A Unified Framework for Concurrent Security: Universal Composability from Stand-alone Non-malleability

Huijia Lin*
Cornell University
huijia@cs.cornell.edu

Rafael Pass†
Cornell University
rafael@cs.cornell.edu

Muthuramakrishnan Venkitasubramaniam‡
Cornell University
vmuthu@cs.cornell.edu

ABSTRACT

We present a unified framework for obtaining Universally Composable (UC) protocols by relying on *stand-alone secure non-malleable commitments*. Essentially all results on concurrent secure computation—both in relaxed models (e.g., quasi-polynomial time simulation), or with trusted set-up assumptions (e.g., the CRS model, the imperfect CRS model, or the timing model)—are obtained as special cases of our framework. This not only leads to conceptually simpler solutions, but also to improved set-up assumptions, round-complexity, and computational assumptions.

Additionally, this framework allows us to consider new relaxed models of security: we show that UC security where the adversary is a *uniform* $\mathcal{PPT}$ but the simulator is allowed to be a *non-uniform* $\mathcal{PPT}$ (i.e., essentially, traditional UC security, but with a non-uniform reduction) is possible without any trusted set-up. This gives the first results on concurrent secure computation without set-up, which can be used for securely computing "computationally-sensitive" functionalities (e.g., data-base queries, "proof of work"-protocols, or playing bridge on the Internet).

Categories and Subject Descriptors

F.1.2 [**Theory of Computation**]: Interactive and reactive computation

General Terms

Theory

*Lin is supported by I3P grant 2006CS-001-0000001-02.

†Pass is supported in part by NSF CAREER Award CCF-0746990, AFOSR Award FA9550-08-1-0197, BSF Grant 2006317 and I3P grant 2006CS-001-0000001-02.

‡Venkitasubramaniam is supported by I3P grant 2006CS-001-0000001-02.

STOC'09, May 31–June 2, 2009, Bethesda, Maryland, USA.

Keywords

secure multi-party computation, non-malleability, universal composability

1. INTRODUCTION

The notion of *secure multi-party computation* allows m mutually distrustful parties to securely compute a functionality $f(\bar{x}) = (f_1(\bar{x}), ..., f_m(\bar{x}))$ of their corresponding private inputs $\bar{x} = x_1, ..., x_m$, such that party P_i receives the value $f_i(\bar{x})$. Loosely speaking, the security requirements are that the parties learn nothing more from the protocol than their prescribed output, and that the output of each party is distributed according to the prescribed functionality. This should hold even in the case that an arbitrary subset of the parties maliciously deviates from the protocol.

The above security guarantees are traditionally formalized using the *simulation paradigm* [26, 27]. The basic idea, which originates in [24], is to say that a protocol π securely realizes f if running π emulates an ideal process 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 by a polynomial-time *simulator* in the ideal process.

Shortly after its conceptualization, strong results were established for secure multi-party computation. Specifically, it was shown that any probabilistic polynomial-time computable multi-party functionality can be securely computed, assuming the existence of enhanced trapdoor permutations [50, 24]. The original setting in which secure multi-party protocols were investigated, however, only allowed the execution of a single instance of the protocol at a time; this is the so called *stand-alone setting*. A more realistic setting, is one which allows the concurrent execution of protocols. In the *concurrent setting*, many protocols are executed at the same time. This setting presents the new risk of a coordinated attack in which an adversary interleaves many different executions of a protocol and chooses its messages in each instance based on other partial executions of the protocol. The strongest (but also most realistic) setting for concurrent security—called *Universally Composable* (UC) security [10, 46, 16], or environmental-security—considers the execution of an unbounded number of concurrent protocols, in an arbitrary, and adversarially controlled, network environment. Unfortunately, security in the stand-alone setting does not

imply security in the concurrent setting. In fact, without assuming some trusted set-up, the traditional simulation-based notion of concurrent security, and in particular UC security, cannot be achieved in general [11, 12, 34].

To circumvent the broad impossibility results, two distinct veins of research can be identified in the literature.

Trusted set-up models: A first vein of work initiated by Canetti and Fischlin [11] and Canetti, Lindell, Ostrovsky and Sahai [14] (see also e.g., [6, 13, 31, 15]) considers constructions of UC-secure protocol using various trusted set-up assumptions, where the parties have limited access to a trusted entity. (See [9] for a recent survey of various different set-up assumptions.) In many situations, however, trusted set-up is hard to come by (or at least expensive). An important question is to identify the weakest possible set-up that allows us to obtain general feasibility results for UC security.

Relaxed models of security: In some situations, trusted set-up is not only expensive, but might not even exist. It is thus imperative to have a notion of concurrent security that can be realized without trusted set-up. Another vein of work considers relaxed models of security such as *quasi-polynomial simulation* [41, 47, 4] or *input-indistinguishability* [38]. These works, however, only provide weak guarantees about the computational advantages gained by an adversary in a concurrent execution of the protocol. As such, currently, there are no known protocols—without trusted set-up—that can be used to securely compute "computationally-sensitive" functionality (such as e.g., private data-base queries, proof-of-work protocols [18], or player bridge or poker on the Internet [24]) in a fully concurrent setting.[1]

In this work we address both of the above research goals by presenting a *unified framework* for the construction of UC secure protocols—both with, and without, trusted set-up. This framework not only provides a conceptually simpler solutions for essentially all general UC-feasibility results (e.g., [14, 13, 6, 31, 15, 29, 28]), but also allows us to (often significantly) improve the round-complexity and the complexity theoretic assumptions. (Interestingly, our new results even improve the round complexity of *stand-alone* secure computation. As far as we know this is the first improvement to the original work of Goldreich, Micali and Wigderson [24], assuming only trapdoor permutations.) More importantly, this framework allows us to consider weaker trusted set-up models (e.g., the existence of a *single* imperfect reference string, or an "unrestricted" timing model) and new relaxed models of security. In particular, we present a new model of concurrent security, called *Non-Uniform UC*, which allows us to achieve—without any trusted set-up—the first "fully-concurrent" secure computation protocol that provides strong guarantees about the computational advantages gained by an adversary. We also complement our positive results with new lower bounds, showing that our results (both with and without trusted set-up) are essentially tight—both in terms of round complexity and in terms of complexity-theoretic assumptions. As such, our framework helps in characterizing models in which UC security is realizable, and also at what cost.

We start by outlining our framework, and then proceed to introduce our new notion of security.

[1]Yet another vein of work considers relaxed notions of concurrency, such as *bounded concurrency* [1, 43, 35, 42]. In this work, we, however, focus only on *full concurrency*, where no restrictions on the number of concurrent executions are made.

1.1 Our Unified Framework

Earlier results on UC secure computation all rely on quite different techniques. Roughly speaking, to prove that a protocol is concurrently secure, one needs to show two different properties: 1) *concurrent simulation*, and 2) *concurrent non-malleability*. Intuitively, concurrent simulation amounts to providing the simulator with a "trapdoor" that allows it to emulate players without knowing their inputs. Concurrent non-malleability, on the other hand, requires showing that an adversary cannot make use of messages received in one execution to "cheat" in another execution; this is often achieved by providing a technique which enables the simulator to have a *different* trapdoors for each player (in a sense an "identity-based" trapdoor using the terminology of [13]), and showing that the trapdoor for one player does not reveal a trapdoor for another.

