Reducing Complexity Assumptions for Statistically-Hiding Commitment

Omer Horvitz[*†] Jonathan Katz[*‡] Chiu-Yuen Koo[*]
Ruggero Morselli[*]

Abstract

Determining the minimal assumptions needed to construct various cryptographic building blocks has been a focal point of research in theoretical cryptography. For most — but not all! — cryptographic primitives, complexity assumptions both necessary and sufficient for their existence are known. Here, we revisit the following, decade-old question: *what are the minimal assumptions needed to construct a statistically-hiding bit commitment scheme*? Previously, it was known how to construct such schemes based on any one-way permutation. In this work, we show that regular one-way functions suffice.

We show two constructions of statistically-hiding commitment schemes from regular one-way functions. Our first construction is more direct, and serves as a "stepping-stone" for our second construction which has improved round complexity. Of independent interest, as part of our work we show a *compiler* transforming any commitment scheme which is statistically-hiding against an honest-but-curious receiver to one which is statistically-hiding against a malicious receiver. This demonstrates the equivalence of these two formulations of the problem.

1 Introduction

A central focus of modern cryptography has been to investigate the weakest possible assumptions under which various cryptographic primitives exist. This direction of research has been quite fruitful, and minimal assumptions are known for a wide variety of primitives: e.g., pseudorandom generators, pseudorandom functions, symmetric-key encryption/message authentication, and digital signatures [24, 13, 14, 23, 27, 29, 32]. In other cases, black-box separation results exist which indicate the difficulty — if not impossibility — of constructing "strong" cryptographic protocols (say, key-exchange) from "weak" building blocks (say, one-way permutations; see [25]).

The above may give the impression that exact characterizations for all primitives of interest (at least in terms of equivalent complexity-theoretic assumptions) are known; however, this is not the case. Questions that remain open (to choose two examples) include the possibility of constructing efficient-prover non-interactive zero-knowledge proofs [4] based on assumptions weaker than trapdoor permutations [9], as well as determining whether constant-round ZK proofs exist based only on the assumption of one-way functions (see [11, Chap. 4]).

Another key cryptographic primitive which has resisted attempts at a full characterization is *statistically-hiding commitment*. Informally, a commitment scheme defines a two-phase interactive protocol between a sender $\mathcal{S}$ and a receiver $\mathcal{R}$: after the *commitment phase*, $\mathcal{S}$ is uniquely bound

[*]Dept. of Computer Science, University of Maryland. {`horvitz,jkatz,cykoo,ruggero`}`@cs.umd.edu`.
[†]Research supported by U.S. Army Research Office award DAAD19-01-1-0494.
[‡]Research supported by NSF CAREER award #0447075.

to (at most) one value which is not yet revealed to $\mathcal{R}$, and in the *decommitment phase* $\mathcal{R}$ finally learns this value. The two security properties hinted at in this informal description are known as *binding* (namely, that $\mathcal{S}$ is bound to at most one value after the commitment phase) and *hiding* (namely, that $\mathcal{R}$ does not learn the value to which $\mathcal{S}$ commits before the decommitment phase). In a statistically-hiding commitment scheme the hiding property holds *even against all-powerful receivers* (i.e., hiding holds information-theoretically), while the binding property is required to hold only for computationally-bounded (say, polynomial-time) senders.

Statistically-hiding commitment schemes are used as a building block in constructions of statistical zero-knowledge arguments [6, 28] or statistically-secure computation protocols (e.g., [2, 26]), and are also useful whenever the receiver is more powerful than the sender. They also have advantages when used within protocols in which certain commitments are never revealed; in this case, it need only be infeasible to violate the binding property *during the period of time the protocol is run*, whereas the committed values will remain hidden *forever* (i.e., regardless of how much time the receiver invests after completion of the protocol). Indeed, this is part of the motivation for statistical zero-knowledge as well. For further discussion, the reader is referred to [30, 31, 28].

Perfectly-hiding[1] commitment schemes were first shown to exist based on specific number-theoretic assumptions [6, 5] or, more generally, based on any collection of claw-free permutations [21] with an efficiently-recognizable index set [16] (see [16] for a definition of a weaker variant of statistically-hiding commitment which suffices for some applications and for which an efficiently-recognizable index set is not needed). Naor, et al. [28], using techniques developed earlier by Ostrovsky, et al. [30, 31], later showed a construction of a perfectly-hiding commitment scheme based on any one-way (almost-everywhere) permutation. Statistically-hiding commitment schemes can also be constructed from any collision-resistant hash function [8, 22] (see [33] for minimal assumptions for the existence of the latter). We remark that results of Simon [36] and Fischlin [10] rule out black-box constructions of collision-resistant hash functions or claw-free permutations based on one-way permutations.

We remark that using "*almost-everywhere* one-to-one" one-way functions [15] and/or "*almost* one-to-one" one-way functions [11, Sect. 3.5] (both of which are known to exist assuming any *regular* one-way function) in the construction of Naor, et al. does *not* result in a statistically-hiding commitment scheme. Prior to the present work, then, the minimal assumptions under which statistically-hiding commitment schemes were known to exist were one-way (almost-everywhere) permutations or the incomparable assumption of collision-resistant hash functions, neither of which are known to exist based on the assumption of regular one-way functions.

1.1 Our Results

A *regular* one-way function f is a one-way function satisfying the additional property that every point in the image of f has the same number of pre-images (but see footnote 2). A variety of conjectured one-way functions are regular; we refer the reader to [17] for examples. In this work we show that statistically-hiding commitment can be based on the existence of any regular one-way function. We show two constructions of the former from the latter: the first is more direct, and serves as a stepping-stone to the latter protocol which achieves better round complexity. Techniques used in designing the second construction may be of independent interest: as part of our work, we show a compiler transforming any commitment scheme which is statistically-hiding against

[1] Very informally, in a statistically-hiding commitment scheme the receiver learns only a negligible amount of information about the sender's committed value, whereas in a perfectly-hiding commitment scheme the receiver learns *nothing*. Note that any perfectly-hiding scheme is also statistically-hiding.

an honest-but-curious (a.k.a. semi-honest) receiver into a statistically-hiding commitment scheme secure against an arbitrarily-malicious receiver. Since our compiler requires only the existence of one-way functions, our result implies an equivalence between the two formulations of the problem. To the best of our knowledge, this is the first time such an equivalence has been demonstrated.

Our results may be viewed an example of the paradigm in which a sequence of works constructs a given primitive from ever-weaker assumptions; e.g., in the cases of pseudorandom generators and universal one-way hash functions/signature schemes (see [11, Chap. 2] and [12, Chap. 6]), constructions were first based on specific, number-theoretic assumptions [3, 21], and then the minimal assumptions were gradually reduced to trapdoor permutations [1] (in the case of signatures), one-way permutations [18, 29], regular one-way functions [17, 34], and (finally) one-way functions [23, 32]. We hope our work will similarly serve as a step toward resolving the question of the minimal assumptions required for statistically-hiding commitment.

2 Preliminaries

Throughout this paper, we let k denote a security parameter. Let X_1 and X_2 be two distributions over a set $\mathcal{X}$. The statistical difference between X_1 and X_2, written $\mathsf{SD}(X_1, X_2)$, is defined as:

$$\mathsf{SD}(X_1, X_2) \stackrel{\text{def}}{=} \frac{1}{2}\sum_{x\in\mathcal{X}} \left|\Pr_{X_1}[x] - \Pr_{X_2}[x]\right|.$$

Two distribution ensembles $X_1 = \{X_1(k)\}_{k\in\mathbb{N}}$ and $X_2 = \{X_2(k)\}_{k\in\mathbb{N}}$ are *statistically indistinguishable* if $\mathsf{SD}(X_1(k), X_2(k))$ is negligible as a function of k. For a function $f : \{0,1\}^* \to \{0,1\}^*$, we let $\mathsf{image}_k(f) \stackrel{\text{def}}{=} \{f(x) \mid x \in \{0,1\}^k\}$. If k is understood, we will simply write $\mathsf{image}(f)$.

Commitment schemes. An interactive bit commitment scheme is defined via a triple of algorithms $(\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$. Looking ahead, $\mathcal{S}$ and $\mathcal{R}_1$ will interact during what is called a *commitment phase*, while $\mathcal{R}_2$ will be used during the (non-interactive) *decommitment phase*. More formally:

- $\mathcal{S}$ (the *sender*) is a probabilistic polynomial time (PPT) interactive Turing machine (ITM) which receives as initial input the security parameter 1^k and a bit b. Following its interaction, it outputs some information decom (the *decommitment*).

