Adaptive Hardness and Composable Security in the Plain Model from Standard Assumptions

Ran Canetti
Tel Aviv University
Canetti@tau.ac.il

Huijia Lin
Cornell University
huijia@cs.cornell.edu

Rafael Pass
Cornell University
rafael@cs.cornell.edu

***Abstract*—We construct the first general secure computation protocols that require no trusted infrastructure other than authenticated communication, and that satisfy a meaningful notion of security that is preserved under universal composition—*assuming only the existence of enhanced trapdoor permutations.* The notion of security fits within a generalization of the "angel-based" framework of Prabhakaran and Sahai (STOC'04) and implies super-polynomial time simulation security. Security notions of this kind are currently known to be realizable only under strong and specific hardness assumptions.**

A key element in our construction is a commitment scheme that satisfies a new and strong notion of security. The notion, security against chosen-commitment-attacks (CCA security), means that security holds even if the attacker has access to a *extraction oracle* that gives the adversary decommitment information to commitments of the adversary's choice. This notion is stronger than concurrent non-malleability and is of independent interest. We construct CCA-secure commitments based on standard one-way functions, and with no trusted set-up. To the best of our knowledge, this provides the first construction of a natural cryptographic primitive requiring *adaptive hardness* from standard hardness assumptions, using no trusted set-up or public keys.

***Keywords*-cryptography; adaptive hardness; secure multi-party computation; composable security**

I. Introduction

The notion of *secure multi-party computation* allows m mutually distrustful parties to securely compute (or, *realize*) a functionality $f(\bar{x})$ of their corresponding private inputs $\bar{x} = x_1, ..., x_m$, such that party P_i receives the ith component of $f(\bar{x})$. Loosely speaking, the security requirements are that the output of each party is distributed according to the prescribed functionality—this is called *correctness*—and that even malicious parties learn nothing more from the protocol than their prescribed output—this is called *privacy*. These properties should hold even in case that an arbitrary subset of the parties maliciously deviates from the protocol.

Soon after the concept was proposed [42], general constructions were developed that appeared to satisfy the intuitive correctness and secrecy for practically any multi-party functionality [42], [20]. These constructions require only authenticated communication and can use any enhanced trapdoor permutation. However, definitions that capture the security properties of secure multi-party computation protocols (and, in fact, of secure cryptographic protocols in general) took more time to develop. Here, the *simulation paradigm* emerged as a natural approach: originally developed for capturing the security of encryption and then extended to Zero-Knowledge [19], [21], this paradigm offers a general and expressive approach that allows capturing a wide variety of requirements and situations in a natural and precise way. The idea is to say that a protocol π securely realizes f if running π emulates an idealized protocol I_f where all parties secretly provide inputs to an imaginary trusted party that computes f and returns the outputs to the parties; more precisely, any "harm" done by a polynomial-time adversary in the real execution of π, could have been done even by a polynomial-time adversary (called a *simulator*) that interacts with parties running I_f . An advantage of the simulation paradigm is its expressiveness: It allows capturing a large variety of security properties in a natural and precise way, simply by formulating the appropriate f. This idea was informally articulated in [20]. Many formulations of this paradigm were proposed, e.g. [16], [1], [31], [5], [18], [37], [6]. A proof that the [20] construction satisfies the [5], [18] definition eventually appeared in [18], demonstrating the realizability of this definition. We call this definition *basic security.*

Basic security indeed seems to be adequate in situations when the protocol is run in isolation. However, it does not provide sufficiently strong *composability* guarantees. Let us explain.

Composable security. A useful notion of security should provide guarantees even in settings where multiple protocols co-exist in the same system and potentially interact with each other, or in other words are *composed* to form a larger system. We distinguish three quite different (and incomparable) properties to consider in such settings:

Concurrent Multi-Instance Security: The security properties relating to the local data and outputs of the *analyzed protocol itself* should remain valid even multiple concurrent instances of the protocol co-exist and are susceptible to coordinated attacks against multiple instances.

Modular analysis: The notion of security should support designing composite protocols in a modular way, while preserving security. That is, there should be a way to deduce

security properties of the overall protocol from security properties of its components. This is essential for asserting security of complex protocols.

Environmental Friendliness: The security properties of *other, potentially unknown protocols* that co-exist in the same system should not be adversely affected by adding the analyzed protocol.

The simulation paradigm suggests a natural approach to formulating composable notions of security: Consider a protocol π that securely realizes a function f (i.e., π emulates the ideal protocol I_f), and let ρ be a protocol that uses subroutine calls to protocol π. Now, since the execution of π should look to an observer just the same as an execution of I_f, the behavior of ρ should, intuitively, remain unchanged when each call to π is replaced by a call to I_f. Therefore, rather than analyzing the protocol ρ that uses (potentially multiple instances of) π, we might as well analyze the simpler system where each instance of π is replaced by an instance of I_f.

Making good of this intuitive approach turns out to be non trivial. Specifically, the definitions of [16], [1] were not shown to have any composability properties, whereas those of [31], [5], [18] only guarantee *non-concurrent composability.* That is, the above three properties related to composable security are guaranteed only when the protocol ρ makes sure that the instances of π run in sequence, with nothing else happening in the rest of the system from the onset of the execution of each instance until all participants of this instance complete their respective local processing of π. This is a significant restriction.

UC security [6] gives a more stringent formulation of the simulation paradigm than basic security, providing a very strong composability property that implies all three composability requirements discussed above. But these strong properties come at a price: Many natural functionalities cannot be realized with UC security in the *plain model,* where the only set-up provided is authenticated communication channels; some additional trusted set-up is necessary [8], [9]. Furthermore, the need for additional trusted set up extends to any protocol that only guarantees a concurrent multi-instance extension of basic security [29].

Security with super-polynomial simulators (SPS) [32] is a relaxation of UC security that allows the adversary in the ideal execution to run in super-polynomial time. Informally, this corresponds to guaranteeing that "any poly-time attack that can be mounted against the protocol can also be mounted in the ideal execution—albeit with super-polynomial resources." Protocols that realize practically any functionality with SPS security in the plain model were shown based on sub-exponential hardness assumptions [32], [2], [27].

Although SPS security is sometimes weaker than basic security, it often provides an adequate level of security. Furthermore, SPS security guarantees concurrent multi-instance security (with super-polynomial simulation). However, SPS security is environmentally friendly only in a very partial way: For other protocols in the systems, it only preserves those security properties that withstand super-polynomial time adversaries. Furthermore, SPS security is not closed under composition (protocol $\rho^{\pi/\phi}$ where each instance of π is replaced by an instance of ϕ is not guaranteed to emulate ρ with SPS security, even if π realizes ϕ with SPS security), thus it is not a convenient basis for modular analysis of protocols.

