On the (Im)Possibility of Tamper-Resilient Cryptography: Using Fourier Analysis in Computer Viruses

Per Austrin[*], Kai-Min Chung[†], Mohammad Mahmoody[†], Rafael Pass[‡], Karn Seth[§]

April 3, 2013

Abstract

We initiate a study of the security of cryptographic primitives in the presence of efficient tampering attacks to the randomness of honest parties. More precisely, we consider p-tampering attackers that may *efficiently* tamper with each bit of the honest parties' random tape with probability p, but have to do so in an "online" fashion. Our main result is a strong negative result: We show that any secure encryption scheme, bit commitment scheme, or zero-knowledge protocol can be "broken" with probability p by a p-tampering attacker. The core of this result is a new Fourier analytic technique for biasing the output of bounded-value functions, which may be of independent interest. We also show that this result cannot be extended to primitives such as signature schemes and identification protocols: assuming the existence of one-way functions, such primitives can be made resilient to $({}^{1}\!/_{\mathrm{poly}(n)})$-tampering attacks where n is the security parameter.

Keywords Tampering, Randomness, Fourier Analysis, Encryption.

[*] Aalto University and KTH Royal Institute of Technology `austrin@kth.se`. Work done while at University of Toronto, funded by NSERC

[†] Cornell {`mohammad,chung`}`@cs.cornell.edu`. Supported in part by NSF Award CNS-1217821.

[‡] Cornell {`rafael`}`@cs.cornell.edu`. Pass is supported in part by a Alfred P. Sloan Fellowship, Microsoft New Faculty Fellowship, NSF Award CNS-1217821, NSF CAREER Award CCF-0746990, NSF Award CCF-1214844, AFOSR YIP Award FA9550-10-1-0093, and DARPA and AFRL under contract FA8750-11-2- 0211. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the US Government

[§] Cornell {`karn`}`@cs.cornell.edu`.

Contents

1 Introduction

A traditional assumption in cryptography is that the only way for an attacker to gather or control information is by receiving and sending messages to honest parties. In particular, it is assumed that the attacker may not access the *internal states* of honest parties. However, such assumptions on the attacker—which we refer to as *physical assumptions*—are quite strong (and even unrealistic). In real-life, an attacker may through a "physical attack" learn some "leakage" about the honest parties' internal states and may even tamper with their internal states. For instance, a computer virus may (e.g., using a, so-called, buffer overflow attack [Ale96, Fry00, PB04]) be able to bias the randomness of an infected computer. Understanding to what extents the traditional physical assumptions can be relaxed, to capture such realistic attacks, is of fundamental importance.

Indeed, in recent years *leakage-resilient cryptography*—that is, the design of cryptographic schemes and protocols that remain secure even when the attacker may receive (partial) leakage about the internal state of the honest parties—has received significant attention (see e.g., [MR04, DP08, AGV09, BKKV10, DHLAW10, DGK$^+$10, KLR12, GR12, LL12, DSK12, Rot12]).

In this work, we focus on understanding the power of *tampering attacks*—that is, attacks where the adversary may partially modify (i.e., tamper with) the internal state of honest parties. Early results in the 1990's already demonstrate that tampering attacks may be very powerful: by just slightly tampering with the computation of specific implementations of some cryptographic schemes (e.g., natural implementations of RSA encryption [RSA78]), Boneh, DeMillo and Lipton [BDL97] demonstrated that the security of these schemes can be broken completely.

Previous works on tamper-resilient cryptography consider tampering of computation [AK96, BDL97,BS97,IPSW06,FPV11,LL12] and tampering with the memory of an honest party who holds a secret (e.g., a signing or a decryption algorithm) [GLM$^+$04,DPW10,LL10,KKS11,LL12,CKM11]. This line of research culminated in strong *compilers* turning any polynomial-size circuit C into a new "tamper-resilient" polynomial-size circuit C'; tamper-resilience here means that having "grey-box" access to C' (i.e., having black-box access while tampering with the computation of C') yields no more "knowledge" than simply having black-box access to C. These works, thus, show how to keep a secret hidden from a tampering attacker. Our focus here is somewhat different. In analogy with recent work of leakage-resilient security, we aim to investigate to what extent a tampering attacker may violate the *security* of a cryptographic protocol by tampering with the internal state of honest parities. Namely, we study the following question:

> *Can security of cryptographic primitives be preserved under tampering attacks to the computation of honest parties?*

For concreteness, let us focus on the security of public-key encryption schemes (but as we shall see shortly, our investigation applies to many more cryptographic tasks such as zero-knowledge proofs and secure computation). Roughly speaking, we require a tamper-resilient encryption scheme to guarantee that ciphertexts conceal the encrypted messages, even if the internal computation of the sender (of the ciphertext) has been tampered with.[1]

[1] Let us remark that the simulation property of tamper-resilient compilers do not necessarily guarantee that if the sender algorithm is compiled into a "tamper-resilient" version, then the encryption scheme is tamper-resilient. This is due to the fact that the simulation property of those compilers only guarantee that an attacker cannot learn more from tampering with the sender strategy than it could have with black-box access to it. But in the case of encryption schemes, it is actually the *input* to the algorithm (i.e., the message to be encrypted) that we wish to hide (as opposed to some secret held by the algorithm). See Appendix C for a more detailed outline of the previous work.

