Electronic Colloquium on Computational Complexity, Report No. 145 (2010)

A Taxonomy of Enhanced Trapdoor Permutations

Ron Rothblum *

September 21, 2010

Abstract

Trapdoor permutations (TDPs) are among the most widely studied building blocks of cryptography. Despite the extensive body of work that has been dedicated to their study, in many setting and applications (enhanced) trapdoor permutations behave unexpectedly. In particular, a TDP may become easy to invert when the inverter is given auxiliary information about the element to be inverted (e.g., the random coins that sampled the element). Enhanced TDPs were defined in order to address the latter special case, but there are settings in which they apparently do not suffice (as demonstrated by the introduction of doubly-enhanced TDPs).

We study the hardness of inverting TDP in natural settings, which reflect the security concerns that arise in various applications of TDPs to the construction of complex primitives (e.g., Oblivious Transfer and NIZK). For each such setting, we define a corresponding variant of the notion of an enhanced TDP such that this variant is hard to invert in this setting. This yields a taxonomy of such variants, which lie between enhanced TDPs and doubly-enhanced TDPs. This work explores this taxonomy and its relation to various applications.

For example, one of the abstract settings that we consider arises in the standard protocol for one-out-of-k oblivious transfer, based on enhanced trapdoor permutations. In the case of $k > 2$, this protocol provides a natural separation between barely enhanced TDPs and a corresponding variant (which belongs to the aforementioned taxonomy). We comment that, for the case of $k = 2$ the standard protocol is secure as is.

*Department of Computer Science and Applied Mathematics, Weizmann Institute of Science, Rehovot, Israel. E-mail: ron.rothblum@weizmann.ac.il. This research was partially supported by the Israel Science Foundation (grant No. 1041/08).

ISSN 1433-8092

1 Introduction

A collection of trapdoor permutations is a collection of efficiently computable permutations that are hard to invert on the average, with the additional property, that each permutation has a trapdoor that makes the permutation easy to invert. Trapdoor permutations are among the most fundamental primitives of cryptography and have been used to construct a variety of schemes and protocols, most notably, public-key encryption [Yao82] and signature schemes [BM92].

Trapdoor permutations were also believed to imply oblivious transfer and (efficient prover) non-interactive zero-knowledge proofs for $\mathcal{NP}$ but it seems that some additional structure is required for these applications (see, e.g. [Gol04, Gol09]). The point is that in these applications, the adversary may get auxiliary information such as the randomness used to sample an image of the permutation, and this auxiliary information may allow inverting the permutation or approximating its hardcore predicate. The phenomenon was first observed in the context of constructing oblivious transfer (see [Gol04]) and later in the context of non-interactive zero-knowledge proofs (see [Gol09]). This led to the introduction of enhanced and doubly-enhanced trapdoor permutations, which suffice for the construction of oblivious transfer and non-interactive zero-knowledge proofs for $\mathcal{NP}$, respectively. These phenomena motivate further study of the hardness requirements from enhanced trapdoor permutations, and on closer examination still more issues arise. For example, while enhanced trapdoor permutations do suffice for one-out-of-two oblivious transfer, we show that the standard construction (as in [Gol04]) needs to be adapted in order to obtain one-out-of-k oblivious transfer, for $k \geq 3$. The motivating question behind this work is asking what added features are necessary and sufficient for applications such as oblivious transfer and non-interactive zero-knowledge proofs for $\mathcal{NP}$.

Our approach is to define a number of abstract scenarios, where each scenario captures the type of information available to the adversary and the information that we wish to keep secret. Each scenario leads to a corresponding notion of an enhanced trapdoor permutations which is hard to invert in that scenario. We study the relations between these variants of enhanced trapdoor permutations as well as the relation to the aforementioned applications, while noting that all these variants of enhanced trapdoor permutations are implied by the doubly-enhanced property.

1.1 Trapdoor Permutations

Loosely speaking, a collection of one-way permutations is a collection of efficiently computable permutations that are hard to invert on the average, i.e., given a random permutation f from the collection and a random element x from its domain, it is infeasible to find $f^{-1}(x)$. Each permutation is represented by an index α and has an associated domain D_α over which it is defined. A natural domain to consider is $\{0,1\}^{|\alpha|}$; however, this is not necessarily the case in general, and our only requirement is that it is possible to efficiently sample elements (uniformly) from this domain. We denote the domain sampler by S, and use the convention that $S(\alpha; r)$ refers to the (deterministic) output of S on input α and the random string r.

A collection of trapdoor permutations (TDP) is a collection of one-way permutations $\{f_\alpha \colon D_\alpha \to D_\alpha\}_\alpha$ for which, each permutation has an associated trapdoor that makes the permutation easy to invert. We require an efficient way to generate an index together with the corresponding trapdoor.

Enhanced TDP. Consider an adversary for inverting a TDP that is given not only an element x but also the random coins that were used to sample x. Goldreich [Gol04] noted that there are

TDPs that can be inverted in such a setting (assuming that TDPs exist)[1] and showed that this issue captures a real security concern in the standard protocol for oblivious transfer (which was believed to work based on any TDP). He therefore defined the notion of an *enhanced* TDP which is a TDP that is infeasible to invert even given the random coins that sampled the element to be inverted. That is, given an index of a permutation α and a random string r it is infeasible to compute $x \stackrel{\text{def}}{=} f_\alpha^{-1}(S(\alpha; r))$.

Hardcore Predicates. A hardcore predicate of a TDP is an efficiently computable predicate, defined over the domain of the permutation that is infeasible to approximate, given only the image of the element. In other words, given α and x it is easy to compute $h(x)$ but given only α and $f_\alpha(x)$ it is infeasible to approximate $h(x)$. Note that since f_α is a permutation, $h(x)$ is information theoretically determined by $\alpha, f_\alpha(x)$ and therefore, $h(x)$ is only hard to approximate in a computational sense. An *enhanced* hardcore predicate of an *enhanced* TDP is naturally defined w.r.t the enhanced security property. That is, based on α and r, it is infeasible to approximate $h(x)$ where $x = f_\alpha^{-1}(S(\alpha; r))$. Goldreich and Levin [GL89] showed a hardcore predicate that holds for (a minor modification of) any TDP. If the TDP is enhanced then the Goldreich-Levin predicate is an *enhanced* hardcore predicate.

1.2 Are Enhanced Hardcore Bits Pseudorandom?

Let $\{f_\alpha \colon D_\alpha \to D_\alpha\}_\alpha$ be a standard TDP with a hardcore predicate h. Suppose that we are given a randomly selected index of a permutation α and two random elements from its domain $y_1, y_2 \in D_\alpha$. By definition, it is infeasible to compute the hardcore bit of the inverse of any single element (i.e. $h(f_\alpha^{-1}(y_j))$. Moreover, intuitively it seems as though these two bits are pseudorandom, and in particular it is infeasible to compute any *relationship* between these two bits.

A simple argument shows that this is indeed the case. To prove this, assume toward a contradiction that there exists an algorithm A that given α, y_1 and y_2 computes $h(f_\alpha^{-1}(y_1)) \oplus h(f_\alpha^{-1}(y_2))$, where $\oplus$ denotes exclusive-or. We use A to construct an adversary A' for the hardcore predicate h. Recall that A' is given α and y_1 and needs to compute $h(f_\alpha^{-1}(y_1))$. The key point is that A' can generate $h(f_\alpha^{-1}(y_2)), y_2$ by itself[2] and then invoke A on α, y_1, y_2 to obtain $b = h(f_\alpha^{-1}(y_1)) \oplus h(f_\alpha^{-1}(y_2))$. Finally, using $h(f_\alpha^{-1}(y_2))$, the adversary A' outputs $b \oplus h(f_\alpha^{-1}(y_2))$ which indeed equals $h(f_\alpha^{-1}(y_1))$.

Surprisingly perhaps, this argument does not extend to the enhanced setting. Suppose that $\{f_\alpha\}$ is an *enhanced* TDP with a domain sampler S and an *enhanced* hardcore predicate h. Given α, r_1 and r_2, is it feasible to compute $h(x_1) \oplus h(x_2)$ where $x_j = f_\alpha^{-1}(S(\alpha; r_j))$? In fact, this may indeed be feasible. The key point, which causes the extension of the proof from the standard setting to the enhanced setting to fail, is that it may not be feasible to generate a sample of the form $(h(x_2), r_2)$ without using the trapdoor. In Appendix A we present an enhanced trapdoor permutation based on quadratic residuosity for which this is the case. In Section 3 we use this property to show that the standard one-out-of-k oblivious transfer protocol is insecure for $k \geq 3$.

