On the (Im)possibility of Basing Oblivious Transfer and Bit Commitment on Weakened Security Assumptions

Ivan Damgård[1], Joe Kilian[2], and Louis Salvail[1]

[1] BRICS, Basic Research in Computer Science, center of the Danish National Research Foundation, Department of Computer Science,University of Århus, Ny Munkegade, DK-8000 Århus C, Denmark.
{ivan,salvail}@daimi.aau.dk
[2] NEC Research Institute, 4 Independence Way, Princeton, NJ 08540.
joe@research.nj.nec.com

Abstract. We consider the problem of basing Oblivious Transfer (OT) and Bit Commitment (BC), with information theoretic security, on seemingly weaker primitives. We introduce a general model for describing such primitives, called *Weak Generic Transfer* (WGT). This model includes as important special cases *Weak Oblivious Transfer* (WOT), where both the sender and receiver may learn too much about the other party's input, and a new, more realistic model of noisy channels, called *unfair noisy channels*. An unfair noisy channel has a known range of possible noise levels; protocols must work for any level within this range against adversaries who know the actual noise level.
We give a precise characterization for when one can base OT on WOT. When the deviation of the WOT from the ideal is above a certain threshold, we show that no information-theoretic reductions from OT (even against passive adversaries) and BC exist; when the deviation is below this threshold, we give a reduction from OT (and hence BC) that is information-theoretically secure against active adversaries.
For unfair noisy channels we show a similar threshold phenomenon for bit commitment. If the upper bound on the noise is above a threshold (given as a function of the lower bound) then no information-theoretic reduction from OT (even against passive adversaries) or BC exist; when it is below this threshold we give a reduction from BC. As a partial result, we give a reduction from OT to UNC for smaller noise intervals.

1 Introduction

A 1 out of 2 Oblivious transfer (1-2 OT) protocol is one by which a sender with 2 bits b_0, b_1 as input can interact with a receiver with a bit c as input. Ideally, the sender should learn nothing new from the protocol, whereas the receiver should learn b_c and nothing more. Several variants of OT exist, but it does not matter much which one we consider, as they are almost all equivalent (see e.g. [8]).

A bit commitment scheme is a pair of protocols Commit and Open executed by two parties, a commiter, C, and a receiver, R. First, C and R execute Commit,

J. Stern (Ed.): EUROCRYPT'99, LNCS 1592, pp. 56–73, 1999.

where C has a bit b as input; R either accepts that a commitment has taken place or rejects. Ideally, the receiver should learn no information about b from this. Later, they may execute Open, after which R returns *accept* 1, *accept* 0 or *reject*. We require our protocols to be *correct*, *private* and *binding*:

Correctness: If both parties follow the protocol, R should always accept with the same value (b) that C wished to commit to.

Privacy: Committing to b reveals nothing to the receiver about b.

Binding: C cannot cause R to accept a commitment, and then be able to execute Open so that R accepts a 1 and also be able to execute Open so that R accepts a 0.

We have described the ideal requirements here. However, usually when building such protocols, one accepts an error that can be made negligibly small as a function of some security parameter k.

A great deal of work has gone into how to implement oblivious transfer and bit commitment based on seemingly weaker primitives. For example, a binary symmetric channel (BSC) is one that allows a sender S to send a bit b_S to a receiver R, such that a bit b_R will be received, which is not necessarily equal to b_S. There is a constant probability $0 < \delta < 1/2$, called the *noise level* of the channel such that each time the channel is invoked, $Pr(b_S \neq b_R) = \delta$. Another, essentially equivalent formulation has S and R receiving random bits b_S and b_R that are individually unbiased but correlated so that $Pr(b_S \neq b_R) = \delta$. Another equivalent formulation has a random bit b transmitted to both parties through independent noisy channels. One motivation for the last two formulations is that one might want to implement noisy channels by a very weak broadcast source, such as a satellite.

Crépeau and Kilian [9] showed that a BSC can be used to implement 1–2 OT with unconditional (information theoretic) security; the efficiency of this reduction was later improved by Crépeau [6], who also directly implemented a bit commitment scheme (indirectly, bit commitment can be based on 1–2 OT).

The reductions given in [9,6] rely on the fact that δ is known exactly by each party. However, in real life it may be possible for one party to surreptitiously alter the noise level of the channel. If the noise is induced by a communications channel then it may be possible to alter the mechanism (say by heating it up or cooling it down), or change the way it uses the mechanism, to change the noise rate. For example, suppose the channel consists of two pieces of optical fibre with a repeater station in between, a very common case in practice. If one party has access to the data received by the repeater station, then he can send and receive a cleaner signal than the other party expects him to. In the case of a noisy broadcast channel, an adversary might send a jamming signal or buy a more sensitive antenna. Note that while it may be hard to hide the fact that one has made a channel noisier, one can always hide the fact that one has made it less noisy, simply by deliberately garbling ones own messages and pretending to hear a more garbled version than one has actually heard.

Such "unfair advantages" are not always devastating for applications to cryptography: Maurer [15] shows that secure key exchange between two parties with

access to a random but noisy signal is possible, even in the presence of an enemy who can receive the signal much more reliably than the honest players. However, this scenario is a game for two parties who trust each other and want to be protected "against the rest of the world." It is natural to ask if we can still make do with unfair channels in case of games between two *mutually distrusting parties*. Unfortunately, the protocols of [9,6] break down in this scenario.

1.1 Our Results

In this paper we propose a general model for two-party primitives where a cheating player can get more information than an honest one; we call this model *Weak Generic Transfer* (WGT). We then consider a number of important subcases and show when they can and cannot be used as a basis for bit commitment and oblivious transfer.

We consider a family of *Weak Oblivious Transfers*, which are 1–2 OT protocols with the following faults: with probability (at most) p a cheating sender will learn which bit the receiver chose to receive, and with probability q a cheating receiver will learn both of the sender's input bits. Note that the honest participants only receive what they are supposed to receive; this extra information cannot be relied on. We call such a protocol a (p, q)-WOT. We give tight results for when one can reduce oblivious transfer to (p, q)-WOT.

In the statement of our results, when we use "reduction" we mean a reduction that is information-theoretically secure against unbounded adversaries, where deviations from the ideal are negligible in a given security parameter.

Theorem 1. *1–2 OT and BC can be reduced to (p, q)-WOT iff $p + q < 1$.*

We also consider a still noisier model, denoted (p, q, ϵ)-WOT, in which with probability *at most* ϵ an honest receiver receives $\overline{b_c}$ instead of b_c (i.e., the incorrect value); a cheating receiver is under no such handicap. In this case, we prove positive and negative results that are no longer tight.

