Degradation and Amplification of Computational Hardness*

Shai Halevi and Tal Rabin
IBM T.J. Watson Research Center,
Hawthorne, NY USA
shaih@alum.mit.edu, talr@us.ibm.com

March 6, 2008

Abstract

What happens when you use a partially defective bit-commitment protocol to commit to the same bit many times? For example, suppose that the protocol allows the receiver to guess the committed bit with advantage ε, and that you used that protocol to commit to the same bit more than $1/\varepsilon$ times. Or suppose that you encrypted some message many times (to many people), only to discover later that the encryption scheme that you were using is partially defective, and an eavesdropper has some noticeable advantage in guessing the encrypted message from the ciphertext. Can we at least show that even after many such encryptions, the eavesdropper could not have learned the message with certainty?

In this work we take another look at amplification and degradation of computational hardness. We describe a rather generic setting where one can argue about amplification or degradation of computational hardness via sequential repetition of interactive protocols, and prove that in all the cases that we consider, it behaves as one would expect from the corresponding information theoretic bounds. In particular, for the example above we can prove that after committing to the same bit for n times, the receiver's advantage in guessing the encrypted bit is negligibly close to $1-(1-\varepsilon)^n$.

Our results for hardness amplification follow just by observing that some of the known proofs for Yao's lemmas can be easily extended also to handle interactive protocols. On the other hand, the question of hardness degradation was never considered before as far as we know, and we prove these results from scratch.

Key words. Commitment schemes, Hardness amplification and degradation, Secrecy amplification and degradation, Oblivious transfer, XOR lemmas.

1 Introduction

This work discusses the effect of running several executions of a cryptographic protocol sequentially, on the secrecy or correctness guarantees of that protocol. An illustrating example to keep in mind is a defective bit-commitment scheme, where the sender may open the commitment in two ways

*A preliminiary version appeared in the proceedings of TCC'08, LNCS 4948, pages 623–640, Springer 2008.

with probability up to δ (binding defect) and the receiver may have probability of up to $(1+\varepsilon)/2$ in guessing the sender's bit (secrecy defect). We ask how does sequential repetition of such a protocol effect ε and δ, in situations where the inputs to the various executions may be dependent.

This question is closely related to the issue of *robust combiners* for cryptographic protocols. Indeed, Damgård et al. considered in [2] just this kind of defective protocols (for both commitment and oblivious transfer), and described how a non-defective protocol can be obtained from them. Two transformations were described in [2], one running many copies of the defective protocol with the same input bit, and the other running many copies with randomly chosen inputs whose exclusive-or equals the original input bit. Damgård et al. proved that in an information-theoretic setting, if the original defects satisfy $\varepsilon + \delta < 1$ then alternating between these two transformations can reduce the secrecy and binding defects to negligible quantities. Given these results, one would like to prove the same result also in the computational setting.

To illustrate the problem with moving to the computational setting, consider using a defective bit-commitment scheme to commit twice to the same input bit. In the information theoretic setting from [2], it is clear that if the commitment scheme has secrecy defect of ε, then using it twice with the same input bit yields a secrecy defect of $1 - (1-\varepsilon)^2 = 2\varepsilon - \varepsilon^2$. In the computational setting, however, the simple hybrid argument that is commonly used to reason about "encrypting the same message many times" can only prove a bound of 2ε on the resulting defect, which is clearly too weak of a bound. (For example, one needs to show that the resulting scheme offers some secrecy, even if the original one has secrecy defect of $\frac{2}{3}$.)

In the specific context of robust combiners for commitment and oblivious-transfer, results similar to those of Damgård et al. were recently proved in the computational setting by Wullschleger [9]. Wullschleger was able to bypass the problem from above to some extent by working in the "honest-but-curious" model and considering a "randomized" variant of these primitives, where the parties execute the protocol on random bits, which are considered outputs of the protocol rather than inputs to it. These variants are known to be equivalent to the standard notions of commitment and oblivious transfer, but since the parties have no inputs then the different executions are truly independent. Using results of Holenstein on hardness amplification of independent executions [5, 6], Wullschleger proved that starting from a defective protocol for the randomized variants, one can obtain a non-defective protocol for the same variant.

It should be noted that the techniques of Holenstein, which are based on the hard-core-set approach of Impagliazzo [7], were originated in the non-interactive setting of one-way functions and do not extend in general to hardness of interactive protocols (see discussion in the appendix). However, they do extend to interactive protocols in the honest-but-curious attack model, so Wullschleger used these techniques to get a secure OT protocol in the honest-but-curious model, and then applied the Goldreich-Micali-Wigderson "compiler" [3] to get a secure protocol against malicious adversaries.

1.1 Our Results

Although sufficient for proving the existence of OT-combiners, Wullschleger's results still leave several open problems: Mainly, they do not answer the fundamental question regarding the effect of sequential repetition on the secrecy and correctness guarantees of protocols (in the face of malicious adversaries). They also do not answer the question of whether the specific transformations that were described by Damgård et al. [2] work also in the computational setting. Finally, since it relies on the GMW "compiler" then Wullschleger's combiner does not preserve in any way the efficiency properties of the original defective scheme. Closing these gaps is the focus of the current work.

Hardness Amplification/Degradation Lemmas. In Section 3 we describe a rather generic setting where one can argue about hardness amplification and degradation of interactive protocols. We formulate and prove two new lemmas, showing that the information theoretic bounds on hardness-degradation (for both secrecy and correctness) carry over also to the computational setting: Lemma 3.2 asserts that the secrecy degradation from "encrypting the same message t times" obeys the bound of $1-(1-\varepsilon)^t$. Similarly, Lemma 3.5 asserts that given t interactive puzzles that are δ-hard to solve, the probability of solving *at least one of them* is at most $1-(1-\delta)^t$. These lemmas can be thought of as mirroring Yao's XOR lemma and Yao's hardness-amplification lemma for one-way functions [10], respectively. The proofs of these hardness-degradation lemmas are similar in their high-level structure to the corresponding hardness-amplification proofs. For Lemma 3.2 we had to prove a new lemma (Lemma 3.3) that plays a role similar to the one played by Levin's "Isolation Lemma" in the proof of Yao's XOR lemma.

We complement the results for secrecy/correctness degradation with results on secrecy/correctness amplification. Specifically, we observe that some (but not all) of the known proofs for Yao's XOR lemma and Yao's hardness-amplification lemma can be used to prove amplification also for interactive protocols.[1]

Improving Defective protocols. We then consider the applicability of our hardness amplification and degradation lemmas to the analysis of the transformations from [2]. Roughly, we prove that these transformations result in a secure protocol whenever the defect parameters of the original protocol satisfy $\varepsilon+\delta \le 1-1/polylog(k)$ (with k the security parameter), but our lemmas cannot be applied to prove security in some cases where $\varepsilon+\delta$ is bounded away from 1 only by a polynomial fraction. In Lemma 4.1, we characterize exactly the range of the defect parameters (ε,δ) for which we can prove that these transformations produce a secure protocol.

2 Notations

The statistical distance between two distributions D_1, D_2 over a countable domain is the scaled sum $|\mathcal{D}_1-\mathcal{D}_2| \stackrel{\text{def}}{=} \frac{1}{2}\sum_x |\mathcal{D}_1(x)-\mathcal{D}_2(x)|$, where the sum is taken over all the elements in the union of the support of the two distributions, and $\mathcal{D}_i(x)$ is the probability mass of x according to the distribution $\mathcal{D}_i$. We use $x \in_R S$ to denote choosing x from S uniformly at random. A positive function is *negligible* if it tends to zero faster than any polynomial, and it is *noticeable* otherwise.

An algorithm is called *efficient* if it runs in probabilistic polynomial time. A two-party protocol is a pair of algorithms, one for each party. We use the following notations to describe a two-party protocol (A, B):

[1]Roughly, the proofs that can be extended are those where the single-instance adversary A runs the multiple-instance adversary A' on just one vector that includes the instance that A wants to solve. In the interactive case, this translates to a "non-rewinding" reduction. See more details in the proofs of Lemma 3.1 and Lemma 3.5 and some discussion in the appendix.

- The communication transcript is denoted $\langle A(a,r_a), B(b,r_b)\rangle$.
- The event where A outputs the string x is denoted $(A(a,r_a), B(b,r_b)) \xrightarrow{A} x$, and similarly $(A(a,r_a), B(b,r_b)) \xrightarrow{B} y$ for the output of B, and $(A(a,r_a), B(b,r_b)) \rightarrow (x,y)$ for the output of both.

In these notations, a, b are the inputs and r_a, r_b are the randomness used by the participants. We often omit the randomness (and sometimes also the input) from these notations. We use $\star$ to denote a "don't care" input or output.