Using the equivalence of pseudorandomness and unpredictability, enhanced hardcore bits may also be predictable in the following sense. Given α, $(r_1, h(x_1))$ and r_2 it may be feasible to predict $h(x_2)$. The types of scenarios that we consider in this work are generalizations of this attack. That is, scenarios in which the adversary is given samples (e.g. r_1 together with $h(x_1)$) that it may not be able to generate by itself and is required to execute a task that is infeasible without the samples

[1] Given any TDP, consider changing its sampling algorithm S to $S'(\alpha; r) \stackrel{\text{def}}{=} f_\alpha(S(\alpha; r))$. The random coins of S' always give away the preimage under f_α of sampled elements.

[2] By selecting $x \leftarrow S(\alpha)$ and outputting $h(x), f_\alpha(x)$.

(e.g. compute $h(x_2)$ based on r_2). For each scenario we consider a corresponding variant of an enhanced trapdoor permutation that is hard in that scenario. We consider connections between these variants while distinguishing between hardness that holds w.r.t a fixed number of samples and hardness that holds w.r.t any (polynomial) number of samples.

1.3 Organization

We start in Section 2 by presenting the formal definitions of trapdoor permutations, hardcore predicates and oblivious transfer. In Section 3, we demonstrate the type of problem encountered when using enhanced trapdoor permutations by analyzing the standard OT protocol. In Section 4, we discuss the aforementioned scenarios in which enhanced trapdoor permutations are not hard to invert or have hardcore predicates that are not hard to predict.

2 Definitions

A function $\epsilon\colon \mathbb{N} \to [0,1]$ is *negligible* if for every polynomial $p(\cdot)$ and all sufficiently large $n \in \mathbb{N}$ it holds that, $\epsilon(n) < \frac{1}{p(n)}$. We use $\mathrm{neg}(n)$ and $\mathrm{poly}(n)$ to respectively denote some unspecified negligible function and polynomial. Throughout this manuscript all polynomials are assumed to be positive.

2.1 Collections of Trapdoor Permutations

Formally, we define a collection of trapdoor permutations (TDP) as follows:

Definition 2.1. *A* TDP *is a collection of permutations $\{f_\alpha : D_\alpha \to D_\alpha\}_\alpha$ together with the following associated probabilistic polynomial-time algorithms:*

1. *An* index sampler *I that given the security parameter 1^n, outputs an index of a permutation, denoted α and a corresponding trapdoor, denoted τ.*
2. *A* domain sampler *S that on input α (the index of a permutation), outputs a uniformly distributed element $x \in D_\alpha$.*
3. *An* evaluation algorithm *F (for Forward) that, given α and x, computes the value of the permutation f_α on x, i.e., outputs $f_\alpha(x)$.*
4. *An* inverting algorithm *B (for Backward) that, given the trapdoor of the permutation τ and an element x, inverts the permutation on x, i.e., outputs $f_\alpha^{-1}(x)$.*

The security requirement is that for every probabilistic polynomial-time algorithm A,

$$\Pr_{\substack{(\alpha,\tau)\leftarrow I(1^n)\\ x\leftarrow S(\alpha)}}\left[A(\alpha,x) = f_\alpha^{-1}(x)\right] = \mathrm{neg}(n) \tag{2.1}$$

where the probability is also over the coin tosses of A.

An enhanced TDP is one for which it is infeasible to invert elements even given the random coins that sampled them:

Definition 2.2. *A TDP* $\{f_\alpha \colon D_\alpha \to D_\alpha\}$ *with domain sampler* S*, is* enhanced *if it for every probabilistic polynomial-time algorithm* A*,*

$$\Pr_{\substack{(\alpha,\tau)\leftarrow I(1^n)\\ r\leftarrow\{0,1\}^{\mathrm{poly}(n)}}}\left[A(\alpha,r) = f_\alpha^{-1}(S(\alpha;r))\right] = \mathrm{neg}(n) \tag{2.2}$$

where once again, the probability is also over the coin tosses of A*.*

A hardcore predicate is an efficiently computable predicate defined over the domain of a TDP, that is infeasible to compute based only on an image of an element:

Definition 2.3. *Let* $\{f_\alpha \colon D_\alpha \to D_\alpha\}_\alpha$ *be a TDP. The predicate* h*, defined over the domain of the permutations, is a* hardcore predicate *if it can be computed efficiently and for every probabilistic polynomial-time algorithm* A*,*

$$\Pr_{\substack{(\alpha,\tau)\leftarrow I(1^n)\\ x\leftarrow S(\alpha)}}\left[A(x) = h(f_\alpha^{-1}(x))\right] = \frac{1}{2} + \mathrm{neg}(n) \tag{2.3}$$

An enhanced hardcore predicate is defined analogously, allowing the adversary access to the random string that sampled the element:

Definition 2.4. *Let* $\{f_\alpha \colon D_\alpha \to D_\alpha\}_\alpha$ *be a TDP with domain sampler* S*. The hardcore predicate* h *is* enhanced *if for every probabilistic polynomial-time algorithm* A*,*

$$\Pr_{\substack{(\alpha,\tau)\leftarrow I(1^n)\\ r\leftarrow\{0,1\}^{\mathrm{poly}(n)}}}\left[A(r) = h(f_\alpha^{-1}(S(\alpha;r)))\right] = \frac{1}{2} + \mathrm{neg}(n) \tag{2.4}$$

Goldreich and Levin [GL89] showed that if $\{f_\alpha\}_\alpha$ is a trapdoor permutation then the trapdoor permutation $g_\alpha(x,s) = (f_\alpha(x), s)$ where $|x| = |s|$, has a hardcore predicate $h(x,s) = \langle x,s\rangle = \sum x_i s_i \bmod 2$. If $\{f_\alpha\}_\alpha$ is an enhanced TDP then h is an *enhanced* hardcore predicate of $\{g_\alpha\}_\alpha$.

2.2 Oblivious Transfer

One-out-of-k oblivious transfer (OT) is an interactive protocol consisting of two parties, a sender S and a receiver R. The input of S is composed of k-bits $\sigma_1, \ldots, \sigma_k$ and the input of R is an index $i \in [k]$. At the end of the protocol, the receiver, R, should output σ_i but learn nothing about the sender's input other than σ_i and the sender, S, should learn nothing about the receiver's input (i.e., i). These privacy requirements should hold in a computational sense, with respect to a security parameter n, (which is given to both parties in unary). We restrict our attention to the "semi-honest" model. In this model, each party acts according to the protocol but may write down anything it sees. We mention that a protocol in the "semi-honest" model can be compiled to a protocol that is secure against malicious adversaries by using zero-knowledge proofs (see [Gol04]).

This formulation of OT was introduced by Even et-al [EGL85]. A three-message protocol for OT based on enhanced trapdoor permutations was given by [EGL85, GMW87]. We refer to this protocol (or actually to its description in [Gol04]) as the standard OT protocol. The standard OT protocol uses an enhanced TDP $\{f_\alpha \colon D_\alpha \to D_\alpha\}_\alpha$ with corresponding algorithms I, S, F, B (recall that I is the index/trapdoor sampler, S is the domain sampler, F computes the permutation and B inverts it using the trapdoor) and an enhanced hardcore predicate h (e.g. the Goldreich-Levin hardcore predicate [GL89]). The protocol is depicted in Figure 1.

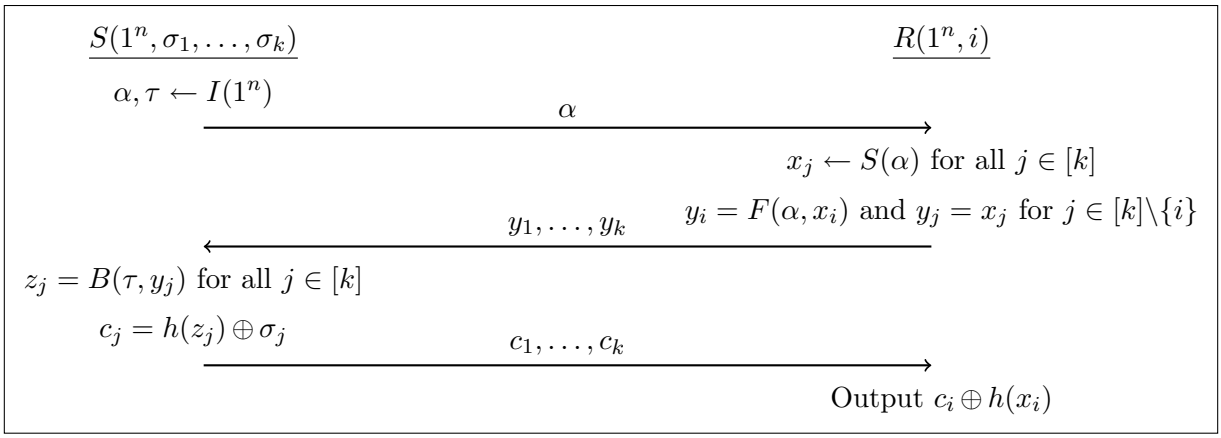

Figure 1: One-out-of-k Oblivious Transfer.

