Black-Box Constructions of Two-Party Protocols from One-Way Functions

Rafael Pass* and Hoeteck Wee**

[1] Cornell University rafael@cs.cornell.edu
[2] Queens College, CUNY hoeteck@cs.qc.cuny.edu

Abstract. We exhibit constructions of the following two-party cryptographic protocols given only black-box access to a one-way function:

- constant-round zero-knowledge arguments (of knowledge) for any language in NP;
- constant-round trapdoor commitment schemes;
- constant-round parallel coin-tossing.

Previous constructions either require stronger computational assumptions (e.g. collision-resistant hash functions), non-black-box access to a one-way function, or a super-constant number of rounds. As an immediate corollary, we obtain a constant-round black-box construction of secure two-party computation protocols starting from only semi-honest oblivious transfer. In addition, by combining our techniques with recent constructions of concurrent zero-knowledge and non-malleable primitives, we obtain black-box constructions of concurrent zero-knowledge arguments for NP and non-malleable commitments starting from only one-way functions.

Key words: black-box constructions, zero-knowledge arguments, trapdoor commitments, parallel coin-tossing, secure two-party computation, non-malleable commitments

1 Introduction

Much of the modern work in foundations of cryptography rests on general cryptographic assumptions like the existence of one-way functions and trapdoor permutations. General assumptions provide an abstraction of the functionalities and hardness we exploit in specific assumptions such as hardness of factoring and discrete log without referring to any specific underlying algebraic structure. The expressive nature of general assumptions means that we could then derive constructions based on a large number of concrete assumptions of our choice, even ones that may not have been considered at the time of designing the

* 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

** most of this work was done while a post-doc at Columbia University, supported in part by NSF Grants CNS-0716245 and SBE-0245014.

protocols. Constructions based on general assumptions may use the primitive guaranteed by the assumption in one of two ways:

Black-box usage: A construction is black-box if it refers only to the input/output behavior of the underlying primitive; we would typically also require that in the proof of security, we can use an adversary breaking the security of the construction as an oracle to break the underlying primitive (c.f. [39, 27]).

Non-black-box usage: A construction is non-black box if it uses the code computing the functionality of the primitive.

Motivated by the fact that the vast majority of constructions in cryptography indeed are black-box, a rich and fruitful body of work initiated in [27] seeks to understand the power and limitations of black-box constructions in cryptography, resulting in a fairly complete picture of the relations amongst most cryptographic primitives with respect to black-box constructions. We stress that the general question of whether we can securely realize tasks via black-box access to a general primitive is not merely of theoretical interest. A practical reason is related to efficiency, as non-black box constructions tend to be less efficient due to the use of general NP reductions to order to prove statements in zero knowledge; this impacts both computational complexity as well as communication complexity. As such, non-black box constructions traditionally only serve as "feasibility" results; moreover, the constructions underlying such feasibility results often do not translate readily into "practical" black-box constructions without a recourse to the use of either specific assumptions or additional general assumptions.

Fortunately, a recent line of work has narrowed - and in several cases even closed - the gap between black-box and non-black-box constructions for many cryptographic tasks. A notable example is the work of Ishai et al. [28, 24], which building on the early work of Kilian's [30], provides a black-box construction of secure multi-party protocols that can tolerate any number of static malicious adversaries, assuming only the existence a semi-honest oblivious transfer protocol (which can in turn be based on homomorphic encryption schemes or enhanced trapdoor permutations); the corresponding non-black-box feasibility result was known since the 1980s [21]. Several other works address improvements in efficiency for two-party protocols and multi-party protocols with an honest majority e.g. [12, 34], as well as public-key encryption schemes secure against chosen-ciphertext attacks and variants thereof [9, 37]. In fact, the recent success we have had with black-box constructions of secure protocols seems to hint that there is perhaps no inherent gap between non-black-box "feasibility" results and black-box "practical" constructions for natural cryptographic tasks: that is,

any feasibility result may also be realized by a practical black-box construction under the *same* assumptions. If so, this would be a stark contrast to black-box versus non-black-box use of the adversary's code in the proof of security in simulation-based notions of security, for which a gap has been established [20, 1].

Upon closer examination, one notices that while the afore-mentioned black-box constructions of secure protocols do improve on the efficiency of previous non-black-box constructions as measured in terms of computational and communication complexity, most (except for [12]) do not match the round complexity of existing non-black-box constructions. Indeed, there are several fundamental constant-round two-party cryptographic tasks, notably zero-knowledge arguments for NP for which we do know how to realize via non-black-box usage of a one-way function [16], but existing black-box constructions either require a super-constant number of rounds or stronger assumptions [21, 19]. This raises the following intriguing question:

> Is there an inherent trade-off between round complexity and either efficiency or computational assumptions in realizing these two-party cryptographic tasks?

Put differently, if we require a constant-round zero-knowledge argument system, must we necessarily turn to a non-black-box construction (thereby incurring a loss in efficiency) or use collision-resistant hash functions (a stronger assumption)? Interestingly, the Feige-Shamir zero-knowledge arguments [16] constitute one of the earliest examples of non-black-box constructions; the same work also presents a non-black-box construction of constant-round trapdoor commitments from one-way functions, for which there is again a gap with respect to existing black-box constructions. Other related tasks with a similar gap include parallel coin-tossing from one-way functions, and secure two-party computation from semi-honest oblivious transfer. In each of these cases, constant-round non-black-box constructions are known [33, 43, 21], whereas existing black-box constructions either *additionally* assuming collision-resistant hash functions or constant-round statistically hiding commitments [19, 13, 33, 34] (which we know cannot be realized via black-box access to a one-way function [25, 42][3]) or by considering protocols with super-constant number of rounds. We summarize these prior results in Figure 1.