3 Amplification/Degradation of Computational Hardness

In this section we prove some lemmas about amplification and degradation of computational hardness for sequential composition of protocols. (By "computational hardness" we roughly mean breaking either the secrecy or correctness of the protocol.) The amplification lemmas are straightforward extensions of Yao's XOR lemma and Yao's hardness-amplification lemma for one-way functions [10, 4], but the degradation lemmas are new.

We deal with two-party protocols, where one player either tries to guess the input of the other party or tries to break the correctness of the protocol (e.g., in a commitment scheme the goal is either to learn the committed bit or to open the commitment in two different ways). We study how the computational-hardness of accomplishing these tasks is amplified or degraded when several copies of the protocol are run sequentially in various settings. We consider the following four scenarios in the setting of two parties A and B, where A has input a.

SECRECY. In this setting player B wants to learn the input of player A.

Amplification We examine the effect of running the protocol t times, where in each invocation player A chooses a random input, subject to the condition that the XOR of the t inputs is A's original input a.

When restricted to the non-interactive case of one-way functions, this is exactly the setting for Yao's XOR lemma [10]. We note that Levin's proof [8] can be easily extended to sequential composition of interactive protocols (see also [4, Lemma 4]).

Degradation We examine the effect of running the protocol t times, but this time player A uses the same input in every run. This "secrecy degradation" setting is dealt with in Lemma 3.2.

CORRECTNESS. In this setting, player A tries to break the correctness of the protocol by outputting some "forbidden value" at the end of the protocol execution (such as two different opening of the commitment).

Amplification We consider the setting where after t runs of the protocol, player A needs to break *all the t executions.*

When restricted to the non-interactive case of one-way functions, this is exactly the setting for Yao's hardness-amplification lemma from weak to strong one-way functions [10]. Here, again, the proof of Canetti et al. [1] can be easily extended to interactive protocols.[2]

Degradation We consider the setting where after t runs of the protocol, player A needs to break *any one of the t executions*. This "hardness degradation" setting is dealt with in Lemma 3.5. (The proof closely mirrors the "hardness amplification" proof from [1].)

3.1 Secrecy Amplification and Degradation

Let (A, B) be an interactive protocol where A has a single-bit input $a \in \{0, 1\}$ (and B may have no input), and let $t = t(k)$ be polynomially bounded. Denote by $(A^t_=, B^t)$ a t-fold sequential repetition of (A, B), where the protocol (A, B) is run t times sequentially, each time with the same input bit a. Also denote by $(A^t_\oplus, B^t)$ a t-fold sequential repetition of (A, B), where the input of A in each run is random and independent, subject to the condition that the XOR of the inputs in all the runs equals to the input bit of $A^t_\oplus$.

Definition 1 (Input Secrecy Defect) *The protocol (A, B) has an ε-bounded secrecy defect with respect to A if, for every efficient B', it holds that $\Pr[(A(a), B') \xrightarrow{B'} a] \leq (1 + \varepsilon)/2 + \mathsf{negl}(k)$, where the probability is taken over the choice of $a \in_R \{0, 1\}$ and the randomness of A and B', k is the security parameter, and negl is a negligible function.*

Lemma 3.1 (Yao's XOR Lemma [10] – Secrecy Amplification) *If (A, B) has an ε-bounded secrecy defect with respect to A and t is polynomially-bounded, then $(A^t_\oplus, B^t)$ has an ε^t-bounded secrecy defect with respect to $A^t_\oplus$.*

Proof (sketch): We observe that Levin's proof of Yao's XOR lemma [8] can be extended also to interactive protocols. (See a description of that proof also in [4, Lemma 4].) The reason that this particular proof extends to the interactive case (whereas the other proofs from [4] do not seem to extend) is that this proof does not need to "rewind" A:

Recall that we assume an adversary B' with advantage better than ε^t when talking to $A^t_\oplus$, and we want to construct an adversary B^* with advantage better than ε when talking to A. In the non-interactive case, we had a "puzzle" that came from A and we could stick that puzzle anywhere in a vector of t puzzles and let B' attempt to solve that vector. We could also stick the same puzzle in many vectors and run B' on all oof them. In the interactive case, on the other hand, once we sent some messages to the real party A, we cannot "take them back" and try another interaction instead.

On a high level, the reduction following Levin's approach proceeds as follows: B^* simulates the interactions between B' and $A^t_\oplus$ for several runs, $i = 1, 2, \ldots$: Starting from the state that B' ended at after the $i - 1$'st run, B^* uses repeated sampling to look for a simulated execution of the i'th run after which B^* still has advantage better than ε^{t-i} in guessing the bit of $A^{t-i}_\oplus$ (where the probability is taken over the remaining runs). It continues in this fashion until it cannot find

[2]Despite the similarities, the hardness-amplification lemma *does not follow* from the results for soundness amplification of interactive proofs. The reason is that in our case the adversary can compute the "forbidden output" at the very end, after all the executions took place. In the IP setting, on the other hand, the prover needs to "convince the verifier" after each execution and before the next one starts.

such an i'th run (or until it gets to the last run). Then it uses the current state of B' as a basis for a single interaction with the "real player" A. If this was the last run then it uses the output of B' as the guess of A's input bit, and otherwise it uses repeated sampling again to estimate the probability that B' outputs one (taken over the remaining runs), and compares that probability to some threshold (that it can also compute using repeated sampling).

Levin's isolation lemma then proves that if at some point B' failed to find an i'th run as above, then there is a threshold that it can set that would give it an advantage better than ε of guessing the input bit of the "real player" A. □

Lemma 3.2 (Secrecy Degradation) *If (A, B) has an ε-bounded secrecy defect with respect to A and t is polynomially-bounded, then $(A^t_=, B^t)$ has an ε'-bounded secrecy defect with respect to $A^t_=$, where $\varepsilon' = 1 - (1-\varepsilon)^t$.*

We emphasize that the simple hybrid argument that is commonly used to reason about "encrypting the same message many times" can be used in this context to prove a bound of $\varepsilon' \leq t\varepsilon$. The difficulty in the proof below is in improving the bound from $t\varepsilon$ to $1-(1-\varepsilon)^t$.

Proof: Let $t = t(k)$ be polynomially bounded, let $\varepsilon = \varepsilon(k)$, and denote $\varepsilon' \stackrel{\text{def}}{=} 1-(1-\varepsilon)^t$. We show that if there exist a randomized adversary B' of time complexity T' such that

$$\Pr_{a,r_a,r_b}[\big(A^t_=(a,r_a), B'(r_b)\big) \stackrel{B'}{\to} a] \geq \frac{1+\varepsilon'+\rho}{2},$$

where $\rho = \rho(k)$ is noticeable, then there exists a randomized adversary B^* of time complexity $T = T' \cdot poly(kt/\varepsilon\rho)$ such that

$$\Pr_{a,r_a,r_b}[(A(a,r_a), B^*(r_b)) \stackrel{B^*}{\to} a] \geq \frac{1+\varepsilon+\varepsilon\rho/4}{2} .$$

An alternative way to write the condition $Pr[(A^t_=(a), B') \stackrel{B'}{\to} a] \geq \frac{1+\varepsilon'+\rho}{2}$ is

$$\Pr[\big(A^t_=(1), B'\big) \stackrel{B'}{\to} 1] - \Pr[\big(A^t_=(0), B'\big) \stackrel{B'}{\to} 1] \ \geq\ \varepsilon' + \rho .$$

Below we always use this alternative formulation.

Consider breaking B' into two parts: the first part B'_1 interacts with $A(a)$ only once and outputs the internal state at the end of this interaction, and the second part B'_2 gets this internal state as input and then interacts with $A(a)$ for $t-1$ more times before outputting a guess for the bit a. Denote by $\mathcal{D}_0, \mathcal{D}_1$ the probability distribution of the internal state s after B'_1 interacts with $A(0)$, $A(1)$, respectively.

$$\mathcal{D}_0 \stackrel{\text{def}}{=} \left\{ s \ : \ (A(0), B'_1) \stackrel{B'_1}{\to} s \right\}, \quad \text{and} \quad \mathcal{D}_1 \stackrel{\text{def}}{=} \left\{ s \ : \ (A(1), B'_1) \stackrel{B'_1}{\to} s \right\}$$

(the notation $\mathcal{D}_0, \mathcal{D}_1$ is interpreted both as a probability distribution and as the corresponding support set). For any given internal state $s \in \mathcal{D}_0 \cup \mathcal{D}_1$, consider the experiment where starting from this internal state s, B'_2 interacts $t-1$ more times with A, but the input of A in all these executions is some bit a' (which may or may not be equal to the input bit a of the first execution). We denote by $p_0(s), p_1(s)$ the probabilities that B' outputs 1 in this experiment when $a' = 0$ and $a' = 1$, respectively. Namely, for every $s \in \mathcal{D}_0 \cup \mathcal{D}_1$ we denote

$$p_0(s) \stackrel{\text{def}}{=} \Pr\left[\left(A_{=}^{t-1}(0), B_2'(s)\right) \stackrel{B_2'}{\to} 1\right], \quad \text{and} \quad p_1(s) \stackrel{\text{def}}{=} \Pr\left[\left(A_{=}^{t-1}(1), B_2'(s)\right) \stackrel{B_2'}{\to} 1\right]$$

We view p_0, p_1 as random variables in $[0, 1]$, where each random variable can be chosen over either of the two probability spaces $\mathcal{D}_0$ or $\mathcal{D}_1$. Below, we use notations such as $\Pr_{\mathcal{D}_0}[p_0 > t]$ to denote the probability that we get $p_0(s) > t$ when setting $s \in_R \mathcal{D}_0$, or $E_{\mathcal{D}_1}[p_1]$ to denote the expected value of $p_1(s)$ taken over the choice $s \in_R \mathcal{D}_1$, etc.