To formalize the semi-honest model, we use the notion of the view of each player. The view of player P with respect to security parameter n is a random variable $View_P(i, (\sigma_1, \ldots, \sigma_k))$ which consists of everything player P sees in the interaction between R on input i and S on input $\sigma_1, \ldots, \sigma_k$, including its own random coin tosses and the received messages. Using this notion we define an OT protocol as follows:

Definition 2.5. *Let $k \geq 2$ be a natural number. (S, R) are a one-out-of-k oblivious transfer (OT) protocol if S and R are interactive probabilistic polynomial-time algorithms and it holds that:*

1. *(Correctness) For every $\sigma_1, \ldots, \sigma_k \in \{0,1\}$ and $i \in [k]$, when $R(1^n, i)$ interacts with $S(1^n, \sigma_1, \ldots, \sigma_k)$, it holds that R outputs σ_i and S outputs nothing.*
2. *(Sender Privacy) There exists a probabilistic polynomial-time simulator Sim_R such that for every $\sigma_1, \ldots, \sigma_k \in \{0,1\}$ and $i \in [k]$, the ensembles $\{Sim_R(i, \sigma_i)\}_{n \in \mathbb{N}}$ and $\{View_R(i, (\sigma_1, \ldots, \sigma_k))\}_{n \in \mathbb{N}}$ are computationally indistinguishable.*
3. *(Receiver Privacy) There exists a probabilistic polynomial-time simulator Sim_S such that for every $\sigma_1, \ldots, \sigma_k \in \{0,1\}$ and $i \in [k]$, the ensembles $\{Sim_S(\sigma_1, \ldots, \sigma_k)\}_{n \in \mathbb{N}}$ and $\{View_S(i, (\sigma_1, \ldots, \sigma_k))\}_{n \in \mathbb{N}}$ are computationally indistinguishable.*

If the output of Sim_S (resp. Sim_R) is identically distributed to the actual view of the sender (resp. receiver) then we say that the receiver (resp. sender) has perfect privacy.

3 Failure of the one-out-of-k OT Protocol for $k \geq 3$

In this section we show that the standard OT protocol fails for $k \geq 3$. We start of by proving that it is indeed correct for $k = 2$. We then proceed to show the problem that arises when trying to extend this proof to $k = 3$, or larger k.

3.1 The Case $k = 2$

Recall that the standard protocol (Figure 1) is based on an enhanced TDP $\{f_\alpha\}_\alpha$ with corresponding algorithms I, S, F, B and an enhanced hardcore predicate h.

When both parties follow the standard protocol the receiver outputs σ_i, thus the protocol is indeed correct. The (prefect) privacy of the receiver (in the semi-honest model) is also immediate

and follows from the fact that f_α is a permutation. We note that correctness and the privacy of the receiver hold for any $k \geq 2$.

The privacy of the sender is less trivial. For sake of simplicity we assume that $i = 1$ and consider an interaction between the receiver, R, and the sender, S given the input σ_1, σ_2. The view of the receiver is:

$$(i = 1, \sigma_1), (r_1, r_2), \left(\alpha, h(S(\alpha; r_1)) \oplus \sigma_1, h(f_\alpha^{-1}(S(\alpha; r_2))) \oplus \sigma_2\right).$$

To prove that privacy of the sender holds, we need to present a simulator for this view. The simulator $Sim_R(i = 1, \sigma_1)$ chooses $(\alpha, \tau) \leftarrow I(1^n)$, samples $r_1, r_2 \leftarrow \{0,1\}^{\mathrm{poly}(n)}$ and outputs:

$$(i = 1, \sigma_1), (r_1, r_2), \left(\alpha, h(S(\alpha; r_1)) \oplus \sigma_1, h(f_\alpha^{-1}(S(\alpha; r_2)))\right).$$

Note that the only difference between the actual view and the output of the simulator is in the last element. However, using the fact that h is an *enhanced* hardcore predicate, $r_2, h(f_\alpha^{-1}(S(\alpha; r_2)))$ and $r_2, h(f_\alpha^{-1}(S(\alpha; r_2))) \oplus \sigma_2$ are computationally indistinguishable, which in turn implies that the actual view is computationally indistinguishable from the output of the simulator.

3.2 The Case $k = 3$

Consider an attempt to extend the proof for the case $k = 2$ to the case $k = 3$. Once again, for simplicity, we assume $i = 1$. The natural extension of the proof is to have the simulator output $h(f_\alpha^{-1}(S(\alpha; r_2)))$ and $h(f_\alpha^{-1}(S(\alpha; r_3)))$ instead of the actual received message $h(f_\alpha^{-1}(S(\alpha; r_2))) \oplus \sigma_2$ and $h(f_\alpha^{-1}(S(\alpha; r_3))) \oplus \sigma_3$. The problem is that it may be easy to distinguish the output of the simulator from the actual received message, using the property described in Section 1.2.

We stress that it is not only the natural extension of the proof to the case $k = 3$ that fails, and the protocol is indeed insecure. To see this (again assuming $i = 1$) recall that R should learn σ_1 but nothing about σ_2 and σ_3. However, based on the protocol R learns $(r_2, b_2 \oplus \sigma_2)$ and $(r_3, b_3 \oplus \sigma_3)$ (where $b_i = h(f_\alpha^{-1}(S(\alpha; r_i)))$). Given that R can also compute $b_2 \oplus b_3$ from r_1 and r_2, it can easily compute $\sigma_1 \oplus \sigma_2$, contradicting the supposed privacy of the sender.

3.3 Fixing the Protocol

One way to fix the standard OT protocol is by using the well known (simple) reduction from general k to $k = 2$. As shown in Section 3.1, one-out-of-two OT can be based on *any* enhanced TDP, hence, the following holds:

Theorem 3.1. *If there exists an enhanced TDP then for any $k \geq 2$, there exists a protocol for one-out-of-k OT.*

An alternate approach, that considers the original protocol, is shown in Section 4.2.2.

4 Problematic Scenarios for Enhanced Trapdoor Permutations

In this section, we discuss different scenarios in which it may be insecure to use enhanced TDPs. We start by presenting these scenarios and the corresponding TDPs and proceed to show connections between these variants of enhanced TDPs.

Throughout this section, we consider an enhanced TDP $\{f_\alpha \colon D_\alpha \to D_\alpha\}_\alpha$ with corresponding algorithms I, S, F, B and a hardcore predicate h. By α we denote a random index from this collection (with respect to a security parameter n). For $j \in \mathbb{N}$, we use r_j to denote uniformly

distributed random coins for the sampling algorithm S and x_j to denote the inverse of the corresponding sampled element, i.e., $x_j \stackrel{\text{def}}{=} f_\alpha^{-1}(S(\alpha, r_j))$. b_j denotes the corresponding hardcore bit, i.e., $b_j \stackrel{\text{def}}{=} h(x_j)$.

4.1 The Scenarios

The attack we presented on the OT protocol in Section 3 was based on the existence of an enhanced TDP with the property that given α, r_1 and r_2, it is feasible to compute $b_1 \oplus b_2$. This means that an adversary that is given a sample of the form (r_1, b_1), can compute the hardcore predicate, i.e., given r_2 (in addition to α and (r_1, b_1)) the adversary can compute $b_2 = h(f_\alpha^{-1}(S(\alpha; r_2)))$. We generalize this type of attack and consider the following scenarios:

1. **Scenario BX:** Based on α and i samples of the form $(r_1, b_1), \ldots, (r_i, b_i)$ it is feasible to invert the permutation, i.e., from r_{i+1} compute $x_{i+1} = f_\alpha^{-1}(S(\alpha; r_{i+1})$.
2. **Scenario BB:** Based on α and i samples of the form $(r_1, b_1), \ldots, (r_i, b_i)$ it is feasible to break the hardcore predicate, i.e., from r_{i+1}, compute $b_{i+1} = h(x_{i+1})$.
3. **Scenario XX:** Based on α and i samples of the form $(r_1, x_1), \ldots, (r_i, x_i)$ it is feasible to invert the permutation, i.e., compute x_{i+1} as in Scenario 1.
4. **Scenario XB:** Based on α and i samples of the form $(r_1, x_1), \ldots, (r_i, x_i)$ it is feasible to break the hardcore predicate, i.e., compute b_{i+1} as in Scenario 2.

Scenarios 1-4 are respectively referred to as Scenarios BX, BB, XX, XB, where the convention is that the first letter represents what the adversary is given (B for hardcore bits and X for preimages) and the second letter represents the goal of the adversary (B to approximate the hardcore bit and X to invert the permutation).

A few immediate relations between the scenarios are depicted in Figure 2, where an arrow from Scenario x to Scenario y means that an adversary in the setting of Scenario x implies an adversary in Scenario y. These relations follow from the fact that b_j can be efficiently computed from x_j. Hardness holds in the opposite direction, that is, an arrow from scenario x to scenario y means that a TDP that is hard in the setting of scenario y is also hard in the setting of scenario x.