Angel-based UC security [38] is a framework for notions of security that allow mitigating these shortcomings of SPS security. Specifically, angel-based security considers a model where both the adversary and the simulator have access to an oracle (an "angel") that allows some judicious use of super-polynomial resources. (In spirit, these resources allow the simulator to "reverse engineer" the adversary.) It is not hard to see that, for any angel, angel-based security implies SPS security, and at the same time is closed under composition, akin to UC security. That is, if π emulates ϕ with respect to some angel then, for any ρ, $\rho^{\pi/\phi}$ emulates ρ with respect to the same angel. This means that angel-based UC security, with any angel, can be used as a basis for modular analysis of protocols. It remains to consider the "environmental friendliness" of this class of notions. Here too angel based security *may* provide an improvement over SPS security: For the other protocols in the system, any security property that holds with respect to adversaries that have access to the specific angel in use will be preserved when adding the analyzed protocol to the system. This means that different angels provide different levels of "environmental friendliness".

Furthermore, angel-based security is potentially realizable: A protocol realizing practically any ideal functionality with respect to some angel, in the plain model, are constructed in [38]. A different construction, based on a different angel, is given in [30]. In [2] it is remarked that their protocol can be shown to be angel-based secure but the specific angel is not given. These protocols are very attractive indeed: They use no trusted initial set up other than authenticated communication, and obtain a meaningful and composable security notion. However, they are all based on strong and non-standard hardness assumptions. This leaves open the following questions:

> Is it possible to realize general functionalities with SPS security in the plain model, under standard hardness assumptions? Furthermore, can we show angel-based security with respect to some angel? Further, can we show that this angel is environmentally friendly, at least for some class of protocols?

A. Our Results

We show an angel with which we can securely realize the ideal commitment functionality, $\mathcal{F}_{\mathsf{com}}$, in the plain model, assuming only one way functions. We then rely on known results [12], [4], [22] to construct a protocol for general functionalities. (Here we need to assume also existence of enhanced trapdoor permutations.)

In contrast with the angels considered so far in the literature, which maintain only global state on the system and are otherwise stateless, our angel is highly interactive and stateful. (In fact, dealing with such angels requires some modification of the original model of [38].) Still, security with respect to our angel implies SPS security.

Furthermore, we demonstrate that our angel provides a partial notion of environmental friendliness, which we refer to as *robustness*:[1] Any attack mounted by an adversary on a *constant-round* protocol that uses our angel can be carried out by a polytime adversary with no angels at all.[2] In fact, we rely on this robustness property to argue that the [12], [4] protocol (using $\mathcal{F}_{com}$) remains secure even with respect to our angel (so that we can rely on this protocol to realize general functionalities).

To formally present and prove our results, we also re-cast the model of [38] within the extended UC (EUC) framework of [11]. Roughly speaking, this notion is identical to standard UC security as in [5], except that all parties have access to an additional global entity. In [11], this entity is used to model global set-up such as a reference string or strong public-key infrastructure. Here, in contrast, we consider a global entity that, as in [38], interacts only with the corrupted parties. This means that the actual protocol uses no trusted infrastructure, and the global entity becomes a means for relaxing the security requirement. We call the global entity, $\mathcal{H}$ a *helper functionality* or an *angel*, and denote the corresponding notion of security $\mathcal{H}$-EUC security.

Main Theorem (Informally Stated): Assume the existence of enhanced trapdoor permutations. Then there exists a subexponential-time computable interactive machine $\mathcal{H}$ such that for any "well-formed" polynomial-time functionality $\mathcal{F}$, there there exists a protocol that realizes $\mathcal{F}$ with $\mathcal{H}$-EUC security, in the plain model.

We emphasize that this is the first protocol to acheive any non-trivial notion of fully concurrent multi-instance security in the plain model and under standard assumptions, let alone closure under composition or environmental friendliness. In particular, it yields the first protocol achieving SPS-security in the plain model based only on standard assumptions.

[1] This notion of robustness is a strengthening of the notion of robustness considered in [27].

[2] In fact, at the cost of increasing the round-complexity of our protocol, the angel can be modified to retain the security of any protocol with an *a-priori* bounded number of rounds.

Intuitively, our helper functionality (i.e., angel) will allow a party P to obtain the decommitment information for commitments *where P is the committer.* This will allow the simulator (i.e., the adversary we need to construct for interacting with the ideal protocol) to extract the decommitment information from commitments made by corrupted parties. Given such extraction help, simulation is relatively easy.

The main challenge is to make sure that the adversary will not be able to use this angel in order to break the security of other commitments, that were made by uncorrupted parties. For this we need commitment schemes that allow a committer to commit securely even when the receiving party has access to such an extraction oracle. That is, we need commitments that bind the identity of the committer to the committed value in a way that does not allow an adversarial receiver to transform a given commitment to one that has a related decommitment value and a different committer identity. This should be the case even when the receiving party has unlimited adaptive access to the extraction oracle. We call this strong property *security against adaptive chosen commitment attack (CCA security).*

B. The Main Tool: CCA-Secure Commitments

The protocols of [38], [30] for angel-based UC security rely on certain, quite specific *adaptive hardness* assumptions, namely assumptions that postulate security even in face of adversaries that have adaptive access to some help information. Indeed, such form of adaptive hardness seems to be inherent in the angel-based approach to security. However, such assumptions appear to be qualitatively stronger than non adaptive ones. A natural question is then whether such an adaptive hardness property can be based on a more standard assumption.

We answer this question positively: We formulate an angel that provides a useful adaptive hardness property, and show that this adaptive hardness property can be based on a standard assumption, specifically existence of one way functions.

The adaptive hardness property is the CCA security property of commitment schemes mentioned above. Roughly speaking, a tag-based commitment scheme (i.e., commitment scheme that take an identifier—called the tag—as an additional input) is said to be *CCA-secure* if the value committed to using the tag id remains hidden even if the receiver has access to a (super-polynomial time) oracle that provides decommitments to commitments using any tag $\mathsf{id}' \neq \mathsf{id}$. CCA-security can be viewed as a natural strengthening of *concurrent non-malleability* [14], [34], [26]—roughly speaking, a commitment scheme is concurrently non-malleable if it is CCA-secure with respect to restricted classes of adversaries that only ask a single parallel—i.e., non-adaptive—decommitment query after completing its interaction with the committer.