The simulation part is usually "easy" to achieve. Consider, for instance, the Common Reference String model—where the players have access to a public reference string that is ideally sampled from some distribution. In this model it is easy to provide the simulator with a single trapdoor; it could, for instance, be the inverse of the CRS through a one-way function. However achieving concurrent non-malleability is significantly harder. In this particular case, [14] solve the problem by embedding the public-key of a CCA-secure encryption scheme in the CRS, but in general, quite different techniques are employed in each model. Yet the same phenomena persists: concurrent simulation is easy to achieve, but concurrent non-malleability requires significantly more work, and often stronger set-up and/or stronger computational assumptions and/or larger round-complexity.

We provide a technique showing that concurrent simulation is sufficient—i.e., it is sufficient to provide the simulator with a *single* trapdoor. In a nutshell, once such a trapdoor is established, concurrent non-malleability (and thus full UC security) is achieved by further relying on a *stand-alone secure non-malleable commitment* [17]. (In a sense, this can be viewed as an analog of the transformation from concurrent zero-knowledge to concurrent non-malleable zero-knowledge of Barak, Prabhakaran and Sahai[3]; the main tool used in their transformation is also a stand-alone secure non-malleable commitment).

To formalize "concurrent simulation" we define the notion of a *UC-puzzle*—which, intuitively, is a protocol with the property that no adversary can successfully complete the puzzle and also obtain a trapdoor, but there exists a simulator who can generate (correctly distributed) puzzles together with trapdoors.

THEOREM 1 (INFORMALLY STATED). Assume the existence of a $t_1(\cdot)$-round UC-secure puzzle Σ using some set-up $\mathcal{T}$, the existence of $t_2(\cdot)$-round "natural" non-malleable commitments and the existence of enhanced trapdoor permutations. Then, for every m-ary functionality f, there exists a $O(t_1(\cdot) + t_2(\cdot))$-round protocol Π—using the same set-up $\mathcal{T}$—that UC-realizes f.

We emphasize that, in previously studied models, $O(1)$-round UC-puzzles are "easy" to construct. As such, Theorem 1 provides a conceptually simple and unified proof of known results, while at the same time reducing the trusted set-up, the computational assumptions and/or the round-complexity. We briefly highlight some results obtained by instantiating our framework with known constructions of non-malleable commitments [17, 44, 32, 33].[2] In all the results below we focus only on *static* adversaries.

UC in the "imperfect" string model. Canetti, Pass and Shelat [15] consider UC security where parties have access to an "imperfect" reference string—called a "sunspot"—that is generated by any arbitrary efficient min-entropy source (obtained e.g., by measurement of some physical phenomenon). The CPS-protocol, however, requires m communicating parties to share m reference strings, each of them generated using fresh entropy.

Our results show that, somewhat surprisingly, a *single* imperfect reference string is sufficient for UC security. This stands in sharp contrast to the general study of randomness extraction, where single-source extraction from arbitrary *efficient* sources is impossible, but extraction from multiple sources is feasible![3]

UC in the timing model. Dwork, Naor and Sahai [19] introduced the *timing model*, where all players are assumed to have access to clocks with a certain drift. In this model, they rely on *delays* and *time-outs* to obtain a $O(1)$-round concurrent zero-knowledge protocol. Kalai, Lindell and Prabhakaran [31] subsequently presented a concurrent secure computation protocol in the timing model; whereas the timing model of [19] does not impose a maximal upper-bound on the clock drift, the protocol of [31] requires the clock-drift to be "small"; furthermore, it requires $\omega(n)$ rounds and an extensive use of delays (roughly $n\Delta$, where Δ is the latency of the network).

Our results establish that UC security is possible also in the "unrestricted" timing model (where the clock drift can be "large"); additionally, we reduce the use of delays to only $O(\Delta)$, and only require an $O(1)$-round protocol; in fact, we also establish lower bounds showing that the *run time* (and thus the use of delays) of our protocol is optimal up to a constant factor.

UC with quasi-polynomial simulation. Pass [41] proposed a relaxation of the standard simulation-based definition of security, allowing for a super polynomial-time, or Quasi-polynomial simulation (QPS). Prabhakaran and Sahai [47] and Barak and Sahai [4] recently obtained general multi-party protocols that are concurrently QPS-secure without any trusted set-up, but rely on strong complexity assumptions.

Our results show how to weaken the complexity assumptions, while at the same time achieving a stronger (and more meaningful) notion of security, which (in analogy with [41]) requires that indistinguishability of simulated and real executions holds also for all of *quasi-polynomial time*; in contrast, [4] only achieves indistinguishability w.r.t distinguishers with running-time smaller than that of the simulator.[4] We complement this results by a lower bound showing that our complexity assumptions, in essence, are necessary to achieve QPS security.

Stand-alone secure multi-party computation. The original construction of *stand-alone* secure m-party computation by Goldreich, Micali and Wigderson relies only on the existence of enhanced trapdoor permutations, but requires $O(m)$-communication rounds. We obtain the first (asymptotic) improvement to the round complexity of this results without strengthening the underlying assumption. By relying on the original DDN construction of non-malleable commitments [17] (see also [32]) we already obtain a $\log \log m$-round secure computation protocol. If, instead relying on the recent construction from [33], we obtain a $O(1)^{\log^* m}$ round protocol.[5]

Our results also establishes that, on top of enhanced trapdoor permutations, no further assumptions are necessary to establish UC secure computation in e.g., the *uniform random string (URS) model* [14] or the *"multi-CRS" model* [28]; earlier results required additional assumptions (e.g., dense crypto systems).

1.2 New Notions of Security

QPS or input-indistinguishability are two notions of concurrent security that can be achieved without trusted set-up. However, neither of them provide strong guarantees about the computational advantages gained by an adversary in an interaction. Consider, for instance, using a secure computation protocol for playing bridge on the Internet. A protocol with QPS security might give an adversary significant computational advantages (potentially all of quasi-polynomial time) in the game of bridge, making it easier for him to win; input-indistinguishability provides even less guarantees. For concrete examples of situations where such advantages could be useful, consider e.g., the "proof-of-work" protocols of Dwork and Naor [18].

Our goal is to establish a notion of concurrent security that provides strong guarantees about computational advantages gained by an adversary, while at the same time being realizable without trusted set-up. In a nut-shell, our

[2] Interesting, for many of our results, we get quite substantial improvements "already" by relying on the original DDN-construction [17].

[3] Note that the results of Trevisan and Vadhan [49] only show that extraction from sources with size bounded by some *fixed* polynomial is possible. In contrast, traditional techniques show that extraction from sources with *arbitrary* polynomial running-time is impossible.

[4] In essence, this means that anything an attacker can learn "on-line" (in poly-time) can be simulated "off-line" (in qpoly-time) in a way that is indistinguishable also "off-line". In this language, [4] only achieves on-line indistinguishability.

[5] Earlier results improve the round-complexity by making stronger assumptions: (1) assuming *dense crypto systems*, Katz, Ostrovsky and Smith [30] achieved $O(\log m)$ rounds; (2) assuming collision-resistant hash-function, and additionally relying on *non-black box simulation* [1], Pass [42] achieved $O(1)$ rounds. Our results only use black-box simulation; as such they are a significant improvement also over any known protocol using black-box simulation (and in particular [30]).

approach can be described as follows: whereas traditional UC security (following the seminal works of [27, 23]) guarantees that an adversary does not learn *anything* more than it could have learnt if the parties directly communicated with a trusted party, a notion of "adequate" security would simply require that the adversary does not learn anything more *about the inputs/outputs* of the parties. As such, an "adequate" notion of security would allow the adversary to learn some bizarre string (e.g., the factorization of a segment of the decimal expansion of π) as long as it is independent of the inputs of the players.