Connection to Doubly-Enhanced TDP Scenario XB is actually the setting of the protocol for non-interactive zero-knowledge proofs for $\mathcal{NP}$. In this protocol, the verifier is presented with samples of the form (r_j, x_j). To prove that the protocol is zero-knowledge, we need to argue that the verifier cannot compute the hardcore predicate, however, if the TDP is vulnerable to an attack in the setting of Scenario XB then the hardcore predicate becomes easy to compute. Goldreich [Gol09], addressed the problem raised there by defining *doubly-enhanced* TDPs and showing that they are sufficient. Recall that a doubly-enhanced TDP is defined as follows:

Definition 4.1. *Let $\{f_\alpha \colon D_\alpha \to D_\alpha\}_\alpha$ be an enhanced TDP with domain sampling algorithm S. This collection is* doubly-enhanced *if there exists a probabilistic polynomial-time algorithm that on input α outputs a pair (r, x) such that r is uniformly distributed as random coins of S and $f_\alpha(x) = S(\alpha; r)$.*

Thus, the "doubly-enhanced" property guarantees the ability to generate samples of the form (r_j, x_j) (from which can be derived also samples of the form (r_j, b_j)). This implies, in particular, that a doubly-enhanced TDP is hard in all the above scenarios w.r.t to polynomially many samples. This

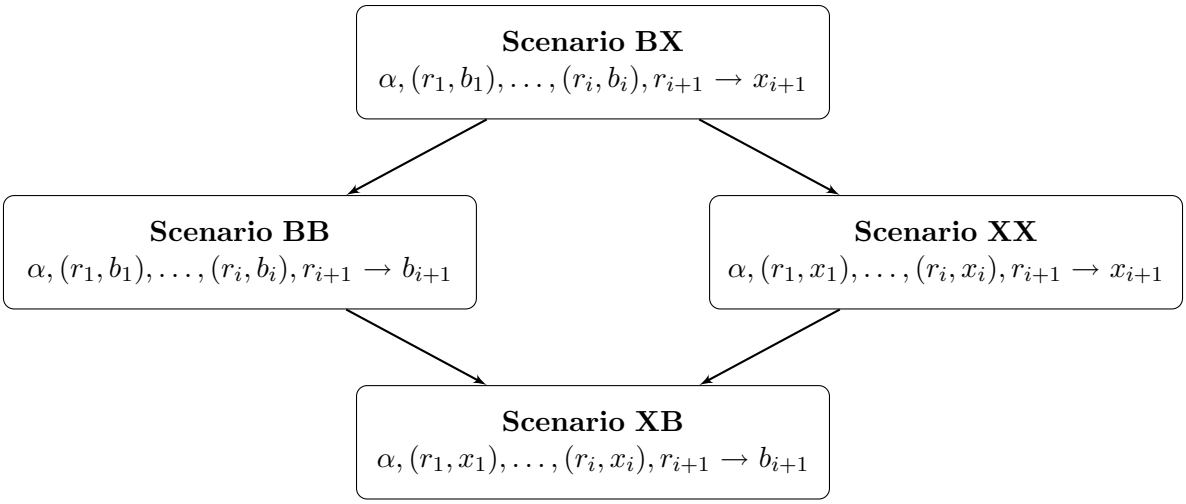

Figure 2: Attacks on Enhanced Trapdoor Permutations.

follows from the fact that any adversary that requires samples (r_j, x_j) or (r_j, b_j) can be converted to an adversary that does not, by simply generating the necessary samples itself (using the algorithm guaranteed by Definition 4.1).

4.2 Hardness of Enhanced TDP w.r.t a Fixed Number of Samples

In this section, and the following one, we show connections between the variants of enhanced TDP that correspond to the scenarios discussed above. We distinguish between hardness that holds for a *fixed* number of samples (discussed in this section) and hardness that holds for polynomially may samples, for *any* polynomial (discussed in Section 4.3).

The results presented in this section are depicted in Figure 3. Each box represents a TDP that is hard in one of the scenarios. Arrows represent connections between these primitives. A solid arrow between from primitive X to Y means that "X is Y"[3] whereas a dotted arrow means that a transformation is required, that is, if there exists X then there exists Y. Some of the arrows are labeled with further restrictions, e.g., "GL", which means that the result holds when using the GL hardcore predicate. Note that all downward pointing arrows follow from the fact that the hardcore bit of an element is efficiently computable.

4.2.1 Scenario BX

Recall that in Scenario BX, the adversary needs to invert the permutation based on samples of the form (r_j, b_j). We first show that *any* enhanced TDP with *any* enhanced hardcore predicate is hard to invert in Scenario BX if at most logarithmically many samples (r_j, b_j) are revealed to the adversary. We proceed by showing that by modifying an enhanced TDP, we can actually obtain an enhanced TDP that is hard to invert even given *polynomially* many samples.

Theorem 4.2. *Let $\{f_\alpha\}_\alpha$ be an enhanced TDP with an enhanced hardcore predicate h. Then, $\{f_\alpha\}$ is hard to invert in the setting of Scenario BX for $i = O(\log n)$ samples.*

[3]We assume that all TDP are by default in the form required for GL. Thus, we do not view the (minor) modification required for the GL hardcore predicate as a transformation.

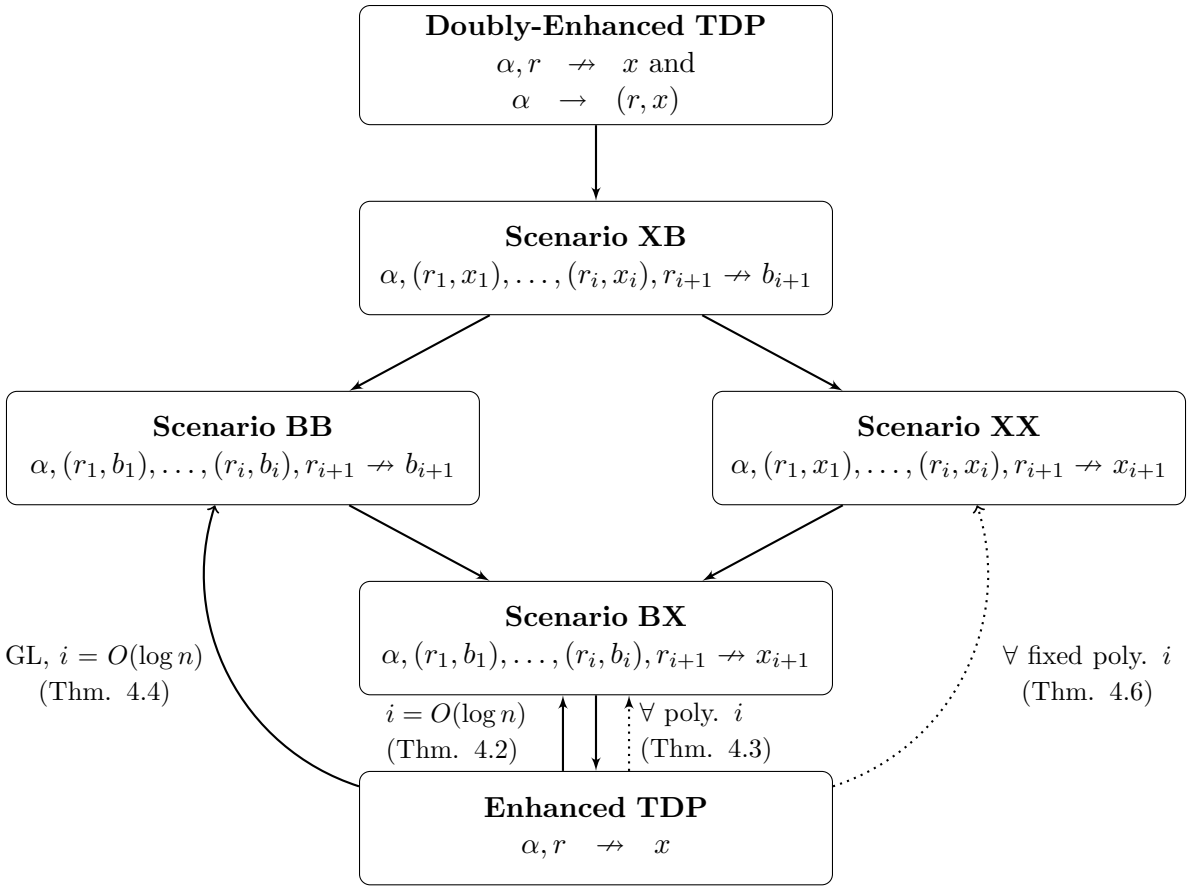

Figure 3: Hardness of Enhanced TDP w.r.t a *fixed* number of samples.

Proof. We use an adversary A for Scenario BX to construct an adversary A' that inverts the permutation in the enhanced setting (i.e. an adversary that inverts based on the random string of the sampling algorithm). The first observation is that A' can enumerate all possible values for logarithmically many hardcore bits. For each sequence of values of hardcore bits, A' runs A to produce a candidate preimage. The second observation is that it is possible to verify the result; that is, A' can check whether a candidate element is indeed the preimage. Details follow.

