# Generalized Environmental Security From Number Theoretic Assumptions

Tal Malkin[1], Ryan Moriarty[2], and Nikolai Yakovenko[3]

[1] Department of Computer Science, Columbia University, tal@cs.columbia.edu*
[2] Department of Computer Science, UCLA, ryan@cs.ucla.edu**
[3] Google, Inc. yakovenko@google.com***

**Abstract.** We address the problem of realizing concurrently composable secure computation without setup assumptions. While provably impossible in the UC framework of [Can01], Prabhakaran and Sahai had recently suggested a relaxed framework called generalized Environmental Security (gES) [PS04], as well as a restriction of it to a "client-server" setting based on monitored functionalities [PS05]. In these settings, the impossibility results do not apply, and they provide secure protocols relying on new non-standard assumptions regarding the existence of hash functions with certain properties.

In this paper, we first provide gES protocols for general secure computation, based on a new, concrete number theoretic assumption called the relativized discrete log assumption (rDLA). Second, we provide secure protocols for functionalities in the (limited) client-server framework of [PS05], replacing their hash function assumption with the *standard* discrete log assumption. Both our results (like previous work) also use (standard) super-polynomially strong trapdoor permutations.

We believe this is an important step towards obtaining positive results for efficient secure computation in a concurrent environment based on well studied assumptions. Furthermore, the new assumption we put forward is of independent interest, and may prove useful for other cryptographic applications.

## 1 Introduction

### 1.1 Background and Motivation

Since its beginnings a few decades ago, theoretical cryptography has developed by formalizing the intuitive notions of security, and basing the strength of protocols realizing these definitions on widely accepted complexity assumptions. Much success was achieved in defining and realizing secure multi-party computation, arguably the most general and

* Supported by NSF Early Career Development (CAREER) Grant CCF-0347839.
** This work was done while the author was at the department of Computer Science, Columbia University
*** This work was done while the author was at the department of Computer Science, Columbia University

important task in cryptography, in various 'stand-alone' settings. As our understanding develops side by side with new emerging needs and applications for cryptography in uncontrolled, distributed environments such as the Internet, new goals and challenges arise. An important current direction in cryptography is to model and realize secure protocols operating in such open settings, requiring concurrent composition. Intuitively, one would like to have protocols that will remain secure even if they are composed arbitrarily with other protocols. Such general composability is often what is required in practical settings. It is important to continue to base this new developing theory on well studied and scrutinized complexity assumptions.

**The UC/ES Framework** Perhaps the most well known definition of security in a composable setting is the Universally Composable (UC) Security paradigm of Canetti [Can01] (an alternative paradigm was proposed by Pfitzmann et. al. [PW00,BPW04]).
The UC security notion is based on the (by now standard) ideal world / real world simulation paradigm. Very roughly, an ideal world is defined where functions are computed by a trusted party. For a protocol to be secure, we require that for every adversary $A$ operating in the real world under a certain environment, there exists an ideal world adversary $S$ (a simulator), working in time polynomial in that of $A$, that can simulate everything that happened in the real world under $A$. This should hold under any environment (which models anything else going on in the world, provides inputs to all parties, watches their interactions, etc). Hence, the UC security notion is also referred to as the Environmental Security (ES) notion.
A major advantage of this framework is that, according to the UC Theorem [Can01], protocols that are secure in this model remain secure even when composed concurrently and arbitrarily (hence the name universally composable security). In particular, consider an arbitrary protocol $\pi$ which uses some ideal calls (using a trusted party) to compute certain functions. Replacing the ideal calls by UC-secure protocols computing the functions is safe in the sense that whatever an adversary $A$ can achieve, can be simulated in polynomial time within the ideal calls model.
Unfortunately, while this UC/ES framework is very appealing and strong in terms of the provided security guarantees, it turned out to be *too* strong. Indeed, many of the most basic cryptographic tasks (such as commitment or secure computation) were proven impossible to realize in this framework, unless additional "trust" assumptions are being made [Can01,CF01,DG03,CKL03,Lin03,Lin04] (e.g., an honest majority, or a common random string available to all parties and selected by a trusted party).

**The gES Framework** Recently, Prabhakaran and Sahai [PS04] introduced a new model of security, generalized Environmental Security (gES). Roughly, this model relaxes the security requirements of the UC/ES setting so as to avoid the impossibility results, while still rendering the model meaningful enough that a protocol secure in this model intuitively implies meaningful guarantees on its actual security if employed "in real life". We discuss the meaningfulness of these guarantees below. This framework is exciting and promising, as it allowed, for the first time, to realize multi-party computation of general functionalities without any setup assumptions, while maintaining security under a pretty general form of composability (see discussion below).

Their idea, roughly, was to perform a thought experiment where the adversary in the ideal world (the simulator) is given super-polynomial computational power (following the approach suggested by Pass [Pas03]). To allow for secure composability, the super-polynomial power in the ideal world is given through a super polynomial angel (oracle), which can answer queries based on its knowledge of who the corrupted parties are. This is the gES notion of security. We refer the reader to [PS04] for more details, but remind that the angel is only required as a tool for the security proof, and is not needed for protocol execution. For a particular angel $\Gamma$, the resulting security model is called $\Gamma$-ES. Using this model, [PS04] showed how to achieve secure multi-party computation of any functionality, against a static adversary (one who cannot corrupt parties adaptively), and without any setup assumptions. More specifically, it is shown in [PS04] that

1. For every Imaginary Angel $\Gamma$, $\Gamma$-ES protocols are universally composable.
2. There exists an Imaginary Angel $\Psi$ (under new complexity assumptions) such that there are $\Psi$-ES protocols for commitment, ZK proofs and any PPT functionality.

