SCIS 2013 The 30th Symposium on
Cryptography and Information Security
kyoto, Japan, Jan. 22 - 25, 2013
The Institute of Electronics,
Information and Communication Engineers

On the Round-Complexity Lower Bound of CCA-Secure Commitments

Susumu Kiyoshima * Yoshifumi Manabe † Tatsuaki Okamoto ‡

Abstract— We study commitment schemes that are secure against chosen commitment attack (CCA-secure commitments). Using an idea behind the impossibility of $o(\log n/\log\log n)$-round black-box concurrent zero-knowledge proofs, we show that if we use black-box reductions to prove CCA security, we cannot construct any $o(\log n/\log\log n)$-round CCA-secure commitment based on falsifiable polynomial-time hardness assumptions.

Keywords: CCA-secure commitment, lower bound, falsifiable assumption, black-box reduction

1 Introduction

1.1 Background

Commitment schemes are fundamental two-party protocols in cryptography, and used as building blocks of various cryptographic protocols.

The basic security requirements of commitment schemes are the *hiding* property and the *binding* property. The hiding property guarantees that the committer can commit to a value while keeping it secret from the receiver. The binding property guarantees that after committing to a value, the committer cannot decommit the commitment to two distinct values.

Due to the importance of commitment schemes, various works proposed commitment schemes with additional security guarantees. An example of such a commitment scheme is a *(concurrent) non-malleable commitment scheme* [DDN00, PR05], which plays important roles in constructing round-efficient general secure multi-party computation (MPC) protocols (i.e., protocols for any functionality) [KOS03, Pas04, LPV09, LP11, Goy11].

Recently, Canetti et al. [CLP10] proposed a commitment scheme that guarantees very strong security: *security against chosen commitment attack* (or *CCA security*). Then, they showed that CCA-secure commitment schemes can be used to construct a universally composable protocol for general secure MPC in the plain model (i.e., without any trusted setup)[1]. Thus, with a CCA-secure commitment scheme, we can construct a general secure MPC protocol that remains secure even when many other protocols are concurrently executed with it. (In contrast, with a non-malleable commitment scheme, we have general secure MPC protocols only in the stand-alone setting.) CCA-secure commitment schemes are also of independent interest. Like CCA security for encryption schemes, CCA security for commitment schemes is a natural and desirable security notion. In particular, CCA security implies concurrent non-malleability.

Roughly speaking, CCA-secure commitment schemes are commitment schemes such that the hiding property holds even against adversary $\mathcal{A}$ that has access to the *committed-value oracle*, where the committed-value oracle $\mathcal{O}$ interacts with $\mathcal{A}$ as an honest receiver in many concurrent sessions of the commit phase, and at the end of each session, $\mathcal{O}$ computes the committed value of this session by brute force and returns it to $\mathcal{A}$.

Canetti et al. [CLP10] showed that for any constant $\epsilon > 0$, there exists an $O(n^\epsilon)$-round CCA secure commitment scheme based on the minimum assumption of the existence of one-way functions. Recently, Lin and Pass [LP12] constructed another CCA secure commitment scheme with the same asymptotic round complexity from the same assumption[2]. Both schemes are based on the concurrent non-malleable commitment scheme of [LPV08].

A natural question is whether we can construct round-efficient CCA-secure commitment schemes. In particular, since there exist constant-round concurrent non-malleable commitment schemes based on the existence of one-way functions [Goy11, LP11], an important question is the following.

> *Is it possible to construct constant-round CCA-secure commitment schemes based on the existence of one-way functions (or other standard assumptions)?*

* Kyoto University (kiyoshima@ai.soc.i.kyoto-u.ac.jp)
† NTT and Kyoto University (manabe.yoshifumi@lab.ntt.co.jp)
‡ NTT and Kyoto University (okamoto.tatsuaki@lab.ntt.co.jp)

[1] Recall that in *universally composable security* [Can01], we cannot construct such a protocol in the plain model [CF01, CKL03]. Thus, Canetti et al. constructed their protocol in a relaxed framework called *UC security with super-polynomial-time helpers* [CLP10].

[2] The advantage of the scheme of [LP12] is that it uses the underlying one-way functions only in a black-box way, i.e., only through their input/output interfaces.

1.2 Our Result

In this paper, we show that it is *impossible* to construct constant-round CCA-secure commitment schemes based on standard assumptions *if we use standard techniques to prove the security.* More precisely, we prove the following theorem.