A first observation is that if the attacker may completely change the computation of the sender, he could simply make the sender send the message in the clear. Thus, to hope for any reasonable notion of tamper-resilient security we need to restrict the attacker's ability to tamper with the computation. In this work we consider tampering attacks only to the randomness of the honest players. Note that even here we need to restrict the tampering ability of the attacker, for otherwise the adversary can effectively make the scheme deterministic by always fixing the randomness of the honest parties to all zeros. But it is well-known that deterministic encryption schemes cannot be semantically secure. Therefore, here we initiate study of the power of *weak* types of tampering attacks to the randomness of the honest parties.

Online Tampering with Randomness. We envision the adversary as consisting of two separate entities: **(1)** a *classical attacker* who interacts with the honest parties only by sending/receiving messages to/from them (without any side-channels), and **(2)** a *tampering circuit* (a.k.a. the "virus") who observes the internal state of the honest parties and may *only* tamper with their randomness (but may not communicate with the outside world, and in particular with the classical attacker). The tampering circuit only gets to tamper with a small fraction of the random bits, and furthermore needs to tamper with them one-by-one, efficiently, and in an on-line fashion. More precisely, we consider a so-called *p-tampering attack*, where the adversary gets to tamper with the random tape of the honest players as follows. The randomness of honest parties is generated bit-by-bit, and for each generated bit x_i the efficient tampering circuit gets to tamper with it with independent probability p having only knowledge of previously generated random bits $x_1, x_2, \dots, x_{i-1}$ (but not the value of the random bits tossed in the future). Roughly speaking, requiring security with respect to p-tampering attacks amounts to requiring that security holds even if the honest players' randomness comes from a *computationally efficient* analog of a Santha-Vazirani (SV) source [SV86]. Recall that a random variable $X = (X_1, \dots, X_n)$ over bit strings is an SV source with parameter δ if for every $i \in [n]$ and every $(x_1, \dots, x_i) \in \{0,1\}^i$, it holds that $\delta \leq \mathsf{P}[X_i = x_i | X_1 = x_1, \dots, X_{i-1} = x_{i-1}] \leq 1 - \delta$. It is easy to see that the random variable resulting from performing a p-tampering attack on a uniform n-bit string is an SV source with parameter $(1-p)/2$; in fact, any SV source is equivalent to performing a *computationally unbounded* p-tampering attack on a uniform n-bit string.

The main focus of this work is on the following question:

Can security be achieved under p-tampering attacks?

Before describing our results we note that previous works on *resettably cryptography* [CGGM00] can be interpreted as achieving tamper resilience against adversaries who tamper with the randomness of the honest parties by *resampling* the randomness of the honest parties and executing them again and again. This is incomparable to our model, since our adversary does not have control over the honest parties' execution (to run them again), but is more powerful in that it could change the value of some of the random bits.

1.1 Our Results

Our main result is a strong negative answer to the question above for a variety of basic cryptographic primitives. A p-tampering attacker can break all of the following with advantage $\Omega(p)$: **(1)** the security of any CPA-secure (public-key or private-key) encryption scheme, **(2)** the zero-knowledge property of any efficient-prover proof (or argument) system for nontrivial languages, **(3)** the hiding property of any commitment scheme, and **(4)** the security of any protocol for computing a "nontrivial" finite function. More formally, we prove the following theorems.

Theorem 1.1 (Impossibility of Encryption)**.** *Let Π be any CPA-secure public-key encryption scheme. Then a p-tampering attacker can break the security of Π with advantage $\Omega(p)$. Moreover, the attacker only tampers with the random bits of the* encryption *(and not the key-generation) and does so* without *knowledge of the encrypted message.*

Recall that in a *multi-message* secure encryption scheme, the encryptions of any two sequences of messages (of the same lengths) are indistinguishable. Theorem 1.1 extends to any private-key encryption scheme that is multi-message secure.

Theorem 1.2 (Impossibility of Zero-Knowledge)**.** *Let (P, V) be an efficient prover proof/argument system for a language $L \in \mathsf{NP}$ such that the view of any p-tampering verifier can be simulated by an efficient simulator with indistinguishability gap $o(p)$, then the language L is in BPP.*

Theorem 1.3 (Impossibility of Commitments)**.** *Let (S, R) be a bit-commitment scheme. Then, either an efficient malicious sender can break the biding with advantage $\Omega(p)$ (without tampering), or an efficient malicious p-tampering receiver can break the hiding with advantage $\Omega(p)$.*

Following [GLM$^+$04] we consider the computation of two-party functions $f : \mathcal{D}_1 \times \mathcal{D}_2 \mapsto \mathcal{R}$ where only one player gets the output. A function f is called trivial in this context, if there is a deterministic single-message (i.e., only one player speaks) protocol for computing f that is information theoretically secure.