- $\mathcal{R}_1$ (the *receiver*) is a PPT ITM which receives the security parameter 1^k as initial input. Following its interaction, it outputs some state information s.

- $\mathcal{R}_2$ (acting as a receiver, in the decommitment phase) is a deterministic poly-time algorithm which receives as input state information s and a decommitment decom; it outputs either a bit b or the distinguished value $\perp$.

Denote by $(\mathsf{decom} \mid s) \leftarrow \langle \mathcal{S}(1^k, b), \mathcal{R}_1(1^k)\rangle$ the experiment in which $\mathcal{S}$ and $\mathcal{R}_1$ interact (using the given inputs and uniformly random coins), and then $\mathcal{S}$ outputs decom while $\mathcal{R}_1$ outputs s. We make the following correctness requirement: for all k, all b, and every pair $(\mathsf{decom} \mid s)$ that may be output by $\langle \mathcal{S}(1^k, b), \mathcal{R}_1(1^k)\rangle$, it is the case that $\mathcal{R}_2(s, \mathsf{decom}) = b$.

The security of a commitment scheme can be defined in two complementary ways. Since we are interested in the case of statistically-hiding commitment, we provide only this definition here.

Definition 1 Commitment scheme $(\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$ is *statistically-hiding* if the following hold:

Statistical hiding. Given an ITM $\mathcal{R}_1^*$, let $\mathsf{view}_{\langle\mathcal{S}(b),\mathcal{R}_1^*\rangle}(k)$ denote the distribution over the view of $\mathcal{R}_1^*$ when interacting with $\mathcal{S}(1^k, b)$ (this view simply consists of the sequence of messages received from $\mathcal{S}$), where this distribution is taken over the random coins of $\mathcal{S}$ and we assume $\mathcal{R}_1^*$ is deterministic without loss of generality. Then we require that for any (even all-powerful) $\mathcal{R}_1^*$ the ensembles $\{\mathsf{view}_{\langle\mathcal{S}(0),\mathcal{R}_1^*\rangle}(k)\}$ and $\{\mathsf{view}_{\langle\mathcal{S}(1),\mathcal{R}_1^*\rangle}(k)\}$ are statistically indistinguishable.

Note that, without loss of generality, it suffices to consider deterministic $\mathcal{R}_1^*$.

Computational binding. Let $\mathcal{S}^*$ be an ITM which takes input 1^k and outputs $(\mathsf{decom}, \mathsf{decom}')$ following its interaction. Then we require that the following is negligible for any PPT $\mathcal{S}^*$:

$$\Pr\left[((\mathsf{decom}, \mathsf{decom}') \mid s) \leftarrow \langle\mathcal{S}^*(1^k), \mathcal{R}_1(1^k)\rangle : \begin{array}{c} \mathcal{R}_2(s, \mathsf{decom}), \mathcal{R}_2(s, \mathsf{decom}') \in \{0,1\} \bigwedge \\ \mathcal{R}_2(s, \mathsf{decom}) \neq \mathcal{R}_2(s, \mathsf{decom}') \end{array}\right],$$

where the probability is taken over the random coins of both $\mathcal{S}^*$ and $\mathcal{R}_1$. $\diamondsuit$

Regular one-way functions. Let $f : \{0,1\}^* \to \{0,1\}^*$ be a function such that, for all x, $|f(x)| = m(|x|)$ for some polynomially-bounded and computable function m. We say that f is *one-way* if it is computable in polynomial time and if the following is negligible for all PPT A:

$$\Pr[x \leftarrow \{0,1\}^k; y = f(x); x' \leftarrow A(1^k, y) : f(x') = y].$$

(Note that this is the classical definition of one-way function [11, Definition 2.2.1].) We additionally say that f is *$\ell(k)$-regular* if, for every x, we have $\left|\{x' \in \{0,1\}^k \mid f(x') = f(x)\}\right| = 2^{\ell(|x|)}$ and, furthermore, $\ell(k)$ is poly-time computable. In other words, for each x there are exactly $2^{\ell(|x|)}$ elements (including x itself) which f maps to the same value.[2] (Some previous definitions of regular one-way functions do not require that ℓ be poly-time computable and the constructions of, e.g., [17] do not rely on this. However, we do not know how to extend our results to the case when ℓ is not poly-time computable.)

Universal hashing and an extended Chernoff bound. Let $\mathcal{H} = \{H_k\}_{k\in\mathbb{N}}$ be a sequence of function families, where each H_k is a family of functions mapping strings of length $m(k)$ to strings of length $\delta(k)$. We assume further that the following can be done in time polynomial in k: (1) selecting a function $h \in H_k$ uniformly at random (we denote this by $h \leftarrow H_k$); (2) given $h \in H_k$ and $x \in \{0,1\}^{m(k)}$, evaluating $h(x)$; and (3) given a string h^*, deciding whether $h^* \in H_k$ or not.

Considering any particular value of k, we say H_k is *k-universal* (following [7]) if for any distinct $x_1, \ldots, x_k \in \{0,1\}^{m(k)}$, and any $y_1, \ldots, y_k \in \{0,1\}^{\delta(k)}$ we have:

$$\Pr_{h\in H_k}[h(x_1) = y_1 \wedge \ldots \wedge h(x_k) = y_k] = 2^{-\delta k}.$$

The following flavor of the Chernoff bound will be useful in our analysis:

Lemma 1 (Extended Chernoff Bound [35, Theorem 5]) *Let X be the sum of (any number of) n-wise independent random variables, each taking values in the interval $[0,1]$, such that $E[X] = \mu$. Then for any $\delta \le 1$ for which $n \ge \lfloor\delta^2\mu e^{-1/3}\rfloor$ we have* $\Pr[|X - \mu| \ge \delta\mu] \le e^{-\lfloor\delta^2\mu/3\rfloor}$.

[2] A more general definition of a regular function allows $f(x)$ to have a number of pre-images within a factor of $p(|x|)$ from $2^{\ell(|x|)}$, for some polynomial p (see [11], p. 79). Our constructions extend to functions satisfying this definition, at the price of increasing the number of rounds.

2.1 Interactive Hashing

Interactive hashing was introduced by Ostrovsky, et al. [30, 31], and used by Naor, et al. [28] to construct a statistically-hiding (actually, perfectly-hiding) commitment scheme based on any one-way permutation. We review interactive hashing, as well as the resulting commitment scheme, in Appendix A. For purposes of self-containment it suffices to note that interactive hashing defines an interactive protocol between a sender $\mathcal{S}$ (with input y) and a receiver $\mathcal{R}$. At the conclusion of the protocol, $\mathcal{S}$ obtains (y_0, y_1, v) satisfying $y_v = y$, and $\mathcal{R}$ obtains (y_0, y_1). For future reference, we denote by $IH(y)$ an execution of the interactive hashing protocol, where $\mathcal{S}$ begins with input y.

As mentioned, interactive hashing is used in [28] to construct a perfectly-hiding commitment scheme based on any permutation f (see Construction 5 in Appendix A). In their commitment scheme, it is relatively easy to see that the hiding property holds for an arbitrary permutation f (regardless of whether f is one-way). The main result of [28] was to prove that their scheme is *computationally binding* when f is a one-way permutation. In fact, examination of their proof shows that it achieves computational binding under a *weaker* condition on f: it suffices for f to be what we call "one-way over its range", defined as follows:

Definition 2 Let $f : \{0,1\}^* \to \{0,1\}^*$ be such that there exists a polynomially-bounded function $m(\cdot)$ such that $|f(x)| = m(|x|)$ for all x. We say f is *one-way over its range* if the following is negligible for all PPT A:

$$\Pr[y \leftarrow \{0,1\}^{m(k)}; x \leftarrow A(1^k, y) : f(x) = y].$$

For future reference, we state the following theorem:

Theorem 1 (Implicit in [28]) *If f is one-way over its range, then Construction 5 is computationally binding.*

We stress that the notion of a function being "one-way over its range" is decidedly *not* equivalent to the (standard) definition of a one-way function given earlier, since in the present case y is a uniformly-random string of length $m(k)$ (as opposed to setting $y = f(x)$ for uniformly-random x). In fact, the constant function (i.e., $f(x) = 0^{m(|x|)}$) is one-way over its range without relying on any computational assumptions. Of course, when such a function is used in Construction 5 the construction no longer satisfies the hiding property.

3 Our Main Construction

Here, we show our main result: a construction of a statistically-hiding commitment scheme based on any regular one-way function. For simplicity of exposition, we drop the explicit dependence on the security parameter k (i.e., we describe the protocol for some fixed value of k) and thus write m, ℓ instead of $m(k), \ell(k)$; recall, however, that these are all polynomially-bounded functions.