It is not hard to construct CCA-secure commitments using trusted set-up (by e.g., relying on known constructions of CCA-secure encryption schemes). It is also quite simple to construct a CCA-secure commitment scheme under an adaptive hardness assumption (such as the existence of adaptively-secure one-way permutations—namely one-way permutations that remain uninvertible even if the adversary has access to a inversion oracle) [33].[3]

Our main technical contribution consists of showing how to construct a CCA-secure commitment based only on one-way functions, and without any trusted set-up.

Theorem (informally stated) Assume the existence of one-way functions. Then, for every $\epsilon > 0$, there exists an $O(n^\epsilon)$-round CCA-secure commitment scheme (where n is the security parameter).

As far as we know this yeilds the first non-trivial primitive whose "adaptive hardness" can be proven based on standard assumptions without set-up. Note that many standard cryptographic primitives (such as signatures [21], pseudo-random function [10] and CCA-secure encryption [41]) consider adaptive attacks where an adversary has access to an oracle breaking the primitive. However, all these cryptographic rely on some trusted set-up (in the case of signatures and CCA-secure encryption, the public-key used needs to be well-formed, whereas in the case of pseudo-random function, the "seed" needs to be uniform and perfectly hidden from the adversary).

Our construction of CCA-secure commitments: Consider the scenario of *concurrent non-malleable commitments* [14], [34], [26]. We consider a man-in-the-middle attacker (MIM) that participates in one interaction "on the left" and many interactions "on the right". In the left interaction it receives a commitment, whereas in the right interactions it provides (potentially) many commitments, acting as a committed. Intuitively, we want to ensure that the values committed to on the right are "independent" of the value committed to on the left. Let us revisit the approach of [26] (which builds on [14]) for constructing such commitments. The basic idea is to have a protocol where the scheduling of the messages depend on the tag of the commitment. The scheduling ensure that for every right interaction with a tag that is different from the left interaction, there exists a point—called a safe point—from which we can "rewind" the right interaction (and extract out the value committed to), without violating the hiding property of the left interaction. It now follow from the hiding property of the left interaction that the values committed to on the right do not depend on the value committed to on the left.

At first sight, it would seem that this type of construction and analysis could be directly used to guarantee also CCA-security: we simply view the extrenal interaction the adversary participates in as a receiver, as the left interaction, and view all the oracle calls as right interactions. Each time a right interaction completes, we "rewind" the appropriate safe-point to extract the decommitment information (without violating the hiding of the left interaction), and feed the decommitment information to the adversary.

But there is a problem with this approach. Recall that in the setting of CCA-security, we need to provide the adversary with the decommitment information at the very moment it completes a commitment to its oracle. If the adversary "nests" its oracle calls, these rewindings become recursive and the running-time of the extraction quickly becomes exponential. (Note that this does not happen in the context of concurrent non-malleability since in that setting it is suffices to extract the committed values at the end of the interaction; it is, thus, enough to rewind the right interactions one at a time, in sequence.) The problem is analogous to the simulation problem in the context of *concurrent zero-knowledge* [15], where a nesting verifier might cause the naive simulation to take exponential time. On a high-level, we resolve this problem using an idea that is similar to that used in the context of concurrent zero-knowledge in the the work of Richardson and Kilian [40]. We present a scheduling of messages (again based on the tag) which ensures that every right interaction has n^ϵ (where n is the security parameter and $\epsilon > 0$) safe-points, instead of just one. Intuitively, having many safe-point ensures that there is always at least one safe-point where the number of nested executions is "small," which in turn ensures that the overall running-time of the extraction is polynomial.

Formalizing this argument turns out to be tricky. The main obstacle is that the techniques used to acheive non-malleability are incompatible with those used in the context of concurrent zero-knowledge:

- In order to use the proof technique of [26], [14], the protocol (roughly speaking) needs to consists of a sequence of *three-message* "slots" with the property that if the second message is rewound the committed value can be efficiently recovered, but if all three messages are rewound, then the committed value remains completely hidden. This is used to argue that there exist points where we can extract the committed value from the rigth interaction, without violating the hiding of the left interaction.
- Known concurrent zero-knowledge protocols [40], [25], [39], [36] (and their analyses) rely on the fact that the intial message in the protocol determines some value; this value can then recovered by rewinding any "chunk" of more than two messages that does not

[3] [33] considered a variant of the notion of CCA-security for non-interactive and perfectly binding commitment scheme (called adaptively-secure commitments). We have extended this definition to general commitment schemes.

contain the initial message. [4] This property does not hold for protocols such as [14], [26] since they rely on the principle that three-message chunks reveal nothing when rewound.

To get around this problem we develop a new concurrent extraction technique (based on the simulator strategies in [40], [36]) which can be applied also to protocols such as [14], [26]. Roughly speaking, instead of determining what slot to rewind based on the number of new executions that started "within" the slot (as in [40], [36]), we treat slots from different executions uniformly and instead decide whether to rewind a slot, based on the number of slots that are contained within it. (This idea is similar to one used in [13] in a different context, but where a similar problem arises). This allows us to extract the decommitment information to all commitments provided by the adversary to its oracle, without violating the hiding of the left interaction, and while ensuring that the (expected) running-time of the extraction procedure is polynomial.

Robust CCA-security: "Plain" CCA-security only guarantees that the security of the particular commitment scheme in question remains intact when the adversary has access to an extraction oracle. As mentioned above, in our final construction we additionally require that an attacker having access to the extraction oracle should not be able to violate the security of *any* constant-round protocol; we call CCA-secure commitments satisfying this property *robust CCA-secure*. Robust CCA-security of our construction follows in essentially the same way as plain CCA-security. Roughly speaking, when extracting the commitments on the "right", we simply have to make sure not to rewind any messages from the constant-round protocol on the "left"; this is possible since the commitment scheme has a super constant number of slots.

II. Definition of CCA-Secure Commitments

A. Notations and Preliminaries

Let N denote the set of all positive integers. For any integer $n \in N$, let $[n]$ denote the set $\{1, 2, \ldots, n\}$, and let $\{0,1\}^n$ denote the set of n-bit long string; furthermore, let ε denote the empty string. We assume familiarity with the basic notions of *Interactive Turing Machines* [21] (ITM for brevity) and interactive *protocols*. Given a pair of ITMs, A and B, we denote by $\langle A(x), B(y)\rangle(z)$ the random variable representing the (joint) output of A and B, on common input z and private input x and y respectively, and when the random input to each machine is uniformly and independently chosen.