Theorem 1.4 (Impossibility of Secure Computation)**.** *The security of any protocol for computing a two-party non-trivial function can be broken with advantage $\Omega(p)$ through a p-tampering attack.*

Tampering with Randomness vs. Imperfect Randomness. Our negative results are closely related to the impossibility result of Dodis et al. [DOPS04] on the "impossibility of cryptography with imperfect randomness", where the security of cryptographic primitives are analyzed assuming that the honest parties only have access to randomness coming from an SV source (as opposed to the randomness being perfectly uniform). [DOPS04] present several strong impossibility results for secure realizability of cryptography primitives in a setting where players only have access to such imperfect randomness. The SV sources they consider for their impossibility results, however, may not be efficiently computable.

The key-difference between tamper-resilient security in our setting and security with imperfect randomness is that we restrict to randomness sources that are *efficiently samplable* through an (online) p-tampering attack; thus achieving tamper-resilient security becomes easier than resilience to imperfect randomness. (Another less important difference is that for primitives with simulation-based security, we allow the simulator to depend on the p-tampering attacker, whereas [DOPS04] the simulator must work for any randomness source; this further makes acheiving tamper-resilient security easier than resilience to imperfect randomness.)

Positive Results. We complement the above negative results by demonstrating some initial positive results: Assuming the existence of one-way functions, for any $p = n^{-\alpha}$, where $\alpha > 0$ is a constant and n is the security parameter, every implementation of signature schemes, identification protocols, and witness hiding protocols can be made resilient against p-tampering attackers. We additionally present a relaxed notion of semantic security for encryption schemes that can be achieved under p-tampering attacks. Since the above mentioned primitives already imply the existence of one-way functions [IL89], therefore preventing against $n^{-\alpha}$-tampering attacks can be achieved for these primitives unconditionally. Finally, we present positive results for tamper-resilient key-agreement and secure computation in the presence of two honest players. We provide the details on this in Appendix B.

1.2 Our Techniques

Our main technical contribution is the development of new methods for biasing Boolean, and more generally, bounded-value functions, using a p-tampering attack.

1.2.1 Biasing Bounded-Value Functions

Our first (negative) result uses elementary Fourier analysis to prove an efficient version of the Santha-Vazirani theorem: Any balanced (or almost balanced) efficiently computable Boolean function f can be biased by $\Omega(p)$ through an efficient p-tampering attack.

Specifically, let U_n denote the uniform distribution over $\{0,1\}^n$ and let $U_n^{\mathsf{Tam},p}$ denote the distribution obtained after performing a p-tampering attack on U_n using a tampering algorithm Tam; more precisely, let $U_n^{\mathsf{Tam},p} = (X_1, \ldots, X_n)$ where with probability $1-p$, X_i is a uniform random bit, and with probability p, $X_i = \mathsf{Tam}(1^n, X_1, \ldots, X_{i-1})$.

Theorem 1.5. (Biasing Boolean Functions: Warm-up). *There exists an oracle machine* Tam *with input parameters n and $\varepsilon < 1$ that runs in time* $\mathrm{poly}(n/\varepsilon)$ *and for every $n \in N$ and $\varepsilon \in (0,1)$, every Boolean function $f : \{0,1\}^n \to \{-1,1\}$, and every $p < 1$, for $\mu = \mathsf{E}[f(U_n)]$ it holds that*

$$\mathsf{E}[f(U_n^{\mathsf{Tam}^f,p})] \geq \mu + p \cdot (1 - |\mu| - \varepsilon).$$

The tampering algorithm Tam is extremely simple and natural; it just greedily picks the bit that maximizes the bias at every step. More precisely, $\mathsf{Tam}^f(x_1, \ldots, x_{i-1})$ estimates the value of $\mathsf{E}_{U_{n-i}}[f(x_1, \ldots, x_{i-1}, b, U_{n-i})]$ for both of $b = 0$ and $b = 1$ by sampling, and sets x_i to the bit b with larger estimated expectation.

Theorem suffices for our impossibility result for tamper-resilient zero-knowledge. For all our remaining impossibility results, however, we need a more general version that also deals with bounded value functions $f : \{0,1\}^n \to [-1,1]$. Our main technical theorem provides such a result.