Construction 1 *Let $f : \{0,1\}^* \to \{0,1\}^*$ be an ℓ-regular function such that $|f(x)| = m$ for all $x \in \{0,1\}^k$. Let $\delta \stackrel{\text{def}}{=} k - \ell - \log k$, and let H_k be a k-universal family of functions mapping m-bit strings to δ-bit strings. Define $\omega \stackrel{\text{def}}{=} \log^2 k$.*

The commitment scheme is defined by a tuple of algorithms $(\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$. The sender $\mathcal{S}$, on input a bit b, interacts with $\mathcal{R}_1$ in ω phases. In each phase i (for $i = 1, \ldots, \omega$):

1. *$\mathcal{S}$ selects $x_i \in \{0,1\}^k$ and $h_i \in H_k$ uniformly at random; it then computes $y_i = h_i(f(x_i))$.*

2. $\mathcal{S}$ *and* $\mathcal{R}_1$ *then run* $IH(h_i|y_i)$, *with* $\mathcal{S}$ *obtaining output* $((h_{i,0}|y_{i,0}),(h_{i,1}|y_{i,1}),v_i)$ *(i.e.,* $h_i|y_i = h_{i,v_i}|y_{i,v_i}$*) and* $\mathcal{R}_1$ *obtaining output* $((h_{i,0}|y_{i,0}),(h_{i,1}|y_{i,1}))$.

At the completion of all ω *phases,* $\mathcal{S}$ *sends* $\hat{v} = b \oplus \bigoplus_{i=1}^{\omega} v_i$ *to* $\mathcal{R}_1$. *Finally,* $\mathcal{S}$ *outputs* $\mathsf{decom} = \{h_i|x_i\}_{1\leq i\leq\omega}$ *and* $\mathcal{R}_1$ *outputs state* $s = \left(\hat{v}, \{h_{i,b}|y_{i,b}\}_{1\leq i\leq\omega; b\in\{0,1\}}\right)$.

In the decommitment phase, $\mathcal{R}_2(s,\mathsf{decom})$ *proceeds as follows: for each* i, *find* v_i *such that* $h_i|h_i(f(x_i)) = h_{i,v_i}|y_{i,v_i}$. *If for some* i *no such* v_i *exists, output* $\perp$. *Otherwise, output* $\hat{v} \oplus \bigoplus_{i=1}^{\omega} v_i$.

It is easy to see that correctness holds if both sender and receiver are honest. We now show that the above gives a statistically-hiding commitment scheme when f is one-way:

Theorem 2 *If* f *is an* ℓ*-regular, one-way function then Construction 1 is a statistically-hiding commitment scheme.*

The theorem follows from the two lemmas proved below.

Lemma 2 *Construction 1 is statistically hiding.*

Proof For a given execution of the scheme, let τ denote the initial transcript resulting from the ω iterations of the interactive hashing sub-protocols; thus, the entire view of $\mathcal{R}_1^*$ consists of τ and the bit $\hat{v}$ sent in the final round. Given a particular (deterministic) $\mathcal{R}_1^*$, we therefore write $(\tau,\hat{v}) \leftarrow \mathsf{view}_{\langle\mathcal{S}(b),\mathcal{R}_1^*\rangle}$ (cf. Definition 1; security parameter k is implicit) to denote the experiment in which $\mathcal{S}$ chooses a uniform random tape and then executes the protocol with $\mathcal{R}_1^*$ using this random tape and the bit b, resulting in view $(\tau,\hat{v})$ for $\mathcal{R}_1^*$. Below, we define a "good" set of initial transcripts Good, and show that:

Claim 1 *With all but negligible probability* $\varepsilon_1(k)$, *we have* $\tau \in \mathsf{Good}$.

Claim 2 *For some negligible* $\varepsilon_2(k)$, *the following holds for all* $\tau^* \in \mathsf{Good}$ *and* $\hat{v}^* \in \{0,1\}$*:*

$$\left|\Pr[\hat{v} = \hat{v}^* \mid \tau = \tau^*, b = 0] - \Pr[\hat{v} = \hat{v}^* \mid \tau = \tau^*, b = 1]\right| \leq \varepsilon_2(k).$$

These claims suffice to prove the lemma, since the statistical difference between the view of $\mathcal{R}_1^*$ when the sender commits to 0 (i.e., $b=0$) and the view of $\mathcal{R}_1^*$ when the sender commits to 1 (i.e., $b=1$) may be bounded as follows:

$$\begin{aligned}
&\frac{1}{2}\sum_{\tau^*,\hat{v}^*}\left|\Pr_{(\tau,\hat{v})\leftarrow\mathsf{view}_{\langle\mathcal{S}(0),\mathcal{R}_1^*\rangle}}[(\tau,\hat{v}) = (\tau^*,\hat{v}^*)] - \Pr_{(\tau,\hat{v})\leftarrow\mathsf{view}_{\langle\mathcal{S}(1),\mathcal{R}_1^*\rangle}}[(\tau,\hat{v}) = (\tau^*,\hat{v}^*)]\right| \\
&\quad\leq\ \varepsilon_1(k) + \frac{1}{2}\sum_{\tau^*\in\mathsf{Good};\hat{v}^*}\left|\Pr_{(\tau,\hat{v})\leftarrow\mathsf{view}_{\langle\mathcal{S}(0),\mathcal{R}_1^*\rangle}}[(\tau,\hat{v}) = (\tau^*,\hat{v}^*)] - \Pr_{(\tau,\hat{v})\leftarrow\mathsf{view}_{\langle\mathcal{S}(1),\mathcal{R}_1^*\rangle}}[(\tau,\hat{v}) = (\tau^*,\hat{v}^*)]\right| \\
&\quad\leq\ \varepsilon_1(k) + \frac{1}{2}\sum_{\tau^*\in\mathsf{Good};\hat{v}^*}\Pr_{(\tau,\hat{v})\leftarrow\mathsf{view}_{\langle\mathcal{S}(0),\mathcal{R}_1^*\rangle}}[\tau = \tau^*]\cdot\varepsilon_2(k) \quad\leq\quad \varepsilon_1(k) + \varepsilon_2(k)
\end{aligned}$$

(i.e., the views are statistically close), where we use the fact that $\Pr_{(\tau,\hat{v})\leftarrow\mathsf{view}_{\langle\mathcal{S}(0),\mathcal{R}_1^*\rangle}}[\tau = \tau^*] = \Pr_{(\tau,\hat{v})\leftarrow\mathsf{view}_{\langle\mathcal{S}(1),\mathcal{R}_1^*\rangle}}[\tau = \tau^*]$ for any τ^*, since the initial transcript τ does not depend on b.

We proceed with the proof of the first claim by defining the set of good initial transcripts. This set is defined via an event Good which depends only on the initial transcript (thus, the abuse of notation should not cause confusion). The i^{th} phase of the interactive hashing sub-protocol defines values $(h_{i,0}|y_{i,0}), (h_{i,1}|y_{i,1})$ as described earlier. For any function $h \in H_k$ and string $y \in \{0,1\}^\delta$, let:

$$\mathsf{Preimages}_h(y) \stackrel{\text{def}}{=} |\{z \mid z \in \mathsf{image}(f) \wedge h(z) = y\}|$$

and let

$$\mathsf{BadPairs} \stackrel{\text{def}}{=} \left\{(h,y) \mid \mathsf{Preimages}_h(y) \leq \frac{k}{2} \bigvee \mathsf{Preimages}_h(y) \geq \frac{3k}{2}\right\}.$$

We say event Good_i occurs if both $(h_{i,0}, y_{i,0}), (h_{i,1}, y_{i,1}) \notin \mathsf{BadPairs}$. Finally, define $\mathsf{Good} \stackrel{\text{def}}{=} \cap_{i=1}^{\omega} \mathsf{Good}_i$ (i.e., Good_i occurs for all i).