The technical Lemma 3.3 below asserts roughly that either there exists some internal state s^* such that $p_1(s^*) - p_0(s^*) > 1 - (1 - \varepsilon)^{t-1}$, or there exists some probability threshold τ such that $\Pr_{\mathcal{D}_1}[p_1 > \tau] - \Pr_{\mathcal{D}_0}[p_1 > \tau] > \varepsilon$. If there exists a state s^* as in the first case, then $B_2'(s^*)$ guesses the input bit of $A_{=}^{t-1}$ with advantage better than $1 - (1 - \varepsilon)^{t-1}$ and we can continue recursively. Otherwise, we can construct B^* roughly as follows: B^* first plays the part of B_1', interacting with $A(a)$ and gets the internal state s. Then, it evaluates $p_1(s)$ (by repeated sampling), outputs 1 if $p_1(s) > \tau$ and 0 otherwise.

The actual statement of the technical lemma below is slightly more complicated, since it also includes the "slackness parameter" ρ that is needed to get the result in a uniform complexity setting. Specifically, in the first case there should be a significant probability of finding a state s^* for which $p_1(s^*) - p_0(s^*) > 1 - (1 - \varepsilon)^{t-1} + \rho$, and in the second case there should be some uniform way of finding the threshold τ.

Lemma 3.3 *Fix any integer t and any $\varepsilon, \rho \in [0, 1]$ such that $\rho < (1 - \varepsilon)^t$. Also let $\mathcal{D}_0, \mathcal{D}_1$ be two probability distributions and let p_0, p_1 be two random variables that are defined over both $\mathcal{D}_0$ and $\mathcal{D}_1$. If $E_{\mathcal{D}_0}[p_1] - E_{\mathcal{D}_1}[p_0] > 1 - (1 - \varepsilon)^t + \rho$, then at least one of the two conditions must hold:*

(i) Either $\Pr_{\mathcal{D}_0}\left[p_1 - p_0 > 1 - (1 - \varepsilon)^{t-1} + \rho\right] \geq \dfrac{\epsilon\rho}{2}$,

(ii) or $E_\tau\left[\Pr_{\mathcal{D}_1}[p_1 > \tau] - \Pr_{\mathcal{D}_0}[p_1 > \tau]\right] > \varepsilon(1 + \rho/2)$*, where the expectation is over choosing τ uniformly at random in the interval* $[1 - (1 - \varepsilon)^{t-1} + \rho,\ 1]$.

We prove Lemma 3.3 later in this section. Using this lemma, we now complete the proof of Lemma 3.2 as follows: from the assertion we have that $E_{\mathcal{D}_0}[p_1] - E_{\mathcal{D}_1}[p_0] > 1 - (1 - \varepsilon)^t + \rho$ so we can apply Lemma 3.3. The adversary B^* will sample $poly(k/\epsilon\rho)$ internal states $s \in_R \mathcal{D}_0$, and for each will evaluate $p_0(s)$ and $p_1(s)$ with accuracy $poly(\rho/t)$ and error $poly(\varepsilon\rho/tk)$. If it finds a state s for which $p_1 - p_0 > 1 - (1 - \varepsilon)^{t-1} + \rho(1 - 1/2t)$ then it uses $B_2'(s)$ as an adversary against the $(t-1)$-sequential repetition $A_{=}^{t-1}(a)$ and continue by recursion.

Otherwise, B^* plays the role of B_1', interacting with $A(a)$ to produce an internal state s. Then B^* evaluates $p_1(s)$ with accuracy $poly(\varepsilon\rho)$ and error $poly(\varepsilon\rho/k)$. It then chooses at random $\tau \in_R [1 - (1 - \varepsilon)^{t-1} + \rho,\ 1]$, and outputs 1 if $p_1(s) > \tau$ and 0 otherwise. It is not hard to see that this algorithm has expected advantage of $\varepsilon(1 + \rho/2) - poly(\varepsilon\rho/k) > \varepsilon(1 + \rho/4)$. □

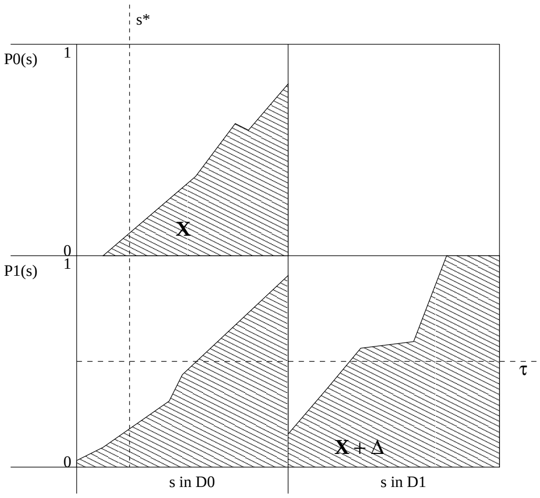

Figure 1: An illustration of Lemma 3.3. We know that the gray area in the lower-right box is more than $X + (1 - (1 - \varepsilon)^t)$. We essentially prove that either there is s^* such that $p_1(s^*) - p_0(s^*) > 1 - (1 - \varepsilon)^{t-1}$ or there is τ such that $\Pr_{\mathcal{D}_1}[p_1 > \tau] - \Pr_{\mathcal{D}_0}[p_1 > \tau] > \varepsilon$.

Proof of Lemma 3.3. The proof relies on the identity $E[X] \equiv \int_0^\infty \Pr[X > \tau]d\tau$, that holds for any non-negative random variable X. In our case, we have $p_0, p_1 \in [0, 1]$ so we can integrate between 0 and 1 (rather than 0 and ∞). Assume that the premise of the lemma holds but condition (i) does not, and we prove that then condition (ii) must hold. For the proof below, denote

$$\mu \stackrel{\text{def}}{=} 1 - (1 - \varepsilon)^{t-1} + \rho$$

If condition (i) does not hold then with all but probability $\varepsilon\rho/2$ over choosing $s \in_R \mathcal{D}_0$, we have $p_1(s) - p_0(s) \le \mu$. This implies that, for all $\tau \in [\mu, 1]$, it holds that

$$\Pr_{\mathcal{D}_0}[p_1 > \tau] \le \Pr_{\mathcal{D}_0}[p_0 > \tau - \mu] + \frac{\varepsilon\rho}{2},$$

and therefore also

$$\int_\mu^1 \Pr_{\mathcal{D}_0}[p_1 > \tau]dt \quad \le \quad \int_0^{1-\mu} \left(\Pr_{\mathcal{D}_0}[p_0 > \tau] + \frac{\varepsilon\rho}{2} \right) dt \quad = \quad \int_0^{1-\mu} \Pr_{\mathcal{D}_0}[p_0 > \tau]dt + \frac{(1-\mu)\varepsilon\rho}{2} \qquad (1)$$

Using this inequality and the premise of the lemma, we can write:

$$\begin{aligned}
1 - (1 - \varepsilon)^t + \rho \quad &< \quad E_{\mathcal{D}_1}[p_1] - E_{\mathcal{D}_0}[p_0] \\
&= \quad E_{\mathcal{D}_1}[p_1] - E_{\mathcal{D}_0}[p_1] + E_{\mathcal{D}_0}[p_1] - E_{\mathcal{D}_0}[p_0] \\
&= \quad E_{\mathcal{D}_1}[p_1] - E_{\mathcal{D}_0}[p_1] \\
&\quad\; + \int_0^\mu \Pr_{\mathcal{D}_0}[p_1 > \tau]d\tau + \int_\mu^1 \Pr_{\mathcal{D}_0}[p_1 > \tau]d\tau - \int_0^{1-\mu} \Pr_{\mathcal{D}_0}[p_0 > \tau]d\tau - \int_{1-\mu}^1 \Pr_{\mathcal{D}_0}[p_0 > \tau]d\tau \\
&\stackrel{\text{Eq. (1)}}{\le} \quad E_{\mathcal{D}_1}[p_1] - E_{\mathcal{D}_0}[p_1] + \int_0^\mu \Pr_{\mathcal{D}_0}[p_1 > \tau]d\tau + \frac{(1-\mu)\varepsilon\rho}{2} - \int_{1-\mu}^1 \Pr_{\mathcal{D}_0}[p_0 > \tau]d\tau
\end{aligned}$$