To formalize such a notion, we consider a new variant of UC security—called *non-uniform UC*, where both the network environment and the adversary are modeled as *uniform* $\mathcal{PPT}$, but the simulator is allowed to be a *non-uniform* $\mathcal{PPT}$— i.e., in essence, UC security, but with a non-uniform reduction. As with traditional UC, our definition guarantees that the running-time of the simulator is within a (fixed) *additive* polynomial term of the running-time of the adversary in an execution-by-execution manner.[6] As such, non-uniform UC guarantees that an adversary "essentially" cannot get even computational advantages by deviating, except for possibly a short non-uniform advice string which is independent of the protocol execution, and in particular of the inputs of the players. As a corollary of Theorem 1, we directly get the following feasibility results.

THEOREM 2 (INFORMALLY STATED). Assume the existence of enhanced trapdoor permutations and evasive promise problems in **BPP**. Then, for any m-party functionality f, there exists a protocol Π that non-uniform UC-realizes f.

Our construction relies on a new complexity theoretic assumption: the existence of an *evasive promise problem in* **BPP**.[7] We remark that, this assumption is implied by a number of standard (and quite weak) assumptions such as the existence of uniform collision-resistant hash-functions, or a scaled-up version of the *worst-case* hardness of $\mathbf{NP} \cap co\mathbf{NP}$ (we prove this formally in the full-version). Perhaps surprisingly, we show that the existence of evasive promise problems in **BPP** is also *necessary* for achieving non-uniform UC security.

We mention that (just as QPS and input-indistinguishability), the notion of non-uniform UC itself is not closed under composition (in contrast to the traditional notion of UC); rather we directly consider security under concurrent executions. We also mention that since the simulator is allowed to get a non-uniform advice, non-uniform UC does not guarantee "plausible deniability" or security under, so called, "chosen-protocol attacks". However, even the traditional notion of UC does not always provide those guarantees (see [41, 13] for more details).[8]

[6]In all our constructions, this polynomial is simply the time needed to *honestly* execute the protocol. As such, our protocols are also *precise* [37] in the running-time (but not size).

[7]This assumption is a generalization (and thus relaxation) of the assumption that there exists an *evasive set in* **P**. Evasive sets (introduced by Goldreich and Krawczyk [22]) are non-empty sets, S, such that no $\mathcal{PPT}$ adversary can find an element in S, except with negligible probability. Interestingly, Goldreich and Krawczyk introduced the notion of evasive sets also in the context of protocols composition. However, they relied on this notion to provide examples of protocols whose security *fails* under composition.

[8]In fact, non-uniform UC can be seen as a natural way of defining concurrent security, *without deniability*: if a protocol is not "deniable", then the view of the adversary represents some "knowledge" that cannot be simulated; what we guarantee is that this new "knowledge" is independent of the actual protocol execution, and the inputs used (and furthermore cannot be used to violate the security of other instances of the protocol).

1.3 Techniques

By relying on previous results [42, 43, 34, 14, 23] the construction of a UC protocol for realizing any multi-party functionality reduces to the task of constructing a zero-knowledge protocol that is *concurrently simulatable* and *concurrently simulation-sound* [48]—namely, even if an adversary receives multiple concurrent *simulated* proofs, it will not be able to prove any false statements. Concurrent simulation is easy to achieve in any model where we have a UC-puzzle. The tricky part is to obtain a zero-knowledge proof that also is simulation-sound.

To achieve this we introduce a new notion of non-malleability for interactive proofs, called *strong non-malleable witness indistinguishable* ($\mathcal{SNMWI}$). Informally, $\mathcal{SNMWI}$ extends the notion of *strong witness indistinguishability* [21] to a man-in-the-middle setting: consider a man-in-the-middle attacker (MIM) that is participating in two interactions, a left interaction where it is acting as a verifier, and a right interaction where it is acting as a prover. $\mathcal{SNMWI}$ requires that whenever the common inputs in the left interaction are indistinguishable, so are the views of *and witnesses* used by the MIM. $\mathcal{SNMWI}$ is related to (and inspired by) the notion of *non-malleable witness indistinguishability*, recently introduced by Ostrovsky, Persiano and Visconti [40], but the actual security requirements are quite different.

As we show, $\mathcal{SNMWI}$ is a relaxation of the notion of *simulation-extractability* [17, 44], and, as such, potentially easier to achieve. In particular, one of our main technical contributions is a construction of $\mathcal{SNMWI}$ arguments of knowledge from any "natural" non-malleable commitment (with only constant overhead in round-complexity).

Using a puzzle and a $\mathcal{SNMWI}$ argument of knowledge, we can now provide a conceptually simple construction of a simulation-sound zero-knowledge argument of any language $L \in \mathbf{NP}$: to prove a statement $x \in L$, the verifier first provides the prover with a puzzle Σ, the prover next commits to a witness w and then provides a $\mathcal{SNMWI}$ argument of knowledge of the fact that it committed to a valid witness for x, or that it has committed to a trapdoor to the puzzle Σ.

1.4 Overview

In Section 2, we define the notion of $\mathcal{SNMWI}$ and outline how it can be constructed based on any "natural" non-malleable commitment. In Section 3, we introduce the notion of a UC puzzle and show how to UC-realize any functionality in any "generalized UC-model" where there exists a UC-puzzle (our general UC model incorporates extensions such as QPS and non-uniform UC, as well as trusted-set up.) The complete proofs will appear in the full version. We defer our lower bounds (showing optimality of our constructions) to the full-version too; roughly speaking, the lower bounds rely on a "reverse" variant of the Canetti-Fischlin [11] impossibility results (sometimes in combination with a use of *universal arguments* [36, 2].)

2. STRONG NON-MALLEABLE $\mathcal{WI}$

We start by defining the notion of strong non-malleable $\mathcal{WI}$ ($\mathcal{SNMWI}$) only for languages with *unique* witnesses; we next extend it to general **NP**-languages. Let R_L be the canonical witness relation for some language L with unique witnesses. Consider a, so-called, tag-based argument system for L—i.e., the prover and the verifier receive a "tag" as an additional common input, besides the statement x. $\mathcal{SNMWI}$ considers a man-in-the-middle execution of the protocol $\langle P_s, V_s\rangle$, in which the adversary A simultaneously participates in two interactions of $\langle P_s, V_s\rangle$, one left and one right interaction. In the left interaction, the adversary A, on auxiliary input z, receives a proof of statement x from P_s on private input y such that $y \in R_L(x)$, using a fixed tag id. In the right interaction, A adaptively chooses a statement $\tilde{x}$ and tag $\tilde{\mathsf{id}}$ and attempts to provide a proof to V_s. Let $\tilde{y}$ denote the witness associated with $\tilde{x}$, unless either of the following happens (a) A fails in the right interaction or (b) $\mathsf{id} = \tilde{\mathsf{id}}$; in this case $\tilde{y}$ is set to $\perp$. Let $\mathsf{mim}^A_{\langle P_s, V_s\rangle}(x, y, z, \mathsf{id})$ denote the random variable that describes the witness $\tilde{y}$ combined with the view of A in the above man-in-the-middle experiment.