Assume toward a contradiction that there exists a probabilistic polynomial-time algorithm A that given $\alpha, (r_1, b_1), \ldots, (r_{i(n)}, b_{i(n)}), r_{i(n)+1}$ outputs $x_{i(n)+1}$ with non-negligible probability. We use A to invert the permutation in the enhanced setting, contradicting the assumption that it is an *enhanced* trapdoor permutation.

Given α and r we want to find $x = f_\alpha^{-1}(S(\alpha; r))$ where S is the domain sampling algorithm of $\{f_\alpha\}$. To do this, we select $r_1, \ldots, r_{i(n)}$ as $i(n)$ uniformly distributed random strings of the sampling algorithm S. We then enumerate over all possible values of hardcore bits of $r_1, \ldots, r_{i(n)}$ by going over all $c_1, \ldots, c_{i(n)} \in \{0, 1\}$. For each choice of $c_1, \ldots, c_{i(n)}$ we run $A(\alpha, (r_1, c_1), \ldots, (r_{i(n)}, c_{i(n)}), r)$ and obtain a candidate x'. For each candidate x', we check whether $f_\alpha(x') = S(\alpha; r)$ and output x' if this is the case. Note that after a polynomial number of iterations we will reach the correct choice of hardcore bits and then A inverts $S(\alpha; r)$ with non-negligible probability. □

Theorem 4.2 states that any enhanced TDP is hard to invert based on logarithmically many samples in Scenario BX. Indeed, this holds for any enhanced TDP and no modification is required.

If we allow modifications, then as shown by the following theorem, we can actually obtain an enhanced TDP that is hard to invert even given polynomially many samples.

Theorem 4.3. *Let $\{f_\alpha \colon D_\alpha \to D_\alpha\}_\alpha$ be an enhanced TDP with a hardcore predicate h_f. The direct product of $\{f_\alpha\}$ with itself, denoted $\{g_{\alpha,\beta} : D_\alpha \times D_\beta \to D_\alpha \times D_\beta\}$ and defined as $g_{\alpha,\beta}(x,y) = (f_\alpha(x), f_\beta(y))$, is an enhanced TDP with an enhanced hardcore predicate $h_g(x,y) = h_f(y)$ that is hard to invert in Scenario BX w.r.t any polynomial $i(\cdot)$.*

Proof. We denote the domain sampling algorithm of $\{f_\alpha\}$ (resp. $\{g_{\alpha,\beta}\}$) by S_f (resp. S_g). Note that $S_g((\alpha,\beta);(r,r')) \stackrel{\text{def}}{=} (S_f(\alpha;r), S_f(\beta,r'))$. We denote the random string of S_g by $r'' = (r,r')$ where r is used to sample the first element (from D_α) and r' is used for the second one (from D_β).

Suppose there exists an adversary A that inverts $\{g_{\alpha,\beta}\}$ based on a polynomial number of samples (r''_j, b''_j), where $r''_j = (r_j, r'_j)$ and $b''_j = h_g\left(g^{-1}_{\alpha,\beta}\left(S_g\left((\alpha,\beta)\,; r''_j\right)\right)\right) = h_f\left(f^{-1}_\beta\left(S_f\left(\beta; r'\right)\right)\right) = b'_j$. We use A to construct an adversary A' that inverts $\{f_\alpha\}$. The key point is that it is possible to generate the necessary samples for A using only the trapdoor of β. Thus, to invert f_α, we generate β together with the corresponding trapdoor. We use this trapdoor to generate samples (r''_j, b''_j) where $b''_j = b'_j = h_f\left(f^{-1}_\beta\left(S_f\left(\beta; r'\right)\right)\right)$ and invoke A to invert $g_{\alpha,\beta}$ and in particular f_α. Details follow.

Given α and r, the adversary A' needs to find $x = f^{-1}_\alpha\left(S_f\left(\alpha; r\right)\right)$. This is done by first sampling an index β together with the corresponding trapdoor. Consider the permutation $g_{\alpha,\beta}$. For this permutation, it is easy to generate samples (r''_j, b''_j), where $r''_j = (r_j, r'_j)$ and $b''_j = h_f\left(f^{-1}_\beta\left(S_f\left(\beta; r'_j\right)\right)\right)$ since the hardcore predicate depends only β, which we can invert. Thus, to invert f_α, we invoke A on the index (α,β), $i(n)$ samples (r''_j, b_j) and the random coins $r'' = (r,r')$ (where r is the random string given as input and r' is an independent random string). By our assumption, with non-negligible probability, the adversary outputs a preimage $x'' = (x,x')$ of $S_g\left((\alpha,\beta)\,;(r,r')\right) = S_f\left(\alpha;r\right), S_f\left(\beta;r'\right)$. In particular we have $x = f^{-1}_\alpha\left(S_f\left(\alpha;r\right)\right)$ and so A' outputs x. □

4.2.2 Scenario BB

Recall that Scenario BB is the setting that causes the standard OT protocol to fail for $k \geq 3$ (see Section 3). We show that the Goldreich-Levin (GL) hardcore predicate [GL89], is unpredictable in this setting as long as at most $i = O(\log n)$ samples (r_j, b_j) are revealed to the adversary. Thus, when implemented using the GL hardcore predicate, the standard OT protocol is secure for $k = O(\log n)$.

Theorem 4.4. *Let $\{f_\alpha \colon D_\alpha \to D_\alpha\}_\alpha$ be an enhanced TDP and assume for simplicity that all elements in D_α are of length n. Let $\{g_\alpha \colon D_\alpha \times \{0,1\}^n \to D_\alpha \times \{0,1\}^n\}_\alpha$ be the enhanced TDP defined as $g_\alpha(x,s) = f_\alpha(x), s$ where $|x| = |s| = n$ and let $h(x,s) \stackrel{def}{=} \langle x,s\rangle = \sum_{i=1}^n x_i s_i \bmod 2$ be the GL hardcore predicate of g. Then h is unpredictable in the setting of Scenario BB for $i = O(\log n)$ samples.*

We denote the domain sampling algorithm of $\{f_\alpha\}$ (resp. $\{g_\alpha\}$) by S_f (resp. S_g). Note that $S_g(\alpha);(r,s)) \stackrel{\text{def}}{=} (S_f(\alpha;r), s)$.

To proof Theorem 4.4, we show that given α and i random strings $(r_1,s_1),\ldots,(r_i,s_i)$ of S_g, the sampling algorithm of $\{g_\alpha\}_\alpha$, it is infeasible to approximate $\bigoplus_{j\in U} b_j$, for any non-empty set $U \subseteq [i]$ (where $b_j = h(x_j,s_j)$ and $x_j = f^{-1}_\alpha(S(\alpha;r_j))$). The theorem follows by applying the computational

XOR lemma[4] for hardcore functions [Gol01, Lemma 2.5.8], which holds for $i = O(\log n)$, and the equivalence of pseudorandomness and unpredictability. Thus it suffices to prove the following:

Proposition 4.5. *Let $i \stackrel{def}{=} i(n)$ be a polynomial. For any adversary A, any polynomial $p(\cdot)$, all sufficiently large n and any non-empty set $U \subseteq [i(n)]$:*

$$\Pr_{\substack{\alpha,\tau \leftarrow I(1^n) \\ (r_1,s_1),\ldots,(r_i,s_i) \leftarrow \{0,1\}^{\mathrm{poly}(n)} \times \{0,1\}^{\mathrm{poly}(n)}}} \left[A\left(\alpha, (r_1,s_1), \ldots, (r_i,s_i)\right) = \bigoplus_{j \in U} b_j \right] = \frac{1}{2} + \frac{1}{p(n)} \qquad (4.1)$$

where $b_j \stackrel{def}{=} h(x_j, s_j) = \langle x_j, s_j \rangle$ and $x_j \stackrel{def}{=} f_\alpha^{-1}(S(\alpha; r_j))$.

Proof. Assume toward a contradiction that this is not the case. That is, there exists an infinite set of n, a set $U = \{j_1, \ldots, j_{\ell(n)}\}$, and an adversary A that computes $\bigoplus_{j \in U} b_j$ based on α and $(r_1, s_1), \ldots, (r_i, s_i)$. The main observation is that $\bigoplus_{j \in U} b_j = \bigoplus_{j \in U} \langle x_j, s_j \rangle = \langle x_{j_1} \circ \cdots \circ x_{j_{\ell(n)}}, s_{j_1} \circ \cdots \circ s_{j_{\ell(n)}} \rangle$ where $\circ$ denotes concatenation.