Consider a fixed, arbitrary phase i. We first bound the probability that $(h_{i,v_i}, y_{i,v_i}) \stackrel{\text{def}}{=} (h_i, y_i)$ is in $\mathsf{BadPairs}$. Note that this event depends solely on the choices of the honest sender in the relevant phase. Let $x_i \in \{0,1\}^k$ be the point chosen by the sender. For all $z \in \mathsf{image}(f) \setminus \{f(x_i)\}$, define the indicator random variable X_z to be 1 iff $h_i(f(x_i)) = h_i(z)$ and let $X \stackrel{\text{def}}{=} \sum_{z \in \mathsf{image}(f) \setminus \{f(x_i)\}} X_z$. For an arbitrary $z \in \mathsf{image}(f) \setminus \{f(x_i)\}$ we have $E[X_z] = 2^{-\delta}$, where the probability is taken over the choice of h_i. It follows that

$$E[X] = (|\mathsf{image}(f)| - 1) \cdot 2^{-\delta} = (2^{k-\ell} - 1) \cdot 2^{\ell + \log k - k} = k - \frac{k}{2^{k-\ell}}.$$

Furthermore, since H_k is a k-universal family, the random variables $\{X_z\}$ are $(k-1)$-wise independent. Thus, by Lemma 1, for k large enough we have

$$\Pr\left[|X - E[X]| \geq \frac{1}{2}E[X]\right] \leq e^{-k/30},$$

where we use the fact that $k/2^{k-\ell} = k/|\mathsf{image}(f)|$ must be negligible since f is one-way. Finally, note that $\mathsf{Preimages}_{h_i}(f(x_i)) = X + 1$; putting everything together, we see that for k large enough:

$$\Pr_{h_i}[(h_i, y_i) \in \mathsf{BadPairs}] = \Pr\left[\mathsf{Preimages}_{h_i}(y_i) \leq \frac{k}{2} \bigvee \mathsf{Preimages}_{h_i}(y_i) \geq \frac{3k}{2}\right] \leq e^{-k/30}. \tag{1}$$

Next, we bound the probability that $(h_{i,v_i}, y_{i,v_i}) \notin \mathsf{BadPairs}$ but $(h_{i,\bar{v}_i}, y_{i,\bar{v}_i}) \in \mathsf{BadPairs}$. Since Equation (1) also holds for a fixed arbitrary y_i and a random h_i, it follows that

$$|\mathsf{BadPairs}| \leq \textstyle\sum_{y_i \in \{0,1\}^\delta} \Pr_{h_i}[(h_i, y_i) \in \mathsf{BadPairs}] \cdot |H_k| \leq 2^\delta \cdot |H_k| \cdot e^{-k/30} \tag{2}$$

for large enough k. Now, conditioned on the view of $\mathcal{R}_1^*$ in all previous phases, $(h_{i,\bar{v}_i}, y_{i,\bar{v}_i})$ is uniquely determined by (h_{i,v_i}, y_{i,v_i}) (since we assume $\mathcal{R}_1^*$ is deterministic). Let ϕ be the function mapping the sender's chosen value (h_{i,v_i}, y_{i,v_i}) to the second value $(h_{i,\bar{v}_i}, y_{i,\bar{v}_i})$ resulting from the interactive hashing protocol. Observe that if $\phi(h,y) = (h', y')$, then it is also the case that $\phi(h', y') = (h, y)$; this is because, for either of these choices, the sender responds with the exact same answer to each of the receiver's queries during the interactive hashing sub-protocol. It follows that ϕ is one-to-one. Letting $\mathsf{MapToBadPairs} \stackrel{\text{def}}{=} \phi^{-1}(\mathsf{BadPairs})$, Equation (2) implies that $|\mathsf{MapToBadPairs}| \leq$

$2^\delta \cdot |H_k| \cdot e^{-k/30}$. Thus:

$$\begin{aligned}
&\Pr\left[(h_{i,v_i}, y_{i,v_i}) \notin \mathsf{BadPairs} \bigwedge (h_{i,\bar{v}_i}, y_{i,\bar{v}_i}) \in \mathsf{BadPairs}\right] \\
&\quad = \Pr\left[(h_{i,v_i}, y_{i,v_i}) \in \mathsf{MapToBadPairs} \setminus \mathsf{BadPairs}\right] \\
&\quad = \sum_{(h,y)\in\mathsf{MapToBadPairs}\setminus\mathsf{BadPairs}} \Pr\left[(h_{i,v_i}, y_{i,v_i}) = (h,y)\right] \\
&\quad = \sum_{(h,y)\in\mathsf{MapToBadPairs}\setminus\mathsf{BadPairs}} \frac{1}{|H_k|} \cdot \frac{\mathsf{Preimages}_h(y)}{2^{k-\ell}} \\
&\quad \leq \sum_{(h,y)\in\mathsf{MapToBadPairs}\setminus\mathsf{BadPairs}} \frac{1}{|H_k|} \cdot \frac{3k/2}{2^{k-\ell}},
\end{aligned}$$

using the fact that $(h,y) \notin \mathsf{BadPairs}$. Continuing:

$$\begin{aligned}
\sum_{(h,y)\in\mathsf{MapToBadPairs}\setminus\mathsf{BadPairs}} \frac{1}{|H_k|} \cdot \frac{3k/2}{2^{k-\ell}} &\leq |\mathsf{MapToBadPairs}| \cdot \frac{1}{|H_k|} \cdot \frac{3k/2}{2^{k-\ell}} \\
&\leq \left(2^\delta e^{-k/30}\right) \cdot \frac{3k/2}{2^{k-\ell}} \quad = \quad \frac{3}{2} \cdot e^{-k/30}. \qquad (3)
\end{aligned}$$

Equations (1) and (3) show that Good_i occurs with all but negligible probability, and thus Good occurs with all but negligible probability as well. This completes the proof of Claim 1.

A proof of Claim 2 follows rather easily. Occurrence of Good implies that, for all i,

$$\frac{1}{3} \leq \frac{\mathsf{Preimages}_{h_{i,v_i}}(y_{i,v_i})}{\mathsf{Preimages}_{h_{i,\bar{v}_i}}(y_{i,\bar{v}_i})} \leq 3.$$

This further implies that, from the receiver's point of view, $\frac{1}{4} \leq \Pr[v_i = 0] \leq \frac{3}{4}$ for all i; moreover, $\left|\Pr[\bigoplus_{i=1}^{\omega} v_i = 0] - \frac{1}{2}\right| \leq 2^{-\omega}$. But this means that for any $\hat{v}^* \in \{0,1\}$ we have

$$\begin{aligned}
\left|\Pr[\hat{v} = \hat{v}^* \mid b = 0] - \Pr[\hat{v} = \hat{v}^* \mid b = 1]\right| &= \left|\Pr[\textstyle\bigoplus_{i=1}^{\omega} v_i = \hat{v}^*] - \Pr[\bigoplus_{i=1}^{\omega} v_i = 1 \oplus \hat{v}^*]\right| \\
&\leq 2^{-\omega+1},
\end{aligned}$$

which is a negligible quantity since $\omega = \log^2 k$. ■

Lemma 3 *Construction 1 is computationally binding.*

Proof We prove computational binding for each of the interactive hashing sub-protocols: namely, for each i we show that it is computationally infeasible for a PPT $\mathcal{S}^*$ to compute $x_{i,0}, x_{i,1}$ such that $h_{i,0}(f(x_{i,0})) = y_{i,0}$ and $h_{i,1}(f(x_{i,1})) = y_{i,1}$. A straightforward hybrid argument then immediately implies the Lemma. In what follows, we omit dependence on i since the value of i is arbitrary.

Each interactive hashing sub-protocol is exactly Construction 5 instantiated with the function $f'(h,x) = (h, h(f(x)))$ where $h \in H_k$ and $x \in \{0,1\}^k$. Once we show that f' is one way over its range, applying Theorem 1 gives the desired result. Toward establishing that f' is one way over its range, we first prove that f' is one way (according to the standard definition). Let A' be a PPT adversary attempting to invert f', and let

$$\begin{aligned}
&\mathsf{Adv}_{A',f'}(k) \stackrel{\text{def}}{=} \\
&\quad \Pr[h \leftarrow H_k; x \leftarrow \{0,1\}^k; (h,y) = f'(h,x); (h',x') \leftarrow A'(1^k, h, y) : f'(h',x') = (h,y)]. \qquad (4)
\end{aligned}$$

Let $\mathsf{Expt}_{A'}(k)$ denote the experiment in the above expression (i.e., $\mathsf{Expt}_{A'}(k)$ denotes the experiment "$h \leftarrow H_k; x \leftarrow \{0,1\}^k; (h,y) = f'(h,x); (h',x') \leftarrow A'(1^k, h, y)$").