Theorem 2. *1–2 OT can be reduced to (p, q, ϵ)-WOT, for the case of passive adversaries, if $p + q + 2\epsilon < .45$. No reductions from 1–2 OT or BC exist if $p + q + 2\epsilon \geq 1$.*

Passive adversaries, also known as "honest but curious" adversaries follow the protocol, but then try to use their view of the protocol execution to violate the security conditions.

Both theorems comprise a constructive result and an impossibility result. The constructive result of Theorem 1 generalizes a theorem of [9], which solves the special cases where either p or q is 0 (or negligible in the security parameter). Brassard/Crépeau [2] and Cachin [4] consider a more general model of WOT, where the extra information that an adversary learns is only specified by a general information measure, but here again the weakness is one-sided: only the receiver learns extra information. Prior to this work, few nontrivial impossibility results of this type were known (see [14] for one such result). These impossibility

results hold even if security against passive cheating is required and the honest players are allowed infinite computing power.

We note that one motivation for the study of these imperfect protocols is that they provide easier to achieve steps for other reductions. For example, our reduction from 1–2 OT to unfair noisy channels first reduces (p, q, ϵ)-WOT to unfair noisy channels.

We finally consider *unfair noisy channels* (UNC). These channels have parameters γ and δ, where $\gamma, \delta \leq 1/2$. The noise level p of this channel is guaranteed to fall into the interval $[\gamma, \delta]$. The protocol must work for any p in this range; however the value of p is *not* known to the honest players (but may be set within this range by the adversary).

Theorem 3. *For $\delta > 2\gamma(1-\gamma)$, neither 1–2 OT nor BC may be reduced to (γ, δ)-UNC. For $\delta < 2\gamma(1-\gamma)$ BC may be reduced to (γ, δ)-UNC. Finally, 1–2 OT may be reduced to (γ, δ)-UNC if $\alpha^3\beta^3(1-\zeta(1-\alpha)) > \frac{0.775+\epsilon(\delta)}{1-\epsilon(\delta)}$, where $\epsilon(\delta) = \frac{\delta^2}{\delta^2+(1-\delta)^2}$, $\alpha = \frac{1-\delta-\gamma}{1-2\gamma}$,$\beta = \frac{1-\gamma}{1-\delta}$, and $\zeta = \frac{1-\gamma}{\delta}$.*

1.2 Techniques Used

All of our impossibility results rely on a general simulation technique that allows us to leverage the result that it is impossible to implement 1–2 OT (information-theoretically) given only a clear channel.

Our upper bounds for (p, q)-WOT and (p, q, ϵ)-WOT use some reductions first used in [9]. The reduction from bit commitment to (γ, δ)-UNC is based on the interactive hashing technique of [16]. The precise hashing method of [16] doesn't work for our application; instead we use families of universal hash functions [10]. Hash functions are ubiquitous in cryptography; two classic results on achieving privacy with universal hash functions are [13] and [1]. For the specifics of our analysis we use bounds on their behaviour implied by the results of [17].

Guide to the Paper In Section 2 we give the general scenario for weak generic transfer. In Section 3 we show impossibility results for reducing 1–2 OT and bit commitment to (p, q)-WOT, (p, q, ϵ)-WOT and (γ, δ)-UNC. In Section 4 we give reductions from 1–2 OT (and hence bit commitment) to (p, q)-WOT and (p, q, ϵ)-WOT. In Section 5 we give a reduction from bit commitment to (γ, δ)-UNC. In Section 6 we in give a reduction from 1–2 OT to (γ, δ)-UNC.

2 The General Scenario: Weak Generic Transfer

In order to show more clearly the basic properties we study, we start with a general scenario that includes as special cases those primitives we later study in greater detail.

First, we describe a specification for standard two party primitives, and then show how to augment these specifications to model interesting deviations from the ideal behaviour of the protocol.

Initially, our scenario includes two players A, B that start with private inputs x_A, x_B, respectively chosen from domains X_A and X_B (the precise nature of these domains has no impact on the following discussion). A specification for a standard two-party primitive is a function output that maps (x_A, x_B) to a probability measure D on $Y_A \times Y_B$. When the primitive is executed with inputs (x_A, x_B), $D = \mathsf{output}(x_A, x_B)$ is computed, (y_A, y_B) is chosen according to D, y_A is sent to A and y_B is sent to B. This framework is powerful enough to model primitives such as OT, 1–2 OT and binary symmetric channels.

To model the possibility that a primitive might inadvertently leak information, we modify D to be a distribution $(Y_A \times Z_A) \times (Y_B \times Z_B)$; $((y_A, z_A), (y_B, z_B))$ are sampled from D. If A is honest, then A receives only y_A, but if A is corrupt, it also receives z_A; B behaves symmetrically.

We can model passive ("honest but curious") adversaries by simply specifying that an adversary $Q \in \{A, B\}$ follows the protocol, ignoring z_Q, and then later learns what it can from the values of z_Q that is obtained. An active adversary may immediately use this extra information in planning its next move.

We have modeled deviations from privacy; we now model deviations in behaviour. Instead of having a single function output, we have a (possibly infinite) set S of functions $\{\mathsf{output}\}$, which contains a "default" output_0. When the protocol is executed, the adversary has the option of choosing output from S; the protocol then behaves as before. If there is no adversary, then the default output_0 is used. We say that S specifies a *Weak Generic Transfer* (WGT). We will assume throughout that A and B have access to the WGT as a black box and can execute it independently as many times as they wish.

A WGT may consist of a protocol where for instance a noisy channel is used several times, and the protocol instructs one player to send the same bit each time. An active cheater may choose not to do so, and so he may behave in a way that is not consistent with any legal input value; in this case we say that he inputs "?." We cannot require in general that a WGT prevents such behaviour: this would require that the cheater was committed to his input, and it is not clear a priori that a WGT implies bit commitment (indeed some WGT's don't, as we shall see). The best we can ask for in case of active cheating is therefore that the following is satisfied:

- For any active cheating strategy followed by A (B), there exists an input value x such that A (B) learns nothing more than what is implied by the view of a passively cheating player with input x.

If this is satisfied, we say that the WGT is *secure against active cheating* (but note that the only security we ask for here is that active cheating does not give any advantage over passive cheating).