$$
\begin{aligned}
&\leq \quad E_{\mathcal{D}_1}[p_1] - E_{\mathcal{D}_0}[p_1] + \int_0^{\mu} \Pr_{\mathcal{D}_0}[p_1 > \tau] d\tau + \frac{(1-\mu)\varepsilon\rho}{2} \\
&= \quad \int_0^{\mu} \Pr_{\mathcal{D}_1}[p_1 > \tau] d\tau + \int_{\mu}^{1} \Pr_{\mathcal{D}_1}[p_1 > \tau] d\tau - \int_{\mu}^{1} \Pr_{\mathcal{D}_0}[p_1 > \tau] d\tau + \frac{(1-\mu)\varepsilon\rho}{2} \\
&\leq \quad \mu \; + \; \int_{\mu}^{1} \left(\Pr_{\mathcal{D}_1}[p_1 > \tau] - \Pr_{\mathcal{D}_0}[p_1 > \tau] \right) d\tau \; + \; \frac{(1-\mu)\varepsilon\rho}{2} \\
&= \quad \mu \left(1 - \frac{\varepsilon\rho}{2}\right) + \int_{\mu}^{1} \left(\Pr_{\mathcal{D}_1}[p_1 > \tau] - \Pr_{\mathcal{D}_0}[p_1 > \tau] \right) d\tau + \frac{\varepsilon\rho}{2}
\end{aligned} \tag{2}
$$

Substituting back $\mu = 1 - (1-\varepsilon)^{t-1} + \rho$, we conclude that

$$
\begin{aligned}
\int_{\mu}^{1} \left(\Pr_{\mathcal{D}_1}[p_1 > \tau] - \Pr_{\mathcal{D}_0}[p_1 > \tau] \right) d\tau \quad &> \quad (1 - (1-\varepsilon)^t + \rho) - (1 - (1-\varepsilon)^{t-1} + \rho)(1 - \frac{\varepsilon\rho}{2}) - \frac{\varepsilon\rho}{2} \\
&= \quad \varepsilon(1-\varepsilon)^{t-1} - \frac{\varepsilon\rho}{2}((1-\varepsilon)^{t-1} - \rho) \\
&= \quad \varepsilon((1-\varepsilon)^{t-1} - \rho) - \frac{\varepsilon\rho}{2}((1-\varepsilon)^{t-1} - \rho) + \varepsilon\rho \\
&= \quad \underbrace{((1-\varepsilon)^{t-1} - \rho)}_{1-\mu} \left(\varepsilon - \frac{\varepsilon\rho}{2}\right) + \varepsilon\rho \; > \; (1-\mu)\left(\varepsilon + \frac{\varepsilon\rho}{2}\right)
\end{aligned}
$$

Hence, the expected value of the difference $\Pr_{\mathcal{D}_1}[p_1 > \tau] - \Pr_{\mathcal{D}_0}[p_1 > \tau]$, taken over a uniform random choice of $\tau \in_R [\mu,\ 1]$, is at least $\varepsilon + \frac{\varepsilon\rho}{2}$. $\square$

3.2 Hardness Amplification and Degradation

Consider an interactive protocol $\mathcal{P} = (A, B)$, and let $R_{\mathcal{P}}$ be a poly-time recognizable relation that describes what it means for A to "break the protocol's correctness". Namely, after the protocol is run and B's output is some string y, a cheating A' is successful if it outputs a string x such that $(x, y) \in R_{\mathcal{P}}$. (For example, (A, B) is a commitment scheme, A is the sender, B's output is the communication transcript y, and $(x, y) \in R_{\mathcal{P}}$ if x contains two different openings that are both consistent with y.)

Let (A^t, B^t) be a t-fold sequential repetition of the protocol (A, B) with A, B having the same input (if any) but independent randomness. Define $\wedge^t(R_{\mathcal{P}})$ and $\vee^t(R_{\mathcal{P}})$ as the AND and OR of the t individual relations, namely

$$
\begin{aligned}
\wedge^t(R_{\mathcal{P}}) &\stackrel{\text{def}}{=} \{(\langle x_1, \ldots, x_t \rangle, \langle y_1, \ldots, y_t \rangle) \; : \; \forall i \leq t,\ (x_i, y_i) \in R_{\mathcal{P}}\}, \\
\vee^t(R_{\mathcal{P}}) &\stackrel{\text{def}}{=} \{(x, \langle y_1, \ldots, y_t \rangle) \; : \; \exists\ i \leq t \ s.t.\ (x, y_i) \in R_{\mathcal{P}}\}
\end{aligned}
$$

In other words, $\wedge^t(R_{\mathcal{P}})$ represents the case that all the t copies must be broken, and $\vee^t(R_{\mathcal{P}})$ represents the case that at least one copy is broken.

Definition 2 ($R_{\mathcal{P}}$-defect) *The protocol $\mathcal{P} = (A, B)$ has a δ-bounded $R_{\mathcal{P}}$-defect with respect to B if for every efficient A' it holds that $\Pr[(A', B) \to (x, y) : (x, y) \in R_{\mathcal{P}}] \leq \delta + \mathsf{negl}(k)$, where the probability is taken over the randomness of A' and B, k is the security parameter, and* negl *is a negligible function.*

Lemma 3.4 (Hardness Amplification) *If $\mathcal{P} = (A, B)$ has a δ-bounded $R_{\mathcal{P}}$-defect with respect to B and t is polynomially bounded then (A^t, B^t) has a δ^t-bounded $\wedge^t(R_{\mathcal{P}})$-defect with respect to B^t.*

The proof is nearly identical to the hardness-amplification proof from [1] for the non-interactive case (and also very similar to the proof for Lemma 3.5 below). Again, the reason that this proof extends to the interactive case (whereas some other proofs of Yao's lemma of weak-to-strong-OWFs do not extend) is that it does not need to "rewind" the player B. $\square$

Lemma 3.5 (Hardness Degradation) *If $\mathcal{P} = (A, B)$ has a δ-bounded $R_{\mathcal{P}}$-defect with respect to B and t is polynomially bounded then (A^t, B^t) has a δ'-bounded $\vee^t(R_{\mathcal{P}})$-defect with respect to B^t, where $\delta' = 1 - (1 - \delta)^t$.*

Proof (sketch): The proof is very similar to the hardness-amplification proof from [1]. Let $t = t(k)$ be polynomially bounded, and let $\delta = \delta(k)$ be a noticeable function and $\delta' = 1 - (1 - \delta)^t$. Assume that there exists a randomized adversary A' of time complexity T' that satisfies the relation $\vee^t(R_{\mathcal{P}})$ with probability $\delta' + \rho'$ for some noticeable quantity $\rho' = \rho'(k)$. We then show that there exists a randomized adversary A^* of time complexity $T^* = T' \cdot poly(kt/\delta'\rho')$ that satisfies $R_{\mathcal{P}}$ with probability $\delta + \rho$, where ρ is the solution to $(1 - \delta - \rho)^t = (1 - \delta)^t - \rho'$. Observe that if ρ' is noticeable and t is polynomial then also ρ is noticeable. Note also that by definition, the success probability of A' is $1 - (1 - \delta - \rho)^t$.

Denote the state of A' after the i'th interaction with B by s_i (with s_0 being the initial state of A'). The adversary A^* begins by playing the role of B in the first interaction. Repeating the first interaction up to $poly(kt/\delta\rho)$ times, A^* is looking for an internal state s_1 after the first interaction such that when proceeding from this state, A' satisfies $R_{\mathcal{P}}$ for one of the last $t - 1$ runs with probability at least $1 - (1 - \delta - \rho)^{t-1}$. (Note that A' can estimate that probability by sampling.)

If A^* succeeds in finding such s_1, then it fixes that internal state and keeps looking for internal states $s_2, s_3, \ldots$ such that when proceeding from s_i, adversary A' satisfies $R_{\mathcal{P}}$ for one of the last $t - i$ runs with probability at least $1 - (1 - \delta - \rho)^{t-i}$. If A^* can find an internal state s_{t-1} from which A' satisfies $R_{\mathcal{P}}$ for the last run with probability $\geq \delta + \rho$ then we are done: A^* just uses A' from this state when interacting with the real B. Otherwise, A^* has some state s_i with $0 \leq i < t - 1$ such that A' satisfies $R_{\mathcal{P}}$ for one of the last $t - i$ runs with probability at least $1 - (1 - \delta - \rho)^{t-i}$, and yet for (almost) all continuation states s_{i+1}, A' only satisfies $R_{\mathcal{P}}$ for one of the last $t - i - 1$ runs with probability less than $1 - (1 - \delta - \rho)^{t-i-1}$.