[3] This is true even if we allow black-box access to semi-honest oblivious transfer, by observing that the impossibility result in [25] extends to enhanced trapdoor permutations with which we could realize constant-round semi-honest oblivious transfer [15].

cryptographic task	black-box OWF	non-black-box OWF	black-box SHC
zero-knowledge argument	$\omega(1)$ [22]	$O(1)$ [16]	$O(1)$ [19]
trapdoor commitments (m bits)	$\tilde{O}(m)$ [13]	$O(1)$ [16]	$O(1)$ [13]
coin-tossing (m coins)	$\tilde{O}(m)$ [3]	$O(1)$ [33]	$O(1)$ [33]

Fig. 1. Round complexity of existing constructions of several cryptographic tasks from one-way functions (OWF) and constant-round statistically hiding commitments (SHC). Here $\tilde{O}(\cdot)$ hides savings of multiplicative factors that are logarithmic in the security parameter. The trapdoor commitment with $\tilde{O}(m)$ rounds from black-box OWF is obtained by combining the [13] protocol with Blum's coin-tossing protocol [3].

Our results. In this work, we answer the afore-mentioned question negatively: we present black-box constructions of constant-round zero-knowledge arguments for NP and several other two-party functionalities under the minimal assumption of one-way functions:

Theorem 1 (informal). *There exist black-box constructions of constant-round zero-knowledge arguments (of knowledge) for* NP*, constant-round trapdoor commitments and constant-round parallel coin-tossing, starting from any one-way function.*

We stress that reducing the computational assumptions for these cryptographic protocols from collision-resistant hash functions to one-way functions is important also in practice; recent attacks on the popular MD4, MD5 and SHA1 hash functions demonstrate that achieving collision-resistance in the heuristic sense is much harder than achieving one-way'ness.

The above constructions may be modified to achieve security against adaptive corruptions in the stand-alone model (c.f. [5]) while maintaining constant round complexity. This improves on the early work of Beaver [2], who provided constructions assuming hardness of factoring. The idea is to have the receiver in the commitment scheme from [13] (which we observe to be adaptively secure) commit to its challenge using our trapdoor commitment scheme.

Secure two-party computation. A series of recent work [28, 34, 24, 10, 29] (building on [30]) provided a black-box construction of secure two-party protocols starting from semi-honest oblivious transfer. The resulting protocol has constant round complexity assuming a constant-round parallel coin-tossing protocol. The following result then follows as an immediate corollary of our coin-tossing protocol:

Theorem 2 (informal). *There exists a black-box construction of constant-round secure two-party computation protocol with respect to static malicious adversaries, starting from any constant-round oblivious transfer protocol secure against static semi-honest adversaries.*

This result also extends to any constant number of parties, while preserving constant round complexity. We point out that in concurrent work, Choi et al. [10] established an analogous statement for adaptive corruptions, using as a building block our trapdoor commitment schemes tolerating adaptive corruptions.

Additional constructions. Combining our techniques with previous work, we also obtain black-box constructions of concurrent zero-knowledge arguments and non-malleable commitments from one-way functions:

Theorem 3 (informal). *There exist black-box constructions of the following cryptographic protocols starting from any one-way function:*

- *concurrent zero-knowledge arguments for* NP *with* $c \log n$ *rounds for any super-constant function* $c(\cdot)$*;*
- *non-malleable commitments with* $O(\log n)$ *rounds, and concurrent non-malleable commitments with* $O(n)$ *rounds.*

The concurrent zero-knowledge argument system follows readily from modifying the challenge-response preamble in our stand-alone zero-knowledge argument system in the manner of [38, 36]. The non-malleable commitment scheme requires substantially more work, combining ideas from our stand-alone zero-knowledge argument system, an encoding scheme from [9] along the messaging scheduling and analysis from [14, 32].

2 Overview of our constructions

We begin with our overview of our constant-round zero-knowledge arguments and trapdoor commitment schemes, which are obtained by applying a compiler to challenge-response protocols with a certain structure.

Challenge-response protocol. Consider a 3-round challenge-response protocol, say between a prover and a verifier with possibly a common input, with the following structure: In the first round, the prover commits to values $v_1, \ldots, v_k$ (bit by bit, in parallel). The verifier responds with a random challenge $e \in \{0,1\}^k$, and the prover responds by opening to some subset of bits in each value $v_1, \ldots, v_k$. Then verifier then decides whether to accept or reject.

SPECIAL SOUNDNESS: For every message in the first round, there exists at most one "easy challenge" $\tilde{e}$ that allows the prover to cheat. For a language, cheating means convincing the verifier to accept a NO instance; for a commitment scheme, cheating means generating an accepting commit phase transcript that can be opened to two different values. Moreover, we require that the "easy challenge" is efficiently recoverable in the following sense: there is an efficient procedure that given the values $v_1, \ldots, v_k$ (along with the common input), outputs a string $\tilde{e}$ such that if an easy challenge exists, it must equal $\tilde{e}$.

LOOK-AHEAD SIMULATION: Roughly speaking, this condition says simulation is easy if we can look ahead and obtain the verifier's challenge e. For a language, this condition stipulates that the protocol is special honest-verifier zero-knowledge [11]: we require that the simulator on input any fixed verifier's challenge e generates an "honest-looking" transcript. Here, honest-looking means computationally indistinguishable from an honest prover-verifier interaction wherein the verifier always sends e. For a commitment scheme, this condition stipulates that there exists a simulator that on input any fixed verifier's challenge e generates an "honest-looking" transcript of the commit phase that can be later opened to any value v. Here, honest-looking means computationally indistinguishable from an honest commitment and opening to the value v wherein the verifier always sends e in the commit phase.