It should be clear that (γ, δ)-UNC, (p, q)-WOT and (p, q, ϵ)-WOT are special cases of WGT. Note that only for the case of (γ, δ)-UNC may an adversary choose between more than one output distribution function.

3 Impossibility Results

The basic question we can ask is now: given a WGT, can we build OT or BC based on it? It is easy to characterize a class of WGT's where the answer is no. We first consider the case where there is only noiseless communication between A and B, and consider any interactive protocol between them, of the following form:

- A starts with input x_A, B starts with input x_B.
- The players exchange a finite number of messages, and the protocol specifies at each stage a probabilistic algorithm for computing the next message, given the input, and all message and random coins seen so far.
- The *view* of a player ($View_A$/ $View_B$) is as usual defined to be the player's input and random coins, along with all messages received from the other player. At the end, A and B compute their results, y_A and y_B from their views by some function, i.e., $y_Q = f_Q(View_Q)$.

It is clear that any such protocol can be seen as a WGT by letting $z_A = View_A$ and $z_B = View_B$; this method for producing y_A, y_B, z_A, z_B from x_A, x_B defines a probability measure $D(x_A, x_B)$, and we define just one output distribution function output which always return $D(x_A, x_B)$. Honest players will ignore anything except for the result specified (Y_A, Y_B), but a passively cheating player may use its entire view to compute extra information.

It is well known (and easy to see) that in a two-player scenario with only noiseless communication, OT and BC with information theoretic security is not possible, even if only passive cheating is assumed, and players are allowed infinite computing power. Hence, OT and BC are not reducible to the above WGT. We call such a WGT *trivial*.

We now show how to "implement" (p, q, ϵ)-WOT in this manner, where $2\epsilon = 1 - p - q$. Consider the following protocol, in which A has input (b_0, b_1) and B has input c.

Protocol $\mathsf{SimNoisyWOT}[p, q]((b_0, b_1), c)$

1. With probability q, A announces (b_0, b_1), B computes b_c and the protocol terminates; otherwise, A announces "pass".
2. If A passes, then with probability $p/(1-q)$, B sends c to A who replies with b_c; otherwise, B chooses b_c at random.

By a straightforward case analysis, B learns both b_0 and b_1 with probability q, A learns c with probability p and B receives an incorrect value of b_c with probability $(1 - p - q)/2 = \epsilon$. Aside from easily simulated messages, such as "pass", the view of each side corresponds to the view it could obtain from an actual run of a (p, q, ϵ)-WOT primitive. Now, suppose we had an 1–2 OT protocol based on a (p, q, ϵ)-WOT primitive. If we replaced each execution of the (p, q, ϵ)-WOT by an execution of the $\mathsf{SimNoisyWOT}[p, q]$ primitive, then the view of each party, taken in isolation, would be unchanged. Since the security of 1–2 OT (at least against passive adversaries) is defined solely in terms of properties of Player

A's view and properties of Player B's view, the resulting protocol would remain secure against passive adversaries. This would give a "mental 1–2 OT" protocol, information-theoretically secure against passive adversaries, a contradiction. Similarly, there is no information-theoretically secure (against both parties) mental bit commitment protocol, even if both parties are guaranteed to follow the Commit protocol; we can in a very similar way derive a contradiction.

The above argument implies the following lemma.

Lemma 4. *There is no reduction from 1–2 OT or BC to (p, q, ϵ)-WOT when $p + q + 2\epsilon \geq 1$, even if only security against passive adversaries is required.*

Remark: The simulation argument was for $p+q+2\epsilon = 1$. If $p+q+2\epsilon > 1$, choose $\epsilon' = (1-p-q)/2 < \epsilon$; the impossibility argument works for (p, q, ϵ')-WOT. Note that a (p, q, ϵ')-WOT primitive also meets the requirements of a (p, q, ϵ)-WOT primitive, since its error rate cannot be higher than ϵ, so a protocol that works for (p, q, ϵ)-WOT must work for (p, q, ϵ')-WOT as well.

We now consider the case of the noisy channel. Consider the following purely mental protocol, in which A has input b.

Protocol SimUNC$[\gamma](b)$

1. A and B pick b_A and b_B respectively, such that $Pr(b_A = 1) = Pr(b_B = 1) = \gamma$.
2. A sends $b' = b \oplus b_A$ to B. B computes $b^* = b' \oplus b_B$, denoting b^* as the received bit, while no output is defined for A.

Consider a WGT which between honest players A and B is a BSC with noise level δ, but where if A or B cheats passively, then some extra information is available and allows to reduce the noise level to γ, seen from the cheater's point of view. Let us call this a (γ, δ)-PassiveUNC. It is similar to but not the same as a (γ, δ)-UNC. It immediately follows from the above protocol that a (γ, δ)-PassiveUNC is trivial if $\delta = 2\gamma(1-\gamma)$, and in fact in general if $\delta \geq 2\gamma(1-\gamma)$. And so there is no reduction of 1–2 OT or BC to (γ, δ)-PassiveUNC in this case, not even if only passive security is required.

Now, suppose we have a reduction from 1–2 OT to a (γ, δ)-UNC, where $\delta = 2\gamma(1-\gamma)$, one secure against active attacks. We compare the following two cases: In case 1 the reduction is attacked by an adversary using the following active cheating strategy for a player $Q \in \{A, B\}$: Q sets the noise level for the UNC to be γ always, and then does the following: Whenever Q is supposed to send a bit through the channel, Q first flips it with probability γ and then sends it. Similarly, whenever Q receives a bit from the channel, Q flips it with probability γ and acts as if that was the bit actually received. In any other cases, Q follows the algorithm specified by the reduction. Case 2: we execute the algorithm of the reduction substituting the (γ, δ)-UNC by a (γ, δ)-PassiveUNC, and the adversary executes a passive attack.

There is no difference between the cases from the honest player's point of view. Observe that in case 1, the adversary following the strategy for Q knows as much about every bit sent and received by his opponent as a passive adversary

knows in case 2. So since the reduction is secure in case 1, it must be secure in case 2, and we have a contradiction. Essentially the same argument works for bit commitment. So we have proved:

Lemma 5. *There is no reduction from 1–2 OT or BC to* (γ, δ)*-UNC when* $\delta \geq 2\gamma(1-\gamma)$.