We now consider a "matrix" M that represent the interaction of A' with B on the remaining $t - i$ runs of the protocol, when A' starts from this state s_i. (We assume that s_i includes all the randomness that A' needs for all the runs.) The columns of M are labeled by all the possibilities for the randomness of B during the $i + 1$'st run, and rows are labeled by all the possibilities for the randomness of B in runs $i + 2, \ldots, t$. Hence, each entry in the matrix corresponds to a particular interaction of A' with B on the remaining $t - i$ runs of the protocol.

Each entry in M is labeled with two bits, where the first bit is 1 if at the end of that interaction A' satisfies $R_{\mathcal{P}}$ for the $i + 1$'st run, and the second bit is 1 if A' satisfies $R_{\mathcal{P}}$ for one of the last $t - i - 1$ runs. By our assumption on the state s_i, we know that a random entry in this matrix is labeled with $(0, 0)$ with probability at most $\gamma = (1 - \delta - \rho)^{t-i}$. Denote $\alpha = (1 - \delta - \rho)^{t-i-1}$ and $\beta = (1 - \delta - \rho)$, so $\alpha \cdot \beta = \gamma$. Then, it must be the case that either M has (sufficiently many) columns where the fraction of entries of the form $(\star, 0)$ is no more than α, or else the conditional probability of a (0,0) entry given that the entry is of the form $(\star, 0)$ is at most (only slightly more than) β.

The failure of A^* to find a continuation state s_{i+1} with sufficient residual success probability indicates that the first case does not hold, so the second case must hold. Hence, in this case A^* uses A' starting from s_i to interact with the real player B, arriving at some state s_{i+1} after this "real interaction." Then, A^* simulates many more runs of A' with B starting from this s_{i+1}. Adversary A^* looks for a run in which A' *does not satisfy* $R_{\mathcal{P}}$ for any of the last $t-i-1$ runs, and uses the output of A' in that run in the hope that it satisfies $R_{\mathcal{P}}$ for the $i+1$'st run. The conditional probability argument from above says that the odds of satisfying $R_{\mathcal{P}}$ for the $i+1$'st run conditioned on not satisfying it for the last $t-i-1$ runs is (only slightly less than) $1-\beta = \delta + \rho$. Indeed, a detailed argument that mirrors the proof of [1, Lemma 1] shows that this algorithm A^* has success probability noticeably larger than δ. □

"Weakly verifiable" relations. Since the proof of Lemma 3.5 is fashioned after the proof from [1], it can be used also in settings where the relation $R_{\mathcal{P}}$ is only "weakly verifiable". Namely, when only B can verify the relation $R_{\mathcal{P}}$ (because it depends on some private output of B). This makes the above lemma applicable also to the setting of commitment with non-perfect functionality, where even an honest opening may be rejected with some probability (depending on the internal randomness of the receiver).

4 Fixing Defective Protocols

In [2], Damgård et al. considered defective two-party protocols such as oblivious-transfer and commitment between *a Sender* and *a Receiver*. They suggested reducing the defect by alternating between two transformations: Roughly, in a "type-R" transformation the parties run t copies of the protocol with the same input bits for the sender, and in a "type-S" transformation the sender chooses t random bits whose exclusive-or equals to its input bit and then the parties run one copy of the protocol for each of these random bits.

Below we assume that the underlying protocol has defect ε for the Sender security and defect δ for the Receiver security (such as the commitment protocol that was described in the introduction). In the information-theoretic setting that was considered in [2], it is clear that applying a type-R transformation results in a protocol with sender defect $1-(1-\varepsilon)^t$ and receiver defect δ^t, and similarly applying a type-S transformation results in a protocol with sender defect ε^t and receiver defect $1-(1-\delta)^t$. It was shown in [2] that as long as $\varepsilon + \delta < 1 - 1/poly(k)$, one can alternate between these transformations several times (with total number of copies polynomial in k) and reduce both defects to negligible quantities in k.

Our lemmas from Section 3 imply that the same bounds on the effect of type-R and type-S transformations hold also in the computational setting. One could hope, therefore, that the alternation strategy from [2] can be proven to work also in this setting. Unfortunately, this is not the case. The reason is the strategy from [2] uses a non-constant number of alternations. The proofs for hardness-amplification and degradation from Section 3 all incur a polynomial blowup in the complexity of the adversary for every alternation, and hence a non-constant number of alternations would cause a super-polynomial blowup in the adversary complexity. In Section 4.1 below we analyze the range of parameters ε, δ for which we can reduce the defect to a negligible amount using only a constant number of alternations.

Relation to Wullschleger's work. As we explained in the introduction, Wullschleger recently was able to extend the results from [2] to the computational setting in the "honest-bu-curious" model, using a technique based on the hard-core-set lemma of Holenstein [5, 6]. Wullschleger's work yield stronger defect-reduction results than the ones that we can obtain from a direct analysis of the transformations of [2]: he is able to fix a defect of $\varepsilon + \delta < 1 - 1/poly(k)$, where we can roughly fix only when $\varepsilon + \delta < 1 - 1/polylog(k)$. However, Wullschleger's technique only applied in the honest-but-curious attack model work, whereas ours applies also in the standard model of a malicious adversary. Also, Wullschleger's results do not shed light on what happens when a defective protocol is run several times on related inputs, and does not say what happens when the original transformations from [2] are used in the computational setting.

4.1 Iterating the Transformations

Below, we prove that repeating the transformations S and R a constant number of times results in a scheme with negligible defects as long as $\varepsilon + \delta$ is bounded away from 1 and, moreover, $\varepsilon + \delta < 1 - \min(\varepsilon, \delta)/polylog(k)$.

We begin by setting a few conventions and notations. First, we can assume without loss of generality that we always alternate between transformations S and R (since applying two successive transformations of the same type with parameters t and t' is the same as just one transformation with parameter tt'). We also assume, without loss of generality, that for $\varepsilon > \delta$ we begin with transformation S and for $\varepsilon \leq \delta$ we begin with transformation R. (Namely, we choose the first transformation to increase the larger value and decrease the smaller one.) This is without loss of generality, since we can always start with a "dummy transformation" with parameter $t = 1$.

With these two assumptions, a chain of transformations is completely characterized by the initial values ε_0, δ_0 and by the sequence of parameters $t_1, t_2, \ldots$ that indicate how many times we repeat the scheme from step i in step $i + 1$. In the analysis below we refer to this representation as a "chain".

Definition 3 (Transformation chains) *A transformation chain (or just chain) is represented by a vector $C = \langle(\varepsilon_0, \delta_0), (t_1, t_2, \ldots, t_\ell)\rangle$ where $\varepsilon_0, \delta_0 \in [0, 1]$ and $t_i \geq 1$ for all i. Given C as above, we can compute the values ε_i, δ_i for each $i = 1, \ldots, \ell$ as follows:*

- *If $\varepsilon_0 \geq \delta_0$ then for even i we set $\varepsilon_{i+1} = 1 - (1 - \varepsilon_i)^{t_{i+1}}$ and $\delta_{i+1} = \delta_i^{t_{i+1}}$, and for odd i we set $\varepsilon_{i+1} = \varepsilon_i^{t_{i+1}}$ and $\delta_{i+1} = 1 - (1 - \delta_i)^{t_{i+1}}$.*
- *If $\varepsilon_0 < \delta_0$ then we swap the even and odd rules.*

It is clear, however, that not every "chain" corresponds to a sequence of transformations that we can use. For example, it is clear that $\prod_i t_i$ must be polynomial in the security parameter k. Moreover, all the ε_i's and δ_i's must be bounded away from 1 (i.e., be at most $1 - 1/poly(k)$), since our defect definitions imply that a defect of $1 - \mathsf{negl}(k)$ is the same as a defect of 1. These conditions are captured in the following definition:

Definition 4 (Confined chains) *A chain $C = \langle(\varepsilon_0, \delta_0), (t_1, t_2, \dots, t_\ell)\rangle$ is* **confined** *if there exist constants $c, c' > 0$ such that (a) $\prod_{i=1}^{\ell} t_i \le k^c$ and (b) for all $i \le \ell$, we have $\varepsilon_i, \delta_i \le 1 - k^{-c'}$.*

Moreover, the reductions proving lemmas 3.1 and 3.2 increase the size of the adversary by a polynomial factor (even if we only use $t = 2$), so we can only apply these transformations a constant number of times. This means that, to get a scheme with negligible defect, we must find a constant-length confined chain that begins with the given $(\varepsilon_0, \delta_0)$ and ends with $\varepsilon_\ell, \delta_\ell = \mathsf{negl}(k)$. The next lemma asserts a necessary and sufficient conditions on $(\varepsilon_0, \delta_0)$ for such a chain to exist.