The compiler. We have the verifier commits to its challenge e in advance before running the challenge-response protocol. Indeed, this approach was adopted in [19, 17] for zero knowledge, and in [13] for trapdoor commitments. The difficulty is that we do not know how to guarantee soundness as there could be a malleability attack (specifically, we do not know how to rule out the possibility that after seeing the verifier's commitment to e, the cheating prover could send some carefully crafted commitments that can be open to a valid accepting response once the verifier opens the commitment to e). This problem can be circumvented in one of three ways:

- Have the verifier commit using a *perfectly hiding* commitment scheme and the prover use a *statistically binding* commitment scheme [19, 13].
- Have the verifier commit using a *trapdoor* commitment scheme and the prover use a *statistically binding* commitment scheme (implicit in [33, 38, 41, 23][4]).

[4] In these works (specifically, the protocols based on one-way functions), the verifier commits to its challenge e, and to reveal the challenge, it sends the string e, along with a zero-

– Have both the prover and verifier commit using a *computationally hiding* commitment scheme, but have the prover prove that it "knows" the values underlying its commitments (e.g., by using a zero-knowledge proof of knowledge) before the verifier opens the commitment to its challenge [17, Sec 4.9.2.2].

We adopt the third approach in this paper. Specifically, we use an extractable commitment scheme, which is informally a commitment scheme with a proof of knowledge property. Such a commitment scheme can be constructed via black-box access to any commitment scheme using cut-and-choose techniques [38, 14]. Note that the first approach cannot work in our setting because there is no black-box construction of constant-round perfectly hiding commitment schemes from one-way functions [25], whereas the second requires a functionality that we are trying to construct.

Towards trapdoor commitments & parallel coin-tossing. For zero-knowledge arguments, Blum's challenge-response protocol for the NP-complete problem Graph Hamiltonicity [4] suffices. On the other hand, for trapdoor commitments, we need to design a new challenge-response protocol because we do not know how to efficiently recover the easy challenge in the [13] protocol. Next, we show how to derive an extractable trapdoor commitment scheme starting from any trapdoor commitment scheme (such as ours), and from there, we obtain a constant-round parallel coin-tossing protocol from the works of [3, 8].

3 Preliminaries

We will use 1^k to denote the security parameter. We refer the reader to [17] for definitions of various cryptographic notions, such as zero knowledge.

Commitment schemes. Recall that a commitment scheme Com is a 2-party protocol between a sender $\mathcal{C}$ and a receiver $\mathcal{R}$. In this paper, we always refer to computationally hiding commitment schemes. The binding property however, may be either statistical or computational. A commitment scheme has a commit phase and an open phase; we only consider commitment schemes where the open phase consists of a single message from the sender to the receiver. We know that there is a black-box construction of a 2-round statistically binding commitment scheme from any one-way function [35, 26].

knowledge proof that the value in the commitment is e; the verifier is effectively using a trapdoor commitment.

Trapdoor commitment schemes. Let $(\mathcal{C}, \mathcal{R})$ be a (computationally hiding) commitment scheme. We say that $(\mathcal{C}, \mathcal{R})$ is a trapdoor commitment scheme if there exists an expected polynomial-time probabilistic oracle machine $\mathcal{S} = (\mathcal{S}_1, \mathcal{S}_2)$ such that for any PPT $\mathcal{R}^*$ and all $v \in \{0,1\}^n$, the output (τ, w) of the following experiments are computationally indistinguishable:

- an honest sender $\mathcal{C}$ interacts with $\mathcal{R}^*$ to commit to v, and then opens the commitment: τ is the view of $\mathcal{R}^*$ in the commit phase, and w is the message $\mathcal{C}$ sends in the open phase.
- the simulator $\mathcal{S}$ generates a simulated view τ for the commit phase, and then opens the commitment to v in the open phase: formally, $\mathcal{S}_1^{\mathcal{R}^*}(1^n, 1^k) \rightarrow (\tau, \text{STATE}), \mathcal{S}_2(\text{STATE}, v) \rightarrow w$.

Extractable commitment schemes. Let $(\mathcal{C}, \mathcal{R})$ be a statistically binding commitment scheme. We say that $(\mathcal{C}, \mathcal{R})$ is an extractable commitment scheme if there exists an expected polynomial-time probabilistic oracle machine (the extractor) E that given oracle access to any PPT cheating sender $\mathcal{C}^*$ outputs a pair (τ, σ^*) such that:

- (simulation) τ is identically distributed to the view of $\mathcal{C}^*$ at the end of interacting with an honest receiver $\mathcal{R}$ in commit phase.
- (extraction) the probability that τ is accepting and $\sigma^* = \perp$ is negligible.
- (binding) if $\sigma^* \neq \perp$, then it is statistically impossible to open τ to any value other than σ^*.

We will also consider extractable commitment schemes that are computationally binding; the definition is as above, except if $\sigma^* \neq \perp$, we only require that it is computationally infeasible to open τ to any value other than σ^*.

4 Extractable commitment schemes.

The basic construction. The following protocol used in the works of [14, 38, 40] (also [30]) yields an extractable commitment scheme, starting from any commitment scheme Com:

PROTOCOL ExtCom.
- Common input: security parameter 1^k.
- Sender's input: a string $\sigma \in \{0,1\}^m$.

COMMIT PHASE.
- The sender commits (using Com) to k pairs of strings $(v_1^0, v_1^1), \ldots, (v_k^0, v_k^1)$ where $(v_i^0, v_i^1) = (\eta_i, \sigma \oplus \eta_i)$ and $\eta_1, \ldots, \eta_k$ are random strings in $\{0,1\}^m$.