Definition 3 (Strongly Non-Malleable $\mathcal{WI}$). *We say that $\langle P_s, V_s\rangle$ is* strongly non-malleable witness-indistinguishable *for R_L if for every non-uniform $\mathcal{PPT}$ man-in-the-middle adversary A, every $\mathsf{id} \in \{0,1\}^*$ and every two sequences of input distributions $\{D^1_n\}_{n\in N}$ and $\{D^2_n\}_{n\in N}$, the following holds: if $\{(x,y,z) \leftarrow D^1_n \ : \ (x,z)\}_{n\in N}$ and $\{(x,y,z) \leftarrow D^2_n \ : \ (x,z)\}_{n\in N}$ are computationally indistinguishable, so are the following ensembles:*

$$\left\{(x,y,z) \leftarrow D^1_n \ : \ \mathsf{mim}^A_{\langle P_s, V_s\rangle}(x,y,z,\mathsf{id})\right\}_{n\in N, \mathsf{id}\in\{0,1\}^*}$$

$$\left\{(x,y,z) \leftarrow D^2_n \ : \ \mathsf{mim}^A_{\langle P_s, V_s\rangle}(x,y,z,\mathsf{id})\right\}_{n\in N, \mathsf{id}\in\{0,1\}^*}$$

When considering a general language, mim is not well defined, as $\tilde{y}$ is not uniquely determined. In this case, whenever A chooses a statement $\tilde{x}$ that does not have a unique witness, simply let $\tilde{y}$ output $\perp$. Furthermore, we only require that the above condition holds for $\mathsf{well\text{-}behaved}$ adversaries A, where A is said to be well-behaved, if, except with negligible probability A only chooses statements $\tilde{x}$ with unique witnesses.

We remark that our notion of $\mathcal{SNMWI}$ is similar in spirit to the notion of *non-malleable witness indistinguishability* ($\mathcal{NMWI}$) recently introduced by Ostrovsky, Persiano, and Visconti [40]. Both notions consider a flavor of non-malleability for $\mathcal{WI}$ argument systems and (informally) require that the "witness in the right interaction" is "independent" of that of the left interaction. The main difference between the notions is that whereas the notion of $\mathcal{NMWI}$ "only" requires this to holds when varying the witness used in the left interaction, but keeping the statement fixed, we also require indistinguishability whenever the statements in the left interactions are indistinguishable (just as the notion of *strong witness indistinguishability* [21]). As such, our notion is interesting—in fact, the most interesting—also when considering statements with unique witnesses, whereas $\mathcal{NMWI}$ vacuously holds. In essence, the notion of $\mathcal{NMWI}$ extends the notion of plain WI to the man-in-the-middle setting, whereas $\mathcal{SNMWI}$ extends strong WI.

2.1 Constructing $\mathcal{SNMWI}$ Protocols

It easily follows (proof is given in the full version) that $\mathcal{SNMWI}$ is a relaxation of the notion of *simulation-extractability* [17, 44]. As such, we directly get that the $O(1)$-round construction of [44] is a $\mathcal{SNMWI}$ argument of knowledge; this construction relies on collision-resistant hash-functions. To minimize assumptions, we turn to provide a new construction of $\mathcal{SNMWI}$ arguments of knowledge based on any "natural" non-malleable commitment.

Protocol $\langle P_s, V_s\rangle$

Common Input: Statement $x \in L$, security parameter n and identity id.

Private Input for Prover: The witness w of statement x, $(w,x) \in R_L$.

Committing Phase:

P_s uniformly chooses σ_1 and σ_2 from $\{0,1\}^{\mathrm{poly}(n)}$.

$P_s \rightarrow V_s$: $\langle C, R\rangle$ commitment to w using randomness σ_1. Let $\mathcal{T}_1$ be the transcript of messages generated.

$P_s \rightarrow V_s$: $\langle C, R\rangle$ commitment to w using randomness σ_2. Let $\mathcal{T}_2$ be the transcript of messages generated.

Proving Phase:

$P_s \leftrightarrow V_s$: a $\langle \hat{P}, \hat{V}\rangle$ proof of the statement: there exist values w, σ_1 and σ_2 s.t $w \in R_L(x)$, and $\mathcal{T}_1$ and $\mathcal{T}_2$ are two valid commitments to w using randomness σ_1 and σ_2 respectively.

Figure 1: $\mathcal{SNMWI}$ protocol $\langle P_s, V_s\rangle$

As in [33], we consider commitment schemes that not only are non-malleable with respect to themselves, but also with respect to arbitrary 5-round protocols. Informally, a commitment scheme is non-malleable with respect to a k-round protocol $\langle B, D\rangle$, if for every man-in-the-middle adversary (interacting with $B(x_1)$ or $B(x_2)$ on the left and an honest receiver of the commitment on the right), it holds that the value it commits to is "computationally independent" of the private input, x_1 or x_2, of B, provided that it cannot distinguish the interactions with $B(x_1)$ or $B(x_2)$. (See [33] for the formal definition). All known non-malleable commitments satisfy this property, or can be easily modified to do so; we thus call such non-malleable commitments *natural*. We next show how to use any natural non-malleable commitment $\langle C, R\rangle$ to construct a $\mathcal{SNMWI}$ argument of knowledge for **NP**. The protocol is surprisingly simple: the prover first uses $\langle C, R\rangle$ to commit to a witness w *twice* (sequentially), and then provides a 4-round (stand-alone) zero-knowledge argument of knowledge of the fact that both commitments are to a valid witness w. The proof of security is, however, more subtle. Roughly, we consider two different adversarial schedulings:

Case 1: the first commitment on the right ends before the ZK argument on the left begins. In this case, indistinguishability of views and witnesses on the right, follows from the non-malleability of $\langle C, R\rangle$ (and standard ZK).

Case 2: the first commitment on the right coincides with the ZK argument on the left. In this case, however, the *second* commitment on the right must come after both the commitments on the left (and as such at most 5

messages remaining on the left, including the choice of the statement). We can now rely on non-malleability with respect to 5-round protocols of $\langle C, R\rangle$ to argue that the views and witnesses on the right are indistinguishable.

The complete proof appears in the full-version. By relying on the construction of (natural) non-malleable commitment from [33], we get that one-way functions imply $O(1)^{\log^* n}$-round $\mathcal{SNMWI}$ arguments of knowledge (If we had only relied on the original construction of [17] (see also [32]) we would have obtained a $\log n$-round construction based on one-way functions.)

3. DEFINITIONS

In this section, we present a generalization of the UC notion of security introduced by Canetti [8]. We first briefly recall the basic definition of secure computation in the UC model [25, 7, 39, 8], and then provide a brief description of the generalized model. We here focus only on *static* adversaries—i.e., players are corrupted upon invocation only.

3.1 Traditional UC

UC considers the execution of a protocol in a "larger world" captured by a special entity, called the *environment*. We assume asynchronous authenticated communication over point-to-point channels; the adversary controls the scheduling of the delivery of all messages exchanged between parties.

The environment is the driver of the execution. The execution of a protocol π with the environment Z, adversary A and trusted party $\mathcal{G}$ proceeds as follows. To start an execution of the protocol π, the environment initiates a *protocol execution session*, identified by a session identifier *sid*, and activates all the participants in that session. Z invokes all parties and assigns a unique identifier to each of them at invocation. An honest party, upon activation, starts executing the protocol π on inputs provided by the environment. Adversarially controlled parties may deviate from the protocol arbitrarily. During the protocol execution, the environment is sees all the outputs of honest parties, and is additionally allowed to communicate with the adversary (in an unrestricted fashion). At the end of the execution, Z finally outputs a bit. Some protocol executions involve "trusted parties" $\mathcal{G}$, who computes certain functionalities for the parties. Let n be the security parameter. We define $\text{EXEC}^{\mathcal{G}}_{\pi,A,Z}(n)$ to be the random variable describing the output of the environment Z resulting from the execution of the above procedure.