Theorem 1.6. (Main Technical Theorem: Biasing Bounded-Value Functions). *There exists an efficient oracle machine* Tam *such that for every $n \in N$, every bounded-value function $f : \{0,1\}^n \to [-1,1]$, and every $p < 1$,*

$$\mathsf{E}[f(U_n^{\mathsf{Tam}^f,p})] \geq \mathsf{E}[f(U_n)] + \frac{p \cdot \mathrm{Var}[f(U_n)]}{5}.$$

Note that in Theorem 1.6 the dependence on the variance of f is necessary because f may be the constant function $f(x) = 0$, whereas for the case of balanced Boolean functions this clearly cannot happen. Let us also point out that we have not tried to optimize the constant $1/5$, and indeed it seems that a more careful analysis could be used to bring it down since for small p the constant gets close to 1.

The greedy algorithm does not work in the non-Boolean case anymore. The problem, roughly speaking, is that a greedy strategy will locally try to increase the expectation, but that might lead to choosing a wrong path. As a "counter-example" consider a function f such that: conditioned on $x_1 = 0$ f is a constant function ε, but conditioned on $x_1 = 1$, f is a Boolean function with average $-\varepsilon$. For such f, the greedy algorithm will set $x_1 = 0$ and achieves bias at most ε, while by choosing

$x_1 = 1$ more bias could be achieved. To circumvent this problem we use a "mildly greedy" strategy: we take only one sample of $f(\cdot)$ by choosing $x'_i, x'_{i+1}, \ldots, x'_n$ at random ($x_1, \ldots, x_{i-1}$ are already fixed). Then, we keep the sampled x'_i with probability proportional to how much the output of f is close to our "desired value", and flip the value of x'_i otherwise.

More precisely, $\mathsf{Tam}(1^n, x_1, \ldots, x_{i-1})$ proceeds as follows:

- Samples $(x'_i, x'_{i+1}, \ldots, x'_n) \leftarrow U_{n-i+1}$ and compute $y = f(x_1, \ldots, x_{i-1}, x'_i, \ldots, x'_n)$.
- Sample $\mathsf{Tam}(1^n, x_1, \ldots, x_{i-1})$ from a Boolean random variable with expectation $y \cdot x'_i$. Namely, output x'_i with probability $\frac{1+y}{2}$, and $-x'_i$ with probability $\frac{1-y}{2}$.

Note that our mildly greedy strategy is even easier to implement than the greedy one: to tamper with each bit, it queries f *only once*.

1.2.2 Impossibility Results for Tamper-Resilience

We employ the biasing algorithms of Theorems 1.2.1 and 1.6 to obtain our negative results using the following blue-print: primitives such as encryption, commitment schemes, secure computation of non-trivial functions, and zero-knowledge for non-trivial languages, need to communicate messages with "high" min-entropy; thus an attacker may apply a seeded extractor to the transcript of the protocol to extract out a single unbiased bit. Next, using the above biasing results, the tampering circuit may "signal" something about the secret state of honest parties by biasing the (originally) unbiased extracted bit toward the secret bit.

As previously mentioned, Dodis et al. [DOPS04] prove impossibility results for the above primitives when security is required to hold also when the honest players' randomness comes from an SV source. The main difference is that our lower bounds hold even with respect to efficiently generated (on-line) SV sources of randomness. As mentioned above, another difference is that for simulation-based definitions of security (such as the one for zero-knowledge proofs), the definitions in [DOPS04] requires the existence of a universal simulator that should handle *any* source of imperfect randomness (without knowledge of the source). In our setting, since we view the tampering attacker as being part of the adversary, we allow the simulator to depend on the tampering circuit (as such our definition is easier to achieve, and so the impossibility result is stronger). On the other hand, our results are weaker than those of [DOPS04] in that we can only rule out protocols where the honest players are polynomial-time (whereas [DOPS04], for instance, rule out zero-knowledge proofs also with inefficient provers).

We start by outlining the general idea behind the lower bound of Theorem 1.2 for zero-knowledge protocols, and next briefly sketch how these ideas are extended to the other primitives of Theorems 1.3, 1.1, and 1.4.

Breaking Zero-Knowledge through Tampering. We prove that no language outside BPP can have efficient-prover zero-knowledge proofs that remain zero-knowledge against any tampering attack performed by the verifier.[2] In particular we show that for any language $L \in \mathsf{NP}$ which remains zero-knowledge against $(1/\operatorname{poly}(n))$-tampering adversaries, one can use the simulator (which simulates the view of the *tampering* verifier) to completely reconstruct the witness w for $x \in L$.