B. Commitment Schemes

A commitment scheme $\langle C, R\rangle$ consists of a pair of $\mathcal{PPT}$ ITMs C and R that interacts in a commit stage and a reveal stage. In this work, we consider commitment schemes $\langle C, R\rangle$ that are statistically binding and computationally hiding. Furthermore, We restrict our attention to commitment schemes where the reveal phase is non-interactive—the committer decommits to value v by simply sending a decommitment pair (v, d). We let $\mathsf{open}_{\langle C,R\rangle}$ denote the function that verifies the validity of (v, d); the receiver accepts (v, d) if $\mathsf{open}(c, v, d) = 1$, and rejects otherwise, where c is the commitment, i.e., the transcript of messages exchanged in the commit phase. Additionally, we consider the following two properties of commitment schemes:

- We say that a commitment c is *accepting* if the receiver accepts the transcript c at the end of the commit stage, and *valid* if there exists a decommitment pair (v, d) such that $\mathsf{open}_{\langle C,R\rangle}(c, v, d) = 1$. Then we say that a commitment scheme is *efficiently checkable*, if every commitment that is accepting w.r.t the honest receiver is valid.
- A *tag-based commitment schemes* [35], [14] is a scheme where, in addition to the security parameter, the committer and the receiver also receive a "tag"—a.k.a. the identity—id as common input.

C. CCA-Secure Commitments

Security under chosen-ciphertext-attacks (CCA security) [41] has been studied extensively in the context of encryption schemes, where the confidentiality of encrypted messages is guaranteed even in the presence of a decryption oracle. We here define an analogous notion for commitment schemes. Roughly speaking, a commitment scheme is CCA-secure (chosen-commitment-attack) secure if the commitment scheme retains its hiding property even if the receiver has access to a a "decommiment oracle". Let $\langle C, R\rangle$ be a tag-based commitment scheme. A decommitment oracle $\mathcal{O}$ of $\langle C, R\rangle$ acts as follows in interaction with an adversary A: it participates with A in many sessions of the commit phase of $\langle C, R\rangle$ as an honest receiver, using identities of length n, chosen adaptively by A. At the end of each session, if the session is *accepting and valid*, it reveals a decommitment of that session to A; otherwise, it sends $\perp$. (Note that when a session has multiple decommitments[5], the decommitment oracle only to returns one of them. Hence, there might exist many valid decommitment oracles.)

Loosely speaking, a tag-based commitment scheme $\langle C, R\rangle$ is said to be CCA-secure, if there *exists* a decommitment oracle $\mathcal{O}$ for $\langle C, R\rangle$, such that, the hiding property of the commitment holds even with respect to adversaries with access to $\mathcal{O}$. More precisely, denote by $A^{\mathcal{O}}$ the adversary A with access to the decommitment oracle $\mathcal{O}$. Let $\mathsf{IND}_b(\langle C, R\rangle, \mathcal{O}, A, n, z)$, where $b \in \{0,1\}$, denote the output of the following probablistic experiment: on common

[4] This is used in the analysis to ensure that the extraction for a particular execution does not get stuck because of some execution that started earlier.

[5] Note that the statistically binding property only guarantees that, with overwhelming probability, the committed value is unique. However, there may still exist many different decommitments.

input 1^n and auxiliary input z, $A^{\mathcal{O}}$ (adaptively) chooses a pair of challenge values $(v_0, v_1) \in \{0,1\}^n$—the values to be committed to—and an identity $\mathsf{id} \in \{0,1\}^n$, and receives a commitment to v_b using identity id. Finally, the experiment outputs the output y of $A^{\mathcal{O}}$; the output y is replace with $\perp$ if during the execution A sends $\mathcal{O}$ any commitment using identity id (that is, any execution where the adversary queries the decommitment oracle on a commitment using the same identity as the commitment it receives, is considered invalid).

Definition 1 (CCA-secure Commitments.)**.** *Let $\langle C, R\rangle$ be a tag-based commitment scheme, and $\mathcal{O}$ a decommitment oracle for it. We say that $\langle C, R\rangle$ is* CCA-secure w.r.t. *$\mathcal{O}$, if for every $\mathcal{PPT}$ ITM A, the following ensembles are computationally indistinguishable:*

- $\{\mathsf{IND}_0(\langle C, R\rangle, \mathcal{O}, A, n, z)\}_{n\in N, z\in\{0,1\}^*}$
- $\{\mathsf{IND}_1(\langle C, R\rangle, \mathcal{O}, A, n, z)\}_{n\in N, z\in\{0,1\}^*}$

We say that $\langle C, R\rangle$ is CCA-secure *if there exists a decommitment oracle $\mathcal{O}'$, such that, $\langle C, R\rangle$ is CCA-secure w.r.t. $\mathcal{O}'$.*

As mentioned in the introduction, CCA-security easily implies concurrent non-malleability [14], [34], [26]. Note that in the definition of CCA-security, we could also have considered an adversary that can select a pair of *sequences* $\vec{v}_0, \vec{v}_1$ of challenge messages (instead of simply a pair (v_0, v_1)). It follows using a standard hybrid argument that CCA-security implies security also in this setting.

D. k-Robust CCA-Secure Commitments

We introduce a strengthening of the notion of CCA-security analougosly to the notion of *robust non-malleable commitments* of [28]. We here consider a man-in-the-middle adversary that participates in an *arbitrary* left interaction with a *limited number of rounds*, while having access to a decommitment oracle.

Definition 2. *Let $\langle C, R\rangle$ be a tag-based commitment scheme, and $\mathcal{O}$ a decommitment oracle for it. We say that $\langle C, R\rangle$ is* k-robust CCA-secure w.r.t. *$\mathcal{O}$, if $\langle C, R\rangle$ is CCA-secure w.r.t. $\mathcal{O}$, and for every $\mathcal{PPT}$ adversary A, there exists a $\mathcal{PPT}$ simulator S, such that, for every $\mathcal{PPT}$ k-round ITMs B, the following two ensembles are computationally indistinguishable.*

- $\{\langle B, A^{\mathcal{O}}(z)\rangle(1^n)\}_{n\in N, z\in\{0,1\}^*}$
- $\{\langle B, S(z)\rangle(1^n)\}_{n\in N, z\in\{0,1\}^*}$

Thus, roughly speaking, $\langle C, R\rangle$ is k-robust if the (joint) output of every k-round interaction, with an adversary having access to the oracle $\mathcal{O}$, can be simulated without the oracle. In other words, having access to the oracle does not help the adversary in participating in any k-round protocols.

We say that a tag-based commitment $\langle C, R\rangle$ is robust CCA-secure if there exists a a decommitment oracle $\mathcal{O}$, such that, $\langle C, R\rangle$ is k-robust CCA-secure w.r.t. $\mathcal{O}$, for every constant k.