Theorem 1. *Let $\langle C, R\rangle$ be a $o(\log n/\log\log n)$-round commitment scheme. If there exists a black-box reduction showing CCA security of $\langle C, R\rangle$ based on a falsifiable polynomial-time hardness assumption, then this assumption is false.*

Below, we explain black-box reductions and falsifiable polynomial-time hardness assumptions.

Black-Box Reductions

Black-box reduction $\mathcal{R}$ is a probabilistic polynomial-time (PPT) interactive Turing machine (ITM). We say that $\mathcal{R}$ *shows the security of a protocol based on an assumption* if for any adversary $\mathcal{A}$ that breaks the security of the protocol, $\mathcal{R}^{\mathcal{A}}$ breaks the assumption.

Almost all proofs in cryptography use black-box reductions. An exception is Barak's non-black-box technique [Bar01] that uses the code of the adversary to construct the simulator of his zero-knowledge argument. (Very recently, Bitansky and Paneth [BP12] showed a new non-black-box technique, which is based on the impossibility of program obfuscation.) However, the use of his technique is currently limited to a few areas.

Remark 1. We note that our result holds *even if constructions are non-black-box.* That is, even if we use the code of the underlying primitive (typically for NP reductions in the general zero-knowledge proof), we cannot construct constant-round CCA-secure commitment schemes (when we use black-box reductions and falsifiable polynomial-time hardness assumptions).

Falsifiable Polynomial-Time Hardness Assumptions

A *falsifiable polynomial-time hardness assumption* [Nao03, GW11] is defined by using a threshold c and an interactive game between a PPT ITM (or *challenger*) Ch and a PPT adversary $\mathcal{A}$. We say that $\mathcal{A}$ *breaks the assumption* if the probability that Ch outputs 1 after interacting with $\mathcal{A}$ is non-negligibly higher than the threshold c. The assumption is true if and only if no PPT $\mathcal{A}$ breaks the assumption.

All standard assumptions are falsifiable polynomial-time hardness assumptions. For example, the existence of one-way functions and the hardness of factoring are ones with threshold $c = 0$, and the decisional Diffie-Hellman (DDH) assumption is one with threshold $c = 1/2$.

1.3 Proof Overview

We briefly explain our proof of Theorem 1.

Assume that there exists a black-box reduction $\mathcal{R}$ showing CCA security of $\langle C, R\rangle$ based on a falsifiable polynomial-time hardness assumption. Then, for any (possibly super-polynomial-time[3]) adversary $\mathcal{A}$, if $\mathcal{A}$ breaks CCA security of $\langle C, R\rangle$, then $\mathcal{R}^{\mathcal{A}}$ breaks the assumption.

To show that the assumption is false, we construct a PPT adversary that breaks the assumption. In particular, we show that $\mathcal{R}$ can break the assumption without accessing any successful adversary $\mathcal{A}$ of CCA security.

Toward this end, we construct a super-polynomial-time adversary $\mathcal{A}^1$ such that (1) $\mathcal{A}^1$ breaks CCA security of $\langle C, R\rangle$ but (2) $\mathcal{R}$ cannot get any "useful information" from oracle access to $\mathcal{A}^1$. Recall that adversaries against CCA security are given access to the committed-value oracle $\mathcal{O}$ and try to break the hiding property of $\langle C, R\rangle$. Then, $\mathcal{A}^1$ does the following.

1. First, $\mathcal{A}^1$ interacts with $\mathcal{O}$ as follows. $\mathcal{A}^1$ concurrently commits to many random values and receives answers from $\mathcal{O}$. If one of these answers is not equal to the value that $\mathcal{A}^1$ committed to, $\mathcal{A}^1$ outputs abort and halts.

2. Then, $\mathcal{A}^1$ breaks the hiding property of $\langle C, R\rangle$ by brute force.

Since $\mathcal{R}$ has only black-box access to $\mathcal{A}^1$, the only way for $\mathcal{R}$ to get useful information from $\mathcal{A}^1$ is to interact with $\mathcal{A}^1$ and receive the Step 2 messages. (Since the Step 1 messages are commitments to random values, they are useless to $\mathcal{R}$, i.e., they can be simulated in polynomial time.) In other words, to get useful information, $\mathcal{R}$ needs to extract the committed values and return them to $\mathcal{A}^1$ in Step 1 (otherwise, $\mathcal{A}^1$ halts before Step 2). Since the running time of $\mathcal{R}$ is polynomial, the hiding property of $\langle C, R\rangle$ implies that the only way for $\mathcal{R}$ to extract these committed values is to rewind $\mathcal{A}^1$ in each session of the commitment. (Since the running time of the challenger Ch is also polynomial, the messages that $\mathcal{R}$ receives from Ch are useless for extracting the committed values.) However, since $\mathcal{A}^1$ commits to the values concurrently, there exists the problem of *recursive rewinding* (i.e., when $\mathcal{R}$ rewinds $\mathcal{A}^1$ in a session, it requires a rewinding of another session, which in turn requires a rewinding of another session, and so on), which occurs in the context of *concurrent zero-knowledge proofs* [DNS04] as well. Then, we use a result in the impossibility of $o(\log n/\log\log n)$-round black-box concurrent zero-knowledge proofs [CKPR02] to show that whenever the running time of $\mathcal{R}$ is polynomial, there exists a session in which $\mathcal{R}$ cannot rewind $\mathcal{A}^1$. Thus, $\mathcal{R}$ cannot extract the committed value of this session, and therefore cannot get useful information from $\mathcal{A}^1$.