Now construct a PPT adversary A (attempting to invert f) as follows:

$\underline{A(1^k, z)}$ // $z = f(x)$ for some $x \in \{0,1\}^k$ chosen at random.
  Choose $h \in H_k$ at random, and set $y = h(z)$;
  Run $A'(1^k, h, y)$ and obtain output h', x';
  Output x'

Note that the distribution over the inputs of A' in the above experiment is identical to the distribution over the inputs of A' in Equation (4). Observe that:

$$\begin{aligned}
\mathsf{Adv}_{A,f}(k) &\stackrel{\text{def}}{=} \Pr[x \leftarrow \{0,1\}^k; z = f(x); x' \leftarrow A(1^k, z) : f(x') = z] \\
&= \sum_{\hat{h},\hat{y}} \Pr[\mathsf{Expt}_{A'}(k) : h(f(x')) = y \bigwedge h = \hat{h}, y = \hat{y}] \cdot \frac{1}{\mathsf{Preimages}_{\hat{h}}(\hat{y})},
\end{aligned}$$

where $\mathsf{Preimages}_h(y)$ is as in the proof of Lemma 2. Let $\mathsf{BadPairs}$ be as in the proof of Lemma 2, and recall that $\Pr[\mathsf{Expt}_{A'}(k) : (h,y) \in \mathsf{BadPairs}] \leq e^{-k/30}$ (for large enough k). We may thus write:

$$\begin{aligned}
\mathsf{Adv}_{A,f}(k) &\geq \sum_{(\hat{h},\hat{y}) \notin \mathsf{BadPairs}} \Pr[\mathsf{Expt}_{A'}(k) : h(f(x')) = y \bigwedge h = \hat{h}, y = \hat{y}] \cdot \frac{1}{\mathsf{Preimages}_{\hat{h}}(\hat{y})} \\
&\geq \frac{2}{3k} \cdot \sum_{(\hat{h},\hat{y}) \notin \mathsf{BadPairs}} \Pr[\mathsf{Expt}_{A'}(k) : h(f(x')) = y \bigwedge h = \hat{h}, y = \hat{y}] \\
&= \frac{2}{3k} \cdot \sum_{\hat{h},\hat{y}} \Pr[\mathsf{Expt}_{A'}(k) : h(f(x')) = y \bigwedge h = \hat{h}, y = \hat{y}] \\
&\quad - \frac{2}{3k} \cdot \sum_{(\hat{h},\hat{y}) \in \mathsf{BadPairs}} \Pr[\mathsf{Expt}_{A'}(k) : h(f(x')) = y \bigwedge h = \hat{h}, y = \hat{y}] \\
&\geq \frac{2}{3k} \cdot \left(\mathsf{Adv}_{A',f'}(k) - \Pr[\mathsf{Expt}_{A'}(k) : (h, f(x)) \in \mathsf{BadPairs}]\right) \\
&\geq \frac{2}{3k} \cdot \left(\mathsf{Adv}_{A',f'}(k) - e^{-k/30}\right),
\end{aligned}$$

for large enough k. Since $\mathsf{Adv}_{A,f}(k)$ is negligible by assumption, it must be the case that $\mathsf{Adv}_{A',f'}(k)$ is negligible as well and thus f' is one way.

We now show that f' is one-way over its range. Consider any PPT algorithm A'' inverting f' "over its range". The advantage of A'' (in this sense) is given by:

$$\begin{aligned}
\mathsf{Adv}^*_{A'',f'} &\stackrel{\text{def}}{=} \Pr[h \leftarrow H_k; y \leftarrow \{0,1\}^\delta; (h',x') \leftarrow A''(1^k, h, y) : f'(h',x') = (h,y)] \\
&= \frac{1}{|H_k| \cdot 2^\delta} \cdot \sum_{h \in H_k} \sum_{y \in \{0,1\}^\delta} \Pr[A'' \text{ inverts } (h,y)],
\end{aligned}$$

where "A'' inverts (h,y)" has the obvious meaning.

Consider now the advantage of A'' in inverting f' in the standard sense:

$$\begin{aligned}
\mathsf{Adv}_{A'',f'} &\stackrel{\text{def}}{=} \Pr[h \leftarrow H_k; x \leftarrow \{0,1\}^k : A'' \text{ inverts } (h, h(f(x)))] \\
&= \frac{1}{|H_k| \cdot 2^k} \sum_{h \in H_k} \sum_{x \in \{0,1\}^k} \Pr[A'' \text{ inverts } (h, h(f(x)))] \\
&= \frac{1}{|H_k| \cdot 2^k} \sum_{h \in H_k} \sum_{z \in \mathsf{image}(f)} 2^\ell \cdot \Pr[A'' \text{ inverts } (h, h(z))],
\end{aligned}$$

using the fact that f is ℓ-regular. Continuing:

$$\begin{aligned}
&\frac{1}{|H_k| \cdot 2^k} \sum_{h \in H_k} \sum_{z \in \mathsf{image}(f)} 2^\ell \cdot \Pr[A'' \text{ inverts } (h, h(z))] \\
&\quad= \frac{2^\ell}{|H_k| \cdot 2^k} \sum_{h \in H_k} \sum_{y \in \mathsf{image}(h(f))} \sum_{z \in h^{-1}(y)} \Pr[A'' \text{ inverts } (h, h(z))] \\
&\quad\geq \frac{2^\ell}{|H_k| \cdot 2^k} \sum_{h \in H_k} \sum_{y \in \mathsf{image}(h(f))} \Pr[A'' \text{ inverts } (h, y)] \\
&\quad= \frac{2^\ell}{|H_k| \cdot 2^k} \sum_{h \in H_k} \sum_{y \in \{0,1\}^\delta} \Pr[A'' \text{ inverts } (h, y)] \\
&\quad= \frac{2^\ell \cdot 2^\delta}{2^k} \mathsf{Adv}^*_{A'',f'} \;=\; \frac{\mathsf{Adv}^*_{A'',f'}}{k}.
\end{aligned}$$

Since $\mathsf{Adv}_{A'',f'}$ is negligible (by one-wayness of f'), $\mathsf{Adv}'^*_{A'',f'}$ is negligible as well. This completes the proof that f' is one-way over its range, and thus completes the proof of the lemma. ■

Round complexity. In characterizing the round complexity of Construction 1, we may first note that all the $\omega = \log^2 k$ interactive hashing sub-protocols may be run *in parallel*. (A proof that the protocol remains statistically hiding in this case will appear in the full version.) The round complexity of the construction is then equal to the round complexity of (each of) the interactive hashing sub-protocols. The round complexity of the latter is linear in the length of $\mathcal{S}$'s input (h, y); since we require H_k to be k-wise independent, the length of $h \in H_k$ is $O(k \cdot \max\{m, \delta\})$, and thus the round complexity is $O(k \cdot \max\{m, \delta\} + \delta) = O(k \cdot \max\{m, \delta\})$. (In fact, the round complexity of interactive hashing to a string of length ℓ can be reduced to $O(\ell/\log k)$ rounds, resulting in a round complexity of $O(k \cdot \max\{m, \delta\}/\log k)$ for the commitment scheme.)

4 Improving the Round Complexity

Here, we show that the round complexity of the construction of the previous section can be improved to $\omega(\delta) = \omega(k)$. The methodology we use to construct our improved solution may be of independent interest: we first show a simple modification of Construction 1 which is proven secure against an *honest-but-curious* (i.e., semi-honest) receiver, and then show how any protocol secure against an honest-but-curious receiver can be *compiled* (based only on one-way functions) to give a protocol secure against a malicious receiver. (We are not aware of any such compiler being demonstrated before in the context of commitment schemes.) Our compiler increases the round complexity by an additive factor of $\omega(1)$. A corollary of our technique is that statistically-hiding commitment schemes secure against a *malicious receiver* exist if and only if there exist statistically-hiding commitment schemes secure against an *honest-but-curious receiver.*

For completeness, we first provide a definition of security against a semi-honest receiver:

Definition 3 Commitment scheme $(\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$ is *statistically-hiding against an honest-but-curious receiver* if the following hold:

Statistical hiding Let $\mathsf{view}_{\langle \mathcal{S}(b), \mathcal{R}_1 \rangle}(k)$ be as in Definition 1. Then we require that the ensembles $\{\mathsf{view}_{\langle \mathcal{S}(0), \mathcal{R}_1 \rangle}(k)\}$ and $\{\mathsf{view}_{\langle \mathcal{S}(1), \mathcal{R}_1 \rangle}(k)\}$ are statistically indistinguishable.

Note that here we consider only the view of the honest receiver $\mathcal{R}_1$. This view now depends on both the random coins of $\mathcal{S}$ as well as the random coins of $\mathcal{R}_1$.

Computational binding As in Definition 1. $\diamondsuit$

4.1 Statistically-Hiding Commitment against Honest-But-Curious Receivers

The construction presented in this section is motivated by Construction 1. Here, however, the hash functions $h_i \in H_k$ are chosen by the receiver and sent to the sender, rather than being chosen by the sender and used as inputs to the interactive hashing sub-protocols. Because the receiver now chooses the h_i, it is essential here that the receiver be semi-honest.