This motivates the following interesting and open question: *Which non-trivial cryptographic primitives (if any) can be implemented based on a WGT assuming only that it is non trivial?*

4 Reducing 1–2 OT to (p, q)-WOT and (p, q, ϵ)-WOT

We now look at the possibility of building a 1–2 OT or commitments from a WOT. A reduction that accomplishes such a task can be thought of as a program that gets the noise levels of a UNC or the error probabilities of a WOT and a security parameter value k as input and then instructs at each point in time one of the players to either send a message in the clear to the other player, or send a bit through the noisy channel. Any information known to the player at the time can be used, together with any number of random bits, to compute the next message to send. We make no assumption on the amount of computation required.

4.1 Preliminaries

For a reduction of 1–2 OT to UNC, let $I_c(k, \delta, \gamma), I_{b_c}(k, \delta, \gamma), I_{b_{1-c}}(k, \delta, \gamma)$ be the expected information that the sender obtains about c, the receiver obtains about b_c, and the receiver obtains about b_{1-c} respectively. We will say that *the reduction works* for values δ, γ, if $\lim_{k\to\infty} I_c(k, \delta, \gamma) = \lim_{k\to\infty} I_{b_{1-c}}(k, \delta, \gamma) = 0$ and $\lim_{k\to\infty} I_{b_c}(k, \delta, \gamma) = 1$ For a reduction of 1–2 OT to (p, q)-WOT, we use the same definitions, but with (δ, γ) replaced by (p, q).

For a reduction of bit commitment to UNC, let $I_b(k, \delta, \gamma)$ be the expected information the receiver obtains about b in the *Commit* protocol, and let $p(k, \delta, \gamma)$ be the probability that the binding condition fails. We will say that *the reduction works* for values δ and γ, if $\lim_{k\to\infty} I_b(k, \delta, \gamma) = \lim_{k\to\infty} p(k, \delta, \gamma) = 0$

We refer to [3] for a more sophisticated definition of 1–2 OT; our protocols meet this definition as well.

The set of pairs for which a reduction works will be called the *range* of the reduction. We will say that a reduction *works efficiently* in a point in its range, if the required convergence in k is exponential, and that the number of calls to the WOT or UNC is polynomial in k. This is usually required for a reduction to be useful in practice, but note that our impossibility results hold even if efficiency is not required.

4.2 Some Useful Reductions

We use the following two known reductions for basing 1–2 OT on (p, q)-WOT. The first is designed to reduce the chance the sender (A) learns too much, while the second is targeted against the chance of the receiver (B). Both reductions are assumed to be given as a black-box a protocol $\mathcal{W}$ implementing (p, q)-WOT and work with security parameter k. S-Reduce is taken from [9], while R-Reduce is more or less folklore.

Protocol S-Reduce$(k, \mathcal{W})$

1. Let (b_0, b_1) resp. c be the input of the sender, resp. the receiver.
2. $\mathcal{W}$ is executed k times, with inputs $(b_{0i}, b_{1i}), i = 1..k$ from the sender and $c_i, i = 1..k$ from the receiver. Here, the b_{0i}'s are uniformly chosen, such that $b_0 = \oplus_{i=1}^{k} b_{0i}$, $b_{1i} = b_{0i} \oplus b_0 \oplus b_1$ and the c_i's are uniformly chosen such that $c = \oplus_{i=1}^{k} c_i$.
3. The receiver computes his output bit as the xor of all bits received in the k executions of $\mathcal{W}$.

Protocol R-Reduce$(k, \mathcal{W})$

1. Let (b_0, b_1) resp. c be the input of the sender, resp. the receiver.
2. $\mathcal{W}$ is executed k times, with inputs $(b_{0i}, b_{1i}), i = 1..k$ from the sender and $c_i, i = 1..k$ from the receiver. Here, $c_i = c$, $b_{01} \oplus ... \oplus b_{0k} = b_0$ and $b_{11} \oplus ... \oplus b_{1k} = b_1$
3. The receiver computes the XOR of all bits received.

Lemma 6. *When given k and a (p, q)-WOT $\mathcal{W}$ as input,* S-Reduce$(k, \mathcal{W})$ *implements a $(p^k, 1 - (1 - q)^k)$-WOT, and* R-Reduce$(k, \mathcal{W})$ *implements a $(1 - (1 - p)^k, q^k)$-WOT protocol. Both protocol produce a WOT secure against active cheating if the given WOT has this property.*

Proof (sketch). First, it follows by inspection that the protocols allow the players to compute the correct output. As for the error probabilities, note that for S-Reduce a bad sender will learn c iff he learns all c_i's, which happens with probability p^k. On the other hand, a bad receiver can learn both b_0 and b_1 if he learns just one pair (b_{0i}, b_{1i}), and this happens with probability $1 - (1 - q)^k$. The case of R-Reduce is similar, but with the chances of sender and receiver reversed. The last claim follows easily: In S-Reduce, security of W means that none of the parties can gain anything from inputing ? to W. And if indeed no ? is input to any W instance, R always behaves consistently with some input c, namely the value $c = \oplus_{i=1}^{k} c_i$. S can behave inconsistently by choosing bad values for his bits, but this will not give him more information on c. The case of R-Reduce is similar. □

4.3 A Reduction to (p, q)-WOT

Lemma 7 shows that the lower bound given by Lemma 4 is tight when $\epsilon = 0$. Lemmata 4 and 7 imply Theorem 1.

Lemma 7. *There exists a reduction for building 1–2 OT from a (p, q)-WOT, the range of which is $\{p, q |\ p + q < 1\}$. It works efficiently for all points in its range.*

Proof. Suppose we start with a (p, q)-WOT $\mathcal{W}$, and apply first R-Reduce$(t, \mathcal{W})$ and then S-Reduce$(t', \mathcal{W})$. We call this combination RS-Reduce. It follows easily that RS-Reduce produces a $((1-(1-p)^t)^{t'}, 1-(1-q^t)^{t'})$-WOT. Of course, we can also apply S-Reduce first, and obtain a $(1-(1-p^t)^{t'}, (1-(1-q)^t)^{t'})$-WOT. We call this combination SR-Reduce.

The strategy for our reduction is to apply repeatedly SR-Reduce or RS-Reduce, in order to reduce as quickly as possible the sum of the error probabilities. When given errors (p, q), we apply RS-Reduce if $p \leq q$, and SR-Reduce otherwise. This will be called one step in the reduction.