As a mental experiment, consider the trapdoor permutation $\{f'_\alpha \colon D_\alpha^\ell \to D_\alpha^\ell\}$ defined as $f'_\alpha(x_1, \ldots, x_{\ell(n)}) = (f_\alpha(x_1), \ldots, f_\alpha(x_{\ell(n)}))$. Using the sampling algorithm $S_{f'}(\alpha; r_1, \ldots, r_{\ell(n)}) = S(\alpha; r_1), \ldots, S(\alpha, r_{\ell(n)})$, the collection $\{f'_\alpha\}$ is in fact an *enhanced* trapdoor permutation[5]. If we apply the GL modification to $\{f'_\alpha\}$ we obtain the enhanced trapdoor permutation $g'_\alpha(x_1, \ldots, x_{\ell(n)}, s_1, \ldots, s_{\ell(n)}) = f_\alpha(x_1), \ldots, f_\alpha(x_{\ell(n)}), s_1, \ldots, s_{\ell(n)}$ with the enhanced hardcore predicate $\langle x_1 \circ \cdots \circ x_{\ell(n)}, s_1 \circ \cdots \circ s_{\ell(n)} \rangle$. By definition of an enhanced hardcore predicate, this means that given $\alpha, r_1, \ldots, r_{\ell(n)}, s_1, \ldots, s_{\ell(n)}$ it is infeasible to approximate $\langle x_1 \circ \cdots \circ x_{\ell(n)}, s_1 \circ \cdots \circ s_{\ell(n)} \rangle$ in contradiction to our assumption. □

4.2.3 Scenario XX

In this scenario, the adversary is given an index α of a permutation, i samples of the form (r_j, x_j), and an additional random string r_{i+1} and needs to invert the permutation on $S(\alpha; r_{i+1})$, i.e., compute x_{i+1}. We show how to transform any enhanced TDP to one that is hard to invert in the setting of Scenario XX *as long as the number of revealed samples is known ahead of time*, that is, first the number of revealed samples is fixed and then we construct a TDP that is hard to invert w.r.t this number of samples. A disadvantage of our technique is that the length of the index increases linearly with the number of samples. In fact, we can only construct an enhanced TDP that is hard to invert given $m^{1-\epsilon}$ samples where m is the length of the new index (for any constant $\epsilon > 0$).

Theorem 4.6. *If there exists an enhanced TDP, then for every polynomial $q(\cdot)$, there exists an enhanced TDP that is hard to invert in the setting of Scenario XX with respect to $q(n)$ samples.*

Proof. Let $\{f_\alpha\}_\alpha$ be an enhanced TDP with corresponding algorithm I, S, F, B.

Construction 4.7. *We construct an enhanced TDP f' with algorithms I', S', F', B' that is hard to invert in Scenario XX with $q(n)$ samples:*

$I'(1^n)$**:** *Invoke $I(1^n)$, the original index sampler, $2q(n) \cdot n$ times to obtain a $(2q(n) \times n)$-sized matrix of indexes $\underline{\alpha} \stackrel{def}{=} \{\alpha_{i,j}\}_{i \in [2q(n)], j \in [n]}$ and a corresponding $(2q(n) \times n)$-sized matrix of trapdoors $\underline{\tau} = \{\tau_{i,j}\}_{i \in [2q(n)], j \in [n]}$. Output $\underline{\alpha}$ as the index and $\underline{\tau}$ as the trapdoor.*

[4] This lemma shows that if it is infeasible to compute the parity of a random subset of logarithmically many hardcore bits, then they are pseudorandom.

[5] If from $\alpha, r_1, \ldots, r_{\ell(n)}$ it is feasible to compute $x_1, \ldots, x_{\ell(n)}$ then it particular it is feasible to compute x_1 from α, r_1.

$S'(\underline{\alpha})$**:** *From each column $j \in [n]$ of the matrix $\underline{\alpha}$, select at random an entry $s_j \in_R [2q(n)]$ and sample an element from the corresponding permutation's domain, $x_j \leftarrow S(\alpha_{s_j,j})$. Output $(s_1, \ldots, s_n, x_1, \ldots, x_n)$.*

$F'(\underline{\alpha}, (s_1, \ldots, s_n, x_1, \ldots, x_n))$**:** *For every $j \in [n]$, compute the permutation $\alpha_{s_j,j}$ on x_j by invoking $y_j = F(\alpha_{s_j,j}, x_j)$. Output $(s_1, \ldots, s_n, y_1, \ldots, y_n)$.*

$B'(\underline{\tau}, (s_1, \ldots, s_n, y_1, \ldots, y_n))$**:** *For every $j \in [n]$, invert the permutation $\alpha_{s_j,j}$ on y_j by invoking $x_j = B(\tau_{s_j,j}, y_j)$. Output $(s_1, \ldots, s_n, x_1, \ldots, x_n)$.*

Using the fact that $\{f_\alpha\}$ is a TDP, $\{f'_{\underline{\alpha}}\}$ forms a collection of permutations and S' samples elements uniformly from the domain, as required. Furthermore, using the trapdoor $\underline{\tau}$, it is easy to invert $f'_{\underline{\alpha}}$.

We show that an A' adversary that inverts $\{f'_{\underline{\alpha}}\}$ in the setting of Scenario XX can be used to construct an adversary A that inverts $\{f_\alpha\}$ in the enhanced setting. Recall that A is given an index α and a random string r and needs to find x s.t. $x = f_\alpha^{-1}(S(\alpha; r))$. We first sketch the high-level idea of the proof and then go into details.

First, A generates an index matrix $\underline{\alpha}$ together with the corresponding trapdoor matrix $\underline{\tau}$. Then, A selects $q(n) + 1$ random strings for the sampling algorithm S'. Note that each random string specifies a single permutation from each column of $\underline{\alpha}$. The first $q(n)$ random strings will be used to construct samples for A', and the last random string will be used (after a modification) as the challenge for A'.

The key point is that for each column of $\underline{\alpha}$, with probability $\frac{q(n)}{2q(n)} = \frac{1}{2}$, there exists an entry that is not used by any of the first $q(n)$ random strings. Thus, with probability $1 - 2^{-n}$, one of the n indexes specified by the last random string was not specified by any of the first $q(n)$ samples. After finding the coordinate (i, j) of such an index in the matrix $\underline{\alpha}$ (or halting if it does not exist), A replaces the j-th block of the last random string by r and the (i, j)-th entry of $\underline{\alpha}$ by α. Since none of the first $q(n)$ random strings use α, the adversary A can invert them to obtain the required $q(n)$ samples for A'. If A' is successful then in particular it inverts $S(\alpha; r)$, hence obtaining the required preimage.

We proceed to describe the proof in detail. Assume toward a contradiction that there exists a probabilistic polynomial-time adversary A' that inverts $\{f'_{\underline{\alpha}}\}$ with non-negligible probability based on $q(n)$ samples. Thus, A' inverts $f_{\underline{\alpha}}$ based on $\underline{\alpha}$ and $q(n)$ samples of the form $\left(s_1^{(k)}, \ldots, s_n^{(k)}, x_1^{(k)}, \ldots, x_n^{(k)}\right), \left(r_1^{(k)}, \ldots, r_n^{(k)}\right)$ (for all $k \in [q(n)]$) where $x_j^{(k)}$ is the inverse of the element sampled by $r_j^{(k)}$ w.r.t the permutation $\alpha_{s_j^{(k)},j}$. To simplify notation, we denote $\alpha(k, j) \stackrel{\text{def}}{=} \alpha_{s_j^{(k)},j}$. We use A' to construct an adversary A that on input α and r computes $f_\alpha^{-1}(S(\alpha; r))$ and operates as follows:

1. For every $k \in [q(n)]$, select $s_1^{(k)}, \ldots, s_n^{(k)} \in_R [2q(n)]$.
2. Select $s'_1, \ldots, s'_n \in_R [2q(n)]$.
3. Find $t \in [n]$ such that $s'_t \notin \{s_t^{(1)}, \ldots, s_t^{(q(n))}\}$. If no such t exists, halt.
4. Sample an index matrix $\underline{\alpha} = \{\alpha_{i,j}\}_{i \in [2q(n)], j \in [n]}$ together with the corresponding trapdoor $\underline{\tau}$ by invoking $I'(1^n)$. Replace $\alpha_{s'_t,t}$ with α ($\tau_{s'_t,t}$ is irrelevant and can be erased).
5. For $k \in [q(n)]$:

(a) Select $r_1^{(k)}, \ldots, r_{q(n)}^{(k)}$ as uniformly distributed random coins for S.

(b) For $j \in [n]$, set $x_j^{(k)} = f_{\alpha(k,j)}^{-1}(S(\alpha(k,j); r_j^k))$.

6. Select n uniformly distributed random strings $r'_1, \ldots, r'_n$ for S. Replace r'_t with r.

7. Invoke A' on index $\underline{\alpha}$, the samples $\left(s_1^{(k)}, \ldots, s_n^{(k)}, x_1^{(k)}, \ldots, x_n^{(k)}\right), \left(r_1^k, \ldots, r_n^{(k)}\right)$ (for all $k \in [q(n)]$) and the challenge $(s'_1, \ldots, s'_n, r'_1, \ldots, r'_n)$. The output of A should be $(s'_1, \ldots, s'_n, x'_1, \ldots, x'_n)$, halt if this is not the case.