Lemma 4.1 *Fix some $\varepsilon_0 = \varepsilon_0(k)$ and $\delta_0 = \delta_0(k)$ such that $\varepsilon_0 + \delta_0 < 1 - 1/poly(k)$. There exist a constant-length confined chain that begins with these $(\varepsilon_0, \delta_0)$ and ends with $\varepsilon_\ell, \delta_\ell = \mathsf{negl}(k)$ if and only if $\varepsilon_0 + \delta_0 \le 1 - \Omega\left(\frac{\min(\varepsilon_0,\delta_0)}{polylog(k)}\right)$.*

Proof: Roughly, the proof considers the quantity $a = \frac{1-\max(\varepsilon,\delta)}{\min(\varepsilon,\delta)}$, and shows that as long as $a = 1 + o(1)$, then each iteration increases the $o(1)$ part of a by at most a factor of $O(\log k)$. Thus, we must have $a \ge 1 + \Omega(1/polylog)$ if we want a to grow beyond $1 + o(1)$ in a constant number of iterations. In the proof below we use the following facts:

1. For every $\alpha > -1$ and $x \ge 1$, $(1+\alpha)^x \ge 1 + \alpha x$.
2. For every $0 \le \alpha \le \frac{1}{2}$ and $1 \le x \le \frac{1}{2\alpha}$, $(1+\alpha)^x \le 1 + 2\alpha x$.
3. For every $0 \le \alpha \le \frac{1}{2}$ and $1 \le x \le \frac{1}{\alpha}$, $(1-\alpha)^x \le 1 - \alpha x/2$.
4. For every $0 \le \alpha \le 1$, $(1-\alpha)^{1/\alpha} < e^{-1} (\approx 0.37)$
5. For every $0 \le \alpha < \frac{1}{2}$, $(1-\alpha)^{1/\alpha} > 1/4$

If ($\Rightarrow$) Assume that, for some constant $c \ge 1$, it holds that $\max(\varepsilon_0, \delta_0) \le 1 - k^{-c}$, and also $\varepsilon_0 + \delta_0 \le 1 - \frac{\min(\varepsilon_0,\delta_0)}{\log^c(k)}$. We show a confined chain of length at most $c + 5$ such that $\varepsilon_{c+5}, \delta_{c+5} = \mathsf{negl}(k)$. Assume that $\max(\varepsilon_0, \delta_0) > k^{-c'}$ for some c' (otherwise we already have $\varepsilon_0, \delta_0 = \mathsf{negl}(k)$), and consider the following procedure for generating such a chain:

$H_0 := \max\{\varepsilon_0, \delta_0\},\;\; L_0 := \min\{\varepsilon_0, \delta_0\}$
$i := 1,\;\; t_1 := \max\{t \in \mathbb{N} : (1 - H_0)^t > k^{-c}\}$ $\quad$ // $t_1 \le \lceil c\ln(k)/H_0 \rceil = O(k^{c'} \log k)$
$H_1 := 1 - (1 - H_0)^{t_1},\;\; L_1 := L_0^{t_1}$ $\quad$ // $\frac{1}{2} \le H_1 < 1 - k^{-c}$

while $(1 - H_i)/L_i < 2k$ **do** $\quad$ // $L_i > (1 - H_i)/2k > k^{-c-1}/2$
$\quad t_{i+1} := \max\{t \in \mathbb{N} : (1 - L_i)^t > k^{-c}\}$ $\quad$ // $t_{i+1} \le \lceil c\ln(k)/L_i \rceil = O(k^{c+1} \log k)$
$\quad H_{i+1} := 1 - (1 - L_i)^{t_{i+1}},\;\; L_{i+1} := H_i^{t_{i+1}}$ $\quad$ // $\frac{1}{2} \le H_{i+1} < 1 - k^{-c}$
$\quad i := i + 1$

8. $t_{i+1} := \lfloor k/(1-H_i) \rfloor$, $H_{i+1} := 1-(1-L_i)^{t_{i+1}}$, $L_{i+1} := H_i^{t_{i+1}}$
9. $t_{i+2} := k$, $H_{i+2} := 1-(1-L_{i+1})^{t_{i+2}}$, $L_{i+2} := H_{i+1}^{t_{i+2}}$

Output the chain $\langle (\varepsilon_0, \delta_0), (t_1, t_2, \ldots, t_{i+2}) \rangle$

We start by establishing some simple invariants that holds throughout all the iterations of the loop.

- For all i we have $L_i + H_i < 1$. This follows since initially we have $L_0 + H_0 < 1$, and if $x+y<1$ then also $(1-(1-x)^t) + y^t < 1$ for all $t \geq 1$ so this property is preserved.
- For all i we have $L_i < \frac{1}{2} < H_i < 1 - k^{-c}$:
  - The condition $H_i < 1-k^{-c}$ follows since the t_i's are chosen specifically to ensure it.
  - On the other hand, we always set $H_i := 1-(1-\alpha)^{t_i}$ for some $\alpha < 1$ and where t_i is chosen as $\max\{t : (1-\alpha)^t > k^{-c}\}$. So either $\alpha > \frac{1}{2}$, in which case $H_i \geq \alpha > \frac{1}{2}$, or $\alpha \leq \frac{1}{2}$, in which case $(1-\alpha)^{\lceil \frac{1}{\alpha} \rceil} > \frac{1}{8} > k^{-c}$ and therefore $t_i \geq \lceil \frac{1}{\alpha} \rceil$, so $H_i > 1-(1-\alpha)^{\lceil \frac{1}{\alpha} \rceil} > 1-e^{-1} > \frac{1}{2}$.
  - Finally, since $H_i > \frac{1}{2}$ and $H_i + L_i < 1$ then $L_i < \frac{1}{2}$.
- Since $L_i < \frac{1}{2}$ then $(1-L_i)^{\frac{1}{L_i}} > 1/4$. Thus $(1-L_i)^{\frac{c \log_2 k}{2L_i}} > k^{-c}$, so $t_{i+1} \geq \frac{c \log_2 k}{2L_i}$.
- Inside the loop, we always have $\frac{1-H_i}{L_i} < 2k$ which means that $L_i > \frac{1-H_i}{2k} > \frac{k^{-c}}{2k} = \frac{1}{2k^{c+1}}$.

We now observe that all the t_i's are polynomially bounded: Recall that $(1-H_0)^{\lceil c \ln(k)/H_0 \rceil} < e^{-c \ln(k)} = k^{-c}$ so we must have $t_1 < \lceil c \ln(k)/H_0 \rceil < ck^{c'} \ln(k) = O(k^{c'} \log k)$ (since we assume that $H_0 \geq k^{-c'}$). Similar argument using $L_i > \frac{1}{2k^{c+1}}$ shows that in Line 5 we have $t_{i+1} = O(k^{c+1} \log k)$.

Next, we consider the quantity $a_i \stackrel{\text{def}}{=} \frac{1-H_i}{L_i}$. First, observe that the condition $\varepsilon_0 + \delta_0 \leq 1 - \frac{\min(\varepsilon_0, \delta_0)}{\log^c(k)}$ (which we can re-write as $H_0 + L_0 \leq 1 - \frac{L_0}{\log^c(k)}$) implies that

$$\frac{1-H_0}{L_0} - 1 \; = \; \frac{1-(H_0+L_0)}{L_0} \geq \frac{1}{\log^c(k)} \; .$$

Next, observe that

$$\frac{1-H_1}{L_1} \; = \; \frac{1-(1-(1-H_0)^{t_1})}{L_0^{t_1}} \; = \; \left(\frac{1-H_0}{L_0} \right)^{t_1} \; > \; \frac{1-H_0}{L_0} \; \geq \; 1 + \frac{1}{\log^c(k)} \; .$$

We now show that in each iteration of the loop, the quantity $a_i - 1$ increases by at least a factor of $\Omega(\log k)$. Denote $b_i \stackrel{\text{def}}{=} \frac{1-L_i}{H_i}$, and note that

$$\begin{aligned} b_i - 1 \; &= \; \frac{1-L_i}{H_i} - 1 \; = \; \frac{1-L_i-H_i}{H_i} \\ &= \; \frac{L_i}{H_i} \cdot \frac{1-L_i-H_i}{L_i} \; = \; \frac{L_i}{H_i} \cdot \left(\frac{1-H_i}{L_i} - 1 \right) \; = \; \frac{L_i}{H_i} \cdot (a_i - 1) \; . \end{aligned}$$

Observe that for each iteration of the loop, we have

$$a_{i+1} \; = \; \frac{1-H_{i+1}}{L_{i+1}} \; = \; \frac{1-(1-(1-L_i)^{t_{i+1}}))}{H_i^{t_{i+1}}} \; = \; \left(\frac{1-L_i}{H_i} \right)^{t_{i+1}} \; = \; b_i^{t_{i+1}}$$

and therefore

$$\begin{aligned}
a_{i+1}-1 &= b_i^{t_{i+1}}-1 \;=\; (1+(b_i-1))^{t_{i+1}}-1 \\
&> [1+t_{i+1}(b_i-1)]-1 \;=\; t_{i+1}(b_i-1) \;=\; t_{i+1}\frac{L_i}{H_i}\cdot(a_i-1) \;>\; t_{i+1}\cdot L_i\cdot(a_i-1) \\
&\geq \frac{c\log_2 k}{2L_i}\cdot L_i\cdot(a_i-1) \;=\; \frac{c}{2}\log_2 k\cdot(a_i-1) \;=\; \Omega(\log k)\cdot(a_i-1)\ .
\end{aligned}$$

We have seen that $a_1-1>\Omega(\frac{1}{\log^c(k)})$ and that $a_{i+1}-1\geq\Omega(\log k)\cdot(a_i-1)$, so after at most $c+1$ iterations of the loop we get $a_i-1\geq\Omega(\log k)>4$.