Ideal Execution. Let $\mathcal{F}$ be an ideal functionality (i.e., a trusted party); we denote by π_{ideal} the protocol accessing $\mathcal{F}$, called as the ideal protocol. In π_{ideal}, parties simply interact with $\mathcal{F}$ with their private inputs, and receive the corresponding outputs from the functionality at the end of the computation. The ideal model execution of the functionality $\mathcal{F}$ is the execution of the ideal protocol π_{ideal} with environment Z, adversary A' and trusted party $\mathcal{F}$. Thus, the output of the execution is $\text{EXEC}^{\mathcal{F}}_{\pi_{\mathsf{ideal}},A',Z}(n)$.

Real Execution. Let π be a multi-party protocol implementing $\mathcal{F}$. The real model execution of π is the execution of π with environment Z and adversary A, whose output is the random variable $\text{EXEC}_{\pi,A,Z}(n)$.

Security as emulation of a real model execution in the ideal model. Loosely speaking, a protocol securely realizes an ideal functionality if it securely emulates the ideal protocol π_{ideal}. This is formulated by saying that for every adversary A in the real model, there exists an adversary A' (a.k.a. *simulator*) in the ideal model, such that no environment Z can tell apart if it is interacting with A and parties running the protocol, or A' and parties running the ideal protocol π_{ideal}. We remark that in the traditional UC model the environment is only allowed to open *one* session. To capture multiple concurrent executions, we consider secure implementations of the *multi-session extension* of the functionality to be implemented (as in [10, 14]). More specifically, let $\hat{\mathcal{F}}$ denote the multi-session extension of $\mathcal{F}$: informally, $\hat{\mathcal{F}}$ runs multiple copies of $\mathcal{F}$, where each copy is identified by a special "sub-session identifier".

3.2 A Generalized Version of UC

In the UC model, the environment is modeled as a non-uniform $\mathcal{PPT}$ machine and the ideal-model adversary (or simulator) as a (uniform) $\mathcal{PPT}$ machines. We consider a generalized version (in analogy with [41, 45]) where we allow them to be in arbitrary complexity classes. Note, however, that the adversary is still $\mathcal{PPT}$. Additionally, we "strengthen" the definition by allowing the environment to output a bit string (instead of a single bit) at the end of an execution. In the traditional UC definition, it is w.l.o.g. enough for the environment to output a single bit [10]; in our generalized version this no longer holds and we are thus forced to directly consider the more stringent version.

We represent a generalized UC model by a 2-tuple $(\mathcal{C}_{\mathsf{env}}, \mathcal{C}_{\mathsf{sim}})$, where $\mathcal{C}_{\mathsf{env}}$ and $\mathcal{C}_{\mathsf{sim}}$ are respectively the classes of machines the environment and the simulator of the general model belong to. We consider only classes, $\mathcal{C}_{\mathsf{env}}$ and $\mathcal{C}_{\mathsf{sim}}$, that are closed under probabilistic polynomial time computation.

DEFINITION 4 ($(\mathcal{C}_{\mathsf{env}}, \mathcal{C}_{\mathsf{sim}})$-UC SECURITY). *Let $\mathcal{F}$ and π_{ideal} be, as defined above, and π be a multi-party protocol. The protocol π is said to realize $\mathcal{F}$ with $(\mathcal{C}_{\mathsf{env}}, \mathcal{C}_{\mathsf{sim}})$-UC security, if for every $\mathcal{PPT}$ machine A, there exists a machine $A' \in \mathcal{C}_{\mathsf{sim}}$, such that, for every $Z \in \mathcal{C}_{\mathsf{env}}$, the following two ensembles are indistinguishable w.r.t $\mathcal{C}_{\mathsf{sim}}$.*

$$\left\{\text{EXEC}_{\pi,A,Z}(n)\right\}_{n\in N} \approx \left\{\text{EXEC}^{\mathcal{F}}_{\pi_{\mathsf{ideal}},A',Z}(n)\right\}_{n\in N}$$

Using the above notation, traditional UC is equivalent to (n.u.$\mathcal{PPT}$,$\mathcal{PPT}$)-UC-security. We let QPS-UC denote (n.u.$\mathcal{PPT}$,$\mathcal{PQT}$)-UC-security[9] (where $\mathcal{PQT}$ denotes probabilistic quasi-polynomial time algorithms), and Non-uniform UC denote ($\mathcal{PPT}$,n.u.$\mathcal{PPT}$)-UC-security.

4. CONSTRUCTIONS

By relying on previous results [42, 43, 34, 14, 23] the construction of a UC secure protocol for realizing any multi-party functionality reduces to the task of constructing a zero-knowledge protocol ssZK that satisfies the following two properties:[10]

[9] We mentioned that this is stronger than the notion of QPS security of [41, 4, 47] which only consider indistinguishability w.r.t $\mathcal{PPT}$; we, in analogy with the notion of *strong QPS* of [41] require indistinguishability to hold also w.r.t $\mathcal{PQT}$.

[10] Formally, this can be modelled as implementing a particular "zero-knowledge" proof of membership functionality.

UC simulation: For every adversary A receiving honest proofs of statements x using witness w, where (x, w) are selected by an "environment" $\mathcal{Z}$, there exists a simulator S (which only get the statements x) such that no $\mathcal{Z}$ can distinguish if it is talking to A or S.

Concurrent simulation-soundness: Even an adversary that receives an unbounded number of concurrently *simulated* proofs, of statements selected by the environment $\mathcal{Z}$, still is not able to prove any false statements.

Below we introduce the notion of a *UC-puzzle*. Then, we show how to use any UC-puzzle and a $\mathcal{SNMWI}$ protocol to construct a zero knowledge protocol $\langle P, V\rangle$ that is UC-simulatable and concurrently simulation-sound.

4.1 UC-puzzles

Roughly speaking, a UC puzzle is a protocol $\langle S, R\rangle$ between two players—a *sender* and a *receiver*—and a $\mathcal{PPT}$-computable relation $\mathcal{R}$, such that the following two properties hold:

Soundness: No efficient receiver R^* can successfully complete an interaction with S and also obtain a "trapdoor" y, such that $\mathcal{R}(\mathsf{TRANS}, y) = 1$, where TRANS is the transcript of the interaction.

Statistical UC-simulation: For every efficient adversary A, participating in a polynomial number of concurrent executions with receivers R (i.e., A is acting as a puzzle sender in all these executions) and at the same time communicating with an environment $\mathcal{Z}$, there exists a simulator S that is able to *statistically* simulate the view of A for $\mathcal{Z}$, while at the same time outputting trapdoors to all successfully completed puzzles.

Formally, let $n \in N$ be a security parameter and $\langle S, R\rangle$ be a protocol between two parties, the sender S and the receiver R. We consider a concurrent puzzle execution for an adversary A. In a concurrent puzzle execution, A exchanges messages with a puzzle-environment $Z \in \mathcal{C}_{\mathsf{env}}$ and participates as a sender concurrently in $m = poly(n)$ puzzles with honest receivers $R_1, \ldots, R_m$. At the onset of a concurrent execution, Z outputs a session-identifier *sid* that all receivers in the concurrent puzzle execution receive as input. Thereafter, the puzzle-environment is allowed to exchange messages only with the adversary A. We compare a *real* and an *ideal* execution.

