{IDEAL}^\Gamma_{\mathcal{F},\mathcal{S}_m^\Gamma,\mathcal{Z}_m^\Gamma}$ consists of m copies of $\mathcal{F}$ and the m simulators $\mathcal{S}_1^\Gamma, \ldots, \mathcal{S}_m^\Gamma$. Finally, for Equation (7) we set up $\mathcal{H}^\Gamma$ to simulate the IDEAL^Γ system involving $\mathcal{F}$, $\mathcal{S}_m^\Gamma$ and $\mathcal{Z}_m^\Gamma$, but exclude $\mathcal{Z}^\Gamma$ and π which are simulated by $\mathcal{Z}_m^\Gamma$.

It remains to fully specify the environments $\mathcal{Z}_\ell^\Gamma$ and the adversary $\mathcal{H}^\Gamma$. It is convenient to describe $\mathcal{Z}_\ell^\Gamma$ in terms of simulating an adversary $\mathcal{H}_\ell^\Gamma$. We shall then let $\mathcal{H}^\Gamma = \mathcal{H}_m^\Gamma$. In Figure 10 and Figure 11, we mimic the specifications of $\mathcal{H}$ and $\mathcal{Z}_\rho$ in [3], but with important differences to accommodate the different environments $\mathcal{S}_\ell^\Gamma$ instead of a single environment $\mathcal{S}$.

We define $\mathcal{H}_0^\Gamma$ to be $\tilde{\mathcal{A}}_\mathcal{C}$. Figure 10 defines $\mathcal{H}_\ell^\Gamma$ for $\ell \geq 1$. It is an adversary (simulator) which is designed

