Fully Simulatable Multiparty Computation

Yevgeniy Dodis* Rafael Pass† Shabsi Walfish‡

September 29, 2004

Abstract

We introduce and realize the notion of *fully simulatable* multiparty computation. Unlike any of the previous models, our notion *simultaneously* enjoys the following features:

- **Main feature:** The simulator does not have *any* extra power over the "real-life" adversary. In particular, it *cannot* program any public parameters or run in super-polynomial time. Thus, our implementation is *fully deniable* for tasks such as authentication and zero-knowledge (unlike the previous solutions in the common reference string model).
- Universal composability (in particular, straight-line simulation).
- No PKI (although there exists one "non-programmable" public key; see below).
- Adaptive security.

We remark that it might seem impossible to realize all (or even the main) of the above features, even for relatively simple tasks such as zero-knowledge [10]. The way we overcome this apparent contradiction is by introducing a polynomial-time, *fully off-line* trusted party T to our model. T publishes a single certified public key pk and never has to participate again in any of the protocols. However, any party P has an option of contacting T and requesting an identity-based secret key sk_P. We stress, though, that no honest party actually *needs* to (and correspondingly *will not*) contact T, while the security will hold even against corrupted parties who do contact T. We believe that the addition of fully off-line T is a minimal and very realistic way to overcome the impossibility results in the "standard" model. Additionally, the introduction of T could naturally support other desirable properties impossible in the standard model (such as optimistic fairness with faulty majority).

The main building block of our construction is the notion of *identity-based chameleon hash functions* [1]. We give an elegant, generic construction of such hash functions from any signature scheme possessing a certain Σ-protocol. By showing several efficient implementations of such protocols, we give the first constructions of identity-based chameleon hash functions without random oracles, which is of independent interest.

1 Introduction

Secure multiparty computation is one of the most general and celebrated achievements of modern cryptography. Originating from the pioneering work of Goldrecih, Micali and Wigderson [9], the security of multiparty computation was defined by the means of the *simulation* paradigm. Namely, first one defines the given functionality in the "ideal" model, where the trusted party is available to perform the task at hand by serving as an "uncorruptible" intermediary between the non-communicating parties. Then one shows the realization of this functionality in the "model", where the trusted party is gone and players can communicate to each other. Finally — and this is the key step — one has to argue that for any polynomial adversary A in the real model there exists an efficient adversary (called the simulator) S in the ideal model which can "computationally reproduce the damage caused by A". Intuitively, this means that A really did not gain anything it could not have gotten in the ideal model — by simply running S in his head.

The existence of such secure multiparty protocols seems remarkable, since the simulator is extremely limited in its capabilities. And, indeed, there are many caveats in various feasibility results. For example, most feasibility results

*dodis@cs.nyu.edu. New York University, Department of Computer Science, 251 Mercer St., New York, NY 10012 USA.
†rafael@nada.kth.se. KTH, Nada, S - 100 44 Stockholm, Sweden.
‡walfish@cs.nyu.edu. New York University, Department of Computer Science, 251 Mercer St., New York, NY 10012 USA.

of the late 80's and early 90's only held in the so called *standalone* setting, where no other functionality can be run concurrently with the one being implemented. Indeed, the main technique of such protocols which allowed the simulator to gain advantage over the real-life adversary was *rewinding*. With many protocols running at the same time, the simulation is easily seen to become exponential. Motivated by these considerations, several researchers proposed various frameworks for achieving various forms of *concurrent composability*, where many protocols can be run at the same time. Among other less important details, the main feature of such models was the (necessary) requirement that the simulator has to be "non-rewinding". Perhaps the most popular model of security along these lines is the *universal composability* (UC) framework of Canetti [4]. The attractive feature of this framework is the composition theorem, which states that universally composable protocols can be concurrently run with arbitrary other protocols.

Unfortunately, the UC framework is so strong, that many tasks such as commitment, zero-knowledge and general multiparty computation cannot be realized without relaxing the model [4, 5, 6]. Recently, several such relaxation have been proposed [7, 12, 2] which we briefly survey before describing our solution.

Common Reference String (CRS) Model. This model assumes that at the system initialization there exists an honestly generated random string R (or more generally, some public key pk whose distribution is not necessarily uniform) available to all the parties. The "real" power of this relaxation, however, does not actually come from the fact that R is truly random — in practice[1] one can find an abundant number of randomly looking strings — but rather from the fact that the simulator S is allowed to generate its own string R' (as long as it looks indistinguishable from R). This seemingly harmless advantage given to S turns out to be very significant. Indeed, the UC security in the CRS model states that whatever A could achieve in the real model, it could have achieved in the ideal model *provided it had the power to generate* R. But in the real world the parties presumably know that R was not generated by the adversary, so it is not really true that the adversary did learn something he could not have learned in the ideal model. For example, consider the task of non-interactive zero-knowledge which is possible in the CRS model. In the ideal model, the verifier simply learns a binary value "yes" or "no" stating whether the statement x is true or false. Clearly, in the ideal model it cannot convincingly transfer this conclusion to some third party (as long as x is "non-trivial"). However, if the party gets an actual non-interactive proof π in the real model, it can now obviously convince any third party that x is true — by simply presenting π, and despite the fact that it could have simulated π had he had the power to generate R! Indeed, Pass [10] formally pointed out that the CRS model does not give any advantages over the standard model for the goal of achieving *deniable* zero-knowledge proofs (where the proof should leave no transferable evidence that the interaction took place).