If we still do not have $a_i>2k$ then we will do another iteration of the loop. In this iteration, we have (as usual) $t_{i+1}\geq\frac{c\log_2 k}{2L_i}$, but now $a_i=\frac{1-H_i}{L_i}>5$, so $t_{i+1}\geq\frac{5c\log_2 k}{2(1-H_i)}$. Therefore, at the end of this iteration we have

$$\begin{aligned}
L_{i+1} &= H_i^{t_{i+1}} \;=\; (1-(1-H_i))^{t_{i+1}} \;\leq\; (1-(1-H_i))^{\frac{5c\log_2 k}{2(1-H_i)}} \\
&< e^{-5c\log_2 k/2} \;=\; e^{-5c\ln(k)/2\ln(2)} \;=\; k^{-5c/2\ln(2)} \;<\; k^{-3c}\ .
\end{aligned}$$

On the other hand, we have (as usual) $H_{i+1}\leq 1-k^{-c}$, and therefore $a_{i+1}=\frac{1-H_{i+1}}{L_{i+1}}>\frac{k^{-c}}{k^{-3c}}=k^{2c}>2k$.

We conclude that the loop terminates after at most $c+2$ iterations, so the chain is indeed of constant length. It is left to show that the chain remains confined in the last two steps, and that L_{i+2},H_{i+2} are both negligible. Once the loop terminates, we have

$$L_{i+1} \;=\; H_i^{\left\lfloor\frac{k}{1-H_i}\right\rfloor} \;<\; (1-(1-H_i))^{\frac{k}{1-H_i}-1} < e^{-k}/H_i < 2e^{-k}.$$

On the other hand, $\frac{1-H_i}{L_i}>2k$ so $L_i<\frac{1-H_i}{2k}$ and therefore

$$H_{i+1} \;=\; 1-(1-L_i)^{t_{i+1}} \;<\; t_{i+1}L_i \;<\; \left\lfloor\frac{k}{1-H_i}\right\rfloor\cdot\frac{1-H_i}{2k} \;\leq\; \frac{1}{2}\ .$$

Finally, after the last step we have

$$H_{i+2} \;=\; 1-(1-L_{i+1})^k \;<\; kL_{i+1} \;<\; 2ke^{-k}, \quad\text{and}\quad L_{i+2} \;=\; (H_{i+1})^k \;<\; 2^{-k}.$$

This concludes the proof of the *if* direction. □

Only if ($\Leftarrow$). Assume that $\varepsilon_0+\delta_0\leq 1-poly(k)$, but $\varepsilon_0+\delta_0\geq 1-o\left(\frac{\min(\varepsilon_0,\delta_0)}{polylog(k)}\right)$, and assume that $\varepsilon_0\geq\delta_0$ (the other case is symmetric). Let $C=\langle(\varepsilon_0,\delta_0),(t_1,t_2,\ldots\rangle$ be a confined chain with constant length.

Instead of analyzing the chain C, it will be more convenient below to analyze an "equivalent chain" C' for which $\delta_i\leq\varepsilon_i$ for all i. We get C' from C as follows: we go over the transformations one at a time, starting from the first transformation, and maintain the invariant that we always have $\delta_i\leq\varepsilon_i$. If after the next transformation we still have $\delta_{i+1}\leq\varepsilon_{i+1}$ then we leave that transformation unchanged. On the other hand, if after the next transformation (of type R with parameter t_i) we have $\delta_{i+1}\geq\varepsilon_{i+1}$ then we break it into two transformation: a type R transformation with parameter t'_i that increases δ and decreases ϵ until they are exactly equal (t'_i could be fractional), and a type S transformation with parameter $t''_i=t_i/t'_i$. In some more detail, instead of computing $\varepsilon_{i+1}=\varepsilon_i^{t_i}$ and $\delta_{i+1}=1-(1-\delta_i)^{t_i}$, we do the following:

- We compute the real number $t_i' < t_i$ such that $\varepsilon_i^{t_i'} = 1-(1-\delta_i)^{t_i'}$,
- We set $\varepsilon_{i+1}' = \varepsilon_i^{t_i'}$ and $\delta_{i+1}' = \varepsilon_{i+1}' = 1-(1-\delta_i)^{t_i'}$,
- We compute $t_i'' = t_i/t_i'$ and then set $\varepsilon_{i+1}'' = 1-(1-(\varepsilon_{i+1}'))^{t_i''}$ and $\delta_{i+1}'' = (\delta_{i+1}')^{t_i''}$.
- We invert the type of all the transformations until the end of the chain.

Formally, what we do is to remove t_i from the chain and replace it with t_i', t_i'' (so we get a chain which is one longer than the original one).

It is clear that the change from above only switches the roles of ε and δ (i.e., we have $\varepsilon_{i+1}'' = \delta_{i+1}$ and $\delta_{i+1}'' = \varepsilon_{i+1}$, and similarly for $i+2, i+3, \ldots$). It should also be noted that the resulting chain does not correspond to transformations that can be applied to the commitment scheme (since we use fractional values for the t_i's), but all the values of ε_i, δ_i are still well defined, and their sum is equal to what it was in C. Finally, the length of C' is at most twice the length of the original C, so C' still has constant length.

From now on, we therefore assume that we have a constant-length confined chain C' that starts from $\delta_0 \leq \varepsilon_0$ and maintains $\delta_i \leq \varepsilon_i$, for all i. Denote the number of transformations in C' by ℓ and assume, without loss of generality, that ℓ is even (since we can always append a last dummy transformation with $t=1$).

Again, we consider the quantity $a_i = \frac{1-\varepsilon_i}{\delta_i}$, and the condition $\varepsilon_0 + \delta_0 \geq 1 - o\left(\frac{\delta_0}{polylog(k)}\right)$ implies that $a_0 - 1 \leq o(1/polylog(k))$. We show that the quantity $a_i - 1$ grows by at most a factor of $O(\log k)$ in every two successive transformations in the chain. It follows that $a_\ell - 1 = (a_0-1)\cdot O(\log^{\ell/2}(k)) = o(1/polylog(k))$, which in particular means that $\varepsilon_\ell + \delta_\ell \geq 1 - o(1) > 1/2$. In more details, we prove by induction that, for every even i, we have $a_i - 1 \leq (8c\log k)^{i/2}\cdot(a_0-1)$, where the constant c is the one from the "confinement" property of the chain C' (namely all the ε_i's and δ_i's are bounded by $1-k^{-c}$).

This holds for $i=0$ by definition, and we now proceed to the induction step. Assume that for some even $i < \ell$ it holds that $1-a_i \leq (1-a_0)\cdot(8c\log k)^{i/2}$. This in particular means that $\varepsilon_i + \delta_i \geq 1-o(1)$, and therefore (since we have $\delta_i \leq \varepsilon_i$) then $\varepsilon_i \geq \frac{1}{2} - o(1)$. We now examine how the quantity $\frac{1-\varepsilon}{\delta}$ evolves over the next two steps.

- The next (odd-numbered) transformation is of type S, so we have $\delta_{i+1} = \delta_i^{t_{i+1}}$ and $\varepsilon_{i+1} = 1-(1-\varepsilon_i)^{t_{i+1}}$. Since $\varepsilon_i > \frac{1}{2} - o(1)$ then $1-\varepsilon_i < 2^{-1/2}$, and since the sequence is confined then $1-\varepsilon_{i+1} \leq k^{-c}$. Thus we have

$$2^{-c\log_2 k} \;=\; k^{-c} \;\leq\; (1-\varepsilon_i)^{t_{i+1}} \;<\; \sqrt{1/2}^{t_{i+1}} \;=\; 2^{-t_{i+1}/2}$$

so it follows that $t_{i+1} < 2c\log k < 1/2(a_i - 1)$ (since $1/(a_i-1) = \omega(polylog(k))$. This means that we have

$$\begin{aligned} a_{i+1} &= \frac{1-\varepsilon_{i+1}}{\delta_{i+1}} = \left(\frac{1-\varepsilon_i}{\delta_i}\right)^{t_{i+1}} = a_i^{t_{i+1}} = (1+(a_i-1))^{t_{i+1}} \\ &\overset{\text{Fact 2}}{<} 1 + 2t_{i+1}(a_i-1) < 1 + 2c\log k\cdot(a_i-1) \end{aligned}$$

Thus $a_{i+1} - 1 < 2c \log k(a_i - 1) = o(1/polylog(k))$. Let us denote $b_{i+1} \stackrel{\text{def}}{=} \frac{1-\delta_{i+1}}{\varepsilon_{i+1}}$, so we have $b_{i+1} - 1 = (a_{i+1} - 1)\delta_{i+1}/\varepsilon_{i+1}$.