- Upon receiving a challenge $e = (e_1, \ldots, e_k)$ from the receiver, the sender opens the commitments to $v_1^{e_1}, \ldots, v_k^{e_k}$.
- The receiver checks that the openings are valid.

OPEN PHASE.

- The sender sends σ and opens the commitments to all k pairs of strings.
- The receiver checks that all the openings are valid, and also that $\sigma = v_1^0 \oplus v_1^1 = \cdots = v_k^0 \oplus v_k^1$.

We sketch the proof (implicit in [14, 38, 40]) that ExtCom is an extractable commitment scheme.

Computationally hiding. The proof proceeds by a hybrid argument. Fix a cheating receiver, σ, σ' and suppose we want to show that $\mathsf{ExtCom}(\sigma)$ and $\mathsf{ExtCom}(\sigma')$ are computationally indistinguishable. In the i'th hybrid distribution, the first i pairs of strings are random shares of σ and the last $k - i$ pairs of strings are random shares of σ'. Suppose we have a distinguisher for the i'th and $i+1$'th hybrids. If the distribution of the bit e_i is noticeably biased, then we can break the hiding property of the underlying commitment Com right away. Otherwise, we can guess e_i with probability roughly $1/2$ and obtain a distinguisher for $\mathsf{Com}(\sigma \oplus \eta_i)$ and $\mathsf{Com}(\sigma' \oplus \eta_i)$.

Extractable. We start with the easier case where Com is statistically binding, upon which ExtCom is also statistically binding. Fix a cheating sender $\mathcal{C}^*$. We construct the extractor E as follows:

1. First, simulate an execution of $\mathcal{C}^*$ by internally emulating an honest receiver $\mathcal{R}$ to obtain a transcript τ of the commit phase. If τ is rejecting, then output $(\tau, \perp)$ and halt.
2. If τ is accepting with some challenge e, then keep rewinding $\mathcal{C}^*$ with random challenges until we receive another accepting response from $\mathcal{C}^*$ with some challenge e'. If $e = e'$, then output $(\tau, \perp)$ and halt. Otherwise, extract a value σ^* from the $\mathcal{C}^*$'s responses to distinct challenges e, e' (by combining the appropriate shares), and output (τ, σ^*).

Now, suppose the probability over e that we obtain an accepting transcript τ is p. Then, the expected number of queries E makes to $\mathcal{C}^*$ is $(1-p)+p\cdot\frac{1}{p} \leq 2$. Also, the failure probability, i.e., the probability that τ is accepting and $e = e'$ is at most $p \cdot \frac{2^{-k}}{p} = 2^{-k}$.

We can still use the same extractor E in the case where Com is computationally binding. Now, if there is a cheating sender that can open the commitment in τ to a different value from σ^*, then we can combine this with the opening to σ^* obtained by E to derive an efficient adversary that breaks the binding property of Com.

The parallel variant. For our compiler, we will actually need an extractable commitment scheme to a string σ for which we can open any subset of the bits in σ without compromising the security (i.e. hiding) of the remaining bits. We may obtain such a scheme $\mathsf{PExtCom}$ by running ExtCom to commit to each bit of σ in parallel. That $\mathsf{PExtCom}$ is hiding follows from the more general fact that the hiding property of commitment schemes is preserved under parallel composition. To show that $\mathsf{PExtCom}$ is extractable, we may use the same extractor E as before, except for a modification in step 2. Note that the receiver's challenge in $\mathsf{PExtCom}$ is a k-tuple of m-bit strings, which again we denote by $e \in (\{0,1\}^m)^k$. Once we obtain responses to two challenges e, e' in Step 2, we proceed as follows: if e' agrees with e in any of the k components, we output $(\tau, \perp)$ and halt. Otherwise, we will be able to extract each of the m bits in the m parallel executions of ExtCom. As before, the expected number of queries E makes to $\mathcal{C}^*$ is at most 2. The failure probability in this case is now at most $m \cdot 2^{-k}$.

5 Zero-knowledge arguments for NP

Look-ahead zero-knowledge proof system. We use as our look-ahead zero-knowledge proof system the parallel repetition variant of Blum's Hamiltonicity protocol [4], which we already know to be special honest-verifier zero-knowledge.

HAMILTONICITY PROTOCOL Π_{HAM}.

- Common input: a graph G on n vertices.
- Prover's input, a cycle h in G

1. The prover picks random permutations π_i over $[n]$ and commits to $v_i = (\pi_i, A_i)$, where A_i denotes the adjacency matrix of the graph $\pi_i(G)$.
2. Upon receiving the verifier's challenge $e = (e_1, \ldots, e_k)$, the prover responds as follows for each $i = 1, \ldots, k$: if $e_i = 0$, it opens the commitment to (π_i, A_i); if $e_i = 1$, it opens the commitment to entries in A_i corresponding to the edges of the cycle $\pi_i(h)$.
3. The verifier checks that the openings are valid and in addition, that $A_i = \pi_i(G_i)$ if $e_i = 0$ and that the open entries correspond to edges of a Hamiltonian cycle if $e_i = 1$.

We just need to verify that the easy challenge is efficiently recoverable:

Special soundness. Given a non-Hamiltonian graph G and the values $v_i = (\pi_i, A_i)$, we can compute $\tilde{e} = (\tilde{e}_1, \ldots, \tilde{e}_k)$ as follows: $\tilde{e}_i$ equals 0 if

$\pi_i(G) = A_i$ and 1 otherwise. It is easy to see that if an easy challenge (that allows the prover to cheat) exists, then it must equal $\tilde{e}$.[5]

The zero-knowledge argument system. The zero-knowledge protocol is as follows:

1. The verifier picks a random $e \in \{0,1\}^k$ and commits to e using Com, a statistically-binding commitment scheme.
2. The prover commits to $v_1, \ldots, v_k$ as in Π_{HAM} using $\mathsf{PExtCom}$.
3. The verifier opens the commitment to e.
4. The prover aborts if the opening to e is not valid. Otherwise, it responds to the challenge e according to Π_{HAM}.
5. The verifier runs the final verification step as in Π_{HAM}.