Remark 1 (On the identity length)**.** *Recall that in the definition of CCA-security, the adversary can pick arbitrary identities* id *for both the left and the right interaction. We may also consider a restricted notion of (robust) CCA-security where the adversary is restricted to use identities of some bounded length. As we show in the full version of the paper, standard techniques [14] can be used to show that any* robust *CCA-secure commitment that is secure for identities of length $\ell(n) = n^{\varepsilon}$ can be turned into a robust CCA-secure commitment (that secure for identities of length* $\mathrm{poly}(n)$*).*

III. Construction of a CCA-Secure Commitment

In this section, we show the following theorem.

Theorem 1. *Assume the existence of one-way functions. Then, for every $\epsilon > 0$, there exists an $O(n^{\epsilon})$-round robust CCA-secure commitment scheme (where n is the security parameter).*

Our CCA-secure commitment $\langle C, R\rangle$ is based on a variant of the concurrent non-malleable commitment protocol of [26], which in turn is based on message scheduling technique of [14]. However, here we use a slightly different message schedule in order to provide more "safe" rewinding slots. We proceed to formally specifying the protocol. For simplicity of exposition, the description below relies on the existence of one-way functions with efficiently recognizable range, but the protocol can be easily modified to work with any arbitrary one-way function (see the full version of the paper for more details). Furthermore, we also rely on 3-round special-sound proofs in our protocol, but the analysis also works also with 4-round proofs. (See the full version for more detials.)

Let ℓ and η be polynomials in the security parameter n. To commit to a value v, the Committer C and the Receiver R, on common input 1^n and the identity $\mathsf{id} \in \{0,1\}^{\ell(n)}$, proceeds in the following three stages in the commit phase.

- *Stage 1:* the Receiver picks a random string $r \in \{0,1\}^n$, and sends its image $s = f(r)$, through a one-way function f with an efficiently recognizable range, to the Committer. The Committer checks that s is in the range of f and aborts otherwise. Additionally, the receiver also sends the first messages r_1, r_2 for two commitments, using a a two-round statistically binding string commitment com.
- *Stage 2:* The Committer provides a commitment c_1 to v using the commitment scheme com and r_1 as the first message; let (v, d) denote the decommitment information. Next, it provides a commtiment c_2 to (v, d) using com and r_2 as first message. We refer to (r_1, c_1) as the *first commitment* and (r_2, c_2) as the *second commitment*.
- *Stage 3:* The Committer proves that

– (r_1, c_1) is a valid commitment to v and (r_2, c_2) is a valid commitment to a decommitment pair for (r_1, c_1),
– *or* s is in the image set of f.

This is proved using $4\ell(n)\eta(n)$ invokations of a special-sound $\mathcal{WI}$ proof where the verifier query has length $3n$.[6] The messages in the proofs are scheduled based on the identity id and relies on scheduling pairs of proofs according to schedules design_0 and design_1 depicted in Figure 1. More precisely, the proof stage consist of $\ell(n)$ phases. In phase i, the committer provides $\eta(n)$ sequential $\mathsf{design}_{\mathsf{id}_i}$ pairs of proofs, followed by $\eta(n)$ sequential $\mathsf{design}_{1-\mathsf{id}_i}$ pairs of proofs.

In the reveal phase, the Committer simply decommits to the first commitment (r_1, c_1). The Receiver accepts if the decommitment is valid and rejects otherwise.

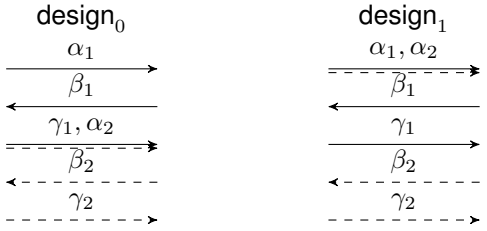

Figure 1. Description of the schedules used in Stage 3 of the protocol. $(\alpha_1, \beta_1, \gamma_1)$ and $(\alpha_2, \beta_2, \gamma_2)$ are respectively the transcripts of a pair of 3-round special-sound proofs.

On the round complexity of $\langle C, R\rangle$: The round complexity of $\langle C, R\rangle$ is $O(\ell(n)\eta(n))$. We will show that $\langle C, R\rangle$ is robust CCA-secure when $\eta(n) = n^\epsilon$ for any constant $\epsilon > 0$. When $\ell(n) = n^{\varepsilon'}$ (which is without loss of generality by Remark 1) we thus obtain a $O(n^{\varepsilon+\varepsilon'})$-round protocol.

It follows using standard techniques that $\langle C, R\rangle$ is a commitment scheme with efficient verifiability.

Proposition 1. *$\langle C, R\rangle$ is a statistically binding commitment scheme with efficient verifiability.*

Proof: It follows using essentially the same proof as in [26] that the protocol is statistically binding and computationally hiding; we refer the reader to [26] for more details.

It remains to show that the validity of an $\langle C, R\rangle$ commitment is efficiently checkable. By construction, an $\langle C, R\rangle$ commitment is valid if and only if the "first commitment" is valid. It then follows from the soundness of Stage 3 of the protocol that, whenever the receiver is accepting (at the end of the commit phase), the first commitment is valid except with negligible probability. Thus, the validity of $\langle C, R\rangle$ is also efficiently checkable. ■

[6] As we shall see later on, the length restriction will facilitate the security proof.

We turn to show that $\langle C, R\rangle$ is a robust CCA-secure commitment when $\eta(n) = n^\epsilon$ for any constant $\epsilon > 0$.

Theorem 2. *Let $\epsilon > 0$ be a constant, and let $\eta(n) = n^\epsilon$. Then $\langle C, R\rangle$ is a robust CCA-secure commitment.*

To show that $\langle C, R\rangle$ is a robust CCA-secure commitment, we need to exhibit a decommitment oracle $\mathcal{O}$ for $\langle C, R\rangle$ such that $\langle C, R\rangle$ is robust CCA-secure w.r.t. $\mathcal{O}$. Consider the following decommitment oracle $\mathcal{O}$: $\mathcal{O}$ acts just as an honest receiver during the commit phase. At the end of every accepting interaction, if the "second commitment" (r_2, c_2) defines a unique value (v, d) such that (v, d) is a valid decommitment for the "first commitment" (r_1, c_2), $\mathcal{O}$ returns (v, d). Otherwise, $\mathcal{O}$ returns the lexicographically first decommitment, or $\perp$ if there does not exists a valid decommitment. The main technical challenge consists of proving the following lemma.