The outline of proving Theorem 1.2 is as follows. We design a p-tampering verifier V^* whose tampering circuit Tam, after getting access to the private input y of the prover, (which is also a

[2] Note that since we require an efficient prover, the languages that we study will fall into NP. Also we only require *weak*-zero knowledge where the simulator's output shall be $o(p)$-indistinguishable from the real transcript.

witness for $x \in L$) tampers with the randomness of the prover, in a way that the honest behavior of the prover P will reveal a bit of $y = (y_1, \ldots, y_m)$. Note that the verifier V^* is able to apply any efficient Boolean function $f(\cdot)$ defined over the transcript τ of its interaction with P (which is essentially the only way the verifier can obtain a "bit" of information about the secret state of P). Thus, the attacker V^* tries to find a pair (f, Tam) where f is an efficient function, Tam is a p-tampering circuit sent to the prover's side, and the output of $f(\tau)$ correlates with y_i where the transcript τ is generated through the interaction with a prover whose randomness comes from the p-tampering source generated by the tampering circuit Tam_y.

The malicious verifier V^* employs the efficient attack of Theorem 1.2.1 as follows. Suppose f is a Boolean function that the verifier can compute over the transcript after its interaction with a tampered prover. Then, assuming f is (almost) unbiased, by Theorem 1.2.1 there is an efficient p-tampering attacker that could bias the output of f by at least $\Omega(p)$ toward any bit of $y = (y_1, \ldots, y_m)$. To get such an unbiased function f we show that (even conditioned on the verifier's randomness) the transcript of any zero-knowledge proof system must have min-entropy at least $\omega(\log n)$, and therefore we can simply take f to be a seeded randomness extractor.

To prove the $\omega(\log n)$ lower bound on the min-entropy of the transcript of any zero-knowledge proof system, we strengthen the result of [GO94] which proved that the prover in any zero-knowledge proof system needs to be randomized (i.e., that the transcript has *positive* min-entropy).

Public-Key Encryption. The CPA security of public-key encryption schemes implies that even after sampling and fixing the encryption key, the encrypted messages should have min-entropy at least $\omega(\log n)$ due solely to the randomness used in the encryption algorithm (otherwise the CPA security can be broken trivially). Thus, similarly to the case of zero-knowledge proofs described above, a Boolean function f chosen from a family of pairwise independent functions would extract an (almost) unbiased bit from the encrypted message. The tampering adversary chooses the function f at random and also would generate a tampering circuit Tam in such a way that by tampering with the bits of the encryption algorithm the average of the function $f(\mathrm{Enc}(\cdot))$ would "signal" the encrypted message $m_b \in \{m_0, m_1\}$. Note that if the tampering circuit Tam gets access to the encrypted message $m_b \in \{m_0, m_1\}$, it can simply use the biasing attack of Theorem 1.2.1 for Boolean functions to bias $f(\mathrm{Enc}(m_b, r))$ towards b by $\Omega(p)$ where r is the randomness used by the encryption algorithm. This would let the adversary guess which message is encrypted with advantage $\Omega(p)$. However, by using the more general attack of Theorem 1.6 we can obtain a tampering attacker who signals the index b of the encrypted message $m_b \in \{m_0, m_1\}$ *without* the knowledge of m_b. This can be done by biasing the *non-Boolean* (yet bounded) function $g(r) = f(\mathrm{Enc}(m_1, r)) - f(\mathrm{Enc}(m_0, r))$ toward 1. The intuition is that now $f(\mathrm{Enc}(m_1, r))$ is biased toward 1 *more than* the amount of bias that is imposed on $f(\mathrm{Enc}(m_0, r))$ toward 1. Since both of these functions were unbiased at the beginning, it is easy to see that after biasing $g(\cdot)$ toward 1 the value of $f(\mathrm{Enc}(\cdot))$ can be used to guess the encrypted message m_b with advantage $1/2 + \Omega(p)$.

Commitment Schemes. If a commitment scheme is statistically binding, then it is easy to show that (conditioned on any receiver's randomness) the transcript of the commitment phase has min-entropy $\omega(\log n)$. In this case, we again follow the same framework for a tampering attack by having the malicious receiver apply a pair-wise independent Boolean function f to the transcript $\tau(r, b)$ of the commitment phase (where r is sender's randomness) and try to bias the output of f in a way that it signals the committed bit b. The problem is that, if the scheme is not statistically binding and the transcript does not have enough entropy, the function $g(r) = f(\tau(r, 0)) - f(\tau(r, 1))$ might be always a constant function for *every* choice of f. In this case we might not be able to signal the bit b by biasing g. Nevertheless, by relying on the binding property of the commitment scheme, we show that any Boolean function f is either good enough already to signal the bit b *without* any tampering, or it holds that the function $g(r) = f(\tau(r, 0)) - f(\tau(r, 1))$ will have enough variance so that the tampering circuit can signal the bit b by biasing g. It is crucial here that a receiver can efficiently *test* whether a chosen function f can to signal b or not. This can be done by simply simulating the commitment phase many times to test the quality of f to signal b.

Secure Function Evaluation. We prove our impossibility result against tamper-resilient secure function evaluation by showing that the task of (tamper-resilient) secure function evaluation of finite functions, where only one party gets the output, implies some form of (tamper-resilient) *weakly* secure commitment. We then observe that our negative result of Theorem 1.3 applies to weakly-secure commitment schemes as well.