This result is very important towards the ultimate goal of reasonable, concurrently secure protocols, without setup assumptions. However, the result of [PS04] is based on a new, non-standard assumption, requiring the existence of a hash function with certain properties regarding distributions of collisions on inputs with the same prefixes (see further discussion in Section 4.1).

Perhaps the most important open problem left in [PS04] is to realize such secure computation without setup assumptions relying on simpler, more standard, easier to analyze, complexity assumptions. Our work provides a big step in this direction.

Very recently (and independently of our work), Barak and Sahai [BS05] have also addressed this problem, and showed how to realize such secure computation (again under a relaxed model where the simulator is super polynomial in the real-world adversary), using reasonably standard assumptions. Namely, assuming the existence of a hash function collection that is collision resistant with respect to super polynomial adversaries, and trapdoor permutations secure against super polynomial adversaries. The main advantage of our solutions, as we shall see, is their simplicity, which will hopefully be useful towards practical implementations, and more importantly, towards distilling a better understanding of this security model, it's meaning, advantages, and limitations.

Finally, we touch upon one more recent framework. Recently, Prabhakaran and Sahai [PS05] suggested a relaxation of gES, introducing

Monitored functionalities and Client-Server Computation. This relaxation aims at achieving secure computation (alas, of a limited class of functionalities) with weaker assumptions, simpler, and more efficient protocols. We do not describe or motivate this framework here, except to note that the assumptions used are still non-standard and hard to work with. On the other hand, the functionalities that can be realized, while limited, avoid the impossibility results in the standard UC model, and thus provide an interesting advance.

**Why is Security in the gES Framework Meaningful?** Before presenting our results, we discuss the meaning of security and composability in the gES framework which we use (the same discussion applies to the framework of [BS05] which also uses super polynomial simulation). In terms of secure computation of a given function, it can be argued (see [PS04,BS05]) that for most applications of secure computation, the ideal model is still "ideal enough" (or "secure enough") even when the adversary is allowed to run in time that is bounded by a specific super polynomial function (depending on the hardness assumption used).[4] Thus, proving that any adversary in the real world can be simulated by such a (strong) ideal model adversary, still provides a meaningful notion of security.

However, it is important to understand the implications of what exactly is guaranteed (and not guaranteed) by the theorem proving universal composability in this framework (be it $\Gamma$-security for some angel $\Gamma$ as used in [PS04] and in this paper, or the security notion suggested by [BS05]). What *is* guaranteed is that, given any protocol secure according to the notion at hand, the protocol remains secure even when composed in an arbitrary manner with any other arbitrary protocols. But, no guarantees are made for protocols that are *not* secure according to the notion at hand.

In particular, consider a protocol $\pi$ which is not secure according to this notion, but enjoys some other weaker security features (e.g., the protocol has some security guarantee in the ideal calls model when the adversary is polynomial time bounded). Now, composing this protocol with other protocols within the new framework (e.g., replacing ideal calls to a function $f$ with a $\Gamma$-ES secure protocol for $f$), may break the original (weaker) security guarantee that $\pi$ had. Indeed, all we know is that anything that happens with adversary $A$ can be simulated in the ideal calls model with an adversary $S$ that has access to the (super-polynomial) $\Gamma$. While the protocol for $f$ was $\Gamma$-ES secure, $\Gamma$ (which is used only as a tool in the analysis) may help "break" some other sub-protocol in $\pi$.[5] In this sense, the notion of composability guaranteed by the general theorems, is not completely "general composition" as defined by [Lin03].

[4] In fact, in many applications, the ideal model is such that even a computationally unbounded adversary cannot cause damage.

[5] In fact, $\pi$ may have been designed specifically with this goal in mind, following a "chosen protocol attack" [KSW97]. For example, if $\Gamma$ is the one used in this paper, namely providing discrete logs (breaking DLA) relative to some primes, $\pi$ could contain a part that relies on the DLA for an appropriate prime.

An argument can be made that one cannot maintain "all possible weak security properties" of insecure protocols under composition (it's not even clear how to define this), and that the (or a) right notion of security is one that guarantees security to those who use it, while insecure protocols naturally will remain insecure under composition. Moreover, one can argue that this is also the case for the standard UC-security framework: security under composition is only guaranteed when composing protocols that were UC-secure to begin with.[6] On the other hand, it seems clear (historically, intuitively, and practically), that security in an ideal calls model against a polynomial time bounded adversary (the notion of security used for UC) is an extremely natural and important security notion. It may be reasonable to assume that anything that is less secure than that is completely insecure (and thus nothing needs to be preserved under composition for protocols that are not secure according to this notion). For sure, it would be desirable to maintain this security property for protocols even when composed with other protocols in a stronger model such as $\Gamma$-ES security. It is important to note that this is, unfortunately, not the case.

At this point we leave the philosophical discussion about the general direction that [PS04] initiated, and [BS05] and the current work follow, and continue to describe our results.

### 1.2 Our Results

We provide an important step towards realizing gES with standard number theoretic assumptions that are concrete, natural, easier to study and analyze, or indeed refute.