To analyse the effect of one step, define $x = q, y = 1 - p$ when $p \leq q$, and $x = p, y = 1 - q$ otherwise. It follows that the difference between the sum of the errors before and after the transformation is

$$f(t, t', x, y) = (1-y^t)^{t'} + 1 - (1-x^t)^{t'} - (1-y+x) = (1-y^t)^{t'} + y - ((1-x^t)^{t'} + x)$$

The constraints we have on p, q imply that $1/2 < y < 1$ and $1 - y \leq x < y$. And we see that the progress we make is the difference between the values of the function $g_{t,t'}(z) = (1-z^t)^{t'} + z$ evaluated at points x and y. The trick is now to choose, given x and y, values of t and t' such that the above difference becomes numerically "large". Note that since we are subtracting the sum before the step from the sum after, the difference is hopefully negative.

In any situation where the error probability sum before a step is greater than 0.2, one of the following three cases apply:

- $y \leq 0.8$**:** This is a case where the smallest of p, q is at least 0.2, so p, q are both "large." In this case, we choose $t = t' = 2$. By direct inspection of $g_{2,2}(x)$, one finds that for any x, y obeying the restrictions, $(g_{2,2}(y) - g_{2,2}(x))/(y - x) \leq -0.1$. Since $y - x = 1 - (p + q)$, this shows that taking one step in this case multiplies the distance from $p + q$ to 1 by a factor of at least 1.1.
- $y > 0.8$**,** $x > .4$**:** In this case, $p + q$ is also "large," but this time because one probability is small and the other is large. In this case, we choose $t = 2$ and $t' = 1$. Again, by direct inspection, one can verify that $(g_{2,1}(y) - g_{2,1}(x))/(y - x) \leq -0.2$. By the same argument as before, we see that in this case, the distance from $p + q$ to 1 is multiplied by at least 1.2 by taking one step.
- $y > 0.8$**,** $x \leq .4$**:** In this case, both p and q and hence $p + q$ are "small." We choose $t = t' = 2$. Observe that for the large y, $g_{2,2}(y)$ approaches y as y approaches 1, while for the small x, $g_{2,2}(x)$ approaches $1 + x$ as x approaches 0. As a result, $(g_{2,2}(y) - g_{2,2}(x))/(1 - y + x) \simeq -1$ for small x and large y, and

is in fact less than -0.2 for the range specified. However, $1 - y + x = p + q$, so we see that in this case, taking one step reduces $p + q$ to at most 0.8 of its previous value.

As soon as we have an error probability sum which is at most 0.2, we start doing steps where we always have $t = t' = 4$. In this case one finds that if the error sum was s before a step it is at most s^2 afterwards.

The overall strategy is now as follows: we first do whatever number of steps is necessary to bring the error probability sum below 0.2. We then do $\log_2(k)$ steps, where k is the security parameter. It follows from the above that the resulting error probability sum is exponentially small in k, at most 0.2^k. The number of calls we make to the WOT is exponential in the number of steps, but since we only take a logarithmic number of steps, the total number of calls is polynomial in k. □

The above argument only considers p, q as being constants. However, even if we have a case where $p + q$ is a function of some parameter n and converges polynomially to 1 in n, e.g. $p(n) + q(n) = 1 - 1/n$, the reduction in the proof still works in time polynomial in n.

4.4 A Reduction to (p, q, ϵ)-WOT

Lemma 4 shows that no reduction of 1-2-OT to (p, q, ϵ)-WOT exists if $p+q+2\epsilon \geq 1$ and this, even in the case of passive adversaries. We show that if $p + q + 2\epsilon < 0.45$, such a reduction does exist. We adapt SR-Reduce to deal with transmission errors. We then characterize triplets (p, q, ϵ) for which 1–2 OT is reducible to (p, q, ϵ)-WOT. The reduction we consider assumes only passive adversaries.

The following error detection phase accepts parameter $l > 0$ and, given a (p, q, ϵ)-WOT $\mathcal{W}$, produces a (p', q', ϵ')-WOT $\mathcal{W}'$ such that $\epsilon' < \epsilon$. As usual b_0 and b_1 denote the two bits to be transmitted and c is the selection bit.

Protocol $\mathsf{ErRed}(l, \mathcal{W})$

1. A chooses $q_0, q_1 \in_R \{0, 1\}$ and B chooses $s \in_R \{0, 1\}$,
2. A sends l times the bits (q_0, q_1) through the (p, q, ϵ)-WOT $\mathcal{W}$ and B selects the bit q_s l times,
3. If B did not receive the same bits $\hat{q}_s$ l times then A and B go to Step 1.
4. B announces $y = 0$ if $s = c$ and $y = 1$ otherwise.
5. A announces r_0 and r_1 such that $b_y = r_0 \oplus q_0$ and $b_{1-y} = r_1 \oplus q_1$, allowing B to compute $b_c = \hat{q}_s \oplus r_s$.

We are now ready to describe a reduction of 1–2 OT to (p, q, ϵ)-WOT which basically inserts calls to ErRed into SR-Reduce (and RS-Reduce). Given positive integers l_0, k, l_1, k', l_2 and a (p, q, ϵ)-WOT $\mathcal{W}_0$, protocol SR_ε produces a new (p', q', ϵ')-WOT $\mathcal{W}$:

Protocol $\mathsf{SR}_\varepsilon(l_0, k, l_1, k', l_2, \mathcal{W}_0)$

- $\mathcal{W} \leftarrow \mathsf{ErRed}(l_2, \mathsf{R\text{-}Reduce}(k', \mathsf{ErRed}(l_1, \mathsf{S\text{-}Reduce}(k, \mathsf{ErRed}(l_0, \mathcal{W}_0)))))$.

$\mathsf{RS}_\varepsilon(l_0, k, l_1, k', l_2)$ is defined the same way except the calls to S-Reduce and R-Reduce are swapped. Similarly to Lemma 6, one can characterize exactly the transformation taking place in a call to $\mathsf{SR}_\varepsilon(l_0, k, l, k', l')$ for any parameters l_0, k, l, k', and l'. In particular, $\mathsf{SR}_\varepsilon(l_0, k, l, k', l')$ transforms a (p, q, ϵ)-WOT into a (p', q', ϵ')-WOT where $p' = 1-(1-(1-(1-p)^{l_0})^k)^{l_1 \cdot k' \cdot l_2}$, $q' = 1-(1-(1-(1-q)^{l_0 \cdot k \cdot l_1})^{k'})^{l_2}$, and ϵ' is of similar but slightly more complicated form. A brute force analysis, using linear programming, shows that SR_ε can be tuned to work at 45% the optimum (the sketch of the proof can be found in [11]).