2 Preliminaries

Notation. By a *negligible function*, denoted as $\mathrm{negl}(n)$, we mean any function which is of the form $n^{-\omega(1)}$. By a *noticeable* function $f(n)$ we mean any function of the form $n^{\Omega(1)}$. We use the notation PPT do denote *probabilistic polynomial time.* We might use the terms "efficient" and PPT interchangeably. For interactive algorithms (A, B) and $C \in \{A, B\}$, by $\mathsf{View}_C\langle A, B\rangle(x)$ we denote the view of C in an execution of the interaction between A, B, where this view includes the common input, the private randomness of C, and the messages exchanged. By $\Delta(X, Y)$ we denote the *statistical distance* between the random variables X, Y defined as $\frac{1}{2}\sum_x |\mathsf{P}[X = x] - \mathsf{P}[Y = x]|$. By $\mathrm{H}_\infty(X)$ we denote the *min-entropy* of the random variable X defined as the largest k such that $\mathsf{P}[X = a] \leq 2^{-k}$ for every a. We call a sequence $\{\mathbf{X}_x\}$ of random variables indexed by $x \in I \subseteq \{0, 1\}^*$ an *ensemble* of random variables. We call two ensembles of random variables $\{\mathbf{X}_x\}$ and $\{\mathbf{Y}_x\}$ (with the same index set) $\alpha(|x|)$-*indistinguishable* if for every polynomial $p(\cdot)$ and every sequence of Boolean circuits C_n of size $p(n)$, we have $\Delta(C_{|x|}(x, \mathbf{X}_x), C_{|x|}(x, \mathbf{Y}_x)) \leq \alpha(|x|)$. we use the term *computationally indistinguishable* to refer to the case where $\alpha(\cdot) = \mathrm{negl}(\cdot)$. For function $f \colon \{0, 1\}^n \mapsto \mathbb{R}$ by $\mathsf{E}[f(\cdot)]$ and $\mathrm{Var}[f(\cdot)]$ we mean $\mathsf{E}[f(U_n)]$ and $\mathrm{Var}[f(U_n)]$.

Definition 2.1 (SV Sources)**.** *The random bit string $X = (X_1, \ldots, X_n)$ is a Santha-Vazirani (SV) source with parameter δ if for every $i \in [n]$ and every $(x_1, \ldots, x_i)$, it holds that $\delta \leq \mathsf{P}[X_i = x_i | X_1 = x_1, \ldots, X_{i-1} = x_{i-1}] \leq 1 - \delta$. It is easy to see that $\mathrm{H}_\infty(X) \geq n \cdot \lg(1 - \delta)$ holds for any SV source $X = (X_1, \ldots, X_n)$ with parameter δ.*

We provide standard definitions of multi-message and CPA secure encryption schemes, interactive (zero-knowledge) proofs, and commitment schemes in Appendix A.

3 Biasing Functions via Online Tampering

In this section we study how much the output of a bounded function can be biased through a tampering attack. First we formally define an online tampering process and a tampering source of randomness (as a result of an online tampering attack performed on a uniform source of randomness).

Definition 3.1. *A distribution $X = (X_1, \ldots, X_n)$ over $\{-1, 1\}^n$ is an (efficient) p-tampering source if there exists an (efficient) tampering algorithm* Tam *such that X can be generated in an online* *fashion as follows: For $i = 1, \ldots, n$,*

$$X_i = \begin{cases} \mathsf{Tam}(1^n, X_1, \ldots, X_{i-1}) & \textit{with probability } p, \\ U_1^i & \textit{with probability } 1-p, \end{cases}$$

where U_1^i denotes a uniformly random bit over $\{-1,1\}$. In other words, with probability p, Tam *gets to tamper the next bit with the knowledge of the previous bits (after the tampering)*[3]*. The tampering algorithm* Tam *might also receive an auxiliary input and use it in its tampering strategy.*[4] *We use $U_n^{\mathsf{Tam},p}$ to denote the p-tampered source obtained by the above tampering process with tampering algorithm* Tam*.*

Note that in the definition above, the tampering algorithm Tam might be completely oblivious to the parameter p. By referring to Tam as a *p-tampering* algorithm, we emphasize on the fact that Tam's algorithm might depend on p.

Remark 3.2. *Every p-tampering source is also a Santha-Vazirani source [SV86] with parameter $\delta = (1-p)/2$. In fact, it is not hard to see that without the efficiency consideration, the two notions are equivalent.*

3.1 Preliminaries: Calculating the Effect of a Single Variable

Recall that the Fourier coefficients of any function $f : \{-1,1\}^n \to [-1,1]$ are indexed by the subsets S of $[n]$ and are defined as $\hat{f}(S) := \mathsf{E}_x[f(x)\chi_S(x)]$, where $\chi_S(x) := \prod_{i\in S} x_i$. Note that the Fourier coefficient of the empty set $\hat{f}(\emptyset)$ is simply the expectation $\mathsf{E}[f(U_n)]$.