Using $\mathcal{A}^1$, we show that the underlying assumption is false as follows. Since $\mathcal{R}$ is a black-box reduction showing CCA security of $\langle C, R\rangle$, and since $\mathcal{A}^1$ breaks CCA security of $\langle C, R\rangle$, $\mathcal{R}^{\mathcal{A}^1}$ breaks the assumption. Then, since $\mathcal{R}$ does not get any useful information from $\mathcal{A}^1$, we can show that $\mathcal{R}$ can break the assumption

[3] Since $\mathcal{R}$ has only oracle access to $\mathcal{A}$, the running time of $\mathcal{A}$ does not make any difference in the behavior of $\mathcal{R}$.

without accessing $\mathcal{A}$[1]. Then, since the running time of $\mathcal{R}$ is polynomial, we conclude that the assumption is false.

2 Preliminary

In this paper, we say that the round complexity of a protocol is $k(n)$ if each party sends $k(n)$ messages.

2.1 CCA-Secure Commitment

First, we recall the definition of CCA-secure commitments [CLP10, LP12]. In the following, we use *tag-based commitment schemes* to denote commitment schemes such that both the committer and the receiver receive a string called a *tag* as an additional input.

Roughly speaking, a tag-based commitment scheme $\langle C, R\rangle$ is *CCA-secure* if the hiding property of $\langle C, R\rangle$ holds even against adversary $\mathcal{A}$ that can interact with the *committed-value oracle* during the interaction with the committer. The committed-value oracle $\mathcal{O}$ interacts with $\mathcal{A}$ as an honest receiver in many concurrent sessions of the commit phase of $\langle C, R\rangle$ using tags chosen adaptively by $\mathcal{A}$. At the end of each session, if the commitment of this session is invalid or has multiple committed values, $\mathcal{O}$ returns $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{O}$ returns the unique committed value to $\mathcal{A}$.

More precisely, let us consider the following probabilistic experiment $\text{IND}_b(\langle C, R\rangle, \mathcal{A}, n, z)$ for each $b \in \{0,1\}$. On input 1^n and auxiliary input z, adversary $\mathcal{A}^{\mathcal{O}}$ adaptively chooses a pair of challenge values $v_0, v_1 \in \{0,1\}^n$ and an n-bit tag $id \in \{0,1\}^n$. Then, $\mathcal{A}^{\mathcal{O}}$ receives a commitment to v_b with tag id, and $\mathcal{A}$ outputs y. The output of the experiment is $\perp$ if during the experiment, $\mathcal{A}$ sends $\mathcal{O}$ any commitment using tag id. Otherwise, the output of the experiment is y. Let $\mathsf{IND}_b(\langle C, R\rangle, \mathcal{A}, n, z)$ denote the output of experiment $\text{IND}_b(\langle C, R\rangle, \mathcal{A}, n, z)$.

Then, CCA security of $\langle C, R\rangle$ is defined as follows.

Definition 1. Let $\langle C, R\rangle$ be a tag-based commitment scheme and $\mathcal{O}$ be the committed-value oracle of $\langle C, R\rangle$. Then, $\langle C, R\rangle$ is *CCA-secure (w.r.t the committed-value oracle)* if for any PPT adversary $\mathcal{A}$, the following are computationally indistinguishable:

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

In the following, we use *left session* to denote the session of the commit phase between the committer and $\mathcal{A}$, and use *right sessions* to denote the sessions between $\mathcal{A}$ and $\mathcal{O}$.

2.2 Falsifiable Polynomial-Time Hardness Assumption

Next, we give a definition of the falsifiable polynomial-time hardness assumptions, which is essentially the same as the definition of the falsifiable assumptions of [GW11].