Proposition 2. *$\langle C, R\rangle$ is robust CCA-secure w.r.t $\mathcal{O}$.*

We provide a high-level overview of the proof below. The formal proof can be found in the full version of the paper.

Proof overview of Proposition 2: We first argue that $\langle C, R\rangle$ is CCA-secure w.r.t. $\mathcal{O}$. Consider the CCA-experiment IND_b, where the adversary A interacts with an honest committer C, and is given access to $\mathcal{O}$. We refer to its interaction with C as the *left interaction*, and its intearctions with $\mathcal{O}$ as the *right interactions*. Recall that proving CCA-security w.r.t. $\mathcal{O}$ amounts to showing that the views of A in experiments IND_0 and IND_1 are indistinguishable (when A has oracle access to $\mathcal{O}$). The main hurdle in showing this is that the oracle $\mathcal{O}$ is not efficiently computable; if it was, indistinguishability would directly follow from the hiding property of the left interaction. However, since $\langle C, R\rangle$ consists of a sequence of special-sound proofs of the committed vaule, the oracle $\mathcal{O}$ can be efficiently implemented by "rewinding" the special-sound proofs in the right interaction. But, the problem is that once we start rewinding the right interactions, A might send new messages also in the left interaction. So, if done naively, this would require us to also rewind the left interaction, which could violate its hiding property.

The crux of the proof is showing how to appropriately rewind the right interactions (so as to extract out the commited values), without violating the hiding property of the left interaction. This implies that the view of A in IND_b can be efficiently emulated, without the oracle $\mathcal{O}$, and hence by the hiding property of the left interaction, A's view is indistinguishable. Towards doing this, we rewind the right interaction only at special points in the interaction; we call such points safe points.

Note that the hiding property of the left interaction remains intact if during the rewinding, one of the following two cases occurs.

- *Case 1:* A does not request any new messages in the

left interaction.

- *Case 2:* The only new messages A requests in the left interaction are *complete* $\mathcal{WISSP}$ proofs.

The fact that the left interaction is still hiding even if case 2 happens follows using exactly the same proof as proof of hiding of $\langle C, R\rangle$. And, obviously, the left interaction remains hiding if case 1 happens.

Roughly speaking, we now say that a prefix ρ of a transcript Δ (which consists of one left interaction and many right interaction) is a safe-point for the k^{th} right interaction if ρ "lies in between" the first two messages α_r and β_r of a $\mathcal{WISSP}$ proof $(\alpha_r, \beta_r, \gamma_r)$ for the kth interaction (i.e., it contains α_r but not β_r), and satisifes that from ρ to the point when γ_r is sent, one of the two cases above occurs in the left interaction. We call $(\alpha_r, \beta_r, \gamma_r)$ the $\mathcal{WISSP}$ *associated* with the safe point ρ in Δ. It follows using a combinatorical argument (similar in spirit, but more complicated than, [14], [26]) that the message scheduling in Stage 3 of $\langle C, R\rangle$ guarantees the following key property:

> Let Δ be any transcript of one left interaction, and many right interactions, of $\langle C, R\rangle$. Then, any right interaction k that 1) has completed, and 2) uses a different identity than the left interaction, has $\Omega(n^\epsilon)$ *non-overlapping* $\mathcal{WISSP}$ that are associated with a safe point in Δ.

We will now use these safe-points to construct a simulator that can efficiently emulates the oracle $\mathcal{O}$. (As mentioned in the Introduction, the reason we need "many" safe-points is to ensure that we can extract out the committed values in all the right interactions while ensuring an expected polynomial running-time.) On a high-level, the simulation emulates $\mathcal{O}$ by following the honest receiver strategy of $\langle C, R\rangle$, until it enounters a "good" safe-point ρ. It then keeps rewinding the execution back to ρ until it obtains another accepting transcript of the right proof associated with ρ. Once two accepting proof transcripts are obtained, the special-soundness property allows the simulator to extract the decommitment information.

More precisely, the simulation is defined recursively in the following manner: On recursion level d, we say that a safe-point ρ of a transcript Δ is "good" (we call this a $d+1$-good safe-point) if the number of right-execution $\mathcal{WISSP}$ proofs—possibly from different interactions—starting in between ρ and the point where γ_ρ is sent in Δ, is smaller than $k_d = M/\eta'^{d+1}$, where M is a (polynomial) upperbound on the total number of messages in Δ, and $\eta' = n^{\varepsilon'}$ for some constant ε' such that $0 < \varepsilon' < \varepsilon$.

Then, on recursion level d, the simulator emulates every right interaction honestly, but as soon as it encounters a $d+1$-good safe-point in the current transcript, it begins "rewinding" the execution back to the safe-point, and invokes itself recursively at level $d+1$. In each rewinding, if it notices that ρ might no longer be a $d+1$-good safe-point, it cancels the rewinding and starts a new rewinding. It continues the process until it gets another transcript where ρ is a $d+1$-good safe-point again; from this second transcript we can extract out (and store) the decommitment information for the right interaction associated with the safe-point. Finally, whenever in the emulation a right interaction completes, the simulator provides A with the decommitment information extracted out (or outputs `fail` if the decommitment information has not been recovered). By cancelling "bad" rewindings (i.e., rewindings that are not $d+1$-good safe-points) we are guaranteeing two properties: 1) we *never* violate the hiding property of the left interaction, and 2) the running-time of the simulation does not blow up.

Let us now briefly argue that the simulator indeed emulates $\mathcal{O}$ both correctly and efficiently, without violating the hiding property of the left interaction.

Hiding property of the left interaction: Since the simulator only rewinds the right interactions from safe-points, and cancels every rewinding in which the point is no longer a safe-point, it follows that the left interaction remains hiding.

Correctness: We argue that for each right interaction that uses an identity that is different from the left interaction, we extract out the decommitment information before the interaction completes sucessfully. First, note that the recursion level is bounded by $c = \log_{\eta'} M$, which is a constant (since is M is polynomial in n and $\eta' = n^{\epsilon'}$). Since each successful right interaction, that uses a different identity than the left interaction, has n^ϵ safe-points (by the key property above), it follows that for each such interaction, there exists some recursive level d, such that the right interaction has at least $n^\epsilon/c > \eta'$ safe-points on level d. But, as the total number of right-proofs that start on level d is bounded by $k_d = M/\eta'^d$ (otherwise, the simulation at this recursive level is cancelled), there must exist one right-proof with an associated safe-point ρ, such that less than M/η'^{d+1} right-proofs start in between ρ and the last message of the proof. Therefore ρ is a $d+1$-good safe-point and will be rewound. Finally, since we continue rewinding until the decommitment information is found, it follows that for each successful right interaction that uses a different identity than the left interaction, the decommitment information is recovered by the simulator.