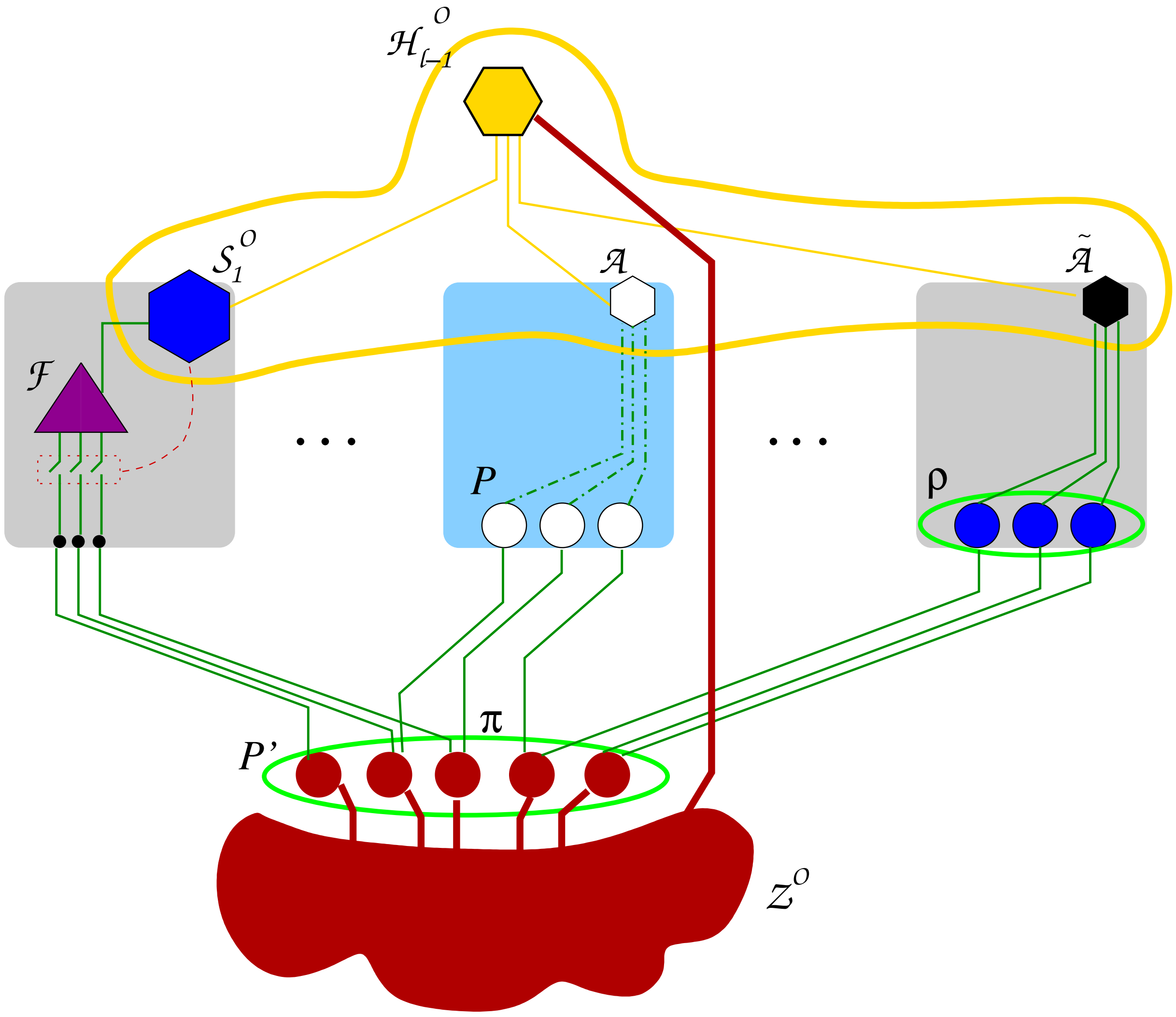

Figure 12: An illustration of $\mathcal{Z}_\ell^\Gamma$. The blue box in the middle is not part of $\mathcal{Z}_\ell^\Gamma$, but shows the adversary $\mathcal{A}$ and the parties P_i it will interact with. The $\ell - 1$ grey boxes to the left of the blue box contain the simulators $\mathcal{S}_j^\Gamma$, $j = 1, \dots, \ell - 1$ (blue hexagon), the ideal functionalities $\mathcal{F}_{(j)}$ (purple triangle) and the dummy parties (small black circles). The grey boxes to the right contain an execution of the protocol ρ (the blue circles indicate the program of the protocol run by the corresponding parties) with the dummy adversary $\tilde{\mathcal{A}}_\mathcal{C}$ (black hexagon). The environment $\mathcal{Z}^\Gamma$ is shown as a red blob at the bottom, along with the simulated parties P_i' (red circles). The adversary $\mathcal{H}_{\ell-1}^\Gamma$ consists of the golden hexagon and the hexagons enclosed by the golden curve. The straight lines show communication paths.

to interact with parties running π^ρ in the $\mathcal{F}^{(\ell+1)}$-hybrid model. In ℓ of the $\ell + 1$ copies of $\mathcal{F}$, it simulates copies of the protocol ρ. For this $\mathcal{H}_\ell^\Gamma$ runs $\mathcal{S}_1^\Gamma, \dots, \mathcal{S}_\ell^\Gamma$ internally. In the copies of ρ (and the $\ell + 1$-st copy of $\mathcal{F}$) $\mathcal{H}_\ell^\Gamma$ lets the environment act on them directly by behaving like a dummy adversary.

$\mathcal{Z}_\ell^\Gamma$ is defined in Figure 11, and graphically illustrated in Figure 12. It is an environment (of polynomial size, as we shall see) which can be used to test the security of the protocol ρ (with dummy adversary $\tilde{\mathcal{A}}_\mathcal{C}$). Thus specifying $\mathcal{Z}_\ell^\Gamma$ also gives a $\mathcal{S}_\ell^\Gamma$ satisfying Equation (5) via the security of ρ. $\mathcal{Z}_\ell^\Gamma$ internally simulates a system with n parties running π^ρ in the $\mathcal{F}^{(\ell)}$-hybrid model, $\mathcal{Z}^\Gamma$ and $\mathcal{H}_{\ell-1}^\Gamma$ (which in turn uses the simulators $\mathcal{S}_1^\Gamma, \dots, \mathcal{S}_{\ell-1}^\Gamma$ to simulate copies of ρ in $\ell - 1$ copies of $\mathcal{F}$). The protocol ρ to be tested is engaged with this internally simulated system: the simulated parties interact with dummy parties running ρ (provided externally) instead of the ℓ-th copy of $\mathcal{F}$. As can be verified from the descriptions in Figure 10 and Figure 11, this sequence of environments satisfies Equation (6).