Definition 2. A *falsifiable polynomial-time hardness assumption* is a pair (Ch, c), where Ch is a PPT ITM called a *challenger* and c is a constant such that $0 \le c < 1$. For any (possibly super-polynomial-time) adversary $\mathcal{A}$, we say that $\mathcal{A}$ *breaks* an assumption (Ch, c) if there exists a polynomial $p(\cdot)$ such that for infinitely many n, we have

$$\Pr\left[\mathsf{output}_{Ch}[\langle Ch, \mathcal{A}\rangle(1^n)] = 1\right] \ge c + 1/p(n) \ .$$

The assumption (Ch, c) is true if and only if no PPT adversary can break (Ch, c).

2.3 Black-Box Reduction

Finally, we recall the definition of black-box reductions from [GW11]. For concreteness, we consider only black-box reductions showing CCA security of a commitment scheme.

Definition 3. A *black-box reduction* is a PPT oracle machine. We say that a black-box reduction $\mathcal{R}$ shows CCA security of a commitment scheme $\langle C, R\rangle$ based on an assumption (Ch, c) if for any (possibly super-polynomial-time) adversary $\mathcal{A}$ that breaks CCA security of $\langle C, R\rangle$, $\mathcal{R}^{\mathcal{A}}$ breaks the assumption (Ch, c).

3 Proof of Theorem 1

In the proof of Theorem 1, we use a technique that Canetti et al.[CKPR02] used to show the impossibility of $o(\log n/\log\log n)$-round black-box concurrent zero-knowledge proofs for non-trivial languages.

First, we recall this technique (for details, see [CKPR02] and Appendix A). For any $k(n) = o(\log n/\log\log n)$-round zero-knowledge proof $\langle P, V\rangle$ and for any PPT black-box simulator $\mathcal{S}$, Canetti et al. constructed a family of cheating verifiers $\{V_{g,h}\}_{g\in G, h\in H}$, where G and H are families of hash functions. Each $V_{g,h}$ executes n^2 sessions of $\langle P, V\rangle$ in a specific schedule R_{n^2} (see Figure 1). The schedule R_{n^2} consists of n *recursive blocks*, and each recursive block consists of n sessions. In each session, $V_{g,h}$ interacts with the prover in the same way as the honest verifier does except that (1) randomness used in this session is determined by using h and a prefix of the transcript (called the *block prefix*) and (2) $V_{g,h}$ decides whether to abort this session by using g and a prefix of the transcript (called the *iteration prefix*)[4]. $V_{g,h}$ accepts a recursive block if and only if $V_{g,h}$ accepted at least $n^{1/2}/4$ sessions in this recursive block. If $V_{g,h}$ rejects a recursive block, $V_{g,h}$ halts. If $V_{g,h}$ accepts all n recursive blocks, $V_{g,h}$ outputs accept. Then, Canetti et al. showed that with overwhelming probability over the choice of $g \in G$, $h \in H$, and randomness of $\mathcal{S}$, if $\mathcal{S}^{V_{g,h}}$ outputs an accepted transcript (i.e., a

[4] Roughly speaking, the block prefix is defined so that whenever $\mathcal{S}$ rewinds $V_{g,h}$ in a recursive block, the randomness used in higher-level recursive blocks is completely changed (and thus $\mathcal{S}$ needs to rewind $V_{g,h}$ in these recursive blocks as well), and the iteration prefix is defined so that whenever $\mathcal{S}$ rewinds $V_{g,h}$ in a session, $V_{g,h}$ aborts this session with a fixed probability (and thus $\mathcal{S}$ needs to rewind $V_{g,h}$ many times until $\mathcal{S}$ gets an accepted transcript of this session).

For any $m \leq n^2$, the schedule R_m is recursively defined as follows.

1. If $m < n$, sessions $1, \ldots, m$ are executed sequentially until they are all completed.
2. Otherwise, for $j = 1, \ldots, k(n)$:

   Message exchange: Each of the first n sessions exchanges two messages.

   Recursive call: If $j < k(n)$, the scheduling $\mathsf{R}_{\lceil (m-n)/(k(n)-1) \rceil}$ is applied recursively on $\lceil (m-n)/(k(n)-1) \rceil$ new sessions.

The set of n sessions that is explicitly executed during the message exchange phase is called a *recursive block*.

Figure 1: Schedule R_m [CKPR02].

transcript in which $V_{g,h}$ outputs accept), there exists a session that was accepted but was not "rewound" in the execution of $\mathcal{S}^{V_{g,h}}$ (since otherwise the running time of $\mathcal{S}$ becomes super-polynomial).[5]

Next, we prove Theorem 1. In the proof, we use the idea behind $\{V_{g,h}\}_{g \in G, h \in H}$.