Efficiency: To prove that the simulation is efficient, consider a simplified scenario where A *never* sends a com commitment that can be decommitted to two different values—i.e., A never manages to violate the statistical binding property of com. We argue that the simulation is efficient in this case. (It suffices to consider this case, since the probability that A violates the statistical binding property of com is negligible, and thus we can always "cut-off" the simulation without loosing "too much".) To prove that the expected running-time of the simulation (in the simplified scenario) is

polynomially bounded, first recall that the recursive depth is a constant c. Secondly, at each recursive level d, there are at most M possible points from which we can rewind and from each of these points, the expected number of rewindings is 1. The latter follows since the simulator only starts rewinds from a point ρ if it is a ℓ+1-good safe-point, and it continues rewinding until ρ becomes a $\ell+1$-good safe-point again; furthermore, in each of the rewindings the simulated view of the adversary is identically distributed to its view in the first execution (this fact relies on us considering the case when all com commitments are well-defined). Thus, the probability that a point is a $\ell+1$-good safe-point (conditioned on it occuring as a prefix in the execution) is the same in the first execution and in the rewindings. Therefore, the expected number of recursive calls starting from any point is 1. We now conclude that the expected total number of rewindings is bounded by $O(M)^c$.

The robustness w.r.t. $\mathcal{O}$ property of $\langle C, R\rangle$ follows using essentially the same proof as above: the key step here is, again, to show that the decommitment oracle $\mathcal{O}$ can be emulated efficiently, without "affecting" the security of the left interaction. Now, however, a rewinding is "safe" only if the adversary does not request *any* new message in the left (constant-round) interaction—that is, the the left interaction is never rewound during the extractions on the right. Roughly speaking, this is achieved by rewinding only those $\mathcal{WISSP}$ proofs that do not interleave with any message in the left interaction (and cancelling every rewinding in which the $\mathcal{WISSP}$ proof interleaves with a left-message).

IV. Realizing Any Functionality

We here sketch how to realize any functionality by relying on our construction of CCA-secure commitments. Let $\langle C, R\rangle$ be a commitment scheme that is robust CCA-secure w.r.t. a decommitment oracle $\mathcal{O}$. Furthermore, assume that $\mathcal{O}$ can be computed in subexponential time; note that our construction of CCA secure commitments can easily be instantiated with bit-commitments using a "scaled-down" the security parameter in order to ensure this property. Consider a helper functionality $\mathcal{H}$ that "breaks" commitments of $\langle C, R\rangle$ in the same way as $\mathcal{O}$ does, subject to the condition that player P_i in a protocol instance sid can only query the functionality on commitments that uses identity (P_i, sid); clearly this functionality can also be implemented in sub-exponential time.

We have now the following theorem:

Theorem 3. *Let ε be any positive constant. Assume the existence of enhanced trapdoor permutations. Then for every well-formed functionality*[7] *$\mathcal{F}$, there exists a $O(n^\varepsilon)$-round protocol Π that $\mathcal{H}$-EUC-emulates $\mathcal{F}$.*

Towards proving the theorem, we first show how to implement the *ideal commitment functionality* $\mathcal{F}_{\mathsf{com}}$ (see [8] for a formal description) in the $\mathcal{H}$-EUC-model, and then show how to realize any functionality using $\mathcal{F}_{\mathsf{com}}$.

Realizing $\mathcal{F}_{com}$: The committer P_i and the receiver P_j, on input $(\texttt{Commit}, (P_i, P_j, sid), v)$ to P_i and $(\texttt{Commit}, (P_i, P_j, sid))$ to P_j, proceed as follows:

- *Stage 1:* the receiver P_j picks a random secret $r \in \{0,1\}^n$, and commits to r using the CCA-secure commitment scheme $\langle C, R\rangle$, and using (P_j, sid) as the identity of the interaction.
- *Stage 2:* the committer P_i commits to the value v using $\langle C, R\rangle$, and using (P_i, sid) as the identity of the interaction.
- *Stage 3:* the committer P_i commits to 0^n using $\langle C, R\rangle$, and using (P_i, sid) as the identity of the interaction.

Finally, on input $(\texttt{Open}, (P_i, P_j, sid))$ to both P_i and P_j, the committer P_i decommits to the value v in a reveal phase, by sending v to P_j and then provides a strongly $\mathcal{WI}$ proof of the statement that either it has committed to v in Stage 2, or that it has committed to a valid decommitment pair (r, d) of the Stage 1 commitment, in Stage 3. The receiver P_j accepts if the proof is convincing, and rejects otherwise.

We defer the formal proof to the full version. Roughly speaking, the proof amounts to showing that commitments are "extractable" and "equivocable"; both can be easily done by using $\mathcal{O}$.

Realizing Any Functionality: First note that to realize any well-formed functionality, it suffices to realize the oblivious transfer functionality $\mathcal{F}_{\mathsf{OT}}$; by previous works [23], [3], [20], [22], this suffices for *unconditionally* implementing any functionality. Next, we would like to rely on the the CLOS-BMR protocol [12], [4]—a *constant-round* protocol that UC-realize $\mathcal{F}_{\mathsf{OT}}$ in the $\mathcal{F}_{\mathsf{com}}$-hybrid model—and simply realize $\mathcal{F}_{\mathsf{com}}$ using our $\mathcal{H}$-EUC secure implementation (described above). If the CLOS-BMR protocol had been a $\mathcal{H}$-EUC realization of $\mathcal{F}_{\mathsf{OT}}$, then the resulting composed protocol would also be $\mathcal{H}$-EUC secure (and we would be done). But recall that UC-security does not necessarily imply $\mathcal{H}$-EUC security (since now the environment is endowed by the super-polynomial oracle $\mathcal{H}$). One way around this problem would be to rely on subexponentially-hard trapdoor permutations in the CLOS-BMR protocol. We take a different route (which dispenses of the extra assumptions): Since the CLOS-BMR protocol is constant-round, we can rely on the *robust* CCA-security property of $\mathcal{O}$ to prove that CLOS-BMR (relying on standard, polynomially-hard, trapdoor permutations) in fact is secure also w.r.t $\mathcal{H}$.

[7] See [12] for a definition of well-formed functionalities.

References

[1] D. Beaver. Foundations of Secure Interactive Computing. In *CRYPTO '91*, pages 377-391, 1991.