First, we provide an instantiation of the assumptions used by [PS04], based on a new assumption, which we call the relativized discrete log assumption (rDLA), as well as a standard (strong) assumption of trapdoor permutations (TDP) secure against super polynomial adversaries. The details of these assumptions are discussed later in the paper. In particular, we obtain $\Omega$-ES protocols for arbitrary functionalities against static adversaries, with no setup assumptions (here $\Omega$ is our imaginary angel, and the security is based on the rDLA). While non-standard, the rDLA is simple to state (intuitively, it says that the DLA over a certain group holds even in the presence of oracles breaking the DLA for other groups), and strictly algebraic/number-theoretic in nature (we work over subgroups of prime order of safe primes, although our assumption and protocols could be considered over other groups). We believe the assumption is easy to understand and think about, and it seems quite reasonable.[7]

[6] For example, if a protocol $\pi$ in the ideal calls model maintained some (weak) notion of security against adversaries bounded by quadratic running time, replacing an ideal call to some $f$ by a UC-secure protocol for $f$ may break that property, as the ideal model adversary in the UC framework is allowed polynomial running time.

[7] We do not claim to be experts in number theory, although several other people that we asked also found the assumption reasonable. Further, the same somewhat philosophical arguments on the plausibility of the assumptions made in [PS04] apply here as well, except that our assumptions are easier to try to attack.

Since this assumption can be framed as an instantiation of the original [PS04] assumption (through a realization of their hash function), our work simplifies and 'cleans' the previous construction, and helps bring the gES model and the [PS04] protocols under more scrutiny, hopefully towards helping to strengthen our belief in its security. We also note that while the [BS05] construction relies on more standard assumptions, and in this sense subsumes our results, our resulting protocols are cleaner, simpler, and hopefully a step towards practically efficient concurrently composable protocols.

Second, we provide an instantiations of the assumptions used by [PS05], based on the standard discrete log assumption (DLA), as well as a standard (strong) assumption of TDP secure against super polynomial adversaries. This allows us to obtain, like [PS05], simpler protocols for a limited class of "server-client" functionalities. Perhaps more importantly, this yields the first results for concurrently secure computation without setup assumptions, under standard computational assumptions.

In sum, our work addresses the arguably most important problem left open by the recent [PS04] pioneering work, and we hope that it provides a useful step towards achieving, or at least understanding, the "holy grail" in this field. Moreover, it provides a significant improvement of the [PS05] results, replacing a new and unstudied assumption by the completely standard DLA. Finally, we believe rDLA, our new assumption, is worth studying independently of the current context, and is likely to find other cryptographic applications.

## 2 Preliminaries

We do not provide here formal definitions of the gES and related models, and refer the reader to the original papers [PS04,PS05] for definitions (as well as further justifications regarding the meaningfulness of the security model).

In all our protocols we use $k$ as the security parameter, and consider functionalities of up to a polynomial $n$ number of parties.

We assume all the parties have unique IDs, which may be adversarially chosen as long as they adhere to the legal format. In this paper, the IDs are of the form $q$ where $q$ is a safe prime, namely $q = 2p + 1$ for a prime $p$ ($p$ is called a Sophie Germain prime).

All adversaries considered in this work are PPT, non-uniform, and static (namely choose the set of parties to corrupt at the onset of computation).

For two distributions $\mathcal{X}$ and $\mathcal{Y}$ we write $X \approx Y$ to denote that they are indistinguishable by PPT circuits (with respect to the security parameter $k$).

## 3 Our Assumptions

In this section we summarize the assumptions that we will use for our results. Section 3.2 describes the rDLA assumption that is new to our work. Section 3.1 describes other assumptions that we will use, which

are standard assumptions. The assumptions previously used by [PS04] and [PS05] appear in Section 4.1 and Section 5.1.
The assumptions we use are related to the discrete log assumption (DLA), which is commonly assumed for different groups. We state our assumptions for subgroups of prime order of $Z_q^*$, specifically for the subgroup $G$ of the quadratic residues, when $q = 2p + 1$ is a safe prime (this can be somewhat extended).

### 3.1 Standard Assumptions

We sketch these assumptions without full formal details.

*The Discrete Log Assumption (DLA)* For any PPT adversary A, consider the following probabilistic experiment: choose a random safe prime $q = 2p + 1$ of length $k$, let $G$ be the subgroup of order $p$ (all quadratic residues) of $Z_q^*$, let $g$ be a generator of $G$, and choose a random $y \in G$. Then, the probability that $A(q, g, y) = x \in Z_p^*$ such that $y = g^x$ is negligible.

*The Discrete-Log-Safe Trapdoor Permutation Assumption (DLS-TDP)* For $n$ polynomial in the security parameter $k$, there exists a family of trapdoor permutations over $\{0, 1\}^n$, that remain secure against adversaries with access to an oracles solving discrete logarithms in the subgroups of $Z_q^*$, for safe primes $q$ of size $k$.
Note that the above assumption is implied by the (more standard assumption of) existence of trapdoor permutations secure against adversaries with super polynomial power $2^{n^\epsilon}$. Indeed, if such strong TDP exist, we can choose $n = k^{1/\epsilon}$. Then in time $2^k$ the discrete log problem can be solved, but the TDP remains secure, implying the DLS-TDP assumption.

### 3.2 The Relativizing Discrete Log Assumption (rDLA)

Let $q = 2p+1$ be a safe prime of size $k$, and let $G$ be the subgroup of size $p$. Then the discrete log problem over $G$ is hard (i.e., no PPT adversary can compute discrete logs with non-negligible probability), even when the adversary has access to oracles that solve the discrete log problem for any input in any other group defined by a safe prime $q' \neq q$ of size $k$. Intuitively, the rDLA assumes some sort of "non-malleability" among different groups, asserting that being able to take discrete logs in all other groups of the same size, will not help an adversary take discrete logs in the given group.

## 4 Achieving gES

In this section we will show how to use rDLA and DLS-TDP to realize secure multi-party computation (for static adversaries) in the gES framework, without any setup assumptions. We do that by showing an instantiation of the assumed hash function in [PS04], proving it satisfies the required properties, and presenting the resulting Angel and protocols for general secure multi-party computation.