- The next (even-numbered) transformation is of type R, so we have $\delta_{i+2} = 1 - (1 - \delta_{i+1})^{t_{i+2}}$ and $\varepsilon_{i+2} = (\varepsilon_{i+1})^{t_{i+2}}$. Recall that we have $\delta_{i+2} \le \varepsilon_{i+2}$ and therefore $\delta_{i+2} < 1/2 < 1 - e^{-1}$, so $(1 - \delta_{i+1})^{t_{i+2}} = 1 - \delta_{i+2} > e^{-1}$, which means that $t_{i+2} < 1/\delta_{i+1}$. Recall also that we have $\varepsilon_{i+1} \ge \varepsilon_i \ge 1/2 - o(1)$, and therefore

$$b_{i+1} - 1 \;=\; \frac{(a_{i+1} - 1)\delta_{i+1}}{\varepsilon_{i+1}} \;=\; \frac{o(1)}{\Theta(1)} \cdot \delta_{i+1} \;<\; \delta_{1+1}/2$$

so $t_{i+2} < 1/\delta_{i+1} < 1/2(b_{i+1} - 1)$. Thus we have

$$b_{i+2} \;=\; (b_{i+1})^{t_{i+2}} \;=\; (1 + (b_{i+1} - 1))^{t_{i+2}} \;\stackrel{\text{Fact 2}}{<}\; 1 + 2t_{i+2}(b_{i+1} - 1)$$

Hence

$$\begin{aligned} b_{i+2} - 1 \;&<\; 2t_{i+2}(b_{i+1} - 1) \;=\; \frac{2t_{i+2}(a_{i+1} - 1)\delta_{i+1}}{\varepsilon_{i+1}} \\ &<\; \frac{2t_{i+2} \cdot 2c \log k(a_i - 1)\ \delta_{i+1}}{\varepsilon_{i+1}} \;=\; \frac{4c \log k\ \delta_{i+1}\ t_{i+2}}{\varepsilon_{i+1}} \cdot (a_i - 1) \end{aligned}$$

In addition, since $\delta_{i+1} < 1/2$ and $1 \le t_{i+2} < 1/\delta_{i+1}$ then from Fact 3 above we get that

$$\delta_{i+2} \;=\; 1 - (1 - \delta_{i+1})^{t_{i+2}} \;>\; \delta_{i+1} t_{i+2}/2$$

and we also know that $\varepsilon_{i+2} \le \varepsilon_{i+1}$. Thus, we have

$$\begin{aligned} a_{i+2} - 1 \;&=\; \frac{(b_{i+2} - 1)\varepsilon_{i+2}}{\delta_{i+2}} \;<\; \frac{(b_{i+2} - 1)\varepsilon_{i+2}}{\delta_{i+1} t_{i+2}/2} \\ &<\; \frac{4\ \delta_{i+1}\ t_{i+2}\ c \log k}{\varepsilon_{i+1}} \cdot (a_i - 1) \cdot \frac{2\varepsilon_{i+2}}{\delta_{i+1} t_{i+2}} \;=\; 8c \log k(a_i - 1) \cdot \frac{\varepsilon_{i+2}}{\varepsilon_{i+1}} \\ &\le\; 8c \log k \cdot (a_i - 1) \;<\; (8c \log k)^{(i+2)/2} \cdot (a_0 - 1) \;=\; o\left(\frac{1}{polylog(k)}\right) \end{aligned}$$

This concludes the proof of the *only if* direction. □

References

[1] R. Canetti, S. Halevi, and M. Steiner. Hardness amplification of weakly verifiable puzzles. In *The 2nd Theory of Cryptography Conference (TCC'05)*, volume 3378 of *Lecture Notes in Computer Science*, pages 17–33. Springer-Velrag, 2005.

[2] I. Damgård, J. Kilian, and L. Salvail. On the (Im)possibility of Basing Oblivious Transfer and Bit Commitment on Weakened Security Assumptions. In *Advances in Cryptography – EUROCRYPT'99*, volume 1592 of *Lecture Notes in Computer Science*, pages 56–73. Springer-Verlag, 1999.

[3] O. Goldreich, S. Micali, and A. Wigderson. Proofs that yiled nothing but their validity and a methodology of cryptographic protocol design. In *27th Annual Symposium on Foundations of Computer Science*, pages 174–187. IEEE, 1986.

[4] O. Goldreich, N. Nisan, and A. Wigderson. On Yao's XOR-lemma. *Electronic Colloquium on Computational Complexity (ECCC)*, 2(50), 1995.

[5] T. Holenstein. Key agreement from weak bit agreement. In *STOC'05*, pages 664–673. ACM, 2005.

[6] T. Holenstein and R. Renner. One-Way Secret-Key Agreement and Applications to Circuit Polarization and Immunization of Public-Key Encryption. In *Advances in Cryptology - CRYPTO'05*, volume 3621 of *Lecture Notes in Computer Science*, pages 478–493, 2005.

[7] R. Impagliazzo. Hard-core distributions for somewhat hard problems. In *FOCS*, pages 538–545, 1995.

[8] L. A. Levin. One-way functions and pseudorandom generators. *Combinatorica*, 7(4):357–363, 1987.

[9] J. Wullschleger. Oblivious-Transfer Amplification. In *Advances in Cryptology - EUROCRYPT'07*, volume 4515 of *Lecture Notes in Computer Science*, pages 555–572. Springer, 2007.

[10] A. C. Yao. Theory and applications of trapdoor functions. In *23rd Annual Symposium on Foundations of Computer Science*, pages 80–91. IEEE, Nov. 1982.

A Hard-core sets and hardness of interactive protocols

In this section we explain the technical difficulty with using arguments based on Impagliazzo's hard-core-set approach [7] in the context of hardness amplification of interactive protocols. Recall that this approach for proving hardness amplification consists of two steps: We begin with the assumption that computing something is somewhat hard (say every efficient algorithm fails to recover x from $y = f(x)$ with at least some noticeable probability δ).

- First we prove that this hardness assumption implies the existence of a hard-core set. Namely a single set S of inputs, of density (almost) δ, such that no efficient algorithm can recover x from $y = f(x)$ on a random $x \in S$ (except with a very small probability).

  The intuition for the existence of such a set is quite clear for the example of inverting one-way functions: It must be the case that all the efficient algorithms fail on the same subset, since if different algorithms were failing on different subsets then one could always get the right answer by trying sufficiently many algorithms (and using the fact that for one-way functions we can verify the right answer when we see it).[3]

[3] Impagliazzo proved that a similar strategy of "asking many different algorithm" can be applied even in cases where we cannot verify the answer. Roughly, one just takes the majority vote.

- In the second step, one uses the existence of a hard-core set as above to prove hardness amplification, roughly arguing that if we choose $x_1, \ldots, x_n$ at random, then at least one of them will be in the hard-core set S (except with probability δ^n) and therefore no efficient algorithm will be able to invert all of the values $y_i = f(x_i)$ except with probability close to δ^n. (Converting this "intuitive argument" to an actual reduction is quite straightforward.)

Consider now applying this argument to hardness of interactive protocols: Let (A, B) be a protocol (where we assume for simplicity that the parties do not have any inputs), and consider for example the task of recovering the randomness of B from the interaction. First notice that in the honest-but-curious attack model we can still use the argument from above: Consider the function that takes as input $x = (r_a, r_b)$ and outputs both r_a and the transcript:

$$f(r_a, r_b) = (r_a,\ \ \langle A(r_a), B(r_b)\rangle).$$

The output of this function captures exactly the view of an honet-but-curious adversary that interacts with B, and hence we can apply the results about hardness amplification for the case of non-interactive functions also to this case.

However, this line of argument does not seem to extend to the malicious attack model. In this case the transcript of the protocol is not an input which is generated by a fixed known procedure, but it actually depends on the actions of the adversary. For every fixed cheating adversary A^* we can still define the function

$$f_{A^*}(r_a, r_b) = (r_a,\ \ \langle A^*(r_a), B(r_b)\rangle)$$

that captures the view of A^* when interacting with B, but it is no longer the same function for all A^*'s. Trying to apply the line of arguments from above, we thus cannot get an analog of the first step: We can no longer "ask many different algorithms" about the transcript, since these "algorithms" are now interactive adversaries that expect to actively participate in the creation of that transcript. We can choose one of these "many different algorithms" and use it to generate the transcript, but we have only one chance to do so. If this turns out to be the wrong algorithm then we cannot go back and try a different one, since B cannot be rewound to run the protocol again.