As we can see, the main reason deniability was lost came from the fact that *the simulator* S *was given an advantage over the real adversary* A, so that one can no longer argue that A did not learn anything it could not have learned in the ideal model. To summarize, the CRS model loses the appeal of the simulation paradigm when the parties know that the CRS was not adversarilly "programmed".[2]

Super-Polynomial Simulation. In this relaxation, originally considered by Pass [11] in the context of zero-knowledge proofs and later by Prabhakaran and Sahai [12] in the general UC context, the simulator is again given given an extra power over the adversary. Specifically, it can run in super-polynomial time. As was observed by [11, 12], for many applications the resulting security notion is strong enough to imply meaningful results. Essentially, this is the case when the ideal-model security is information-theoretic, or the allowed super-polynomial running time of the simulator is still smaller than the super-polynomial hardness assumption used to argue the security of the system. However, this new framework — termed relaxed Environmental Security (rES) by [12] — once again loses many of the advantages of the UC framework. Most importantly, it is no longer closed under concurrent composition, which was one of the main desiderata for modern multiparty computation. This is not surprising, since when arguing the security of the composed system the real-life adversary has to run the simulator for a smaller sub-system, which it can no longer do.

PKI Model. In this model, it is assumed that every party has a certified public key known to all other parties. In particular, this implicitly assumes the introduction of a trusted authority C who can certify all the keys and whom everybody trusts. As was very recently shown by Barak, Canetti, Nielsen and Pass [2], one can achieve universally composable protocols in the PKI model, where the simulator no longer has an extra power over the

[1] In theory, of course, coin-flipping is one of the tasks which are impossible to achieve in the basic UC framework [6].

[2] For example, if the CRS is chosen based on some physical phenomenon such as the location of sunspots.

adversary (however, unlike our solution below, this guarantee holds only against a *static* adversary). Of course, the main disadvantage of this approach is the fact that every user has to register his key and get correct keys of all the other players.

Angel-based Security. In this model, recently introduced by Prabhakaran and Sahai [12], all the parties — including the adversary and the simulator — are given oracle access to the so called [12] "imaginary angel" G. This model is correspondingly called G-ES (or G-UC) since it is shown to be universally composable, for any angel G. Unfortunately, when G is efficient, G-UC model becomes a regular UC model, and all the impossibility results come back. On the other hand, [12] observe that super-polynomial angels G can conceivably overcome the impossibility results of the regular UC model. In fact, based on non-standard computational assumption, [12] construct an exponential angel G^* such that one can securely implement any multiparty functionality in the G^*-UC (non-adaptive) model. Moreover, honest parties never need to utilize the angel. On a positive side, this shows that if the angel G^* indeed existed in the real model *and* did not seem to violate the security of the functionality in the ideal model, then the adversary A did not learn anything beyond what he could have learned (with the help of G^*) in the ideal model. On a negative side, since G^* is not efficiently computable, it is not clear what is the practical relevance of this conclusion in the model when everybody knows that there is no such G^*. To summarize, this model fairly lifts the power of *both* the simulator and the adversary, but does not guarantee simulatability when the adversary clearly cannot utilize this "extra power" in the real model.

1.1 Our Model

Our model improves the model of angel-based security in several significant ways, making the simulation guarantees meaningful in the actual real-life model.

First, we introduce an *efficient* trusted party T who can be viewed as a "real" analog of the "imaginary" angel. At system initialization, T will pick a public key pk and a corresponding master secret key msk. Now, every party P has an option of contacting P to obtain an *identity-based* secret key sk_P *efficiently* computed by T using msk (and, presumably, infeasible to compute using pk only). In particular, no party ever needs to call T twice, and the secret keys sk_P are completely independent from anything going on in the protocol (which means that it is indeed harmless to provide those keys in the ideal model too). Moreover, in our model and implementation *no honest party P will ever need to (and therefore will not) obtain this secret key sk_P*. This is a big advantage over the PKI model, where every party needs to contact the certificate authority C and learn the corresponding secret keys of all the other parties. In our model, the honest parties only need to know *one* global key pk and the non-cryptographic identities of all the other parties (which is always assumed in the UC framework anyway).