### 4.1 The Assumptions and Angel of [PS04]

The constructions of [PS04] rely on the following assumptions[8]:
Assume there exists a hash function $\mathcal{H} : \{0,1\}^k \to \{0,1\}^l$. The input to $\mathcal{H}$ is of the form $(\mu, r, x, b) \in \mathcal{J} \times \{0,1\}^{k_1} \times \{0,1\}^{k_2} \times \{0,1\}$, where $\mathcal{J}$ is the set of IDs of the parties (each party is assumed to have a unique ID, possibly chosen adversarially), and $k_1$, $k_2$, and $l$ are all polynomial in $k$. It is assumed that $\mathcal{H}$ satisfies the following.

**A1** (Collisions and Indistinguishably): For every $\mu \in \mathcal{J}$ 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 $\{\mathcal{D}_{r'}^{\mu'}|\mu' \in \mathcal{J}, r' \in \{0,1\}^{k_1}\}$, these distributions still remain indistinguishable. (Intuitively, this assumption states that there are collisions in the hash function, which are indistinguishable from a random hash of a 0 or a 1).

**A2** (Difficult to find collisions with same prefix): For all PPT circuits $M$ and every id $\mu \in \mathcal{J}$, 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 $\{\mathcal{D}_{r'}^{\mu'}|\mu' \neq \mu, r' \in \{0,1\}^{k_1}\}$. (Note that without this last requirement, insisting that finding collisions remains difficult even when given sampling access to collisions for other $\mu'$, a hash function satisfying these properties could be constructed under standard assumptions).

Additionally, [PS04] also rely on the following assumption:

**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 $\mu$ and $r$.

As discussed above (and in [PS04]), A3 can be replaced by the (stronger, but more standard and natural looking) assumption of TDP secure against super-polynomial adversaries.

*The Angel $\Psi$:* [PS04] use the following imaginary angel $\Psi$. On query $(\mu, r)$, $\Psi$ checks whether $\mu$ is one of the corrupted parties. If so, $\Psi$ outputs a sample from $D_\mu^r$. If not, $\Psi$ returns $\perp$.

### 4.2 Our Hash Function

We propose the following hash function $\mathcal{H}_0$, point its correspondence to the [PS04] hash function, and prove that (under our assumptions) it realizes their required assumptions A1,A2,A3.

[8] This text is extracted almost verbatim from [PS04].

**Defining $\mathcal{H}_0$** $\mathcal{H}_0 : \{0,1\}^k \to \{0,1\}^l$ is defined as follows. The input to $\mathcal{H}_0$ is of the form $(q, g_0, g_1, x, b)$, where:

- $q = 2p + 1$ is a safe prime (namely, $p$ is a prime as well) of length $k_1$, where $k_1$ is polynomially related to $k$. (This corresponds to the party ID $\mu$).[9]
- $g_0, g_1$ are generators of $G = QR(Z_q^*)$. Equivalently, each of $g_0$, $g_1$ is a quadratic residue not equal to 1 in $Z_q^*$. (This corresponds to $r$).
- $x \in Z_p^*$
- $b \in \{0,1\}$.

The output of $\mathcal{H}_0$ is then defined as:

$$\mathcal{H}_0(q, g_0, g_1, x, b) = g_b^x \bmod q.$$

We note that it is easy to efficiently check whether the input is of the correct form, by using primarily testing, and testing whether $g_i$ is a quadratic residue by computing its Legendre symbol. It is also easy to generate inputs to $\mathcal{H}_0$; Choosing random generators $g_0, g_1$ of the QR subgroup can be done by simply choosing a random element of $Z_q^*$ and squaring it. Choosing a safe prime $q$ is easy assuming that safe primes (or Sophie Germain primes) are dense.

**Satisfying A1, A2 and A3** We next show that, instantiating $\mathcal{H}$ with $\mathcal{H}_0$, A1 is satisfied unconditionally, A2 is satisfied if rDLA holds, and A3 is satisfied if the DLS-TDP assumption holds.

**Lemma 1.** *$\mathcal{H}_0$ satisfies A1.*

*Proof.* We need to show that for all safe primes $q$ of size $k_1$ and all $g_0, g_1 \neq 1$ quadratic residues in $Z_q^*$, there exists a distribution $\mathcal{D}_{g_0,g_1}^q$ over $\{(x,y,z) | g_0^x = g_1^y = z \mod q\}$, such that

$$\{(x,z) | (x,y,z) \leftarrow \mathcal{D}_{g_0,g_1}^q\} \approx \{(x,z) | x \leftarrow Z_p^*, z = g_0^x \mod q\}$$
$$\{(y,z) | (x,y,z) \leftarrow \mathcal{D}_{g_0,g_1}^q\} \approx \{(y,z) | y \leftarrow Z_p^*, z = g_1^y \mod q\}$$

We take $\mathcal{D}_{g_0,g_1}^q$ to be a distribution that outputs $(x,y,z)$ where $x \in Z_p^*$ is chosen at random, $z = g_0^x \mod q$, and $y \in Z_p^*$ is the unique element satisfying $z = g_1^y \mod q$. Then, it is clear that the above distributions are identical (and in particular indistinguishable for any distinguisher, even if given sampling access to other $\mathcal{D}_{g_0',g_1'}^{q'}$). □