Construction 2 *Let $f : \{0,1\}^* \to \{0,1\}^*$ be an ℓ-regular function such that $|f(x)| = m$ for all $x \in \{0,1\}^k$. Let $\delta \stackrel{\text{def}}{=} k - \ell - \log k$, and let H_k be a k-universal family of functions mapping m-bit strings to δ-bit strings. Define $\omega \stackrel{\text{def}}{=} \log^2 k$.*

The commitment scheme is defined by a tuple of algorithms $(\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$. The sender $\mathcal{S}$, on input a bit b, interacts with $\mathcal{R}_1$ in ω phases. In each phase i (for $i = 1, \ldots, \omega$):

1. *$\mathcal{R}_1$ selects $h_i \in H_k$ uniformly at random and sends h_i to the sender.*
2. *$\mathcal{S}$ chooses $x_i \leftarrow \{0,1\}^k$ and computes $y_i = h_i(f(x_i))$. $\mathcal{S}$ and $\mathcal{R}_1$ then run $IH(y_i)$, with $\mathcal{S}$ obtaining output $(y_{i,0}, y_{i,1}, v_i)$ (i.e., $y_i = y_{i,v_i}$) and $\mathcal{R}_1$ obtaining output $(y_{i,0}, y_{i,1})$.*

At the completion of all ω phases, $\mathcal{S}$ sends $\hat{v} = b \oplus \bigoplus_{i=1}^{\omega} v_i$ to $\mathcal{R}_1$. Finally, $\mathcal{S}$ outputs $\mathsf{decom} = \{x_i\}_{1 \leq i \leq \omega}$ and $\mathcal{R}_1$ outputs state $s = \left(\hat{v}, \{h_i\}_{1 \leq i \leq \omega}, \{y_{i,b}\}_{1 \leq i \leq \omega; b \in \{0,1\}}\right)$.

In the decommitment phase, $\mathcal{R}_2(s, \mathsf{decom})$ proceeds as follows: for each i, find v_i such that $h_i(f(x_i)) = y_{i,v_i}$. If for some i no such v_i exists, output $\perp$. Otherwise, output $\hat{v} \oplus \bigoplus_{i=1}^{\omega} v_i$.

As in the previous section, each of the ω phases may be run in parallel; doing so results in a round complexity of $|y_i| = \delta$. We remark that it would be sufficient for the receiver to select only a single hash function h (which would then be used by the sender in all the phases). Since this modification does not affect the round complexity, and since a proof is slightly simpler when independent hash functions are used in each phase, we are content with the above description.

Theorem 3 *If f is an ℓ-regular, one-way function then Construction 2 is a statistically-hiding commitment scheme against an honest-but-curious receiver.*

Proof The proof is substantially similar to the proof of Theorem 2, and we therefore only provide a sketch here. First consider the hiding property: using the notation introduced in the proof of Lemma 2, note that $(h_i, y_i) \notin \mathsf{BadPairs} \cup \mathsf{MapToBadPairs}$ with all but negligible probability (recall that h_i is chosen at random by a semi-honest receiver); statistical hiding then follows easily. As for binding, the proof of Lemma 3 essentially shows that — with all but negligible probability over choice of $h \in H_k$ — the function f_h defined by $f_h(x) = h(f(x))$ is one-way over its range. Application of Theorem 1 then completes the proof of binding. ■

4.2 Obtaining Security against a Malicious Receiver

We now demonstrate a compiler which converts any statistically-hiding commitment scheme against an honest-but-curious receiver to a statistically-hiding commitment scheme (i.e., where hiding holds even against a malicious receiver). Our compiler increases the round complexity of the original protocol by only an additive factor of $\omega(1)$. Furthermore, since our compiler requires only the existence of one-way functions (which are implied by the existence of a statistically-hiding commitment scheme against honest-but-curious receivers), we obtain:

Theorem 4 *The existence of a statistically-hiding commitment scheme against honest-but-curious receivers implies the existence of a statistically-hiding commitment scheme.*

Our compiler uses a coin-tossing protocol and zero-knowledge (ZK) proofs (in a way similar to [19]) to "force" honest behavior on the part of the receiver. However, we do not require "simulatable" coin-tossing (as in [2, 19, 26]) or ZK proofs of correctness following each round (as in [19]); instead, we show that a weaker variant of coin-tossing along with a single ZK proof at the end suffice. (The latter in particular is essential for obtaining a round-efficient compiler.)

Informally, our compiler proceeds as follows: the receiver first uses a statistically-*binding* commitment scheme to commit to a sufficiently-long string r_1, and the sender responds with a string r_2 of the same length. The sender and receiver then execute the original protocol, with the receiver using $r_1 \oplus r_2$ as its random tape and the sender committing to a *random* bit b'. At the conclusion of the original protocol, the receiver uses a ZK proof to show that each of the messages it sent during the course of the protocol is consistent with the messages sent by $\mathcal{S}$ as well as the random tape $r_1 \oplus r_2$ (we stress that r_1 is never revealed to $\mathcal{S}$). Finally (assuming $\mathcal{S}$ accepts the proof), $\mathcal{S}$ concludes the protocol by sending $b' \oplus b$ (where b is the bit that $\mathcal{S}$ wants to commit).

Before giving a formal description and proof of security for our compiler, some brief remarks are in order: first, note that one-way functions are sufficient for both statistically-binding commitment [27] as well as zero-knowledge proofs [20, 27]. Unfortunately, the best known round complexity for ZK proofs (with negligible soundness error) based on one-way functions is $\omega(1)$ [20] — the constant-round ZK proof of Goldreich and Kahan [16], for example, requires a statistically-hiding commitment scheme, the very primitive we are trying to construct! (Note also that ZK *arguments* will not do for our application, since the receiver may be all-powerful.) Second, since we have the receiver provide a ZK proof of correctness only at the conclusion of the protocol we must take into account the fact that the receiver may cheat during the course of the protocol, learn some information about the bit committed to by $\mathcal{S}$, and then abort (since it will be unable to provide a successful ZK proof with all but negligible probability). To prevent such an occurrence, we have $\mathcal{S}$ commit to a random bit b'; thus, the only portion of the transcript that depends on the committed bit of $\mathcal{S}$ (i.e., the final bit $b \oplus b'$) is sent *after* the receiver successfully proves correctness of its actions. We now provide a detailed description of our compiler.

Construction 3 *Given as input commitment protocol* $(\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$*, we construct a commitment protocol* $(\mathcal{S}^*, \mathcal{R}_1^*, \mathcal{R}_2^*)$*. During the commitment phase, the sender* $\mathcal{S}^*$ *with input bit b interacts with the receiver* $\mathcal{R}_1^*$ *as follows:*

1. *Let $\ell = \ell(k)$ denote the length of the random tape used by $\mathcal{R}_1$. Then $\mathcal{R}_1^*$ uses a (possibly interactive) statistically-binding commitment scheme to commit to a random string $r_1 \in \{0,1\}^\ell$. Let $(\mathsf{com}, \mathsf{decom})$ denote the resulting commitment (known to both $\mathcal{S}^*$ and $\mathcal{R}_1^*$) and decommitment (known to $\mathcal{R}_1^*$). In response, $\mathcal{S}^*$ sends a random string $r_2 \in \{0,1\}^\ell$. This defines a string $r \stackrel{\text{def}}{=} r_1 \oplus r_2$ which is known to $\mathcal{R}_1^*$.*

2. *$\mathcal{S}^*$ chooses a random bit b', and then $\mathcal{S}^*$ and $\mathcal{R}_1^*$ run protocols $\mathcal{S}(b')$ and $\mathcal{R}_1$, respectively, where the latter is run using random tape r. Note that $\mathcal{R}_1^*$ is entirely deterministic in this stage of the protocol. At the conclusion of this stage, $\mathcal{S}^*$ outputs decom' while $\mathcal{R}_1^*$ outputs s.*

3. *After completion of the above, $\mathcal{R}_1^*$ provides a ZK proof (with negligible soundness error) that it acted correctly throughout the previous stage. Formally, $\mathcal{R}_1^*$ proves that there exists (decom, r_1) such that com is a commitment to r_1 and all the messages sent by $\mathcal{R}_1^*$ in the previous stage are consistent with the messages sent by $\mathcal{S}^*$ and the random tape $r = r_1 \oplus r_2$.*

4. *If $\mathcal{S}^*$ rejects the proof given by $\mathcal{R}_1^*$, it aborts. Otherwise, $\mathcal{S}^*$ sends $\hat{b} = b \oplus b'$ and outputs decom'; the receiver $\mathcal{R}_1^*$ outputs $(s, \hat{b})$.*

In the decommitment phase, $\mathcal{R}_2^*$ proceeds as follows: it runs $\mathcal{R}_2(s, \mathsf{decom}')$ to obtain a bit b' (if the output of $\mathcal{R}_2$ is $\perp$, then $\mathcal{R}_2^*$ outputs $\perp$ as well), and then outputs $\hat{b} \oplus b'$.