Real execution. In the real execution, the adversary A on input 1^n, interacts with a puzzle-environment $Z \in \mathcal{C}_{\mathsf{env}}$ and participates as a sender in m interactions using $\langle S, R\rangle$ with honest receivers that receive input *sid* (decided by Z). The adversary A is allowed to exchange arbitrary messages with environment Z when participating in puzzle interactions with the receivers as a sender. We assume without loss of generality that, after every puzzle-interaction, A honestly sends TRANS to Z, where TRANS is the puzzle-transcript. Finally, Z outputs a string in $\{0,1\}^*$. We denote this by $\text{REAL}_{A,Z}(n)$.

Ideal execution. Consider $A' \in \mathcal{C}_{\mathsf{sim}}$ in the ideal-model that has a special output-tape (not accessible by Z). In the ideal execution, A' on input 1^n interacts with puzzle-environment Z. We denote the output of Z at the end of the execution by $\text{IDEAL}_{A',Z}(n)$.

DEFINITION 5 (UC-PUZZLE). *A pair $(\langle S, R\rangle, \mathcal{R})$ is a $(\mathcal{C}_{\mathsf{env}}, \mathcal{C}_{\mathsf{sim}})$-secure UC-puzzle for a polynomial time computable relation $\mathcal{R}$ and model $(\mathcal{C}_{\mathsf{env}}, \mathcal{C}_{\mathsf{sim}})$, if the following conditions hold.*

- **Soundness:** *For every malicious $\mathcal{PPT}$ receiver A, there exists a negligible function $\nu(\cdot)$ such that the probability that A, after an execution with R on common input 1^n, outputs y such that $y \in \mathcal{R}(\mathsf{TRANS})$ where TRANS is the transcript of the messages exchanged in the interaction, is at most $\nu(n)$.*
- **Statistical Simulatability:** *For every adversary $A \in \mathcal{C}_{\mathsf{adv}}$ participating in a concurrent puzzle execution, there is a simulator $A' \in \mathcal{C}_{\mathsf{sim}}$ such that for all puzzle-environments $Z \in \mathcal{C}_{\mathsf{env}}$, the ensembles $\{\text{REAL}_{A,Z}(n)\}_{n\in N}$ and $\{\text{IDEAL}_{A',Z}(n)\}_{n\in N}$ are statistically close over $n \in N$ and whenever A' sends a message of the form TRANS to Z, it outputs y in its special output tape such that $y \in \mathcal{R}(\mathsf{TRANS})$.*

In other words, we require that no adversarial receiver can complete a puzzle with a trapdoor, but there exists a simulator (which does not rewind the environment it runs in) that can generate *statistically indistinguishable* puzzle transcripts joint with trapdoors. We highlight that the puzzle protocol $\langle S, R\rangle$ may make use of trusted set-up.

As we show, UC-puzzles in a trusted set-up model $\mathcal{T}$ are *sufficient* for achieving UC secure computation with set-up $\mathcal{T}$. This result also holds in generalized versions of the UC framework (which allow us to consider QPS and non-uniform UC security).

4.2 Simulation-Sound Zero Knowledge (ssZK)

We now provide the construction of a concurrently simulation-sound protocol ssZK, based on any $\mathcal{SNMWI}$ argument of knowledge protocol and a UC-puzzle. As mentioned earlier, this will conclude that UC secure computation is feasible in any model where there exists a UC puzzle. Let $\langle P_s, V_s\rangle$ be a $\mathcal{SNMWI}$ argument of knowledge protocol of **NP** and let $\langle S, R\rangle$ be a UC-puzzle. The protocol $\langle P, V\rangle$ for an NP language L, proceeds as follows. On common input the statement x and identity id, and the private input $w \in R_L(x)$, the prover P first executes the puzzle $\langle S, R\rangle$ with the verifier V; the prover here acts as the puzzle receiver. Next P commits to w using a perfect binding commitment scheme com[11], and finally, P provides a proof, using $\langle P_s, V_s\rangle$, of the statement that either it has committed to a valid witness of x, or a trapdoor of the puzzle.

Soundness of $\langle P, V\rangle$ directly follows from the argument of knowledge property of $\mathcal{SNMWI}$ and the soundness of the puzzle $\langle S, R\rangle$. To perform simulation, simply first run the simulator for the puzzle, next commit to the trapdoor (obtained by the puzzle simulator), and use this trapdoor as a "fake" witness in the $\mathcal{SNMWI}$ argument.

It only remains to show that $\langle P, V\rangle$ is simulation sound. Consider an adversary A that receives an unbounded number of concurrently simulated proofs, of statements x selected by an environment $\mathcal{Z}$ that also outputs valid witnesses w for x. We need to show that A will not be able to prove

[11]For simplicity, we here assume a perfectly binding commitment scheme; it is actually sufficient to use a statistically binding commitments.

any false statements, even if it is attempting to do so while receiving the simulated proofs.

First note that, in the *real* execution, when A does not receive any simulated proof, the probability that A can commit to a "fake-witness" (i.e., the trapdoor to the puzzle) is negligible; this follows from the argument of knowledge property of $\langle P_s, V_s \rangle$ and the soundness of the puzzle $\langle S, R \rangle$. Now consider a hybrid experiment H which is identical to the real execution, but where all the puzzles sent out by A (i.e., where A is acting as a challenger) are simulated; the rest of the execution is identical (i.e., we still never use the trapdoor from the puzzles). It directly follows from the *statistical* simulation property of $\langle S, R \rangle$ that A still only commits to a fake-witness with negligible probability.[12] Next, consider the following sequence of hybrid experiments. Let $q(n)$ denote an upper-bound on the number of concurrent executions initiated by A. Let H_i denote a hybrid experiment where all puzzles are simulated, the first i proofs seen by A are generated using the real witnesses (output by $\mathcal{Z}$), and the rest are simulated using the fake-witness. Note that by definition $H_0 = H$. We can now apply the definition of $\mathcal{SNMWI}$ to conclude that the probability that A commits to a fake witness is essentially the same between two intermediary hybrids; we conclude that also in $H_{q(n)}$ the probability that A committed to fake witness is negligible. It follows by the soundness of $\langle P_s, V_s \rangle$ that A thus (almost) never is able to prove any false statements in $H_{q(n)}$, which concludes that $\langle P, V \rangle$ is simulation-sound (since in $H_{q(n)}$ A receives only simulated proofs). The formal proof appears in the full-version.

Protocol $\langle P, V \rangle$

Common Input: Statement $x \in L$, security parameter n, identity id, session-id sid.

Private Input for Prover: The witness w of statement x, $(w, x) \in R_L$.

Preamble:

P $\leftrightarrow$ V: an interaction using $\langle S, R \rangle$ on input 1^n, where P is the receiver and V is the sender. Let TRANS be the transcript of the messages exchanged.

Committing Phase:

P uniformly chooses $r' \in \{0,1\}^{poly(n)}$.

P $\rightarrow$ V: $c = \mathsf{com}(w, r')$.

Proving Phase:

P $\leftrightarrow$ V: a $\langle P_s, V_s \rangle$ proof using identity id of the statement

either there exists values w, r' s.t $c = \mathsf{com}(w, r')$ and $(x, w) \in R_L$

or there exists values u, r' s.t $c = \mathsf{com}(u, r')$ and $u \in R_{L'}(\mathsf{TRANS})$.