Proof (of Theorem 1). Let $\mathcal{R}$ be a black-box reduction showing CCA security of $\langle C, R \rangle$ based on a falsifiable polynomial-time hardness assumption (Ch, c). Then, for any (possibly super-polynomial-time) adversary $\mathcal{A}$ that breaks CCA security of $\langle C, R \rangle$, there exists a polynomial $p(\cdot)$ such that for infinitely many n, we have

$$\Pr\left[\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}} \rangle(1^n)] = 1\right] \geq c + 1/p(n) \ .$$

First, let us consider the following family $\{\mathcal{A}^1_{g,h}\}_{g \in G, h \in H}$ of super-polynomial-time adversaries against CCA security of $\langle C, R \rangle$. In the left session, $\mathcal{A}^1_{g,h}$ honestly interacts with the left committer with randomly chosen challenge values. In the right sessions, $\mathcal{A}^1_{g,h}$ interacts with $\mathcal{O}$ in n^2 sessions in the schedule R_{n^2}. In the i-th right session, $\mathcal{A}^1_{g,h}$ chooses random $v_i \in \{0,1\}^n$ and commits to v_i. The randomness used in this session is generated as in [CKPR02] (i.e., using h and the block prefix) and $\mathcal{A}^1_{g,h}$ decides whether to abort this session as in [CKPR02] (i.e., using g and the iteration prefix). $\mathcal{A}^1_{g,h}$ accepts the i-th session if and only if $\mathcal{O}$ returns v_i at the end of this session. $\mathcal{A}^1_{g,h}$ accepts a recursive block if and only if $\mathcal{A}^1_{g,h}$ accepted at least $n^{1/2}/4$ sessions in this recursive block. If $\mathcal{A}^1_{g,h}$ rejects a recursive block, $\mathcal{A}^1_{g,h}$ halts. If $\mathcal{A}^1_{g,h}$ accepts all n recursive blocks, $\mathcal{A}^1_{g,h}$ computes the committed value v in the left session by brute force and outputs v. (Thus, $\mathcal{A}^1_{g,h}$ breaks CCA security of $\langle C, R \rangle$.)

Next, let us consider the following family $\{\mathcal{A}^2_{g,h}\}$ of PPT adversaries. $\mathcal{A}^2_{g,h}$ is the same as $\mathcal{A}^1_{g,h}$ except that if $\mathcal{A}^2_{g,h}$ accepts all n blocks, $\mathcal{A}^2_{g,h}$ outputs a random string v.

Then, we show that with overwhelming probability over the choice of g and h, the following are computationally indistinguishable:

- $\{\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}^1_{g,h}} \rangle(1^n)]\}_{n \in \mathbb{N}}$
- $\{\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}^2_{g,h}} \rangle(1^n)]\}_{n \in \mathbb{N}}$

Since $\mathcal{A}^1_{g,h}$ and $\mathcal{A}^2_{g,h}$ differ only in the last message v, we can show the indistinguishability by showing that in the interaction with Ch, $\mathcal{R}^{\mathcal{A}^1_{g,h}}$ does not receive v from $\mathcal{A}^1_{g,h}$. Assume for contradiction that $\mathcal{R}^{\mathcal{A}^1_{g,h}}$ receives v from $\mathcal{A}^1_{g,h}$. Then, since $\mathcal{A}^1_{g,h}$ outputs v only if $\mathcal{A}^1_{g,h}$ accepts all n recursive blocks, and since the running time of Ch and that of $\mathcal{R}$ are polynomial, as in [CKPR02], we can show that there exists a session that was accepted but was not rewound (we also use the fact that $\langle C, R \rangle$ is $o(\log n / \log\log n)$ round). Then, since $\mathcal{A}^1_{g,h}$ accepts a session only if $\mathcal{A}^1_{g,h}$ received the committed value of this session, $\mathcal{R}$ must have sent the committed value of this session to $\mathcal{A}^1_{g,h}$ without rewinding $\mathcal{A}^1_{g,h}$. Since the running time of Ch and that of $\mathcal{R}$ are polynomial, this contradicts the hiding property of $\langle C, R \rangle$ (i.e., we can use Ch and $\mathcal{R}$ to break the hiding property of $\langle C, R \rangle$). We thus conclude that $\mathcal{R}$ does not receive v from $\mathcal{A}^1_{g,h}$, and therefore we conclude that the indistinguishability holds.