Note that $|\mathcal{H}_\ell^\Gamma| = \sum_{j=1}^{\ell} |\mathcal{S}_j^\Gamma| + O(m) \le \ell s + O(m)$, which is polynomial in the security parameter. Hence $\mathcal{Z}_\ell^\Gamma$ is also of polynomial size, because the rest of the system simulated by $\mathcal{Z}_\ell^\Gamma$ involves up to m copies of $\mathcal{F}$ and ρ, π^ρ and $\mathcal{Z}^\Gamma$.[11] Further note that if $\mathcal{S}_i^\Gamma = \mathcal{S}^\Gamma$ independent of the environment $\mathcal{Z}_\ell^\Gamma$, then $\mathcal{H}^\Gamma$ is also independent of the environment $\mathcal{Z}^\Gamma$

To complete the proof, we observe that Equations (4) and (7) follow directly from the definition of $\mathcal{Z}_1^\Gamma$ (using $\mathcal{H}_0^\Gamma = \tilde{\mathcal{A}}_C$) and $\mathcal{H}^\Gamma = \mathcal{H}_m^\Gamma$ respectively.

$\square$

B.1 Extension of the setting

Above, for simplicity we presented the above theorem and proof in a setting where the REAL^Γ setting has no ideal functionalities present, and the IDEAL^Γ setting had $\mathcal{F}$ as the only kind of ideal functionalities. However, indeed, the environments considered may include other functionalities within them (as functionalities are merely polynomial time programs). Further, it is not difficult to see that the theorem (as well as the proof) extends to the case when the parties in the REAL^Γ and IDEAL^Γ settings have access to additional ideal functionalities. This lets the protocols *and* functionalities to be defined in terms of other functionalities.

C Proof of Theorem 4

PROOF (SKETCH). Given an adversary $\mathcal{A}^\Psi$, we sketch how to construct the simulator $\mathcal{S}^\Psi$ such that for all environments $\mathcal{Z}^\Psi$, we have $\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^\Psi,\mathcal{Z}^\Psi} \approx \text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^\Psi,\mathcal{Z}^\Psi}$.

As is usual, $\mathcal{S}^\Psi$ internally simulates $\mathcal{A}^\Psi$. If both the sender C and receiver R running the protocol COM are corrupted, $\mathcal{S}^\Psi$ lets $\mathcal{A}^\Psi$ interact with them directly. We analyze the other three cases below.

Both C, R honest In the commit phase, until the last message the bit b is not used. So $\mathcal{S}^\Psi$ can follow the protocol exactly, playing both C and R. However in the last step (Step 7), it sends out a random bit as b'. In the reveal phase on receiving (REVEAL, b) from $\mathcal{F}_{\text{COM}}$, $\mathcal{S}^\Psi$ simulates C sending b to R. Then it must simulate the interaction between C $\mathcal{F}_{\widetilde{\text{ZK}}}$ and R with the statement $\exists t : f(t) = r_R \oplus r_C \wedge b' = B(t) \oplus b$ as common input. Note that in this interaction all that $\mathcal{A}^\Psi$ sees is the message PROVEN from $\mathcal{F}_{\widetilde{\text{ZK}}}$. So $\mathcal{S}^\Psi$ sends that message to $\mathcal{A}^\Psi$. Now the only difference between the executions $\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^\Psi,\mathcal{Z}^\Psi}$ and $\text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^\Psi,\mathcal{Z}^\Psi}$ is that $B(f^{-1}(r_R \oplus r_C)) \oplus b$ may not be equal to b' in the former (where as it is always the case in the latter). But from the security of the hard core predicate (which by assumption on $\mathcal{T}$ holds even against adversaries with access to the Imaginary Angel Ψ), it can be shown that the environment cannot distinguish between the two executions.