**Lemma 2.** *If rDLA holds, then $\mathcal{H}_0$ satisfies A2.*

*Proof.* We need to prove that, informally, for every safe prime $q$ of size $k_1$, and for randomly generated $g_0, g_1$, it is hard to output a collision $(x,y)$ such that $g_0^x = g_1^y$. This should hold even for an adversary with access to such collision distributions for other safe primes $q'$ of length $k_1$. Indeed, any PPT circuit $M$ that outputs collisions for $q$ (given random $g_0, g_1$) with non-negligible probability, can be converted to a PPT $M'$

[9] As usual, we consider the party IDs to be unique, and possibly adversarially chosen, subject to the required format.

which finds the discrete log in the subgroup $G$ of quadratic residues with non-negligible probability. Specifically, given a generator $g$ and a random element $z \in G$, $M'$ can proceed by choosing $g' = g^r$ for a random $r$, and running $M(g', z)$. If $M$ outputs a collision $(x, y)$ such that $g'^x = z^y$, $M'$ can output $rxy^{-1} \mod p$. If such an $M$ exists, this contradicts the DLA for the corresponding subgroup of $Z_q^*$. If there is such an $M$ that uses access to distributions $\mathcal{D}^{q'}_{g'_0, g'_1}$ for $q' \neq q$, since these distributions can easily be simulated using an oracle that provides discrete logarithms in the QR subgroup of $Z_{q'}^*$, this contradicts the rDLA. □

**Lemma 3.** *If the DLS-TDP assumption holds, then A3 holds.*

*Proof.* This follows immediately from the fact that for all safe primes $q$ and subgroup generators $g_0, g_1$, providing collisions $(x, y, z)$ such that $g_0^x = g_1^y = z$ is equivalent to providing the discrete logarithm of $g_0$ with respect to $g_1$. □

### 4.3 Our Angel $\Omega$

Following the $\Gamma - ES$ Angel model, we use a super-polynomial imaginary angel we call $\Omega$, that breaks the security of already corrupted parties. Specifically, $\Omega$ is the following. On a query $(q, g_0, g_1)$ (of the usual format), $\Omega$ checks whether $q$ is the ID of one of the corrupted parties. If so, $\Omega$ returns $a$ such that $g_0 = g_1^a \mod q$ (that is, return the discrete log of $g_0$ with respect to $g_1$). If $q$ is the ID of a party that is not corrupted, $\Omega$ returns $\perp$.
We remark that we could have used the imaginary angel $\Psi$ from [PS04], instantiated with our hash function $\mathcal{H}_0$. Then, the resulting imaginary angel would have outputted a distribution of collisions for the given $q$, $g_0$ and $g_1$, instead of the discrete logarithm $a$. These outputs are clearly equivalent, and we chose to present our angel $\Omega$ as above since it is a somewhat simpler, cleaner choice.

### 4.4 Putting the Pieces Together: Secure Multi-Party Computation in the $\Omega$-ES Model

We have shown how to realize the hash function, angel, and assumptions required for the constructions of [PS04] using the number theoretic assumption rDLA, and the DLS-TDP (or TDP secure against super polynomial adversaries). This allows the main result from [PS04], namely general secure multi-party computation secure against static adversaries without any setup assumptions, to go through in our setting.

**Theorem 1.** *If rDLA and the DLS-TDP assumption hold, there is a protocol that $\Omega$-ES realizes any multi-party functionality against static adversaries.*

*Proof.* This follows immediately from the previous sections, together with Theorem 3 in [PS04]. All of the protocols necessary for the proof are instantiated directly and the proofs follow from the realization of the assumptions. □

For illustration, and some self-containment, we sketch the general outline of the secure multi-party computation protocol, and present the resulting construction of the basic building blocks, the functionality $\mathcal{F}_{\widetilde{\mathrm{COM}}}$ (in Figure 1) and the protocol BCOM (in Figure 2) that realizes it, within our framework.

*Basic Commitment Semi-Functionality* $\mathcal{F}_{\widetilde{COM}}$ Following the construction from [PS04], we first implement the basic IDEAL commitment semi-functionality $\mathcal{F}_{\widetilde{\mathrm{COM}}}$[10]. We instantiate the REAL protocol BCOM from [PS04] with our hash function and then prove that BCOM $\Omega$-ES realizes the semi-functionality $\mathcal{F}_{\widetilde{\mathrm{COM}}}$. Since $\mathcal{F}_{\widetilde{\mathrm{COM}}}$ is not fully ideal, we need to prove that the commitment is binding separately.
With the help of the $\Omega$ angel, for every PPT adversary $A^{\Omega}$, we can demonstrate a PPT simulator $S^{\Omega}$ such that no PPT environment can distinguish between the REAL interaction with $A^{\Omega}$, and the IDEAL interaction with $S^{\Omega}$. This proves that BCOM is an $\Omega$-ES realization of $\mathcal{F}_{\widetilde{\mathrm{COM}}}$.
To show that the semi-functionality $\mathcal{F}_{\widetilde{\mathrm{COM}}}$ is binding for a corrupt sender $C$ for any environment, we rely upon the rDLA. We show a polynomial reduction from the problem in which a machine $M$ breaks the binding of the commitment scheme $\mathcal{F}_{\widetilde{\mathrm{COM}}}$, to a problem where an adversary uses $M$ to break the security of the rDLA. By assumption, the later is impossible with better than negligible probability, so the commitment is binding.

**Functionality $\mathcal{F}_{\widetilde{\mathbf{COM}}}$**

The parties are sender $C$ and receiver $R$, with adversary $\mathcal{S}^{\Omega}$. The security parameter is $k$, and $q_C, q_R$ are safe primes of size $k_1$ polynomial in $k$.
COMMIT PHASE:

1. $\mathcal{F}_{\widetilde{\mathrm{COM}}}$ picks random quadratic residues $g_0, g_1 \leftarrow QR(q_R)$ and sends them to $C$.
2. $\mathcal{F}_{\widetilde{\mathrm{COM}}}$ receives $c$ from $C$.
3. $\mathcal{F}_{\widetilde{\mathrm{COM}}}$ sends the message COMMIT to $R$.