Intuitively, the point of these id-based secret keys is that the dishonest parties P can (and probably will) obtain those values sk_P already in the real model, which makes it both meaningful and "legal" to supply the corresponding values sk_P to the simulator, whenever the party P gets corrupted. With the proper design of T and the corresponding multiparty protocol, we will demonstrate that the simulator can achieve full black-box simulation: whatever a polynomial-time adversary can obtain in the real protocol with the help of id-based keys sk_P for all corrupted P, the corresponding polynomial-time simulator can achieve via straight-line simulation in the ideal model, and once again with the help of exactly the same id-keys sk_P. Thus, *using identical resources to what is actually available to him in the real model*, the real-life adversary A might have as well "transported" himself to the ideal model, and simply run the simulator S there, while still producing the same effect as in the real model! In particular, our simulator does not program the public key pk, run in super-polynomial time or call some imaginary angels which do not actually exist in the real model. Moreover, our protocol withstands adaptive corruptions (unlike, for example, the protocols of [2, 12]). Additionally, the introduction of T could naturally support other desirable properties impossible in the standard model (such as optimistic fairness with faulty majority).

To summarize, we minimally and realistically changed the standard UC model by introducing an efficient, fully off-line party T, which allowed us to achieve a meaningful notion of simulation in the real model.

1.2 Overview of Our Techniques

We briefly report on our techniques. First, we notice that it is sufficient to implement the commitment functionality in a fully simulatable manner, as the rest of the protocol of [7] does not utilize the programmability of the CRS. Then

we show how to *efficiently* build (interactive, 4-round) fully simulatable commitments from any family of *identity-based chameleon hash functions* (ID-CHash) [1].[3] This protocol is inspired by the corresponding protocol of Damgard and Nielsen [8] for regular UC commitments, although the details are somewhat different.

Identity Based Chameleon Hash Functions. ID-based chameleon hash functions are identity-based analogs of trapdoor (or equivocable) commitment schemes. In such regular schemes, there exists a trapdoor information td associated with the public key pk, which allows one to open a given commitment value c to an arbitrary message m (which, of course, is impossible without td). In the ID-based setting, one also commits with respect to a given identity P, and there correspondingly exists a different trapdoor td_P which allows one to equivocate all commitments w.r.t. P. On the other hand, knowing many values td_P does not allow one to equivocate commitment w.r.t. a new identity Q. So far, ID-CHash functions were used primarily for the purposes of chameleon id-based signature schemes [1, 13], and all the constructions were in the random oracle model.

We give an elegant construction of ID-CHash functions based on any signature scheme possessing a Σ-protocol proving that "the prover knows the signature of a given message m". We then construct a variety of such protocols for various signature schemes: both generic (i.e., starting from any signature scheme), as well as efficient number-theoretic schemes (e.g., based on the strong RSA assumption and the signature of [3]). This gives several constructions of ID-CHash functions *in the standard model*. Ironically, the random oracle based constructions of [1, 13] are also special cases of our construction (applied to specific random oracle based signature schemes).

References

[1] G. Ateniese and B. de Medeiros. Identity-based Chameleon Hash and Applications. *Proc. of Financial Cryptography*, 2004. Available at `http://eprint.iacr.org/2003/167/`.

[2] B. Barak, R. Canetti, J. Nielsen and R. Pass. Universally composable protocols with relaxed set-up assumptions. In *Proc. of FOCS*, 2004, to appear.

[3] J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *Conference on Security in Communication Networks (SCN)*, 2002.

[4] R. Canetti. Universally Composable Security: A New paradigm for Cryptographic Protocols. In *Proc. of FOCS*, pages 136–145, 2001.

[5] R. Canetti and M. Fischlin. Universally Composable Commitments. In *Proc. of Crypto*, pages 19–40, 2001.

[6] R. Canetti, E. Kushilevitz and Y. Lindell. On the Limitations of Universally Composable Two-Party Computation Without Set-Up Assumptions. In *Proc. of Eurocrypt*, Springer-Verlag (LNCS 2656), pp. 68–86, 2003.

[7] R. Canetti, Y. Lindell, R. Ostrovsky and A. Sahai. Universally Composable Two-Party and Multi-Party Secure Computation. In *Proc. of STOC*, pp. 494–503, 2002.

[8] I. Damgard and J. Nielsen. Perfect Hiding and Perfect Binding Universally Composable Commitment Schemes with Constant Expansion Factor. In *Proc. of Crypto*, Springer-Verlag, pp. 581–596, 2002.

[9] O. Goldreich, S. Micali and A. Wigderson. How to Solve any Protocol Problem. In *Proc.of STOC*, 1987.

[10] R. Pass. On Deniabililty in the Common Reference String and Random Oracle Model. In *Proc. of Crypto*, LNCS 2729, pp. 216–337, 2003.

[11] R. Pass. Simulation in Quasi-Polynomial Time and Its Application to Protocol Composition. In *Proc. of EuroCrypt*, LNCS 2656, pages 160–176, 2003.

[12] M. Prabhakaran and A. Sahai. New Notions of Security: Achieving Universal Composability without Trusted Setup. In *Proc. of STOC*, 2004.

[13] F. Zhang, R. Safavi-Naini and W. Susilo. ID-Based Chameleon Hashes from Bilinear Pairings. Available at `http://eprint.iacr.org/2003/208/`.

[3] Similarly to [7], we also need augmenting non-committing encryption protocols; see [7].