R honest, C corrupted Here we shall use the hiding property of the commitment scheme C and the soundness of $\mathcal{F}_{\widetilde{\text{ZK}}}$ (Lemma 9). The idea is that $\mathcal{S}^\Psi$ can *extract* the bit being committed to by the corrupted sender C. Consider the simulator $\mathcal{S}^\Psi$ which plays the part of the receiver and $\mathcal{F}_{\widetilde{\text{ZK}}}$, and talks to C (corrupted by $\mathcal{A}^\Psi$). $\mathcal{S}^\Psi$ starts off following the programs of $\mathcal{F}_{\widetilde{\text{ZK}}}$ and the honest receiver. But after receiving r_C, instead of sending r_R, the value to which c is a commitment, $\mathcal{S}^\Psi$ picks a random string $u \leftarrow \{0,1\}^k$ and sends

[11] If we let the size of the simulator depend on the environment, say be equal to that of the environment, here we will run against a problem as the size of $\mathcal{Z}_\ell^\Gamma$ will double as ℓ increases by 1, restricting us to $m = O(\log k)$ before the security guarantees fail.

$\tilde{r}_R = f(u) \oplus r_C$. Then it simulates an interaction with $\mathcal{F}_{\widetilde{\text{ZK}}}$, with the statement $(\exists r' : c = \mathsf{C}(\tilde{r}_R; r'))$ as common input: it simply sends the message PROVEN to C acting as $\mathcal{F}_{\widetilde{\text{ZK}}}$. When C sends back b', $\mathcal{S}^\Psi$ calculates $b^* = b' \oplus B(u)$ (this is the *extracted* commitment). $\mathcal{S}^\Psi$ then sends b^* to $\mathcal{F}_{\text{COM}}$ to commit to R. Later, if C sends b as reveal, then $\mathcal{S}^\Psi$ plays the part of $\mathcal{F}_{\widetilde{\text{ZK}}}$ and interacts with C as it tries to prove that $(\exists t : f(t) = \tilde{r}_R \oplus r_C \wedge b' = B(t) \oplus b)$. If the simulated $\mathcal{F}_{\widetilde{\text{ZK}}}$ accepts the proof, then $\mathcal{S}^\Psi$ must make R accept b too. For this $\mathcal{S}^\Psi$ sends REVEAL instruction to $\mathcal{F}_{\text{COM}}$, which will send (REVEAL, b^*) to R.

Note that R is not corrupted and the soundness condition of $\mathcal{F}_{\widetilde{\text{ZK}}}$ (Lemma 9) holds. Hence if the protocol continues beyond Step 3, except with negligible probability, f is indeed a permutation. This has two consequences: firstly in the simulation, the sender can reveal only to the bit extracted by the simulator (using the soundness of $\mathcal{F}_{\widetilde{\text{ZK}}}$ again). Secondly, $\tilde{r}_R$ is uniformly randomly distributed, independent of c. Then it can be shown that distinguishing between the executions $\text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^\Psi,\mathcal{Z}^\Psi}$ and $\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^\Psi,\mathcal{Z}^\Psi}$ enables one to distinguish between the following experiments: in one experiment it is given (c, r_R) where r_R is a random string and c a random commitment to r_R, and in the other it gets $(c, \tilde{r}_R)$ where c is as before, but $\tilde{r}_R$ is an independently and uniformly chosen random string. From the hiding property of C (assumed to hold against PPT circuits with access to the Imaginary Angel Ψ (with any arbitrary set of parties corrupted)) a distinguisher will have negligible advantage in distinguishing between these two experiment. It follows that $\text{IDEAL}^{\Psi}_{\mathcal{F}',\mathcal{S}^\Psi,\mathcal{Z}^\Psi} \approx \text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^\Psi,\mathcal{Z}^\Psi}$.