For every prefix $x_{\le i} = (x_1, \ldots, x_i)$, let $f_{x_{\le i}} : \{-1,1\}^{n-i} \to [-1,1]$ be the restriction of f on $x_{\le i}$, i.e., $f_{x_{\le i}}(x_{i+1}, \ldots, x_n) := f(x_1, \ldots, x_n)$. We note that the variables of $f_{x_{\le i}}$ are named $(x_{i+1}, \ldots, x_n)$ and thus the Fourier coefficients of $f_{x_{\le i}}$ are $\hat{f}_{x_{\le i}}(S)$'s with $S \subseteq \{i+1, \ldots, n\}$. The following basic identity can be proved by straightforward calculation.

$$\hat{f}_{x_1}(\emptyset) = \hat{f}(\emptyset) + \hat{f}(\{1\}) \cdot x_1. \tag{1}$$

Recall that $\hat{f}(\emptyset)$ and $\hat{f}_{x_1}(\emptyset)$ are simply expectations. One interpretation of the above identity is that $\pm\hat{f}(\{1\})$ is the change of expectation when we set $x_1 = \pm 1$. This is thus useful for analyzing the bias introduced as the result of a tampering attack.

Using the above identity with a simple induction, we can express $f(x)$ as a sum of Fourier coefficients of restrictions of f. Namely, $f(x)$ equals to the expectation $\hat{f}(\emptyset)$ plus the changes in expectation when we set x_i bit by bit.

Lemma 3.3. *For every $x \in \{-1,1\}^n$, it holds that*

$$f(x) = \hat{f}(\emptyset) + \sum_{i=1}^{n} \hat{f}_{x_{\le i-1}}(\{i\}) \cdot x_i.$$

[3] In a stronger variant of tampering attacks, the attacker might be completely stateful and memorize the original values of the previous bits before and after tampering and also the places where the tampering took place, and use this extra information in its future tampering. Using the weaker stateless attacker of Definition 3.1 only makes our negative results stronger. Our *positive* results hold even against stateful attackers.

[4] The auxiliary input could, e.g., be the information that the tampering algorithm receives about the secret state of the tampered party; this information might not be available at the time the tampering circuit is generated by the adversary.

Proof. By expanding $\hat{f}_{x_{\leq j}}(\emptyset) = \hat{f}_{x_{\leq j-1}}(\emptyset) + \hat{f}_{x_{\leq j-1}}(\{j\}) \cdot x_j$, (implied by Equation (1)) and a simple induction on j it follows that:

$$f(x) = \hat{f}_{x_{\leq j}}(\emptyset) + \sum_{i=j+1}^{n} \hat{f}_{x_{\leq i-1}}(\{i\}) \cdot x_i,$$

which proves the lemma. □

As a corollary, the above lemma implies that the sum of Fourier coefficients (of restrictions of f) in absolute value is at least $|f(x)| - |\hat{f}(\emptyset)|$.

Corollary 3.4. *For every $x \in \{-1,1\}^n$, it holds that*

$$\sum_{i=1}^{n} \left| \hat{f}_{x_{\leq i-1}}(\{i\}) \right| \geq |f(x)| - |\hat{f}(\emptyset)|.$$

Proof. We have

$$\sum_{i=1}^{n} \left| \hat{f}_{x_{\leq i-1}}(\{i\}) \right| = \sum_{i=1}^{n} \left| \hat{f}_{x_{\leq i-1}}(\{i\}) \cdot x_i \right| \geq$$
$$\left| \sum_{i=1}^{n} \hat{f}_{x_{\leq i-1}}(\{i\}) \cdot x_i \right| = |f(x) - \hat{f}(\emptyset)| \geq |f(x)| - |\hat{f}(\emptyset)|$$

where both inequalities follow by the triangle inequality, and the second equality uses Lemma 3.3. □

3.2 Warmup: The Boolean Case

A seminal result by Santha and Vazirani [SV86] shows that for every balanced Boolean function f (e.g., a candidate "extractor"), there exists a p-tampering source X that biases the output of f by at least p. We now present a strengthening of this result that additionally shows that if the function f is efficiently computable, then the source X could be an efficient p-tampering one (and only needs to use f as a black box). In the language of extractors, our result thus proves a strong impossibility result for deterministic randomness extraction from "efficient" Santha-Vazirani sources. Our proof of the generalized result is quite different (and in our eyes simpler) than classic proofs of the Santha-Vazirani theorem and may be of independent interest.