[2] B. Barak and A. Sahai. How To Play Almost Any Mental Game Over The Net - Concurrent Composition via Super-Polynomial Simulation. FOCS 2005: 543-552

[3] D. Ben-Or, S. Goldwasser and A. Wigderson. Completeness Theorems for Non-Cryptographic Fault-Tolerant Distributed Computation. In *20'th STOC*, pages 1–10, 1988.

[4] D. Beaver, S. Micali and P. Rogaway The Round Complexity of Secure Protocols. In *22'th STOC*, pages 503–513, 1990.

[5] R. Canetti. Security and Composition of Multiparty Cryptographic Protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[6] R. Canetti. Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *42nd FOCS*, pages 136–145, 2001.

[7] R. Canetti. Security and composition of cryptographic protocols: a tutorial (part I). In *SIGACT News 37(3)*, pages 67-92, 2006.

[8] R. Canetti and M. Fischlin. Universally Composable Commitments. In *Crypto2001*, Springer LNCS 2139, pages 19–40, 2001.

[9] R. Canetti, E. Kushilevitz and Y. Lindell. On the Limitations of Universally Composable Two-Party Computation Without Set-Up Assumptions. In *Eurocrypt2003*, Springer LNCS 2656, pages 68–86, 2003.

[10] Oded Goldreich and Shafi Goldwasserw and Silvio Micali. How to construct random functions. In *J. ACM, Vol. 33, No. 4*. Pages 792-807. 1986

[11] R. Canetti and Yevgeniy Dodis and Rafael Pass and Shabsi Walfish. Universally Composable Security with Global Setup. In *TCC 2007*.

[12] R. Canetti, Y. Lindell, R. Ostrovsky and A. Sahai. Universally Composable Two-Party and Multy-Party Computation. In *34th STOC*, pages 494–503,2002.

[13] Yi Deng, Vipul Goyal, Amit Sahai: Resolving the Simultaneous Resettability Conjecture and a New Non-Black-Box Simulation Strategy. FOCS 2009.

[14] D. Dolev, C. Dwork and M. Naor. Non-Malleable Cryptography. *SIAM Jour. on Computing*, Vol. 30(2), pages 391–437, 2000.

[15] C. Dwork, M. Naor and A. Sahai. Concurrent Zero-Knowledge. In *30th STOC*, pages 409–418, 1998.

[16] S. Goldwasser and L. Levin. Fair Computation of General Functions in Presence of Immoral Majority. In *CRYPTO '90*, pages 77–93, 1990.

[17] O. Goldreich. *Foundation of Cryptography – Basic Tools*. Cambridge University Press, 2001.

[18] O. Goldreich. *Foundation of Cryptography – Basic Applications*. Cambridge University Press, 2004.

[19] S. Goldwasser, S. Micali. Probabilistic Encryption. JCSS 28(2), pages 270–299, 1984.

[20] O. Goldreich, S. Micali and A. Wigderson. Proofs that Yield Nothing But Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *JACM*, Vol. 38(1), pp. 691–729, 1991.

[21] S. Goldwasser, S. Micali and C. Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM Jour. on Computing*, Vol. 18(1), pp. 186–208, 1989.

[22] Y. Ishai, M. Prabhakaran and A. Sahai. Founding Cryptography on Oblivious Transfer - Efficiently. *CRYPTO'08*, pages 572–591, 2008.

[23] J. Kilian. A Note on Efficient Zero-Knowledge Proofs and Arguments. In *24th STOC*, pages 723–732, 1992.

[24] E. Kushilevitz, Y. Lindell and T. Rabin. Information-theoretically secure protocols and security under composition. In *STOC 2006*, pages 109–118, 2006.

[25] J. Kilian and E. Petrank. Concurrent and Resettable Zero-Knowledge in Poly-logarithmic Rounds. In *33rd STOC*, pages 560–569, 2001.

[26] H. Lin, R. Pass, and M. Venkitasubramaniam. Concurrent Non-Malleable Commitments from One-way Functions. In *TCC 2008*.

[27] H. Lin, R. Pass, and M. Venkitasubramaniam. A Unified Framework for Concurrent Security: Universal Composability from Stand-alone Non-malleability. In *STOC 2009*.

[28] H. Lin, and R. Pass. Non-Malleability Amplification. In *STOC 2009*.

[29] Y. Lindell. Lower Bounds for Concurrent Self Composition. in *1st TCC*, 2004.

[30] Tal Malkin, Ryan Moriarty, Nikolai Yakovenko. Generalized Environmental Security from Number Theoretic Assumptions. In *TCC 2006* pages 343-359.

[31] S. Micali and P. Rogaway. Secure computation. Unpublished manuscript, 1992. Preliminary version in *CRYPTO'91*, Springer-Verlag (LNCS 576), pages 392–404, 1991.

[32] R. Pass. Simulation in Quasi-Polynomial Time and Its Application to Protocol Composition. In *EuroCrypt2003*, Springer LNCS 2656, pages 160–176, 2003.

[33] Pandey, Omkant and Pass, Rafael and Vaikuntanathan, Vinod. Adaptive One-Way Functions and Applications. In *Crypto 2008*, pages 57–74, 2008.

[34] R. Pass, A. Rosen. Bounded-Concurrent Secure Two-Party Computation in a Constant Number of Rounds. In *44th FOCS*, pages 404–413, 2003.

[35] R. Pass and A. Rosen. Concurrent Non-Malleable Commitments. In *46th FOCS*, pages 563–572, 2005.

[36] R. Pass and M. Venkitasubramaniam. On Constant-Round Concurrent Zero-Knowledge. In *TCC '08*, pages 553–570, 2008.

[37] B. Pfitzmann and M. Waidner: A Model for Asynchronous Reactive Systems and its Application to Secure Message Transmission. In *IEEE Symposium on Security and Privacy 2001*, pages 184-193, 2001

[38] M. Prabhakaran and A.Sahai. New notions of security: achieving universal composability without trusted setup. In *STOC 2004*, pages 242-251, 2004.

[39] M. Prabhakaran, A. Rosen and A. Sahai. Concurrent Zero-Knowledge with Logarithmic Round Complexity. *Proceedings of the 43rd annual IEEE symposium on Foundations of Computer Science (FOCS 2002)*, pages 366-375, 2002.

[40] R. Richardson and J. Kilian. On the Concurrent Composition of Zero-Knowledge Proofs. In *EuroCrypt99*, Springer LNCS 1592, pages 415–431, 1999.

[41] Charles Rackoff, Daniel R. Simon: Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack. CRYPTO 1991: 433-444.

[42] A. Yao. How to Generate and Exchange Secrets. In *27th FOCS*, pages 162–167, 1986.