8. Output x'_t.

We first argue that A is indeed efficient. The only step that appears problematic is inverting in step 5b however this can be done efficiently since we only invert permutations for which we have the corresponding trapdoor.

Next, note that A halts in step 3 with probability at most 2^{-n}. This is because the probability that $s'_t \in \left\{s_1^{(1)}, \ldots, s_t^{(q(n))}\right\}$ is at most $\frac{q(n)}{2q(n)} = \frac{1}{2}$ for each $t \in [n]$ and therefore the probability this happens *for all* t is at most 2^{-n}. This also means that although the distribution of samples that A is invoked on is not precisely the same distribution on which A is guaranteed to operate, the two distributions are statistically close. Thus, with non-negligible probability, A outputs $x'_1, \ldots, x'_n$ such that $f_{\alpha_{s'_j,j}}(x'_j) = S(\alpha_{s'_j,j}, r'_j)$ for all $j \in [n]$. In particular, $f_{\alpha_{s'_t,t}}(x'_t) = S(\alpha_{s'_t,t}, r'_t)$. Since $\alpha = \alpha_{s'_t,t}$ and $r = r'_t$ we have that $f_\alpha(x'_t) = S(\alpha, r)$, i.e., x'_t is a preimage as required. □

4.3 Hardness of Enhanced TDP w.r.t Polynomially Many Samples

In this section we continue to establish connections between enhanced TDP that are hard to invert in the different scenarios introduced in Section 4.1. In this section we focus on TDPs that are hard w.r.t *any* polynomial number of samples. This is in contrast to Section 4.2 in which we focused on hardness w.r.t a *fixed* number of samples.

The results presented in this section are depicted in Figure 4 using the same conventions as in Figure 3. We re-emphasize that throughout this section (and in particular in Figure 4) we consider TDP for which hardness (of inverting the permutation or of computing the hardcore predicate) holds w.r.t any (polynomial) number of samples.

4.3.1 Scenario XX vs. Scenario XB

Recall that in scenarios XX and XB, the adversary is required to invert the permutation or compute the hardcore predicate based on samples (r_j, x_j). An appealing aspect of Scenario XX is that it does not involve a hardcore predicate at all. In the author's master's thesis [Rot10, Appendix B], it is shown that a TDP that is hard to invert in the setting of Scenario XB suffices for the efficient prover non-interactive zero-knowledge protocol described in [Gol09]. In fact, it is shown that the only property of the TDP that is used in [Gol09] is hardness to invert in Scenario XB, which is seemingly a weaker assumption than the existence of doubly-enhanced TDP.

We proceed to show that the GL hardcore predicate of a TDP that is hard to invert in Scenario XX is unpredictable in Scenario XB. Thus, the GL hardcore predicate is hard to approximate even if the adversary is given polynomially many samples of the form (r_j, x_j).

Theorem 4.8. *Let $\{f_\alpha\}_\alpha$ be a TDP that is hard to invert in the setting Scenario XX. Then, the GL hardcore predicate, $h(x,s) = \langle x, s\rangle$, w.r.t the enhanced TDP $g_\alpha(x,s) = (f_\alpha(x), s)$ (where $|x| = |s|$), is hard to approximate in the setting of Scenario XB.*

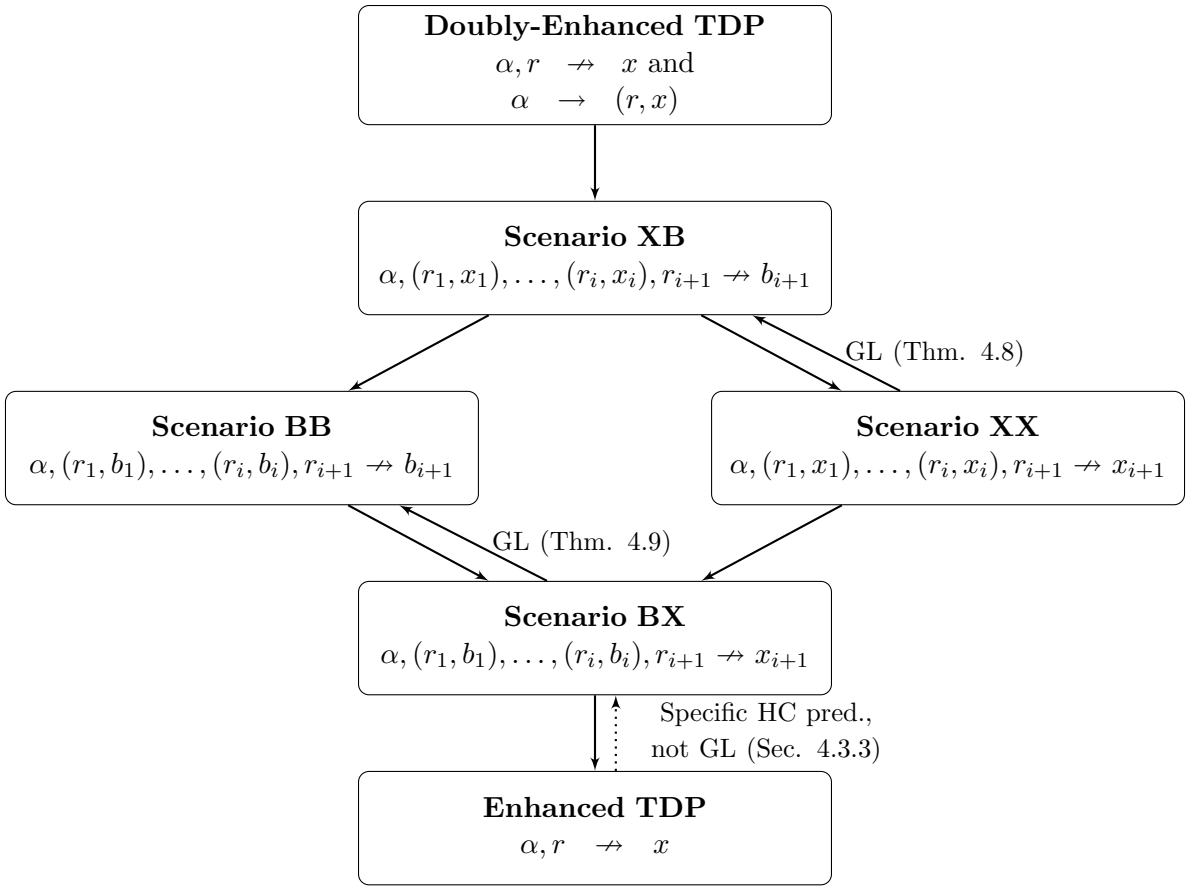

Figure 4: Hardness of Enhanced TDP w.r.t *any* (polynomial) number of samples.

Proof. The proof that h is an (enhanced) hardcore predicate of $\{g_\alpha\}$ (see [Gol01]), reduces the problem of inverting $\{g_\alpha\}$ to approximating h. We adapt the proof to show that inverting $\{g_\alpha\}$ given i samples of the form (r_j, x_j) reduces to approximating h given i' samples of the same form, where i' is related to i (via the advantage of the approximation).

We denote the domain sampling algorithm of $\{f_\alpha\}$ (resp. $\{g_\alpha\}$) by S_f (resp. S_g). We use the convention that random strings of S_f are denoted by r or r_j and those of S_g by (r, s) or (r_j, s_j) such that $S_g(\alpha); (r, s)) \stackrel{\text{def}}{=} (S_f(\alpha; r), s)$.

Let α be an index of a random permutation (w.r.t security parameter n), r a random string for S_f and x the corresponding preimage, i.e., $x = f_\alpha^{-1}(S_f(\alpha; r))$. We denote by $O(s)$ a machine that on input s returns $\langle x, s\rangle$ with probability $\frac{1}{2} + \epsilon$. The GL proof, describes an algorithm H that on input α, r and oracle access to O outputs x with probability $\frac{1}{\text{poly}(n, \frac{1}{\epsilon})}$. The number of oracle queries made by H is $q(n, \frac{1}{\epsilon})$, a polynomial in both n and $\frac{1}{\epsilon}$.

Assume toward a contradiction that there exists an adversary A that given $\alpha, ((r_1, x_1)), \ldots, ((r_i, x_i)), r_{i+1}, s$ computes $b_{i+1} = \langle x_{i+1}, s\rangle$ with advantage ϵ. We use A to construct an adversary A' that inverts $\{f_\alpha\}$ in the setting of Scenario XX using $i(n)$ blocks of $q(n, \frac{1}{\epsilon})$ samples. Thus, A' gets as input a permutation α, samples $(r_1, x_1), \ldots, (r_{i \cdot q(n, \frac{1}{\epsilon})}, x_{i \cdot q(n, \frac{1}{\epsilon})}$ and r' and find $x' = f_\alpha^{-1}(S(\alpha; r'))$. This is done by invoking $H(\alpha, r')$. The major issue is how to answer the oracle calls made by H. To answer the j-th oracle query, $O(s^{(j)})$, A' invokes A on α, the j-th block of samples that it is given as input and $(r', s^{(j)})$ and with probability $\frac{1}{2} + \frac{1}{\text{poly}(n, \frac{1}{\epsilon})}$ obtains $\langle x_{i+1}, s_{i+1}\rangle$. □

Note that the proof of Theorem 4.8 uses, in an essential way, the fact that $\{f_\alpha\}_\alpha$ is hard to invert given *any* polynomial number of samples because the number of samples is related to the advantage of the approximator.