For every $g \in G$ and $h \in H$, since $\mathcal{A}^1_{g,h}$ breaks CCA security of $\langle C, R \rangle$, there exists a polynomial $p(\cdot)$ such that for infinitely many n, we have

$$\Pr\left[\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}^1_{g,h}} \rangle(1^n)] = 1\right] \geq c + 1/p(n) \ .$$

Then, from the above indistinguishability, for random $g \in G$ and $h \in H$, we have

$$\begin{aligned} &\Pr\left[\mathsf{output}_{Ch}[\langle Ch, \mathcal{R}^{\mathcal{A}^2_{g,h}} \rangle(1^n)] = 1\right] \\ &\geq c + 1/p(n) - \mathsf{negl}(n) \\ &\geq c + 1/\mathsf{poly}(n) \end{aligned}$$

with overwhelming probability over the choice of g and h. Since the running time of $\mathcal{R}$ and that of $\mathcal{A}^2_{g,h}$ are polynomial, this fact implies that the assumption (Ch, c) is false. □

4 Conclusion

In this paper, we showed that if we use black-box reductions to prove CCA security, we cannot construct $o(\log n / \log\log n)$-round CCA secure commitment schemes based on falsifiable polynomial-time hardness assumptions. We note that in [KMO12], the authors proposed a constant-round CCA-secure commitment scheme based on the existence of one-way functions *that are secure against sub-exponential-time adversaries*. This scheme gets around the result of this paper by using a sub-exponential-time hardness assumption.

[5] More precisely, Canetti et al. showed that in the execution of $\mathcal{S}^{V_{g,h}}$, there exists a *useful block prefix*.

References

[Bar01] Boaz Barak. How to go beyond the black-box simulation barrier. In *FOCS*, pages 106–115. IEEE Computer Society, 2001.

[BP12] Nir Bitansky and Omer Paneth. From the impossibility of obfuscation to a new non-black-box simulation technique. In *FOCS*. IEEE Computer Society, 2012.

[Can01] Ran Canetti. Universally composable security: A new paradigm for cryptographic protocols. In *FOCS*, pages 136–145. IEEE Computer Society, 2001.

[CF01] Ran Canetti and Marc Fischlin. Universally composable commitments. In Joe Kilian, editor, *CRYPTO*, volume 2139 of *Lecture Notes in Computer Science*, pages 19–40. Springer, 2001.

[CKL03] Ran Canetti, Eyal Kushilevitz, and Yehuda Lindell. On the limitations of universally composable two-party computation without set-up assumptions. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 68–86. Springer, 2003.

[CKPR02] Ran Canetti, Joe Kilian, Erez Petrank, and Alon Rosen. Black-box concurrent zero-knowledge requires (almost) logarithmically many rounds. *SIAM J. Comput.*, 32(1):1–47, 2002.

[CLP10] Ran Canetti, Huijia Lin, and Rafael Pass. Adaptive hardness and composable security in the plain model from standard assumptions. In *FOCS*, pages 541–550. IEEE Computer Society, 2010.

[DDN00] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. *SIAM J. Comput.*, 30(2):391–437, 2000.

[DNS04] Cynthia Dwork, Moni Naor, and Amit Sahai. Concurrent zero-knowledge. *J. ACM*, 51(6):851–898, 2004.

[Goy11] Vipul Goyal. Constant round non-malleable protocols using one way functions. In Lance Fortnow and Salil P. Vadhan, editors, *STOC*, pages 695–704. ACM, 2011.

[GW11] Craig Gentry and Daniel Wichs. Separating succinct non-interactive arguments from all falsifiable assumptions. In Lance Fortnow and Salil P. Vadhan, editors, *STOC*, pages 99–108. ACM, 2011.

[KMO12] Susumu Kiyoshima, Yoshifumi Manabe, and Tatsuaki Okamoto. Constant-round black-box construction of composable protocols. Unpublished manuscript, 2012.

[KOS03] Jonathan Katz, Rafail Ostrovsky, and Adam Smith. Round efficiency of multi-party computation with a dishonest majority. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 578–595. Springer, 2003.

[LP11] Huijia Lin and Rafael Pass. Constant-round non-malleable commitments from any one-way function. In Lance Fortnow and Salil P. Vadhan, editors, *STOC*, pages 705–714. ACM, 2011.

[LP12] Huijia Lin and Rafael Pass. Black-box constructions of composable protocols without set-up. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 461–478. Springer, 2012.

[LPV08] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. Concurrent non-malleable commitments from any one-way function. In Ran Canetti, editor, *TCC*, volume 4948 of *Lecture Notes in Computer Science*, pages 571–588. Springer, 2008.