The analysis. Completeness is straight-forward.

Computational soundness. Suppose there exists a cheating prover P^* (WLOG deterministic) that convinces the verifier to accept a non-Hamiltonian graph G with probability $\epsilon = 1/\operatorname{poly}(k)$. Intuitively this means that P^* on input $\mathsf{Com}(e)$ predicts e with probability roughly $\epsilon \gg 2^{-k}$, which must contradict the hiding property of Com. More formally, fix the graph G, and we know that with probability $\epsilon/2$ over e, P^* succeeds with probability $\epsilon/2$. Let Γ denote the set of such challenges e, so $|\Gamma| \geq \frac{\epsilon}{2} \cdot 2^k$, and consider the procedure A that on input a commitment $\mathsf{Com}(e)$:

1. sends $\mathsf{Com}(e)$ to P^*;
2. uses the extractor for $\mathsf{PExtCom}$ with P^* as the cheating sender to obtain commitments to $v_1, \ldots, v_k$ along with the values $v_1, \ldots, v_k$.
3. computes a candidate easy challenge $\tilde{e}$ from $v_1, \ldots, v_k$ and outputs $\tilde{e}$.

It is easy to see that for all $e \in \Gamma$, $\Pr[A(\mathsf{Com}(e)) \to e] \geq \frac{\epsilon}{2} - \operatorname{neg}(k) \geq \frac{\epsilon}{4}$. By using a non-uniform reduction, we may WLOG assume that $0^k \in \Gamma$. Now, the sets of strings Γ' in the output of $A(\mathsf{Com}(0^k))$ that occurs with probability at least $\frac{\epsilon}{8}$ is at most $\frac{8}{\epsilon}$. Since $|\Gamma'| < |\Gamma|$, there must exist a string, say 1^k, that lies in Γ but outside Γ'. Now,

$$\begin{aligned} 1^k \notin \Gamma' \quad &\Rightarrow \quad \Pr[A(\mathsf{Com}(0^k)) \to 1^k] \leq \tfrac{\epsilon}{8} \\ 1^k \in \Gamma \quad &\Rightarrow \quad \Pr[A(\mathsf{Com}(1^k)) \to 1^k] \geq \tfrac{\epsilon}{4} \end{aligned}$$

This yields a distinguisher for $\mathsf{Com}(0^k)$ and $\mathsf{Com}(1^k)$, which contradicts the hiding property of Com.

[5] Note that determining whether an easy challenge exists is NP-hard, since we must determine whether A_i contains a cycle.

Zero-knowledge. The zero-knowledge simulator is virtually identically to that in the Goldreich-Kahan protocol [19]. Roughly speaking, upon receiving the verifier's commitment to e, the prover sends the cheating verifier V^* dummy commitments. If the verifier aborts, we are basically done. Otherwise, we learn the challenge e and then we could use the honest-verifier zero-knowledge simulator to complete the simulation. As in [19], we will need to estimate the probability that V^* aborts on dummy commitments.

Argument of knowledge. We may obtain a zero-knowledge argument of knowledge for NP by instantiating the Feige-Shamir protocol [16] with the trapdoor commitment scheme, which we present in the next section.

6 Trapdoor commitments

We construct a "look-ahead trapdoor commitment". This is a statistically binding commitment scheme wherein the commit phase comprises a 3-round challenge-response protocol. In addition, the scheme will be "look-ahead trapdoor" in the following sense: if we fix the receiver's challenge in the challenge-response phase, then we may generate a simulated transcript for the commit phase which we may later open to both a 0 and a 1. Moreover, the transcript together with either bit b is computationally indistinguishable from a legitimate commitment to b followed by an opening to b. We note similar constructions appear in [31, 30]. In addition, we stress that we cannot use the challenge-response protocol in [13] because we do not know how to efficient compute the easy challenge in that protocol.[6]

Look-ahead trapdoor bit commitment. To commit to a bit σ. Again, we fix some statistically binding commitment scheme Com.

COMMIT PHASE.

- Each v_i is a 2×2 0,1-matrix given by

$$\begin{pmatrix} v_i^{00} & v_i^{01} \\ v_i^{10} & v_i^{11} \end{pmatrix} = \begin{pmatrix} \eta_i & \sigma \oplus \eta_i \\ \eta_i & \sigma \oplus \eta_i \end{pmatrix}$$

[6] Roughly speaking, easy challenges in [13] are the first-round messages in Naor's commitment scheme [35] that allow a computationally unbounded sender to cheat, i.e. strings of the form $G(a) \oplus G(b) \in \{0,1\}^k$ ranging over all $a, b \in \{0,1\}^k$, and where $G : \{0,1\}^{k/3} \to \{0,1\}^k$ is a pseudorandom generator.

where η_i is a random bit. The sender commits to $v_1, \ldots, v_k$ using Com (bit by bit in parallel). In addition, the sender prepares $(a_1^0, a_1^1), \ldots, (a_k^0, a_k^1)$ where a_i^β is the opening of the commitment to $v_i^{0\beta}, v_i^{1\beta}$ (i.e., either the left or right column of v_i).

- Upon receiving a challenge $e = (e_1, \ldots, e_k)$ from the receiver, the sender responds with $a_1^{e_1}, \ldots, a_k^{e_k}$.
- The receiver checks that the openings are valid and that $v_i^{0e_i} = v_i^{1e_i}$ for $i = 1, 2, \ldots, k$ (i.e., every column that is open contains two equal bits).