Lemma 8. *1–2 OT can be implemented given any (p, q, ϵ)-WOT that satisfies $p + q + 2\epsilon \leq 0.45$.*

The above bound is not tight especially whenever one of $p + q$ and ϵ is small. In particular, SR_ε works for all $(p, q, 0)$ such that $p + q < 1$ and for all $(0, 0, \epsilon)$ such that $\epsilon < \frac{1}{2}$. A natural question arises: Is it possible to use a different method for choosing parameters l_0, k, l_1, k' and l_2 such that reduction SR_ε works also for $p + q + 2\epsilon \gg 0.45$? The following lemma suggests that if one wants to get closer to the bound $p + q + 2\epsilon = 1$, one has to find a different reduction.

Lemma 9. *There exists triplets (p, q, ϵ) that satisfy $p + q + 2\epsilon \leq 0.70$ such that SR_ε and RS_ε does not work for any value of parameters l_0, k, l_1, k' and l_2.*

Proof (sketch). Let $p = q = 0.2$ and $\epsilon = 0.15$ be the parameters of a WOT that satisfies $p + q + 2\epsilon = 0.7$. It can be shown that whatever the parameters l_0, k, l_1, k' and l_2 are, the reduction always generates an intermediary simulatable triplet. □

Lemma 9 suggests that introducing noise in a WOT might lead to a primitive that is strictly weaker than 1–2 OT even for non-simulatable but noisy WOT. However, the gap between the bounds could be narrowed down by finding a better simulation and/or a new reduction. It is unknown to us if such a gap necessarily exists.

5 Reducing Bit Commitment to (γ, δ)-UNC

5.1 Preliminaries

Our commitment protocol makes extensive use of t-universal hash functions, first introduced in [10]; we use the following slightly stronger notion that has become more or less standard. Given a domain D and a range R, a t-universal family of hash functions is a distribution $\mathcal{H}$ on a set of functions $\{h_i\}$ such that for any distinct $X_1, \ldots, X_t \in D$, if h is chosen according to $\mathcal{H}$, the induced distribution on $(h(X_1), \ldots, h(X_t))$ is uniform over R^t. For our application, $D = \{0,1\}^k$, $R = \{0,1\}^l$, for some k, l. For any k and l, there exists a t-universal family of hash functions whose functions may be represented using $poly(k, t)$ bits, and for which the operations of sampling h from the distribution and computing $h(X)$ may be performed in $poly(k, t)$ time. Hence, we speak of one party "sending" a function, abstracting all representational details.

Given two bit-sequences X and X', let $d(X, X')$ denote their Hamming distance, i.e., the number of places where they differ. We use distance as shorthand for Hamming distance.

There is a huge body of literature on universal hash functions and their use in cryptography. Despite superficial differences, our method is motivated by that of [16].

5.2 What We Need To Achieve

Note that it suffices to produce a protocol for committing to $x = r$ for a random bit r; as a standard trick one can commit to $y = b$ by revealing $b' = b \oplus r$ and defining $y = x \oplus b'$. For the rest of the discussion, we analyze the case of committing to random values. We also allow the receiver to reject even though the commiter followed the protocol, but only with probability negligible in the security parameter, k.

5.3 The Protocol

On a high level, our (weak) commitment protocol consists of the following steps. First, C sends string X over the noisy channel to C. R queries C about the value of $h_i(X)$ for $i = 1, 2$. Finally, C chooses a hash function h and designates $h(X)$ as the random committed value. To reveal a bit, C sends X to R. R accepts if X is close to the received value and is consistent with the queried hash values.

Protocol $\mathsf{Commit}(\gamma, \delta, k)$

Define d_0 by $\gamma(1-d_0)+(1-\gamma)d_0 = \delta$ and let $d_1 = (d_0+\gamma)/2$, $d = (d_1+\gamma)/2$ and $\ell = \lfloor \lg(\sum_{i=1}^{\lfloor dk \rfloor} \binom{k}{i}) \rfloor$. We let $d^* = (d_1 + d)/2$, and define ℓ^* as ℓ, where we replace dk by d^*k. Note that, by a standard argument, it follows that $\ell - \ell^* > ck$ for some constant c as k grows sufficiently large.
Let $\mathcal{H}, \mathcal{H}_1$ and $\mathcal{H}_2$ be canonical $64k$-universal families of hash functions from $\{0,1\}^k$ to $\{0,1\}, \{0,1\}^{\ell^*}$ and $\{0,1\}^{\ell-\ell^*}$, respectively.

1. C uniformly chooses $X = x_1, \ldots, x_k \in \{0,1\}^k$ and sends X to R over the (γ, δ) channel. Denote by $X' = x'_1, \ldots, x'_k$ the string received by R.
2. For $i = 1$ to 2
   R chooses $h_i \leftarrow \mathcal{H}_i$ and sends h_i to C.
   C sends $y_i = h_i(X)$ to R.
3. C chooses $h \leftarrow \mathcal{H}$ and sends h to C. The committed bit is defined as $h(X)$.

Protocol $\mathsf{Open}(\gamma, \delta, k)$

Let $X, X', y_1, y_2, h, h_1, h_2, d_0, d_1$ and d be as in the execution of Commit for the bit to be opened, and let $\delta' = \gamma(1 - d_1) + (1 - \gamma)d_1$.

1. C sends X to R.
2. R rejects if $y_i \neq h_i(X)$ for any i or if $d(X, X') \geq \delta' k$ locations. Otherwise, R accepts the committed value of $h(X)$.

5.4 Analysis of the Protocols

We first observe that the protocol behaves correctly if both parties are honest. For the rest of this discussion, "negligible" means smaller than $1/k^c$, for any c, as k grows sufficiently large and "almost always" means with probability negligibly close to 1. Proofs of some of the Lemmata below are sketched in the appendix.

Lemma 10. *If C and R both correctly execute* Commit *and* Open*, then R accepts the value $r = h(X)$ almost always (where h and X are as generated by C during* Commit*.*

We next show that the commiter has only a negligible probability of breaking the commitment scheme.

Lemma 11. *Regardless of C's strategy for generating $X, (h_1, y_1), (h_2, y_2)$ during* Commit*, there will almost certainly be at most one string, denoted X^*, that C can send R with a nonnegligible probability of acceptance.*

Hence, C is committed to $h(X^*)$. Note that C can ensure that $h(X^*)$ is not random, but this does not constitute a break of the commitment scheme. In the reduction from a standard commitment protocol to a random-bit commitment protocol, C and only C benefits from the randomness of the committed bit.

Before proving the lemma, we first define a set of viable X^* that C can reasonably send during the Open protocol.