REVEAL PHASE:

1. $\mathcal{F}_{\widetilde{\mathrm{COM}}}$ receives $(b, x)$ from $C$.
2. $\mathcal{F}_{\widetilde{\mathrm{COM}}}$ checks if $c = g_b^x$. If so, then send message (REVEAL,$b$) to $R$ and adversary $S^{\Omega}$.

**Fig. 1.** The basic commitment functionality $\mathcal{F}_{\widetilde{\mathrm{COM}}}$

[10] A semi-functionality is one that is not fully ideal, therefore its ideal properties must be proved separately. See [PS04]

**Protocol BCOM**

The parties are sender (committer) $C$ and receiver $R$. The security parameter is $k$, and $q_C = 2p_C + 1, q_R = 2p_R + 1$ are safe primes of size $k_1$ polynomial in $k$.

COMMIT PHASE:

1. $R$ picks random quadratic residues $g_0$, $g_1 \leftarrow QR(q_R)$ and sends them to $C$.
2. $C$ chooses $x \leftarrow Z^*_{p_R}$ and computes $c = g_b^x$. $C$ requests $\mathcal{F}_{\mathrm{ENC}}$ to send $c$ to $R$.
3. $R$ receives $c$ from $\mathcal{F}_{\mathrm{ENC}}$ and accepts the commitment.

REVEAL PHASE:

1. $C$ requests $\mathcal{F}_{\mathrm{ENC}}$ to send $(b,x)$ to $R$, and $R$ receives.
2. $R$ checks if $c = g_b^x$. If so, he accepts $b$ as the reveal.

**Fig. 2.** The basic commitment protocol BCOM that $\Omega$-ES realizes $\mathcal{F}_{\widetilde{\mathrm{COM}}}$

*Building the Rest Of the Tools* We follow [PS04] directly to build the rest of the tools need for secure multi-party computation. In the $\mathcal{F}_{\widetilde{\mathrm{COM}}}$-hybrid model we build a multi-bit commitment semi-functionality $\mathcal{F}^*_{\widetilde{\mathrm{COM}}}$[11] and a zero knowledge semi-functionality $\mathcal{F}_{\widetilde{ZK}}$, with corresponding realizations BCOM* and BZK. Proving that that these protocols realize their functionalities does not require the use of angels. The angel $\Omega$ is only only to prove the realization of the basic commitment semi-functionality.
The protocols BCOM* and BZK are then used to build the protocol COM, realizing the standard (fully ideal) commitment functionality $\mathcal{F}_{\mathrm{COM}}$. In order to realize $\mathcal{F}_{\mathrm{COM}}$, however, we need to use the DLS-TDP assumption. In the $\mathcal{F}_{\mathrm{COM}}$-hybrid model, there are known protocols for zero knowledge and for a broadcast channel. The zero knowledge functionality $\mathcal{F}_{ZK}$ has a realization due to Canetti and Fischlin [CF01]. The broadcast channel functionality $\mathcal{F}_{BC}$ is due to Goldwasser and Lindell in [GL02]. The proof for both of these realizations are information theoretic, and so they hold in the $\Omega$-ES model, and do not rely on angels.
With the help of $\mathcal{F}_{\mathrm{COM}}$, $\mathcal{F}_{ZK}$ and $\mathcal{F}_{BC}$, we can now realize the protocol OM-CP for the ideal functionality for one-to-many commit-and-prove, $\mathcal{F}_{CP}^{1:M}$. Again, the proof of this construction does not rely upon the existence of angels.

[11] In our model we could also use a Pedersen commitment for the multi-bit commitment, but we felt that our single bit commitment was easier to understand and that our proofs are much simpler if we just instantiate the hash function from [PS04], rather than create completely new protocols. Using the Pedersen commitment could be of independent interest.

Now we have all of the tools needed to perform general multiparty computation against static adversaries.

*Secure Multi-Party Computation* We can now build general MPC following [CLOS02,PS04]. The proofs in [CLOS02] are information theoretic and will therefore carry over to the $\Omega$-ES model.
Following the result of Lemma 2 in [PS04], using the DLS-TDP assumption, we can create a protocol for any functionality $\mathcal{F}$ that is secure against all semi-honest static adversaries. A full explanation appears in [CLOS02].
Now following Lemma 3 in [PS04], we know there exists a compiler that can turn any protocol secure against semi-honest static adversaries into a protocol secure against all static adversaries. The proof relies on having access to a functionality for one-to-make commit-and-prove. Since we have the protocol OM-CP that $\Omega$-ES realizes one-to-many commit-and-prove functionality $\mathcal{F}_{CP}^{1:M}$, we now have a way of performing any multiparty computation.

## 5 Monitored Functionalities and Client-Server Computation Based on DLA

In [PS05] Prabhakaran and Sahai were able to show how to do any multiparty computation in a "client server model" called "client-server computation". This work is done in the UC/ES framework, but there are many limitations on this model. In the model one party is dedicated as the "server" and all the other parties are "clients". The client receives as output a function of its input and the server's input, while the server receiver as output the client's input. There is, however, the extra security limitation that the server's input to the function is not necessarily independent of the client's input, unless the client had never used that input previously (for more precise details, the reader is referred to [PS05]).
While "client-server computation" is implied by our earlier results in this paper of any multi-party computation we are able to achieve this on much simpler assumptions. With just the standard Discrete Log Assumption and DLS-TDP Assumption against non-uniform adversaries we can show how to do any type of client-server computation (as before, the DLS-TDP assumption is implied by the more standard TDP against super-polynomial adversaries).
The significance of this work is not in the power of the model, but in the fact that we were able to work past inherent restrictions in the UC model using such widely accepted assumptions.
We achieve these results in a similar fashion as in Section 4. We instantiation the hash function used in [PS05], but this time we need only rely on the DLA to do so.