OPEN PHASE.

- The sender sends σ. In addition, it chooses a random $\gamma \in \{0, 1\}$, sends γ, opens the commitments to $v_i^{\gamma 0}, v_i^{\gamma 1}$ for $i = 1, 2, \ldots, k$ (i.e., either the top rows or the bottom rows of all the matrices).
- The receiver checks that all the openings are valid, and also that $\sigma = v_1^{\gamma 0} \oplus v_1^{\gamma 1} = \cdots = v_k^{\gamma 0} \oplus v_k^{\gamma 1}$.

Analysis. It is straight-forward to show that the commitment scheme is computationally hiding.

Special soundness. Suppose we have a cheating sender that generates a transcript for the commit phase that can be successfully open to both a 0 and a 1. It must be the case that every matrix v_i contains at least one column with two unequal bits; call that column $\tilde{e}_i$. Then, the cheating sender will get caught in the commit phase unless $e = \tilde{e} = (\tilde{e}_1, \ldots, \tilde{e}_k)$. Moreover, given $v_1, \ldots, v_k$ it is easy to compute $\tilde{e}$.

Look-ahead trapdoor. We construct a simulator as follows:

- Given the challenge e, pick a random $\beta \in \{0, 1\}$, and prepare the matrices v_i as follows:

$$\begin{pmatrix} v_i^{00} & v_i^{01} \\ v_i^{10} & v_i^{11} \end{pmatrix} = \begin{pmatrix} \eta_i & \beta \oplus \eta_i \\ \eta_i & \bar{\beta} \oplus \eta_i \end{pmatrix} \text{ if } e_i = 0; \text{ and}$$

$$\begin{pmatrix} v_i^{00} & v_i^{01} \\ v_i^{10} & v_i^{11} \end{pmatrix} = \begin{pmatrix} \beta \oplus \eta_i & \eta_i \\ \bar{\beta} \oplus \eta_i & \eta_i \end{pmatrix} \text{ otherwise;}$$

  where η_i is a random bit. When the receiver sends e, open the commitments to $v_i^{0e_i}$ and $v_i^{1e_i}$ like the honest sender.
- To open to σ, send $\gamma = \beta \oplus \sigma$, and open the commitments to $v_i^{\gamma 0}, v_i^{\gamma 1}$ for $i = 1, 2, \ldots, k$.

The trapdoor bit commitment scheme. The construction and the analysis is completely analogous to the zero-knowledge protocol. The verifier begins by committing to a random challenge $e \in \{0,1\}^k$ using a statistically-binding commitment Com, and then we proceed according to the look-ahead scheme except the prover commits using ExtCom. Completeness is again straight-forward. Establishing computational binding is analogous to establishing computational soundness for the zero-knowledge protocol; we transform any cheating sender a distinguisher for Com by arguing that it must on input $\mathsf{Com}(e)$ predict e with noticeable probability. Trapdoor simulation is again based on the Goldreich-Kahan simulation strategy [19].

Extension to multiple bits. We claim that by running the trapdoor bit commitment scheme in parallel, we obtain a trapdoor commitment scheme for multiple bits, with the additional property that we can open the commitment to any subset of the bits without compromising the security of the remaining bits. We know that parallel repetition preserves the hiding and binding properties of commitment schemes. To see that the parallel version is still trapdoor, observe that we may still use the Goldreich-Kahan simulation strategy and that the look-ahead simulation property is preserved under parallel repetition.

7 Parallel coin-tossing

We present a constant-round parallel coin-tossing protocol in this section. Using the composition theorem in [6] and the results of [3, 8], it is sufficient to implement the ideal string commitment functionality $\mathcal{F}_{\mathsf{Com}}$ (shown in Fig 2) with stand-alone security a la [21, 5, 18] in constant rounds. Moreover, by the results of [7], it suffices to construct a constant-round extractable trapdoor commitment scheme.

Extractable trapdoor commitment scheme. We provide a general construction of an extractable trapdoor commitment scheme $\mathsf{ExtTDCom}$ starting from any trapdoor commitment scheme TDCom: simply instantiate the protocol ExtCom with the trapdoor commitment scheme TDCom. Specifically, the sender in $\mathsf{ExtTDCom}$ on input a string $\sigma \in \{0,1\}^m$, commits to k pairs of strings $(v_1^0, v_1^1), \ldots, (v_k^0, v_k^1)$ (with $v_1^0 \oplus v_1^1 = \cdots = v_k^0 \oplus v_k^1 = \sigma$) using TDCom, by treating the k pairs of strings as a single string of length $2km$. The trapdoor property is straight-forward: if we could equivocate on the commitment to the string $(v_1^0, v_1^1), \ldots, (v_k^0, v_k^1)$, then we could easily equivocate on the commitment to σ. The extractable property is already established in Section 4.

Functionality $\mathcal{F}_{\mathsf{Com}}$

1. Upon receiving input $(\texttt{Commit}, sid, P_j, x)$ from P_i where $x \in \{0,1\}^m$, internally record the tuple (P_i, P_j, x) and send the message (sid, P_i, P_j) to the adversary; When receiving $(\texttt{ok})$ from the adversary, output $(\texttt{Receipt}, sid, P_i)$ to P_j. Ignore all subsequent $(\texttt{Commit}, ...)$ inputs.
2. Upon receiving a value $(\texttt{Open}, sid)$ from P_i, where a tuple (P_i, P_j, x) is recorded, send (x) to the adversary; When receiving $(\texttt{ok})$ from the adversary, output $(\texttt{Open}, sid, x)$ to P_j.