4.3.2 Scenario BX vs. Scenario BB

The proof of Theorem 4.8 can be modified by replacing the (r_j, x_j) samples with (r_j, b_j) to prove the following theorem:

Theorem 4.9. *Let $\{f_\alpha\}_\alpha$ be a TDP that is hard to invert in the setting of Scenarion BX for any polynomial $i(\cdot)$. The GL hardcore predicate, $h(x,s) = \langle x, s\rangle$, w.r.t the enhanced TDP $g_\alpha(x,s) = (f_\alpha(x), s)$ (where $|x| = |s|$), is unpredictable in the setting of Scenario BB.*

4.3.3 Scenario BX

Theorem 4.3 constructs an enhanced TDP that is hard to invert in Scenario BX given any polynomial number of samples, based on any enhanced TDP. Therefore, it is relevant also in this section, when discussing scenarios in which the adversary is given polynomially many samples.

We stress that Theorem 4.3 constructs an enhanced TDP with a *specific* hardcore predicate (that is not the GL hardcore predicate). Therefore, Theorem 4.9 (that holds only for GL) *cannot* be applied to the constructed enhanced TDP to produce a TDP that is hard in Scenario BB.

Acknowledgments

I would like to thank my M.Sc. advisor, Oded Goldreich, for suggesting the research area of enhanced trapdoor permutations and for many helpful discussions and comments regarding this work.

References

[BM92] Mihir Bellare and Silvio Micali. How to sign given any trapdoor permutation. *JACM*, 39(1):214–233, 1992.

[BY96] Mihir Bellare and Moti Yung. Certifying permutations: Noninteractive zero-knowledge based on any trapdoor permutation. *J. Cryptology*, 9(3):149–166, 1996.

[EGL85] Shimon Even, Oded Goldreich, and Abraham Lempel. A randomized protocol for signing contracts. *CACM: Communications of the ACM*, 28, 1985.

[FLS90] U. Feige, D. Lapidot, and A. Shamir. Multiple non-interactive zero knowledge proofs based on a single random string. In IEEE, editor, *Proceedings of the 31st Annual Symposium on Foundations of Computer Science*, pages 308–317, St. Louis, MS, 1990. IEEE Computer Society Press.

[GL89] Oded Goldreich and Leonid A. Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32. ACM, 1989.

[GMW87] Oded Goldreich, Silvio Micali, and Avi Wigderson. How to play any mental game or A completeness theorem for protocols with honest majority. In *STOC*, pages 218–229. ACM, 1987.

[Gol01] Oded Goldreich. *Foundations of Cryptography. Volume I: Basic Tools.* Cambridge University Press, 2001.

[Gol04] Oded Goldreich. *Foundations of Cryptography: Volume 2: Basic Applications.* Cambridge University Press, 2004.

[Gol09] Oded Goldreich. Basing non-interactive zero-knowledge on (enhanced) trapdoor permutations: The state of the art. *http://www.wisdom.weizmann.ac.il/~oded/PSBookFrag/nizk-tdp.ps*, November 2008 (revised October 2009).

[NY90] Moni Naor and Moti Yung. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *Proceedings of the 22th Annual ACM Symposium on Theory of Computing, STOC 1990*, pages 427–437, 1990.

[Rot10] Ron Rothblum. On homomorphic encryption and enhanced trapdoor permutations. Master's thesis, submitted to the Feinberg Graduate School, Weizmann Institute of Science, 2010.

[Yao82] A. C. Yao. Theory and applications of trapdoor functions (extended abstract). In *23th Annual Symposium on Foundations of Computer Science (FOCS '82)*, pages 80–91, Los Alamitos, Ca., USA, November 1982. IEEE Computer Society Press.

A An Enhanced TDP vulnerable in Scenario BB

In this section, we present a variant of the enhanced TDP suggested by Goldreich in [Gol04, Appendix C.1.]. Our TDP has the interesting property that it is completely vulnerable to an attack in Scenario BB. For this section we assume familiarity with basic number theory. Sufficient background is provided in [Gol01, Gol04].

Let N be a Blum integer, Q_N the set of quadratic residues modulo N and M_N the set of all integers in $\{1, \ldots, \lfloor \frac{N}{2} \rfloor\}$ with Jacobi symbol 1 modulo N. We define the predicate $QR_N \colon Z_N^* \to \{0,1\}$ to equal 1 if x is a quadratic residue modulo N and 0 otherwise.

Construction A.1. *(A factoring-based enhanced TDP)*

$I(1^n)$: *Uniformly at random select primes P and Q such that $2^{n-1} \leq P, Q \leq 2^n$ and set $N = PQ$. Select a random element $y \in_R M_N$. The index is (N, y) and the trapdoor is (P, Q).*

$S(N, y)$: *Select $r \in_R Z_N^*$. Set $z = y \cdot r^2 \bmod N$. If $z \leq \lfloor \frac{N}{2} \rfloor$ output z and otherwise output $N - z$.*

$F((N, y), x)$: *Set $z = x^2 \bmod N$. If $z \leq \lfloor \frac{N}{2} \rfloor$ output z and otherwise output $N - z$.*

$B((N, y), x)$: *Given the factorization of N it is possible to invert this permutation (for details see [Gol01, Gol04]).*

Note that Construction A.1 differs from one suggested in [Gol04] only in that the index includes an additional element y and the sampler that now multiplies by y. Indeed, as shown in [Gol04], F_N defines a permutation over M_N. The same argument can be applied to show that each element in M_N has exactly four preimages under $F_{N,y}$, therefore S samples uniformly from M_N.

We proceed by showing an *enhanced* hardcore predicate for the permutation. In particular this implies that this is an enhanced trapdoor permutation. Consider the predicate $h_{N,y}(x) = QR_N(F((N, y), x))$ (i.e., $h_{N,y}(x) = 1$ if and only if the image of x under $F_{N,y}$ is a quadratic

residue). Given x this predicate is easy to compute[6]. However, assuming the quadratic residuosity assumption, we show that this predicate is an *enhanced* hardcore predicate by showing that given $(N, y), r$, it is infeasible to approximate $QR_N(S(N, y; r))$.

The key observation is that multiplying by r^2 preserves the quadratic residuosity property whereas multiplying by $-r^2$ complements it (i.e., $y \cdot r^2$ is a quadratic residue if and only if y is a quadratic residue and $-y \cdot r^2$ is a residue if and only if y is a non-residue). Thus, given N, y and r it is easy to check whether y and $S(N, y; r)$ have the same QR_N value, i.e. compute $QR_N(y) \oplus QR_N(S(N, y; r))$, by checking whether S multiplies y by r^2 or by $-r^2$..

The above implies a reduction to the quadratic residuosity problem. Consider an adversary A that on input $(N, y), r$, computes $QR_N(S(N, y; r))$ with probability $\frac{1}{2} + \epsilon$. We use A to construct an adversary A' to the quadratic residuosity problem as follows. Given N and y, the adversary A' need to find $QR_N(y)$. This is done by selecting $r \in_R Z_N^*$, computing $b = QR_N(y) \oplus QR_N(S(N, y; r))$ (as described in the previous paragraph) and outputting $A((N, y), r) \oplus b$. With probability $\frac{1}{2} + \epsilon$ this equals $QR_N(S(N, y; r)) \oplus \Big(QR_N(y) \oplus QR_N(S(N, y; r)) \Big)$ which in turn equals $QR_N(y)$.

Thus, based on the quadratic residuosity assumption, Construction A.1 is an enhanced TDP. However, we argue that the enhanced hardcore bits are not pseudorandom. Indeed, the TDP is completely vulnerable to the attack as in Scenario BB. This follows from the fact that given N, r_1, r_2, it is easy to compute:

$$\Big(QR_N(y) \oplus QR_N(S(N, y; r_1)) \Big) \oplus \Big(QR_N(y) \oplus QR_N(S(N, y; r_2)) \Big)$$

which equals $QR_N(S(N, y; r_1)) \oplus QR_N(S(N, y; r_2))$.

[6] If $F((N, y), x) = x^2 \bmod N$, then $h_{N,y}(x) = 1$. Otherwise it must be that $F((N, y), x) = N - x^2 \bmod N$ which implies that $h_{N,y}(x) = 0$.

ECCC ISSN 1433-8092
http://eccc.hpi-web.de