### 5.1 The Assumptions and Angel of [PS05]

The assumptions made by [PS05] and their imaginary angel, are similar but weaker versions of those used by [PS04]. In particular, the ID of the party is not necessary as an input to the hash function, nor needed by the angel. Intuitively, the reason is that the role played by a certain party (e.g., a server vs a client) does not change across different executions. The imaginary angel can thus decide whether to answer the query in a useful manner based on the identity of the corrupted parties, without requiring the ID. Details follow.

The constructions of [PS05] rely on the following assumptions.

Assume there exists a hash function $\mathcal{H} : \{0,1\}^k \to \{0,1\}^l$. The input to $\mathcal{H}$ is of the form $(r,x,b) \in \{0,1\}^{k_1} \times \{0,1\}^{k_2} \times \{0,1\}$, where $k_1, k_2$, and $l$ are all polynomial in $k$. It is assumed that $\mathcal{H}$ satisfies the following properties:

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

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

Further, given sampling access to $\mathcal{D}_r$ to a distinguisher, these distributions still remain indistinguishable.

**A′2** (Difficult to find collisions with same prefix): For all PPT circuits $M$, for a random $r \leftarrow \{0,1\}^{k_1}$, the probability that $M(r)$ outputs $(x,y)$ such that $\mathcal{H}(r,x,0) = \mathcal{H}(r,y,1)$ is negligible.[12]

Additionally, for most of their results, [PS05] also use the following assumption:

**A′3** There exists a family of trapdoor permutations which remains secure even when the adversary is given sampling access to $\mathcal{D}_r$ for all $r$.

Note that each of these assumptions is weaker than (i.e., implied by) the corresponding assumption from [PS04] described in Section 4.1.

*The Angel $\Gamma$:* [PS04] use the following imaginary angel $\Gamma$. On query $r$, $\Gamma$ checks whether the server is corrupted or not. If so, $\Gamma$ returns $\perp$. If not, $\Gamma$ outputs a sample from $\mathcal{D}_r$.

### 5.2 Our Hash Function

Here, we instantiate the hash function using exactly the same hash function $\mathcal{H}_0 : \{0,1\}^k \to \{0,1\}^l$ as we defined in Section 4.2, that is,

$$\mathcal{H}_0(q, g_0, g_1, x, b) = g_b^x \bmod q.$$

However, this time $(q, g_0, g_1)$ correspond to $r$ from the [PS05] constructions (rather than $q$ corresponding to an ID). This means, for example, that assumption A′2 (difficulty of collision finding) is required to hold when $q$ (as well as the generators) is chosen randomly (not necessarily for every $q$).

[12] Notice that here, unlike the corresponding assumption A2 from Section 4.1, $M$ does not get access to oracles for the collision finding distributions for with other parameters. This is what will allow us to realize this requirement relying only on DLA, and not rDLA.

### Satisfying A′1, A′2, A′3

We show that using our $\mathcal{H}_0$ to instantiate the hash function, A′1 is satisfied unconditionally, A′2 is satisfied if (a standard) DLA holds, and A′3 is satisfied if DLS-TDP holds. This will follow easily using the same arguments as we used in Section 4.2.

**Lemma 4.** *$\mathcal{H}_0$ satisfies A′1.*

*Proof.* We showed in Section 4.2 that $\mathcal{H}_0$ satisfies A1. Since A1 implies A′1 (in fact, they are equivalent here), it immediately follows that $\mathcal{H}_0$ satisfies A′1. □

**Lemma 5.** *If DLA holds, then $\mathcal{H}_0$ satisfies A′2.*

*Proof.* Similarly to the proof of Lemma 2, if a PPT circuit $M$ outputs collisions for a random $q, g_0, g_1$, it can be converted to a PPT circuit $M'$ that computes discrete logs random elements of the quadratic residue subgroup of randomly chosen primes. This contradicts DLA.[13] □

**Lemma 6.** *If the DLS-TDP assumption holds, then A′3 holds.*

*Proof.* This is identical to Lemma 3. □

## 5.3 Our Angel $\Delta$

Our imaginary angel $\Delta$ will first check if the server $\mathcal{S}$ is corrupted. If $\mathcal{S}$ is corrupted $\Delta$ will return $\perp$ on any query. If $\mathcal{S}$ is not corrupted then on input $(q, g_0, g_1)$ $\Delta$ will compute the discrete logarithm of $g_0$ with respect to $g_1$, namely return $a$ such that $g_0 = g_1^a \mod q$.

## 5.4 Putting the Pieces Together: Main Theorems for This Model

By proving that our hash function realizes the properties required by [PS05], all their results automatically translate to hold under our angel $\Delta$, using our assumptions. This allows to achieve the first protocols in a (partially) composable model under very standard and widely acceptable assumptions such as the DLA, without any trusted setup, and avoiding the impossibility results of the UC model.
Below we briefly discuss the resulting theorems, and refer the reader to [PS05] for definitions and in-depth discussion of the functionalities and semi-functionalities achieved, as well as the security model. In the full version of the paper we will explicitly present the resulting protocols.