Fig. 2. Ideal String Commitment Functionality

The coin-tossing protocol. For self-containment, we present the coin-tossing protocol based directly on $\mathsf{ExtTDCom}$.

1. Party 1 chooses a random $s_1 \in \{0,1\}^m$ and commits to s_1 using $\mathsf{ExtTDCom}$. Party 2 aborts with output $\perp$ if the commitment protocol fails.
2. Party 2 chooses $s_2 \in \{0,1\}^m$ and sends s_2 to Party 1.
3. If Party 1 receives an invalid message from Party 2, then Party 1 aborts. Otherwise, Party 1 opens the commitment to s_1. Party 2 aborts with output $\perp$ if the opening is invalid.
4. Output: both parties output $s_1 \oplus s_2$.

The high level proof strategy is as follows.

- If Party 1 is corrupted, we will use the extractor for $\mathsf{ExtTDCom}$ to extract s_1 and then set $s_2 = s_1 \oplus s$ (where s is the string chosen by the trusted party).
- If Party 2 is corrupted, we will use the trapdoor commitment property so that upon receiving s_2 from Party 2, the simulator can open the commitment to $s_1 = s \oplus s_2$.

8 Non-malleable commitments

We begin by describing a commitment scheme satisfying some strong notions of extractability and hiding, based on an encoding scheme from [9].

An intermediate construction. To commit to a string v with parameter 1^ℓ (ℓ is the length of the identities):

COMMIT PHASE.

1. The receiver commits to a random subset $S \subset [10k]$ of size k using Com.

2. The sender picks random $\alpha_1, \ldots, \alpha_k \in \mathrm{GF}(2^n)$ and set $s_j = p(j), j \in [10k]$ where $p(x) = v + \alpha_1 x + \ldots + \alpha_k x^k$. (Note that $(s_1, \ldots, s_{10k})$ encodes v under the Reed-Solomon code.) The sender then commits to $(s_1, \ldots, s_{10k})$ a total of 2ℓ times using $\mathsf{PExtCom}$ sequentially.
3. The receiver opens the commitment to S.
4. The sender opens the 2ℓ commitments to the value s_j for all $j \in S$.
5. The receiver checks that for each $j \in S$, the 2ℓ commitments to s_j open to the same value.

OPEN PHASE.

1. The sender sends v and opens the commitments to $(s_1, \ldots, s_{10k})$ in the first execution of $\mathsf{PExtCom}$.
2. The receiver computes the codeword $w = (w_1, \ldots, w_{10k})$ that agrees with $(s_1, \ldots, s_{10k})$ in at least $9k$ positions, and checks that $(s_1, \ldots, s_{10k})$ is a codeword corresponding to v and that for all $j \in S$, $s_j = w_j$.

We sketch the properties satisfied by this commitment scheme, and defer the analysis to the full version of this paper.

Extractability. There exists expected polynomial-time probabilistic oracle machines $E_1, E_2, \ldots, E_{2\ell}$ such that for all $i = 1, 2, \ldots, 2\ell$, the machine E_i given oracle access to any PPT cheating sender $\mathcal{C}^*$ outputs a pair (τ, σ^*) such that

- (simulation) τ is identically distributed to the view of $\mathcal{C}^*$ at the end of interacting with an honest receiver $\mathcal{R}$ in commit phase.
- (strong extraction) the pair (τ, σ^*) is computationally indistinguishable from the view of $\mathcal{C}^*$ at the end of interacting with an honest receiver $\mathcal{R}$ in commit phase, together with the committed value implicitly specified by the view.

We will also require that the machine E_i extracts from the i'th execution of $\mathsf{PExtCom}$, for $i = 1, \ldots, 2\ell$.

Hiding. We require that the commitment scheme is (computationally) hiding even against a PPT cheating receiver $\mathcal{R}^*$ that may request for an arbitrary number of additional commitments to $(s_1, \ldots, s_{10k})$ using $\mathsf{PExtCom}$, along with the openings to s_j for each $j \in S$ in these additional commitments.

We stress that the notion of extractability above is stronger than that in Section 3. In particular, it guarantees that if there is no valid opening for the commit phase transcript τ, then the extractor must output $\sigma^* = \perp$.

Achieving non-malleability. To obtain a non-malleable commitment scheme from the previous construction, we just need to schedule the messages in the 2ℓ copies of $\mathsf{PExtCom}$ according to the message scheduling in [14]. It follows from the analysis in [32] that the resulting $O(n)$-round commitment scheme is one-many non-malleable. By further applying the results in [14, 32], we obtain a $O(\log n)$-round non-malleable commitment and a $O(n)$-round concurrent non-malleable commitment.