Definition 12. *Given $X, (h_1, y_1), (h_2, y_2)$. We say that X^* is* viable *if it differs from X in at most d^*k places and $y_i = h_i(X^*)$ for $i = 1, 2$.*

Proposition 13. *If X^* is not viable, then C will accept X^* with negligible probability, where the probability is taken over the behavior of the noisy channel.*

Proof (of Lemma 11). (Sketch) We can view the process of generating (h_i, y_i) as progessively constraining and shrinking the viable set S. Initially, the viable set S consists of those strings whose distance from X is at most d^*k .

We use the following result by Rompel [17]

Lemma 14. *([17]) Let $X_1, ..., X_n$ be a set of t-wise independent random variables taking 0/1 values. Let $X = \sum_i X_i$, and $\mu = E(X)$. Then for any $A > 0$, we have that $Pr(|X - \mu| > A) < O((\frac{t\mu + t^2}{A^2})^{t/2})$.*

Fix any string $y \in \{0,1\}^{\ell^*}$, and define X_i as $X_i = 1$ iff the i'th viable string is mapped to y by h_1; $X_i = 0$ otherwise. Then clearly, $\mu = 1$. We apply the above lemma with $t = 4k$ and $A = t^2$, and we say that y is bad if its preimage under h_1 has more than t^3 viable strings in it. The lemma can be used because the X_i's by construction are $64k > t$-wise independent. It immediately implies that the probability that y is bad is at most $2^{-t/2}$. The probability that ANY y is bad is at most $2^{\ell^*} \leq 2^k$ times larger, so since we have chosen $t = 4k$, even this

last probability becomes exponentially small (in k). So except with exponentially small probability, at most $t^3 = 64k$ viable strings remain. The final constraint added is the value of $h_2(X)$. Since h_2 is $64k$-universal and $\ell - \ell^* > ck$, we can view this constraints as assigning at least ck random bits to each string $X^* \in S$. In order for two strings $X_1^*, X_2^* \in S$ to both remain viable, they must both receive the same bit sequence; the probability of this occurring for any such pair is negligible.

Finally, we show that after Commit, R can predict r with only a small advantage.

Lemma 15. *At the conclusion of* Commit*, the expected amount of information R holds about $h(X)$ is exponentially small in k.*

6 Reducing 1–2 OT to (γ, δ)-UNC

We first reduce 1–2 OT to (γ, δ)-PassiveUNC by a reduction secure against passive adversaries. The reduction is a straightforward adaptation of a reduction of Crépeau and Kilian [9] that builds 1–2 OT from a BSC. The same procedure is then shown to reduce 1–2 OT to (γ, δ)-UNC. Bit commitments can finally be used to tolerate active adversary for the same price.

In the appendix, reduction WOTfromPassiveUNC is described. Given a (δ, γ)-PassiveUNC, it produces a $(p(\delta, \gamma), q(\delta, \gamma), \epsilon(\delta))$-WOT $\mathcal{W}$ that can be used in reductions SR_ε and RS_ε. Using lemma 8, 1–2 OT can be obtained from any (δ, γ)-PassiveUNC that satisfies $p(\delta, \gamma) + q(\delta, \gamma) + 2\epsilon(\delta) \leq 0.45$. Unlike the bit commitment case, we were not able to show that as soon as the unfairness of the PassiveUNC is not simulatable then 1–2 OT is possible. Nevertheless, the next lemma gives a partial answer leaving a "grey" area of values for γ, δ where neither the impossibility result, nor our reduction applies. Due to space limitations, we refer the reader to [11] for the proof of next lemma.

Lemma 16. *There exists a reduction secure against passive cheating of 1–2 OT to (γ, δ)-PassiveUNC such that $\alpha^3\beta^3(1 - \zeta(1 - \alpha)) > \frac{0.775+\epsilon(\delta)}{1-\epsilon(\delta)}$ where $\epsilon(\delta) = \frac{\delta^2}{\delta^2+(1-\delta)^2}$, $\alpha = \frac{1-\delta-\gamma}{1-2\gamma}$,$\beta = \frac{1-\gamma}{1-\delta}$, and $\zeta = \frac{1-\gamma}{\delta}$.*

To give a numerical example, when $\delta = .075$, one can reduce 1–2 OT to (γ, δ)-PassiveUNC for $\gamma \approx .06$; no such reduction is possible for $\gamma < .039$.

The reduction is also secure when a (γ, δ)-UNC is used instead of a (γ, δ)-PassiveUNC. The following straightforward lemma establishes this fact.

Lemma 17. *Any secure reduction to (γ, δ)-PassiveUNC against passive adversaries is also a secure reduction to (γ, δ)-UNC given it produces the correct output for any noise level in the interval $[\gamma \ldots \delta]$.*

Intuitively, the adversary maximizes the information he gets by reducing the noise level of a (δ, γ)-UNC to γ. In this case, the information obtained is the same

as if a (γ, δ)-PassiveUNC was used. If in addition, the reduction to PassiveUNC produces the right output for any noise level in $[\gamma \dots \delta]$ then a (γ, δ)-UNC can be used instead. It is easy to verify that WOTfromPassiveUNC is a reduction working for any noise level in the interval $[\gamma \dots \delta]$.

Any reduction of 1–2 OT to UNC that is secure against passive cheating can handle the case of active cheating as well by proceeding along the same lines as [5]. To briefly sketch the construction, we first note that once A and B get a bit commitment scheme, they can prove in ZK that they follow the protocol honestly [12]. They can also use the bit commitment scheme in a cut and choose manner for showing that the bits sent and received through the channel are used according the protocol description. The result being that from an UNC and a bit commitment scheme, a committed UNC is built. From this point, general ZK techniques are used to make sure that no active cheating occurs. Using lemma 17 and the above argument leads to the following corollary:

Corollary 18. *Lemma 16 applies against active adversaries for both the* (γ, δ)*-PassiveUNC and the* (γ, δ)*-UNC.*