To show $\text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^\Psi,\mathcal{Z}^\Psi} \approx \text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^\Psi,\mathcal{Z}^\Psi}$ we consider an intermediate situation, where $\mathcal{F}_{\text{COM}}$ is replaced by a functionality $\mathcal{F}'$ which behaves like $\mathcal{F}_{\text{COM}}$, except that in the reveal stage it accepts a bit b from the sender and sends (REVEAL, b) to the receiver, irrespective of what bit it received during the commitment phase. (To be precise, $\mathcal{S}^\Psi$ is also slightly modified so that it sends (REVEAL, b) to $\mathcal{F}'$ (rather than just REVEAL)).

Note that R is not corrupted and the soundness condition of $\mathcal{F}_{\widetilde{\text{ZK}}}$ (Lemma 9) holds. Hence if the protocol continues beyond Step 3, except with negligible probability, f is indeed a permutation. Conditioned on that, the statement defined in Step 2 of the Reveal Phase is true only if $b = b^*$. So if $b \neq b^*$ the probability that C can make $\mathcal{F}_{\widetilde{\text{ZK}}}$ in Step 2 accept the proof is negligible . Thus $\text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^\Psi,\mathcal{Z}^\Psi} \approx \text{IDEAL}^{\Psi}_{\mathcal{F}',\mathcal{S}^\Psi,\mathcal{Z}^\Psi}$.

Now, if $\mathcal{Z}^\Psi$ can distinguish between the two executions $\text{IDEAL}^{\Psi}_{\mathcal{F}',\mathcal{S}^\Psi,\mathcal{Z}^\Psi}$ and $\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^\Psi,\mathcal{Z}^\Psi}$, it can distinguish between the following experiments: in one experiment it is given (c, r_R) where r_R is a random string and c a random commitment to r_R, and in the other it gets $(c, \tilde{r}_R)$ where c is as before, but $\tilde{r}_R$ is an independently and uniformly chosen random string (here we use the fact that f is a permutation, except with negligible probability, thanks to Lemma 9). From the hiding property of C (assumed to hold against PPT circuits with access to the Imaginary Angel Ψ (with any arbitrary set of parties corrupted)) a distinguisher will have negligible advantage in distinguishing between these two experiment. It follows that $\text{IDEAL}^{\Psi}_{\mathcal{F}',\mathcal{S}^\Psi,\mathcal{Z}^\Psi} \approx \text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^\Psi,\mathcal{Z}^\Psi}$.

C honest, R corrupted The simulator in this case is the same as that in the case when both C and R are honest. The proof of indistinguishability is also almost the same, except now the receiver is corrupt, and so M cannot choose r_R. Suppose c^* is the value of the first message in the protocol such that conditioned on $c = c^*$ the difference $|(\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^\Psi,\mathcal{Z}^\Psi} | c = c^*) - (\text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^\Psi,\mathcal{Z}^\Psi} | c = c^*)|$ is maximized. Now consider the machine M_{c^*} which gets r^* such that $c^* = \mathsf{C}(r^*; r')$ as a non-uniform advice. (If no such r^* exists, then the probability that the execution proceeds beyond Step 6 is negligible, and hence the difference between the two executions is negligible.) M_{c^*} takes (f, r, h) as input where f is randomly drawn from $\mathcal{T}_k$, $r \leftarrow \{0,1\}^k$ and h is either a random bit (Experiment 1) or $h = B(f^{-1}(r))$ (Experiment 2). It tries to distinguish between the two experiments as follows. M_{c^*} starts the system at the point c^* has been sent as the first message in the protocol. At Step 3 it sends the message PROVEN to R. At Step 4, it sends $r_C = r \oplus r^*$,