[LPV09] Huijia Lin, Rafael Pass, and Muthuramakrishnan Venkitasubramaniam. A unified framework for concurrent security: universal composability from stand-alone non-malleability. In Michael Mitzenmacher, editor, *STOC*, pages 179–188. ACM, 2009.

[Nao03] Moni Naor. On cryptographic assumptions and challenges. In Dan Boneh, editor, *CRYPTO*, volume 2729 of *Lecture Notes in Computer Science*, pages 96–109. Springer, 2003.

[Pas04] Rafael Pass. Bounded-concurrent secure multi-party computation with a dishonest majority. In László Babai, editor, *STOC*, pages 232–241. ACM, 2004.

[PR05] Rafael Pass and Alon Rosen. Concurrent non-malleable commitments. In *FOCS*, pages 563–572. IEEE Computer Society, 2005.

[Ros06] Alon Rosen. *Concurrent Zero-Knowledge*. Springer-Verlag New York Inc, 2006.

A Details of $V_{g,h}$ [CKPR02]

In this section, we explain $V_{g,h}$ in [CKPR02] in a little more detail. (Some of the following text is taken from [Ros06].) Let $\langle P, V \rangle$ be a zero-knowledge proof for language L with round-complexity $k(n) = o(\log n / \log \log n)$, and $\mathcal{S}$ be a black-box concurrent simulator for $\langle P, V \rangle$ with polynomial running time $t_{\mathcal{S}}(\cdot)$. (Queries of $\mathcal{S}$ are

partial execution transcripts ending with prover messages, which are answered by the next cheating verifier's messages.) Let G be a family of $t_{\mathcal{S}}(n)$-wise independent hash functions mapping a $\mathsf{poly}(n)$-bit long string into a single bit so that for every $\alpha \in \{0,1\}^{\mathsf{poly}(n)}$ we have

$$\Pr\left[g(\alpha) = 1 \mid g \xleftarrow{\mathsf{U}} G\right] = n^{-1/2n} \ ,$$

and H be a family of $t_{\mathcal{S}}(n)$-wise independent hash functions mapping a $\mathsf{poly}(n)$-bit long string to a $\rho_V(n)$-bit string so that for every $\alpha \in \{0,1\}^{\mathsf{poly}(n)}$ and $\beta \in \{0,1\}^{\rho_V(n)}$ we have

$$\Pr\left[h(\alpha) = \beta \mid h \xleftarrow{\mathsf{U}} H\right] = 2^{-\rho_V(n)}$$

(where $\rho_V(n)$ is the number of random bits used by an honest verifier V on an input $x \in \{0,1\}^n$).

A.1 An Informal Description of $V_{g,h}$

$V_{g,h}$ executes n^2 sessions of $\langle P, V\rangle$ in a specific schedule R_{n^2} (see Figure 1). The schedule R_{n^2} consists of n *recursive blocks*, and each recursive block consists of n sessions. Thus, each session s is uniquely determined by identity $(\ell, i) \in \{1,\ldots,n\}\times\{1,\ldots,n\}$, where $\ell = \ell(s)$ is the index of the recursive block to which s belongs and $i = i(s)$ is the index of s within the n sessions that belong to the ℓ-th recursive block.

Definition 4 (Identifiers of next message)**.** The schedule defines a mapping from partial execution transcripts ending with a prover message to the identifiers of the next verifier message, i.e., the session and round number to which the next verifier message belongs. Recall that such partial execution transcripts correspond to queries of a black-box simulator and so the mapping defines the identifier of the answer. For such a query $\overline{q}$, we denote by $\pi_{\mathrm{sn}}(\overline{q}) = (\ell, i) \in \{1,\ldots,n\} \times \{1,\ldots,n\}$ the session to which the next verifier message belongs, and by $\pi_{\mathrm{msg}}(\overline{q}) = j \in \{1,\ldots,k(n)\}$ its index within the verifier's messages in this session.

In each session, $V_{g,h}$ interacts with a prover in the same way as the honest verifier does except that (1) randomness used in this session is determined by using h and a prefix of the transcript (called the *block prefix*) and (2) $V_{g,h}$ decides whether to abort this session by using g and a prefix of the transcript (called the *iteration prefix*), where the block prefix and the iteration prefix are defined below.