Figure 2: Simulation sound zero-knowledge protocol $\langle P, V \rangle$

4.3 Some instantiations of UC puzzles

By Theorem 1, it suffices to provide a UC puzzle to demonstrate feasibility of UC secure computation. In this section, we briefly outline some simple constructions of UC-puzzles in various models. The complete proofs (as well as more examples of models) appear in the full-version.

Non-Uniform UC. In the non-uniform UC model, we consider a uniform poly-time adversary, but a non-uniform simulator. Assume, for simplicity, the existence of an evasive set Δ in **BPP**. Recall that a set Δ is evasive if, Δ is non-empty, but no $\mathcal{PPT}$ algorithm can find an element in Δ [22]. Then, the empty protocol is a valid puzzle; the "trapdoor" is simply an element from a Δ. By definition, no $\mathcal{PPT}$ can find a trapdoor, but a non-uniform simulator can simply get an element from Δ as non-uniform advice. In the actual construction, we show it suffices to have an evasive promise-problem in **BPP**[13] to construct a puzzle.

THEOREM 6. *Assume the existence of enhanced trapdoor permutations, an t-round $\mathcal{SNMWI}$ protocol and an evasive promise problem in* **BPP**. *Then, for every well-formed ideal functionality $\mathcal{F}$[14], there exists a $O(t)$-round protocol π that realizes $\hat{\mathcal{F}}$ with Non-Uniform UC-security.*

We also complement the feasibility result, by showing that evasive promise-problems are necessary for achieving non-uniform UC security.

THEOREM 7. *If there exists a protocol Π that securely realizes the ideal commitment functionality $\mathcal{F}_{\mathsf{com}}$[15] with Non-Uniform UC-security, then there exists an evasive promise problem in* **BPP**.

UC with QPS. In the QPS model, the simulator is allowed to run in quasi-poly time, but the adversary is only poly-time. Let f be a one-way function that is hard for poly-time, but easy to break in quasi-poly time. Consider the puzzle consisting of the challenger sending $s = f(r)$ (for a random r) and then providing a witness hiding argument of the fact that s is in the range of f. The trapdoor is a string r' s.t. $f(r') = s$; clearly no poly-time adversary can find such a string, but by definition, a quasi-poly simulator can (by breaking f).

THEOREM 8. *Assume the existence of enhanced trapdoor permutations, an t-round $\mathcal{SNMWI}$ protocol secure w.r.t $\mathcal{PQT}$ and one-way functions that can inverted w.p. 1 in $\mathcal{PQT}$. Then, for every well-formed ideal functionality $\mathcal{F}$, there exists a $O(t)$-round protocol π that realizes $\hat{\mathcal{F}}$ with QPS UC-security.*

In the full version we show that a weaker assumption suffices (if additionally assuming the existence of collision-resistant hashfunctions). Namely, that there exists a $\mathcal{PPT}$ interactive machine M, that is "easy" for $\mathcal{PQT}$ machines and "hard" for $\mathcal{PPT}$ machines. More precisely, there is a $\mathcal{PQT}$ machine P such that the output of M in an interaction with P is 1

[12] Statistical (as opposed to computational) indistinguishability is required here as the values committed to are not efficiently computable.

[13] Formally, a promise problem $\Delta = (\Delta_Y, \Delta_N)$ is *evasive*, if for all n, $\Delta_Y \cap \{0,1\}^n \neq \emptyset$ and for every $\mathcal{PPT}$ machine M, there is a negligible function $\nu(\cdot)$, such that,

$$Pr[M(1^n) \in \{0,1\}^n \backslash \Delta_N] \leq \nu(n)$$

[14] See [14] for a formal definition of well-formed functionalities.

[15] See [14] for a formal definition of $\mathcal{F}_{\mathsf{com}}$.

with high probability, but for all $\mathcal{PPT}$ machines P^*, the output of M is 1 with at most negligible probability. We also show this weaker assumption is necessary to achieve QPS UC-security.

The (Imperfect) Reference String model. In the common reference string (CRS) model all parties have access to a reference string ("the CRS") sampled according to some pre-specified distribution. To establish a puzzle, consider a CRS selected as $c = g(s)$ where s is a random string and g is a pseudo-random generator; the trapdoor is a string s' s.t. $c = g(s')$. Clearly no adversary can find such a string, but a simulator setting up the CRS can easily obtain a trapdoor. The same construction also establishes that UC security is possible in the *uniform reference string model*, where the CRS is a uniform random string. A similar construction proves feasibility in the multi-CRS model [28] as well, where there are multiple reference strings that all parties have access to and the adversary is allowed to corrupt at most half of them. In this model, the trapdoor is inverse of at least $\frac{1}{2}$ of the reference strings under the pseudo-random generator g.

A variant of this puzzle (essentially implicit in [15]) is also sufficient to establish that a *single* 'imperfect" reference string [15]; roughly speaking, here the trapdoor is a "short" description of the reference string. This model considers the ideal functionality $\mathcal{F}_{\mathsf{sun}}$ that sets the reference string by sampling uniformly from an efficient distribution D (that has sufficient min-entropy) which is decided by the adversary A.[16] As in [15], we consider (μ, d, t)-conforming adversaries, i.e. the sampling algorithm D set up by the adversary outputs reference strings of length n, has description size at most $d(n)$, and generates an output within $t(n)$ steps and has min-entropy at least $\mu(n)$.

THEOREM 9. *Assume the existence of enhanced trapdoor permutations and collision-resistant hash-functions. Then, for every well-formed ideal functionality $\mathcal{F}$, there exists a $O(1)$-round protocol π in the $\mathcal{F}_{sun}$-hybrid that realizes $\hat{\mathcal{F}}$ with UC-security w.r.t (μ, d, t)-conforming adversaries where $\mu(n) - d(n) > n^{\epsilon}$ for some $\epsilon > 0$.*

The Timing model. The timing model was originally introduced by Dwork, Naor and Sahai [19] in the context of constructing concurrent zero-knowledge protocols. More recently, Kalai, Lindell and Prabhakaran [31], showed how to achieve secure computation in the timing model. In the timing model, we consider networks that have a (known) maximum latency Δ (i.e., the time it takes for a message to be computed, sent and delivered over the network). We also assume that each party possesses a local clock, which is partially synchronized with other clocks; that is, the relative drift between clocks of honest parties is bounded by some constant factor ϵ. To make use of the timing model, all parties have access to the constructs $\mathsf{delay}\ \lambda$ (which delays a message by time λ) and $\mathsf{time\text{-}out}\ \lambda$ (abort if the next-message is not received within time t). We can construct a UC-puzzle in this model if assuming that the environment delays all its messages by time δ: simply require the challenger to send $s = f(r)$ (for a random r and one-way function f) and provide a witness hiding argument *of knowledge* (WH AOK) of the fact that s is in the range of f; the trapdoor is a string r' s.t. $f(r') = s$. Here, the receiver is required to complete the WH AOK within time $\delta = \delta(\epsilon, \Delta)$. Clearly, no $\mathcal{PPT}$ adversary acting as a receiver can find a trapdoor. To extract a trapdoor, the simulator simply rewinds the adversary in the WH AOK protocol. Since, the adversary is required to finish executing the WH POK within δ steps, and every other message in the network (i.e., messages from the environment) are delayed by δ, the simulator can rewind the adversary without rewinding messages from the environment.