We claim the following result:

Theorem 5 *If $(\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$ is a statistically-hiding commitment scheme against an honest-but-curious receiver, then $(\mathcal{S}^*, \mathcal{R}_1^*, \mathcal{R}_2^*)$ as generated by the above compiler is a statistically-hiding commitment scheme (i.e., even against a malicious receiver).*

Proof We provide only a sketch of the proof here, but the omitted details are straightforward. Correctness of $(\mathcal{S}^*, \mathcal{R}_1^*, R_2^*)$ is easy to see, so we first consider the hiding property.

Claim 3 *If $\Pi = (\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$ is statistically-hiding against an honest-but-curious receiver, then $\Pi^* = (\mathcal{S}^*, \mathcal{R}_1^*, \mathcal{R}_2^*)$ is statistically-hiding even against a malicious (all-powerful) receiver.*

Proof Let $\mathcal{R}_1^{**}$ denote a malicious receiver who interacts with $\mathcal{S}^*$ running Π^*, and assume that $\mathcal{R}_1^{**}$ is deterministic without loss of generality. We show the existence of a randomized procedure ψ_0 (which is not computable in polynomial time) with the following properties: on input an element distributed according to $\mathsf{view}^{\Pi}_{\langle \mathcal{S}(b), \mathcal{R}_1 \rangle}(k)$ (i.e., the view of honest-but-curious $\mathcal{R}_1$ interacting with $\mathcal{S}(b)$ in an execution of Π), it outputs an element whose distribution is statistically close to the distribution $\left\langle \mathsf{view}^{\Pi^*}_{\langle \mathcal{S}^*(b), \mathcal{R}_1^{**} \rangle}(k), 0 \right\rangle$ (where we abuse notation and let $\mathsf{view}^{\Pi_1^*}_{\langle \mathcal{S}^*(b), \mathcal{R}_1^{**} \rangle}(k)$ denote the view of $\mathcal{R}_1^{**}$ *except for the final bit* in an execution with $\mathcal{S}^*$ where $\mathcal{S}^*$ uses random bit $b' = b$). A procedure ψ_1 can be defined similarly, with the property that on input an element distributed according to $\mathsf{view}^{\Pi}_{\langle \mathcal{S}(b), \mathcal{R}_1 \rangle}(k)$ it outputs an element whose distribution is statistically close to the distribution $\left\langle \mathsf{view}^{\Pi^*}_{\langle \mathcal{S}^*(b), \mathcal{R}_1^{**} \rangle}(k), 1 \right\rangle$. Statistical closeness of $\mathsf{view}^{\Pi}_{\langle \mathcal{S}(0), \mathcal{R}_1 \rangle}(k)$ and $\mathsf{view}^{\Pi}_{\langle \mathcal{S}(1), \mathcal{R}_1 \rangle}(k)$, along with the construction of the compiler, completes the proof of the claim.[3]

Procedure ψ_0, on input a tuple $(m_1, \ldots, m_i, r)$ (where $m_1, \ldots, m_i$ denote the messages of the sender $\mathcal{S}$ and r denotes the random coins used by honest-but-curious $\mathcal{R}_1$), proceeds as follows:

[3] A slight subtlety here is that this informal description disregards the case when $\mathcal{R}_1^{**}$ aborts (e.g., by failing to give a convincing ZK proof). However, it is not hard to see that this is without loss of generality, since the view of $\mathcal{R}_1^{**}$ is independent of the committed bit of $\mathcal{S}^*$ if $\mathcal{R}_1^{**}$ aborts at any point during the protocol.

1. ψ_0 first runs the statistically-binding commitment scheme with $\mathcal{R}_1^{**}$ to obtain a commitment com. Next, ψ_0 "breaks" this commitment (running in exponential time, if necessary) to obtain a string r_1 to which com corresponds. It then sends the string $r_2 = r \oplus r_1$ to $\mathcal{R}_1^{**}$.

2. Next, ψ_0 interacts with $\mathcal{R}_1^{**}$ by sending messages $m_1, \ldots, m_r$ in response to the messages of $\mathcal{R}_1^{**}$.

3. After sending the m_i, the procedure acts as a verifier in an execution of the ZK proof with $\mathcal{R}_1^{**}$. If the proof succeeds, then ψ_0 concludes the simulation by sending the final bit 0.

We now argue (somewhat informally) that procedure ψ_0 satisfies the properties sketched above; see also footnote 3. Indeed, it is not difficult to see that the view generated by the above procedure matches the view claimed unless either (1) $\mathcal{R}_1^{**}$ violates the binding property of the commitment scheme; or (2) $\mathcal{R}_1^{**}$ is able to give a successful ZK proof for a false statement. Since both of these events occur with only negligible probability (even when considering an all-powerful $\mathcal{R}_1^{**}$), the claimed properties hold. ■

We next consider the binding property.

Claim 4 *If $\Pi = (\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$ is computationally-binding, then $\Pi^* = (\mathcal{S}^*, \mathcal{R}_1^*, \mathcal{R}_2^*)$ is computationally-binding as well.*

Proof Given a PPT sender $\mathcal{S}^{**}$ violating the binding property of Π^* with non-negligible probability, we construct a PPT sender $\hat{\mathcal{S}}$ violating the binding property of Π with non-negligible probability. $\hat{\mathcal{S}}$ is defined as follows:

1. $\hat{\mathcal{S}}$ interacts with an honest receiver $\mathcal{R}_1$, and runs a copy of $\mathcal{S}^{**}$ internally. It begins by sending a random commitment to the string $r_1 = 0^\ell$ to $\mathcal{S}^{**}$, who responds with a string $r_2 \in \{0,1\}^\ell$.

2. $\hat{\mathcal{S}}$ then relays messages faithfully between $\mathcal{R}_1$ and $\mathcal{S}^{**}$. At the conclusion of this phase, no more messages are sent to $\mathcal{R}_1$.

3. Finally, $\hat{\mathcal{S}}$ simulates a ZK proof of correct behavior with $\mathcal{S}^{**}$ acting as the potentially-dishonest verifier. ($\mathcal{S}^{**}$ then sends a final bit, which $\hat{\mathcal{S}}$ ignores.)

Note that if $\mathcal{S}^{**}$ is able to output valid decommitments to two different bits, then $\hat{\mathcal{S}}$ does so as well. It remains to argue that the probability that $\mathcal{S}^{**}$ outputs two valid decommitments in an interaction with $\hat{\mathcal{S}}$ (as above) is negligibly-close to the probability that $\mathcal{S}^{**}$ outputs two valid decommitments in an interaction with an honest $\mathcal{R}_1^*$. This follows by consideration of the following sequence of experiments (in what follows, let $\Pr_i[\mathsf{NoBind}]$ denote the probability that $\mathcal{S}^{**}$ outputs two valid decommitments in experiment i):

Experiment 0. This is the original experiment, where $\mathcal{S}^{**}$ interacts with $\mathcal{R}_1^*$.

Experiment 1. Here, we have $\mathcal{R}_1^*$ act exactly as in Experiment 0, except that it simulates the final ZK proof of correct behavior. By the ZK property of the proof system (against computationally-bounded verifiers), $|\Pr_0[\mathsf{NoBind}] - \Pr_1[\mathsf{NoBind}]|$ is negligible.

Experiment 2. Now, we have $\mathcal{R}_1^*$ act as in the previous experiment, except that its initial commitment is to 0^ℓ rather than to a random $r_1 \in \{0,1\}^\ell$. Computational hiding of the commitment scheme implies that $|\Pr_2[\mathsf{NoBind}] - \Pr_1[\mathsf{NoBind}]|$ is negligible.

To complete the proof, note that Experiment 2 corresponds exactly to an interaction of $\mathcal{S}^{**}$ with $\hat{\mathcal{S}}$. ■

The preceding claims imply the stated theorem. ■

Acknowledgments

The second author thanks Yan Zong Ding for his helpful comments and encouragement. We also thank the anonymous referees for comments that improved the presentation.

References

[1] M. Bellare and S. Micali. How to sign given any trapdoor permutation. *J. ACM*, 39(1):214–233, 1992.

[2] M. Blum. Coin flipping by phone. In *IEEE COMPCOM*, 1982.

[3] M. Blum and S. Micali. How to generate cryptographically-strong sequences of pseudorandom bits. *SIAM J. Computing*, 13(4):850–864, 1984.