References

1. BARAK, B. How to go beyond the black-box simulation barrier. In *FOCS* (2001), pp. 106–115.
2. BEAVER, D. Adaptive zero knowledge and computational equivocation. In *STOC* (1996), pp. 629–638.
3. BLUM, M. Coin flipping by telephone. In *CRYPTO* (1981), pp. 11–15.
4. BLUM, M. How to prove a theorem so no one else can claim it. In *Proc. ICM* (1986).
5. CANETTI, R. Security and composition of multiparty cryptographic protocols. *J. Cryptology 13*, 1 (2000), 143–202.
6. CANETTI, R. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS* (2001), pp. 136–145.
7. CANETTI, R., AND FISCHLIN, M. Universally composable commitments. In *CRYPTO* (2001), pp. 19–40.
8. CANETTI, R., AND RABIN, T. Universal composition with joint state. In *CRYPTO* (2003), pp. 265–281.
9. CHOI, S. G., DACHMAN-SOLED, D., MALKIN, T., AND WEE, H. Black-box construction of a non-malleable encryption scheme from any semantically secure one. In *TCC* (2008), pp. 427–444.
10. CHOI, S. G., DACHMAN-SOLED, D., MALKIN, T., AND WEE, H. Simple, black-box constructions of adaptively secure protocols. In *TCC* (2009). to appear.
11. CRAMER, R., DAMGÅRD, I., AND SCHOENMAKERS, B. Proofs of partial knowledge and simplified design of witness hiding protocols. In *CRYPTO* (1994), pp. 174–187.
12. DAMGÅRD, I., AND ISHAI, Y. Constant-round multiparty computation using a black-box pseudorandom generator. In *CRYPTO* (2005), pp. 378–394.
13. DI CRESCENZO, G., AND OSTROVSKY, R. On concurrent zero-knowledge with pre-processing. In *CRYPTO* (1999), pp. 485–502.
14. DOLEV, D., DWORK, C., AND NAOR, M. Nonmalleable cryptography. *SIAM J. Comput. 30*, 2 (2000), 391–437.
15. EVEN, S., GOLDREICH, O., AND LEMPEL, A. A randomized protocol for signing contracts. In *CRYPTO* (1982), pp. 205–210.
16. FEIGE, U., AND SHAMIR, A. Zero knowledge proofs of knowledge in two rounds. In *CRYPTO* (1989), pp. 526–544.
17. GOLDREICH, O. *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001.
18. GOLDREICH, O. *Foundations of Cryptography: Volume II, Basic Applications*. Cambridge University Press, 2004.
19. GOLDREICH, O., AND KAHAN, A. How to construct constant-round zero-knowledge proof systems for NP. *J. Cryptology 9*, 3 (1996), 167–190.

20. GOLDREICH, O., AND KRAWCZYK, H. On the composition of zero-knowledge proof systems. *SIAM J. Comput. 25*, 1 (1996), 169–192.
21. GOLDREICH, O., MICALI, S., AND WIGDERSON, A. How to play any mental game or a completeness theorem for protocols with honest majority. In *STOC* (1987), pp. 218–229.
22. GOLDREICH, O., MICALI, S., AND WIGDERSON, A. Proofs that yield nothing but their validity for all languages in NP have zero-knowledge proof systems. *J. ACM 38*, 3 (1991), 691–729. Prelim. version in FOCS '86.
23. GOYAL, V., MORIARTY, R., OSTROVSKY, R., AND SAHAI, A. Concurrent statistical zero-knowledge arguments for NP from one way functions. In *ASIACRYPT* (2007), pp. 444–459.
24. HAITNER, I. Semi-honest to malicious oblivious transfer - the black-box way. In *TCC* (2008), pp. 412–426.
25. HAITNER, I., HOCH, J., REINGOLD, O., AND SEGEV, G. Finding collisions in interactive protocols – a tight lower bound on the round complexity of statistically-hiding commitments. In *FOCS* (2007), pp. 669–679.
26. HÅSTAD, J., IMPAGLIAZZO, R., LEVIN, L. A., AND LUBY, M. A pseudorandom generator from any one-way function. *SIAM J. Comput. 28*, 4 (1999), 1364–1396.
27. IMPAGLIAZZO, R., AND RUDICH, S. Limits on the provable consequences of one-way permutations. In *STOC* (1989), pp. 44–61.
28. ISHAI, Y., KUSHILEVITZ, E., LINDELL, Y., AND PETRANK, E. Black-box constructions for secure computation. In *STOC* (2006), pp. 99–108.
29. ISHAI, Y., PRABHAKARAN, M., AND SAHAI, A. Founding cryptography on oblivious transfer - efficiently. In *CRYPTO* (2008), pp. 572–591.
30. KILIAN, J. Founding cryptography on oblivious transfer. In *STOC* (1988), pp. 20–31.
31. KILIAN, J. On the complexity of bounded-interaction and noninteractive zero-knowledge proofs. In *FOCS* (1994), pp. 466–477.
32. LIN, H., PASS, R., AND VENKITASUBRAMANIAM, M. Concurrent non-malleable commitments from any one-way function. In *TCC* (2008), pp. 571–588.
33. LINDELL, Y. Parallel coin-tossing and constant-round secure two-party computation. *J. Cryptology 16*, 3 (2003), 143–184.
34. LINDELL, Y., AND PINKAS, B. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *EUROCRYPT* (2007), pp. 52–78.
35. NAOR, M. Bit commitment using pseudorandomness. *J. Cryptology 4*, 2 (1991), 151–158.
36. PASS, R., TSENG, W., AND VENKITASUBRAMANIAM, M. Unconditional characterizations of concurrent zero knowledge. Manuscript, 2008.
37. PEIKERT, C., AND WATERS, B. Lossy trapdoor functions and their applications. In *STOC* (2008). to appear. Also, Cryptology ePrint Archive, Report 2007/279.
38. PRABHAKARAN, M., ROSEN, A., AND SAHAI, A. Concurrent zero knowledge with logarithmic round-complexity. In *FOCS* (2002), pp. 366–375.
39. REINGOLD, O., TREVISAN, L., AND VADHAN, S. Notions of reducibility between cryptographic primitives. In *TCC* (2004), pp. 1–20.
40. ROSEN, A. A note on constant-round zero-knowledge proofs for NP. In *TCC* (2004), pp. 191–202.
41. ROSEN, A. *The Round-Complexity of Black-Box Concurrent Zero-Knowledge*. Ph.D., Weizmann Institute of Science, May 2004.
42. SIMON, D. R. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *EUROCRYPT* (1998), pp. 334–345.
43. YAO, A. C.-C. How to generate and exchange secrets (extended abstract). In *FOCS* (1986), pp. 162–167.