We say that an adversary is ϵ-drift preserving if for every pair of honest parties P_0 and P_1 (and including the environment), the relative-drift of P_0 and P_1's clock between any two successive "events" is bounded by ϵ (see [31] for more details). The environment is said to be δ-delaying, if every message sent from the environment is delayed by at least δ. Similarly, a protocol is said to be δ-delayed if every message sent in the protocol is delayed by at least δ. We show the following feasibility theorem in this model.

THEOREM 10. *Let $\epsilon > 1$ and $\Delta > 0$ be constants. Assume the existence of enhanced trapdoor permutations, a $2\epsilon^2\Delta$-delayed t-round $\mathcal{SNMWI}$ protocol. Then, for every well-formed ideal functionality $\mathcal{F}$, there exists a $O(t)$-round protocol π that realizes $\hat{\mathcal{F}}$ with (timed) UC-security in networks that have maximum latency Δ w.r.t ϵ-drift preserving adversaries and $2\epsilon^2\Delta$-delaying environments.*

We complement this result by showing that to achieve feasibility, it is necessary for the environment to be $O(\Delta)$-delaying and the protocol to execute for at least $O(\Delta)$ steps.

Stand-alone/Parallel model. We model *stand-alone* secure computation as a UC-model with a restricted environment and show feasibility. In this model, the environment is restricted to exchange messages with the adversary either before a protocol execution begins or after it completes; in particular, it is not allowed to exchange messages during the protocol execution. This can be viewed as a variant of the timing model. Indeed, as it turns out, the puzzle we consider for this model is similar to the timing model which involves the sender sending $s = f(r)$ for random r followed by a WH-POK that s was computed correctly. We achieve simulation by rewinding the adversary and extracting the witness for s in the argument-of-knowledge sub-protocol. We remark that in a rewinding, the simulator does not have to simulate any message from the environment, since the environment is not allowed to interact with the adversary during puzzle interactions. The *parallel* model of computation is a generalization of the stand-alone computation model, where the adversary is restricted to run all protocol executions in parallel (in a lock-step fashion). The same puzzle as for the stand-alone model establishes feasibility in this model as well.

THEOREM 11. *Assume the existence of enhanced trapdoor permutations, a t-round $\mathcal{SNMWI}$ protocol. Then, for every well-formed ideal functionality $\mathcal{F}$, there exists a $O(t)$-round protocol π that realizes $\hat{\mathcal{F}}$ with parallel security.*

5. ACKNOWLEDGEMENTS

We are very grateful to Ran Canetti for many helpful and delightful conversations.

[16] In [15], they let the environment set the distribution. Here, for simplicity we let the adversary choose the distribution.

6. REFERENCES

[1] B. Barak. How to go Beyond the Black-Box Simulation Barrier. In *42nd FOCS*, pages 106–115, 2001.

[2] B. Barak and O. Goldreich. Universal Arguments and their Applications. In *17th CCC*, pages 194–203, 2002.

[3] B. Barak and M. Prabhakaran and A. Sahai. Concurrent Non-Malleable Zero Knowledge. In *47th FOCS*, pages 345-354, 2006.

[4] B. Barak, A. Sahai. How To Play Almost Any Mental Game Over The Net - Concurrent Composition via Super-Polynomial Simulation. In *46th FOCS*, pages 543-522, 2005.

[5] B. Barak and R. Pass. On the Possibility of One-Message Weak Zero-Knowledge. In *TCC 2004*, pages 121-132.

[6] B. Barak, R. Canetti, J. B. Nielsen, and R. Pass. Universally Composable Protocols with Relaxed Set-Up Assumptions. In *45th FOCS*, pages 186-195, 2004.

[7] D. Beaver. Foundations of Secure Interactive Computing. In *Crypto91*, pages 377–391, 1991.

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

[9] R. Canetti. Obtaining Universally Composable Security: Towards the Bare Bones of Trust. In *Asiacrypt*, pages 88-112, 2007.

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

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

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

[13] R. Canetti, Y. Dodis, R. Pass, S. Walfish. Universally Composable Security with Global Setup. In *4th TCC*, 2007

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

[15] R. Canetti, R. Pass, A. Shelat. Cryptography from Sunspots: How to Use an Imperfect Reference String. In *48th FOCS*, pages 249-259, 2007.

[16] Y. Dodis, S. Micali: Parallel Reducibility for Information-Theoretically Secure Computation. In *CRYPTO 2000*, pages 74-92.

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

[18] C. Dwork, M. Naor. Pricing via Processing or Combatting Junk Mail. In *Crypto 1992*, pages 139-147, 1992.

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

[20] A. Feige and A. Shamir. How to Prove Yourself: Practical Solutions to Identification and Signature Problems. In *Crypto86*, LNCS 263, pages 181–187, 1987.

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

[22] O. Goldreich and A. Kahan. How to Construct Constant-Round Zero-Knowledge Proof Systems for NP. *Jour. of Cryptology*, Vol. 9, No. 2, pages 167–189, 1996.

[23] 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.

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

[25] S. Goldwasser and L. Levin. Fair Computation of General Functions in Presence of Immoral Majority. In *CRYPTO'90,* LNCS 537, pages 77–93, 1990.

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

[27] 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.

[28] J. Groth and R. Ostrovsky. Cryptography in the Multi-string Model. In *CRYPTO 2007*, pages 323-341, 2007.

[29] J. Katz. Universally Composable Multi-Party Computation using Tamper-Proof Hardware. In *Eurocrypt 2007*, pages 115-128, 2007.

[30] J. Katz, R. Ostrovsky and A. Smith. Round Efficiency of Multi-Party Computation with a Dishonest Majority, In *EuroCrypt2003*. LNCS 2656 pages 578–595, 2003.

[31] Y. T. Kalai, Y. Lindell, and M. Prabhakaran. Concurrent general composition of secure protocols in the timing model.In *37th STOC*, pages 644-653, 2005.

[32] H. Lin, R. Pass, and M. Venkitasubramaniam. Concurrent Non-Malleable Commitments from One-way Functions. In *TCC 2008*, pages 571-588, 2008.

[33] H. Lin, and R. Pass. Non-Malleability Amplification. In *41st STOC*, 2009.

[34] Y. Lindell. Bounded-Concurrent Secure Two-Party Computation Without Setup Assumptions. In *35th STOC*, pages 683-692, 2003.

[35] Y. Lindell. General Composition and Universal Composability in Secure Multi-Party Computation. In *44th FOCS*, pages 394–403, 2003.

[36] S. Micali. CS Proofs. *SIAM Jour. on Computing*, Vol. 30 (4), pages 1253–1298, 2000.

[37] S. Micali and R. Pass. Local Zero Knowledge. In *38th STOC*, pages 306-315, 2006.

[38] S. Micali, R. Pass, A. Rosen. Input-Indistinguishable Computation. In *47th FOCS*, pages 367-378 , 2006.

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

[40] R. Ostrovksy, G. Persiano, I. Visconti. Concurrent Non-Malleable Witness Indistinguishability and its applications. ECCC TR06-095.

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

[42] R. Pass. Bounded-Concurrent Secure Multi-Party Computation with a Dishonest Majority. In *36th STOC*, 2004, pages 232-241, 2004.

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

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

[45] M. Prabhakaran and A. Sahai. New notions of security: achieving universal composability without trusted setup. In *36th STOC*, pages 242-251, 2004.

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

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

[48] A. Sahai. Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security. In *40th FOCS*, pages 543-553, 1999.

[49] L. Trevisan and S. Vadhan. Extracting Randomness from Samplable Distributions. In *FOCS 2000*.

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