[4] M. Blum, A. De Santis, S. Micali, and G. Persiano. Non-interactive zero-knowledge. *SIAM J. Computing*, 20(6):1084–1118, 1991.

[5] J.F. Boyar, S.A. Kurtz, and M.W. Krentel. Discrete logarithm implementation of perfect zero-knowledge blobs. *Journal of Cryptology*, 2(2):63–76, 1990.

[6] G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *J. Computer and System Sciences*, 37(2):156–189, 1988.

[7] J.L. Carter and M.N. Wegman. Universal classes of hash functions. *J. Computer and System Sciences*, 18(2):143–154, 1979.

[8] I. Damgård, T. Pedersen, and B. Pfitzmann. On the existence of statistically-hiding bit commitment and fail-stop signatures. In *Advances in Cryptology — Crypto '93*, volume 773 of *Lecture Notes in Computer Science*, pages 250–165. Springer, 1994.

[9] U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero-knowledge proofs under general assumptions. *SIAM J. Computing*, 29(1):1–28, 1999.

[10] M. Fischlin. On the impossibility of constructing non-interactive statistically-secret protocols from any trapdoor one-way function. In *RSA Cryptographers' Track*, volume 2271 of *Lecture Notes in Computer Science*, pages 79–95. Springer, 2002.

[11] O. Goldreich. *Foundations of Cryptography, vol. 1: Basic Tools*. Cambridge University Press, 2001.

[12] O. Goldreich. *Foundations of Cryptography, vol. 2: Basic Applications*. Cambridge University Press, 2004.

[13] O. Goldreich, S. Goldwasser, and S. Micali. On the cryptographic applications of random functions. In *Advances in Cryptology — Crypto '84*, volume 196 of *Lecture Notes in Computer Science*, pages 276–288. Springer, 1985.

[14] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986.

[15] O. Goldreich, R. Impagliazzo, L. Levin, R. Venkatesan, and D. Zuckerman. Security preserving amplification of hardness. In *Proc. 31st Annual Symposium on Foundations of Computer Science*, pages 318–326. IEEE, 1990.

[16] O. Goldreich and A. Kahan. How to construct constant-round zero-knowledge proof systems for NP. *Journal of Cryptology*, 9(3):167–190, 1996.

[17] O. Goldreich, H. Krawczyk, and M. Luby. On the existence of pseudorandom generators. *SIAM J. Computing*, 22(6):1163–1175, 1993.

[18] O. Goldreich and L.A. Levin. Hard-core predicates for any one-way function. In *Proc. 22nd Annual ACM Symposium on Theory of Computing*, pages 25–32. ACM, 1989.

[19] O. Goldreich, S. Micali, and A. Widgerson. How to play any mental game — a completeness theorem for protocols with honest majority. In *Proc. 19th Annual ACM Symposium on Theory of Computing*, pages 218–229. ACM, 1987.

[20] O. Goldreich, S. Micali, and A. Widgerson. Proofs that yield nothing but their validity or all languages in NP have zero-knowledge proof systems. *J. ACM*, 38(1):691–729, 1991.

[21] S. Goldwasser, S. Micali, and R.L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM J. on Computing*, 17(2):281–308, 1988.

[22] S. Halevi and S. Micali. Practical and provably-secure commitment schemes from collision-free hashing. In *Advances in Cryptology — Crypto '96*, volume 1109 of *Lecture Notes in Computer Science*, pages 201–215. Springer, 1996.

[23] J. Håstad, R. Impagliazzo, L.A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[24] R. Impagliazzo and M. Luby. One-way functions are essential for complexity-based cryptography. In *Proc. 30th Annual Symposium on Foundations of Computer Science*, pages 230–235. IEEE, 1989.

[25] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *Proc. 21st Annual ACM Symposium on Theory of Computing*, pages 44–61. ACM, 1989.

[26] Y. Lindell. Parallel coin-tossing and constant-round secure two-party computation. *Journal of Cryptology*, 16(3):143–184, 2003.

[27] M. Naor. Bit commitment using pseudorandomness. *Journal of Cryptology*, 4(2):151–158, 1991.

[28] M. Naor, R. Ostrovsky, R. Venkatesan, and M. Yung. Perfect zero-knowledge arguments for NP using any one-way permutation. *J. Cryptology*, 11(2):87–108, 1998.

[29] M. Naor and M. Yung. Universal one-way hash functions and their cryptographic applications. In *Proc. 21st Annual ACM Symposium on Theory of Computing*, pages 33–43. ACM, 1989.

[30] R. Ostrovsky, R. Venkatesan, and M. Yung. Secure commitment against a powerful adversary. In *STACS '92*, volume 577 of *Lecture Notes in Computer Science*, pages 439–448. Springer, 1992.

[31] R. Ostrovsky, R. Venkatesan, and M. Yung. Fair games against an all-powerful adversary. In *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, volume 13, 1993.

[32] J. Rompel. One-way functions are necessary and sufficient for secure signatures. In *Proc. 22nd Annual ACM Symposium on Theory of Computing*, pages 387–394, 1990.

[33] A. Russel. Necessary and sufficient conditions for collision-free hashing. *J. Cryptology*, 8(2):87–100, 1995.

[34] A. De Santis and M. Yung. On the design of provably-secure cryptographic hash functions. In *Advances in Cryptology — Eurocrypt '90*, volume 473 of *Lecture Notes in Computer Science*, pages 412–431. Springer, 1991.

[35] J.P. Schmidt, A. Siegel, and A. Srinivasan. Chernoff-Hoeffding bounds for applications with limited independence. *SIAM J. Discrete Math.*, 8(2):223–250, 1995.

[36] D. Simon. Finding collisions on a one-way street: Can secure hash functions be based on general assumptions? In *Advances in Cryptology — Eurocrypt '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345. Springer, 1998.

A A Brief Review of Interactive Hashing

We review the interactive hashing protocol of [30, 31, 28], which is used in [28] to construct a perfectly-hiding commitment scheme based on the existence of any one-way permutation. In what follows, we let $x \cdot y$ denote $\sum_{i=1}^{m} x_i y_i \bmod 2$ for $x, y \in \{0,1\}^m$.

Construction 4 (Interactive hashing) *The protocol is defined by algorithms $\mathcal{S}$ and $\mathcal{R}$, where $\mathcal{S}$ begins with an m-bit value y (with m known to $\mathcal{R}$), and proceeds as follows:*

1. *The parties interact in $m-1$ stages. In stage i (for $i = 1, \ldots, m-1$), $\mathcal{R}$ chooses $r_i \in \{0,1\}^{m-i}$ uniformly at random and sends the "query" $q_i = 0^{i-1}1r_i$ to $\mathcal{S}$ (in case $\mathcal{R}$ aborts, $\mathcal{S}$ simply takes q_i to be some default value); in response, $\mathcal{S}$ sends $c_i = q_i \cdot y$.*

2. *At the conclusion of the above, there are exactly two strings $y_0, y_1 \in \{0,1\}^m$ satisfying the system of equations $\{q_i \cdot X = c_i\}_{1 \le i \le m-1}$; let y_0 denote the lexicographically smaller of the two. Both parties compute (y_0, y_1), and $\mathcal{S}$ chooses v such that $y = y_v$.*

We define the output of the protocol to be (y_0, y_1, v) for $\mathcal{S}$ and (y_0, y_1) for $\mathcal{R}$.

The above protocol was used in [28] to construct a perfectly-hiding commitment scheme:

Construction 5 (A perfectly-hiding commitment scheme) *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a length-preserving permutation. The scheme is defined by algorithms $(\mathcal{S}, \mathcal{R}_1, \mathcal{R}_2)$ defined as follows: $\mathcal{S}(1^k, b)$ chooses $x \in \{0,1\}^k$ uniformly at random, computes $y = f(x)$, and then executes $IH(y)$ with $\mathcal{R}_1$; this protocol results in output (y_0, y_1, v) for $\mathcal{S}$ and (y_0, y_1) for $\mathcal{R}_1$. The commitment phase concludes by having $\mathcal{S}$ send $\hat{v} = v \oplus b$ to $\mathcal{R}_1$. Finally, $\mathcal{S}$ outputs $\mathsf{decom} = x$ while $\mathcal{R}_1$ outputs state $s = (y_0, y_1, \hat{v})$.*

In the decommitment phase, $\mathcal{R}_2((y_0, y_1, \hat{v}), x)$ proceeds as follows: if $f(x) = y_0$, output $\hat{v}$; if $f(x) = y_1$, outputs $\hat{v} \oplus 1$; otherwise, output $\perp$.