[13] Note that the last step in the proof of Lemma 2, dealing with the access $M$ has to collisions for other $Z^*_{q'}$ is not necessary here, due to the weakened assumption A′2.

*Monitored Commitment, Zero Knowledge Proof, and Commit and Prove under DLA.* In [PS05] Prabhakaran and Sahai achieve protocols that $\Gamma$-ES-realize the monitored functionalities $\langle \mathcal{F}_{\widetilde{\mathrm{COM}}} \rangle$, $\langle \mathcal{F}_{\widetilde{ZK}} \rangle$ and $\langle \mathcal{F}_{\widetilde{CAP}} \rangle$ given in [PS05] under assumptions A$'$1 and A$'$2. Thus we can achieve the protocols for these monitored functionalities with just the Discrete Log Assumption. While these are a means to achieve "client-server computation" they are also of independent interest. The protocols will remain the same as in [PS05] except with $\mathcal{H}$ instantiated with our DL-based $\mathcal{H}_0$.

**Theorem 2.** *Under the Discrete Log Assumption, protocols COM, ZK and CAP $\Delta$-ES-realize monitored functionalities $\langle \mathcal{F}_{\widetilde{COM}} \rangle$, $\langle \mathcal{F}_{\widetilde{ZK}} \rangle$ and $\langle \mathcal{F}_{\widetilde{CAP}} \rangle$*

*Client-Server Computation Under DLA and DLS-TDP Assumption.* In [PS05] Prabhakaran and Sahai achieve "client-server computation" under A$'$1, A$'$2 and A$'$3 and angel $\Gamma$. Thus we can achieve "client-server computation" with the Discrete Log Assumption and the DLS-TDP Assumption (or the stronger assumption of TDP secure against super polynomial adversaries). The Client-Server Computation Protocol (CSC) is the same as [PS05].

**Theorem 3.** *There is a protocol which $\Delta$-ES-realizes monitored functionality $\langle \mathcal{F}_{\widetilde{CSC}} \rangle$ against static adversaries, under the Discrete Log Assumption and the DLS-TDP Assumption.*

## Acknowledgments

We are grateful to Boaz Barak, Ran Canetti, Yehuda Lindell, Manoj Prabhakaran, and Amit Sahai for useful discussions. In particular, it was during a conversation with Boaz, Amit, and Manoj that Manoj suggested we try to present our results through an instantiation of the [PS04] hash function, rather than reproving our protocols from our assumptions. We also thank Zeph Grunschlag, Stephen Miller, Rafi Ostrovsky, and Carl Pomerance for helpful discussions and pointers regarding the feasibility of rDLA. Finally, we thank the anonymous referees for suggestions about the presentation.

## References

[BPW04] Michael Backes, Birgit Pfitzmann, and Michael Waidner. A general composition theorem for secure reactive systems. In *Theory of Cryptography – TCC 2004*, pages 336–354, 2004.

[BS05] Boaz Barak and Amit Sahai. How to play almost any mental game over the net – concurrent composition via super-polynomial simulation. In *Proc. of the 46th Annu. IEEE Symp. on Foundations of Computer Science*, 2005. To appear.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *Proc. of the 42nd Annu. IEEE Symp. on Foundations of Computer Science*, pages 136–145, 2001.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In *Advances in Cryptology – CRYPTO 2001*, pages 19–40, 2001.

[CKL03] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In *Advances in Cryptology – EUROCRYPT 2003*, pages 68–86, 2003.

[CLOS02] Ran Canetti, Yehuda Lindell, Rafail Ostrovsky, and Amit Sahai. Universally composable two-party and multi-party secure computation. In *Proc. of the 34th Annu. ACM Symp. on the Theory of Computing*, pages 494–503, 2002.

[DG03] Ivan Damgård and Jens Groth. Non-interactive and reusable non-malleable commitment schemes. In *Proc. of the 35th Annu. ACM Symp. on the Theory of Computing*, pages 426–437, 2003.

[GL02] Shafi Goldwasser and Yehuda Lindell. Secure computation without agreement. In *Proc. of the 16th International Conference on Distributed Computing (DISC)*, pages 17–32, 2002.

[KSW97] John Kelsey, Bruce Schneier, and David Wagner. Protocol interactions and the chosen protocol attack. In *Proc. of 5th International Security Protocols Workshop*, volume 1361 of *Lecture Notes in Computer Science*, pages 91–104. Springer, 1997.

[Lin03] Yehuda Lindell. General composition and universal composability in secure multi-party computation. In *Proc. of the 44rd Annu. IEEE Symp. on Foundations of Computer Science*, pages 394–403, 2003.

[Lin04] Yehuda Lindell. Lower bounds for concurrent self composition. In *Theory of Cryptography – TCC 2004*, pages 203–222, 2004.

[Pas03] Rafael Pass. Simulation in quasi-polynomial time, and its application to protocol composition. In *Advances in Cryptology – EUROCRYPT 2003*, pages 160–176, 2003.

[PS04] Manoj Prabhakaran and Amit Sahai. New notions of security: achieving universal composability without trusted setup. In *Proc. of the 36th Annu. ACM Symp. on the Theory of Computing*, pages 242–251, 2004.

[PS05] Manoj Prabhakaran and Amit Sahai. Relaxing environmental security: Monitored functionalities and client-server computation. In *Theory of Cryptography – TCC 2005*, pages 104–127, 2005.

[PW00] Birgit Pfitzmann and Michael Waidner. Composition and integrity preservation of secure reactive systems. In *Proc. of the 7th Annu. ACM Conference on Computer and Communications Security (CCS '03)*, pages 245–254, 2000.