Definition 5 (Block prefix)**.** The block prefix of a query $\overline{q}$ satisfying $\pi_{\mathrm{sn}}(\overline{q}) = (\ell, i)$ is the prefix of $\overline{q}$ that is answered with the first verifier message of session $(\ell, 1)$, i.e., the first main session in block ℓ. More formally, $bp(\overline{q}) = (b_1, a_1, \ldots, b_\gamma, a_\gamma)$ is the block prefix of $\overline{q} = (b_1, a_1, \ldots, b_t, a_t)$ if $\pi_{\mathrm{sn}}(bp(\overline{q})) = (\ell, 1)$ and $\pi_{\mathrm{msg}}(bp(\overline{q})) = 1$. The block prefix will be said to correspond to recursive block ℓ.

Definition 6 (Iteration prefix)**.** The iteration prefix of a query $\overline{q}$ satisfying $\pi_{\mathrm{sn}}(\overline{q}) = (\ell, i)$ and $\pi_{\mathrm{msg}}(\overline{q}) = j > 1$ is the prefix of $\overline{q}$ that ends with the $(j-1)$-st prover message in session (ℓ, n), i.e, the n-th main session in black ℓ. More formally, $ip(\overline{q}) = (b_1, a_1, \ldots, b_\delta, a_\delta)$ is the iteration prefix of $\overline{q} = (b_1, a_1, \ldots, b_t, a_t)$ if a_δ is of the form $\mathsf{p}^{(n)}_{j-1}$, where $\mathsf{p}^{(n)}_{j-1}$ denotes the $(j-1)$-st prover message in the n-th main session of block ℓ. This iteration prefix is said to correspond to the block prefix of $\overline{q}$.

$V_{g,h}$ accepts a recursive block if and only if $V_{g,h}$ accepted at least $n^{1/2}/4$ sessions in this recursive block. If $V_{g,h}$ rejects a recursive block, $V_{g,h}$ halts. If $V_{g,h}$ accepts all n recursive blocks, $V_{g,h}$ outputs accept.

A.2 A Formal Description of $V_{g,h}$

On query $\overline{q} = (b_1, a_1, \ldots, a_{t-1}, b_t, a_t)$, the verifier $V_{g,h}$ acts as follows. (Without loss of generality, we assume that at the beginning of $\langle P, V\rangle$, the verifier sends a fixed initiation message.)

1. First, $V_{g,h}$ checks if the execution transcript given by the query is legal (i.e., corresponds to a possible execution prefix), and halts with a special error message if the query is not legal.

2. If a_t is the form $\mathsf{p}^{(n)}_{k(n)}$ (i.e., in case query $\overline{q}$ ends with the last prover message of the n-th main session of a recursive block), $V_{g,h}$ checks whether the transcript $\overline{q} = (b_1, a_1, \ldots, b_t, \mathsf{p}^{(n)}_{k(n)})$ contains the accepting conversations of at least $n^{1/2}/4$ main sessions in the block that has just been completed. In case it does not, $V_{g,h}$ halts with a special error message.

3. Next, $V_{g,h}$ determines the block prefix $bp(\overline{q}) = (b_1, a_1, \ldots, b_\gamma, a_\gamma)$ of query $\overline{q}$. It also determines the identifiers of the next-message $(\ell, i) = \pi_{\mathrm{sn}}(\overline{q})$ and $j = \pi_{\mathrm{msg}}(\overline{q})$, the iteration prefix $ip(\overline{q}) = (b_1, a_1, \ldots, b_\delta, \mathsf{p}^{(n)}_{j-1})$, and the $j-1$ prover messages of session i appearing in query $\overline{q}$, which we denote by $\mathsf{p}^{(i)}_1, \ldots, \mathsf{p}^{(i)}_{j-1}$.

4. If $j = 1$, then $V_{g,h}$ answers with the verifier's fixed initiation message for session i.

5. If $j > 1$, then $V_{g,h}$ determines $b_{i,j} = g(i, ip(\overline{q}))$.

   (a) If $b_{i,j} = 0$, then $V_{g,h}$ sets $\mathsf{v}^{(i)}_j = \mathsf{abort}$.

   (b) If $b_{i,j} = 1$, then $V_{g,h}$ determines $r_i = h(i, bp(\overline{q}))$ and computes the message $\mathsf{v}^{(i)}_j = V(x, r_i; \mathsf{p}^{(i)}_1, \ldots, \mathsf{p}^{(i)}_{j-1})$ that would have been sent by the honest verifier on common input x, random tape r_i, and prover's messages $\mathsf{p}^{(i)}_1, \ldots, \mathsf{p}^{(i)}_{j-1}$.

   (c) Finally, $V_{g,h}$ answers with $\mathsf{v}^{(i)}_j$.