References

1. C. H. Bennett, G. Brassard, C. Crépeau, and U. M. Maurer. "Generalized privacy amplification". *IEEE Trans. Info. Theory*, vol. 41, no.6, pp. 1915–1923, 1995.
2. G. Brassard and C. Crépeau. "Oblivious transfers and privacy amplification". *EUROCRYPT '97*, LNCS series, vol. 1223, pp. 334-347, 1997.
3. G. Brassard, C. Crépeau, and M. Sántha."Oblivious Transfer and Intersecting Codes". *IEEE Trans. Info. Theory*, vol. 42, No. 6, pp.1769–1780, 1996.
4. C. Cachin. "On the Foundations of Oblivious Transfer", *EUROCRYPT '98*, LNCS series, vol. 1403, pp. 361-374.
5. C. Crépeau. "Verifiable disclosure of secrets and applications". *EUROCRYPT '89*, LNCS series, vol. 434, pp.150-154, 1998.
6. C. Crépeau. "Efficient Cryptographic Protocols based on Noisy Channels", *EUROCRYPT '97*, LNCS series, vol.1233, pp.306-317.
7. C. Crépeau. Private communication, 1998.
8. C. Crépeau."Equivalence between two flavours of oblivious transfer",*CRYPTO '87*, LNCS series, pp.350-354, 1987.
9. C. Crépeau and J. Kilian."Achieving Oblivious Transfer using Weakened Security Assumptions", *FOCS 88*, pp.42-52, 1988.
10. L. Carter and M. Wegman. "Universal Classes of Hash Functions". *JCSS*, 18, pp. 143–154, 1979.
11. I. Damgård, J. Kilian, and L. Salvail, "On the (Im)possibility of basing Oblivious Transfer and Bit Commitment on Weakened Security Assumptions", BRICS report available from `http://www.brics.dk/Publications/`, 1998.
12. O. Goldreich, S. Micali, and A. Wigderson, "Proofs That Yield Nothing but Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems", J. Assoc. Comput. Mach., vol. 38, pp. 691–729, 1991.
13. J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. "Construction of a pseudo-random generator from any one-way function". Technical Report TR-91-068, International Computer Science Institute, Berkeley, CA, 1991.

14. J. Kilian. "A general completeness theorems for 2-party games". *STOC '91*, pp. 553–560, 1991.
15. U.M. Maurer."Secret Key Agreement by Public Discussion from Common Information", *IEEE Trans. Info. Theory*, vol. 39, p.733-742, 1993.
16. M. Naor, R. Ostrovsky, R. Venkatesan, and M. Yung. "Perfect zero-knowledge arguments for NP using any one-way permutation". *Journal of Cryptology*, vol. 11, 1998.
17. J. Rompel. *Techniques for Computing with Low-Independence Randomness*. PhD-thesis, MIT, 1990.

A WOT from PassiveUNC and Proofs from Section 5

In this appendix, we first give the reduction of WOT to PassiveUNC and second, we provide the proofs from section 5. Protocol $\mathsf{WOTfromPassiveUNC}(b_0, b_1)(c)$

1. A picks $x, y \in_R \{0,1\}$,
2. A sends (xx, yy) through the PassiveUNC(γ, δ) and B receives $(\hat{x}\hat{x}', \hat{y}\hat{y}')$,
3. If B receives $(\hat{x} \oplus \hat{x}', \hat{y} \oplus \hat{y}') \notin \{(0,1),(1,0)\}$ then they go to step 1.
4. B announces w such that
   - $w = 0$ if $((\hat{x} \oplus \hat{x}' = 0) \wedge (c = 0)) \vee ((\hat{y} \oplus \hat{y}' = 0) \wedge (c = 1))$
   - $w = 1$ if $((\hat{x} \oplus \hat{x}' = 0) \wedge (c = 1)) \vee ((\hat{y} \oplus \hat{y}' = 0) \wedge (c = 0))$
5. A announces
   - $(a, b) = (x \oplus b_0, y \oplus b_1)$ if $w = 0$,
   - $(a, b) = (y \oplus b_0, x \oplus b_1)$ if $w = 1$,
6. B computes
   - $b_0 = a \oplus \hat{x}$ if $c = 0$ and $w = 0$,
   - $b_0 = a \oplus \hat{y}$ if $c = 0$ and $w = 1$,
   - $b_1 = b \oplus \hat{y}$ if $c = 1$ and $w = 0$,
   - $b_1 = b \oplus \hat{x}$ if $c = 1$ and $w = 1$.

Proof Sketch of Lemma 10 By inspection, if R accepts it will always recover $r = h(X)$ (assuming C is good), and R will only reject if X and X' differ in at least $\delta' k$ places. Now, since $\delta < 2\gamma(1-\gamma)$, $d_0 < d_1 < \gamma$ and hence $\delta' > \delta$. However, for each i, $x'_i \neq x_i$ with independent probability at most δ, so by a standard Chernoff bound, the probability that $x'_i \neq x_i$ in $\delta' k$ is negligible in k. Note that by the universality of $\mathcal{H}$, r is distributed uniformly over $\{0,1\}$. □

Proof Sketch of Proposition 13 Clearly, R will reject if $y_i \neq h_i(X^*)$. Suppose that X^* differs from X in $d^* k$ places and the channel flips each bit with probability at least γ. Then X^* and X' will differ in at least $\delta^* k$ expected places, where $\delta^* = \gamma(1-d^*) + (1-\gamma)d^* > \delta'$. By a standard Chernoff bound, they will almost always differ in more than $\delta' k$ places, causing R to reject. □

Proof Sketch of Lemma 15 First, we conceptually give R the value of $d(X', X)$; this can only help R. Let set S denote those Xs of the given distance; after receiving X', each $X \in S$ is equally likely. We first observe that for some constant $c_1 > 0$, $H(X', X) - dn \geq c_1 n$ almost always; it follows that for some constant $c_2 > 0$, $|S|/2^\ell \geq 2^{c_2 k}$.

Now, after receiving X', R can obtain $h_1(X), h_2(X)$. Note that we cannot assume these functions are chosen randomly. Still, conceptually, we can view R as flipping its random coins (if it uses any) and then constructing a (quite shallow) decision tree. Each interior vertex v is labelled with a hash function h to be sent to C; the edges from v to its children correspond to the possible answers C might give (2^{ℓ^*} possibilities in the first level, $2^{\ell-\ell^*}$ in the second).

For every vertex v in the tree we define the set S_v as those $X \in S$ that are consistent with the sequence of hash functions and answers given on the path from the root vertex to v. We can view Step 2 of Commit as a traversal from the root of the tree to a leaf l.

By a simple probability argument, the probability that a given leaf l is reached is $|S_l|/|S|$, and the conditional distribution on X is uniform over S_l. Since the tree has only 2^ℓ leaves, the probability of reaching a leaf l with $|S_l| < 2^{c_2 k/2}$ is at most $2^{-c_2 k/2}$; we can safely ignore this event. On the other hand, in every case where $|S_l| \geq 2^{c_2 k/2}$, it follows immediately from the privacy amplification result in [1] that R's expected information about $h(X)$ is exponentially small.

□