and at Step 7, $b' = h \oplus b$. Now, in Experiment 1 this is identical to the case of $(\text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^{\Psi},\mathcal{Z}^{\Psi}} | c = c^*)$ and in Experiment 2, identical to $(\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^{\Psi},\mathcal{Z}^{\Psi}} | c = c^*)$. Thus M_{c^*} has a distinguishing probability exactly equal to $|(\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^{\Psi},\mathcal{Z}^{\Psi}} | c = c^*) - (\text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^{\Psi},\mathcal{Z}^{\Psi}} | c = c^*)|$. By the choice of c^*, this is at least $|\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^{\Psi},\mathcal{Z}^{\Psi}} - \text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^{\Psi},\mathcal{Z}^{\Psi}}|$. Thus if that difference is non-negligible we have a machine $M = M_{c^*}$ which can violate the assumption on the trapdoor permutation family. Hence we conclude $|\text{HYB}^{\Psi,\mathcal{F}_{\widetilde{\text{ZK}}}}_{\text{COM},\mathcal{A}^{\Psi},\mathcal{Z}^{\Psi}} - \text{IDEAL}^{\Psi}_{\mathcal{F}_{\text{COM}},\mathcal{S}^{\Psi},\mathcal{Z}^{\Psi}}|$ is negligible. □

D Appendix to Section 3

Functionality $\mathcal{F}^{\ell}_{\text{OT}}$

$\mathcal{F}^{\ell}_{\text{OT}}$ parameterized by integers ℓ and m, and running with an oblivious transfer sender T, a receiver R and an adversary $\mathcal{S}^{\Psi}$, proceeds as follows.

1. Upon receiving a message $(x_1, ..., x_\ell)$ from T, where each $x_j \in \{0,1\}^m$, record the tuple $(x_1, ..., x_\ell)$.
2. Upon receiving a number $i \in \{1, \ldots, \ell\}$ from R, send x_i to R, notify $\mathcal{S}^{\Psi}$, and halt. (If no message from T was previously received, then send nothing to R.)

(a) The oblivious transfer functionality, $\mathcal{F}^{\ell}_{\text{OT}}$

Protocol SOT

SOT is parameterized by integers ℓ and m, and a security parameter k. The parties are an oblivious transfer sender T and a receiver R.

1. Given input $(x_1, ..., x_\ell)$, party T draws $(f, f^{-1}) \leftarrow \mathcal{T}_k$ and sends f to the receiver R.
2. Given input i, and having received f from T, receiver R chooses $y_1, ...y_{i-1}, r, y_{i+1}, ..., y_\ell \leftarrow \{0,1\}^k$, computes $y_i = f(r)$, and sends $(sid, y_1, \ldots, y_\ell)$ to T.
3. Having received $(y_1, \ldots, y_\ell)$ from R, the sender T sends $(x_1 \oplus B(f^{-1}(y_1)), \ldots, x_\ell \oplus B(f^{-1}(y_\ell)))$ to R, where $B(\cdot)$ is a hard-core predicate for f.
4. *Output:* Having received $(b_1, \ldots, b_\ell)$ from T, the receiver R outputs $b_i \oplus B(r)$.

(b) The static, semi-honest Oblivious Transfer protocol

Figure 13: Oblivious Transfer- functionality $\mathcal{F}_{\text{OT}}$ and protocol SOT [7]

$\mathsf{Comp}(\boldsymbol{\Pi})$

Let the session ID of the protocol be sid. Party P_i proceeds as follows (the code for all other parties is analogous):

1. **Initiation of $\mathcal{F}_{\text{CP}}^{\textbf{1:M}}$ instances:** On initiating a session of the protocol $\mathsf{Comp}(\Pi)$ for the first time with a set $\mathcal{P}$ of parties, party P_i instantiates copies of $\mathcal{F}_{\text{CP}}^{\text{1:M}}$ as follows:
   - For each $P_j \in \mathcal{P}$, $j \neq i$, it instantiates a copy of $\mathcal{F}_{\text{CP}}^{\text{1:M}}$ with session ID sid_{ij}, denoted by $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_{ij}]$. $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_{ij}]$ is parametrized by the identity relation (i.e., $R = \{(x, y) \mid x = y\}$; thus this copy of $\mathcal{F}_{\text{CP}}^{\text{1:M}}$ functions as a regular commitment functionality).
   - It instantiates another copy, $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_i]$ parametrized by the relation $R = \big\{((m, s_i, \overline{m}), (\overline{x}_i, r_i^i)) \mid m = \Pi(\overline{x}_i, r_i^i \oplus s_i, \overline{m})\big\}$ where $\Pi(\overline{x}, r, \overline{m})$ stands for the message produced by the protocol Π on input $\overline{x}$, random tape r and history $\overline{m}$.
2. **Random tape generation:** For every party P_j, the parties run the following procedure in order to choose a random tape for P_j:
   (a) P_i chooses $r_i^j \leftarrow \{0,1\}^k$. and sends $(\text{COMMIT}, \mathcal{P}, r_i^j)$ to $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_{ij}]$.
   (b) P_i receives $(\text{RECEIPT}, P_h, \mathcal{P})$ from $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_{hi}]$, for every other party $P_h \in \mathcal{P}$. P_i also receives $(\text{RECEIPT}, P_j, \mathcal{P})$ from $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_j]$, where P_j is the party for whom the random tape is being chosen. P_i then uses $\mathcal{F}_{\text{CP}}^{\text{1:M}}$ to decommit to its value r_i^j. That is, P_i sends (PROVE, r_i^j) to $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_{ij}]$ (which is parametrized by the identity relation).
   (c) P_i receives (PROVEN, r_h^j) messages for every $h \neq j$ and defines the string $s_j = \bigoplus_{h \neq j} r_k^j$. (The random tape for P_j is defined by $r_j = r_j^j \oplus s_j$.)

   When choosing a random tape for P_i, the only difference for P_i is that it sends its random string r_i^i to $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_i]$ and it does not decommit (as is understood from P_j's behavior above).
3. **Activation due to new input:** When activated with input x, party P_i proceeds as follows.
   (a) *Input commitment:* P_i sends $(\text{COMMIT}, \mathcal{P}, x)$ to $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_i]$ and adds x to the list of inputs $\overline{x}_i$ (this list is initially empty and contains P_i's inputs from all the previous activations of this copy of $\mathsf{Comp}(\Pi)$). (At this point all other parties P_j receive the message $(\text{RECEIPT}, P_i, \mathcal{P})$ from $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_i]$. Then P_i proceeds to the next step below.)
   (b) *Protocol computation:* Let $\overline{m}$ be the series of Π-messages that were broadcast in all the activations of Π until now ($\overline{m}$ is initially empty). P_i runs the code of Π on its input list $\overline{x}_i$, messages $\overline{m}$, and random tape r_i (as generated above). If Π instructs P_i to broadcast a message, P_i proceeds to the next step (Step 3c).
   (c) *Outgoing message transmission:* For each outgoing message m that P_i sends in Π, P_i sends $(\text{PROVE}, (m, s_i, \overline{m}))$ to $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_i]$. Recall that $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_i]$ is parametrized by a relation which checks if m is indeed the correct next message produced by Π on input sequence $\overline{x}$ and random tape $r_i = s_i \oplus r_i^i$ on history $\overline{m}$ (note that $\overline{x}$ and r_i^i have to be sent to $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_i]$ in the commit-phase).
4. **Activation due to incoming message:** Upon receiving a message $(\text{PROVEN}, (m, s_j, \overline{m}))$ from $\mathcal{F}_{\text{CP}}^{\text{1:M}}[sid_j]$ party P_i first verifies that the following conditions hold:
   - s_j is the random string that is derived in the random tape generation for P_j above.
   - $\overline{m}$ equals the series of Π-messages that were broadcast in all the activations until now. (P_i knows these messages because all parties see all messages sent.)

   If any of these conditions fail, then P_i ignores the messages. Otherwise, P_i appends m to $\overline{m}$ and proceeds as in Steps 3b and 3c above.
5. **Output:** Whenever Π generates an output value, $\mathsf{Comp}(\Pi)$ generates the same output value.

Figure 14: The compiled protocol $\mathsf{Comp}(\Pi)$ [7]