Theorem 1.2.1 (Restated)**.** (Biasing Boolean Functions: Warm-up). *There exists an oracle machine* Tam *with input parameters n and $\varepsilon < 1$ that runs in time* $\mathrm{poly}(n/\varepsilon)$ *and for every $n \in N$ and $\varepsilon \in (0,1)$, every Boolean function $f : \{0,1\}^n \to \{-1,1\}$, and every $p < 1$, for $\mu = \mathsf{E}[f(U_n)]$ it holds that*

$$\mathsf{E}[f(U_n^{\mathsf{Tam}^f,p})] \geq \mu + p \cdot (1 - |\mu| - \varepsilon).$$

Proof of Theorem 1.2.1. Let us first present a proof with an inefficient tampering algorithm achieving bias $p \cdot (1 - |\mu|)$; next, we show how to make it efficient while not loosing much in bias. On input $x_{\leq i-1} = (x_1, \ldots, x_{i-1})$, Tam sets $x_i = \mathrm{sgn}(\hat{f}_{x_{\leq i-1}}(\{i\}))$. By Equation (1), $\hat{f}_{x_{\leq i-1}}(\{i\})$ corresponds to the change in expectation of $f_{x_{\leq i-1}}$ when setting the value of x_i. This amounts to greedily choosing the x_i that increases the expectation. Let $X = U_n^{\mathsf{Tam},p}$. By applying Lemma 3.3 and the linearity of expectations, we have

$$\mathsf{E}[f(X)] = \hat{f}(\emptyset) + \sum_{i=1}^{n} \mathsf{E}_X \left[\hat{f}_{X_{\leq i-1}}(\{i\}) \cdot X_i\right] =$$

$$\hat{f}(\emptyset) + \sum_{i=1}^{n} \mathsf{E}_{X_{\leq i-1}} \left[\hat{f}_{X_{\leq i-1}}(\{i\}) \cdot \mathsf{E}[X_i | X_{\leq i-1}]\right].$$

Since Tam tampers with the i'th bit with independent probability p, therefore $\mathsf{E}[X_i|X_{\leq i-1}] = p \cdot \operatorname{sgn}(\hat{f}_{X_{\leq i-1}}(\{i\})$, and so it holds that

$$\mathsf{E}[f(X)] = \hat{f}(\emptyset) + p \cdot \sum_{i=1}^{n} \mathsf{E}_X \left[\left|\hat{f}_{X_{\leq i-1}}(\{i\})\right|\right] =$$

$$\hat{f}(\emptyset) + p \cdot \mathsf{E}_X \left[\sum_{i=1}^{n} \left|\hat{f}_{X_{\leq i-1}}(\{i\})\right|\right] \geq \hat{f}(\emptyset) + p \cdot (1 - \hat{f}(\emptyset))$$

where the last inequality follows by Corollary 3.4.

Note that the above tampering algorithm Tam in general is not efficient since computing $\hat{f}_{x_{\leq i-1}}(\{i\})$ exactly may be hard. However, we show that Tam may approximate $\hat{f}_{x_{\leq i-1}}(\{i\})$ using $M = \Theta(\frac{n^2}{\varepsilon^2} \cdot \log \frac{n}{\varepsilon})$ samples, and set x_i according to the sign of the approximation of $\hat{f}_{x_{\leq i-1}}(\{i\})$, while still inducing essentially the same bias. This clearly can be done efficiently given oracle access to f. As before, let $X = U_n^{\mathsf{Tam}^f,p}$ denote the corresponding p-tampering source. To lower bound $\mathsf{E}[f(X)]$, we note that the only difference from the previous case is that $\mathsf{Tam}(1^n, x_{\leq i-1})$ is no longer always outputting $\operatorname{sgn}(\hat{f}_{x_{\leq i-1}}(\{i\}))$. Nevertheless, we claim that for every $x_{<i}$ it holds that

$$\hat{f}_{x_{\leq i-1}}(\{i\}) \cdot \mathsf{E}[X_i | X_{\leq i-1} = x_{\leq i-1}] \geq p \cdot \left(|\hat{f}_{x_{\leq i-1}}(\{i\})| - \varepsilon/n\right)$$

since either (i) $|\hat{f}_{x_{\leq i-1}}(\{i\})| \geq \varepsilon/2n$ in which case (by a standard Chernoff bound) Tam outputs $\operatorname{sgn}(\hat{f}_{x_{\leq i-1}}(\{i\}))$ with probability at least $1 - \varepsilon/2n$, or (ii) $|\hat{f}_{x_{\leq i-1}}(\{i\})| < \varepsilon/2n$ in which case the inequality holds no matter what Tam outputs since $|\mathsf{E}[X_i|X_{\leq i-1} = x_{\leq i-1}]| \leq p$. A lower bound on $\mathsf{E}[f(X)]$ then follows by the same analysis as before.

$$\begin{aligned}\mathsf{E}[f(X)] &\geq \hat{f}(\emptyset) + p \cdot \sum_{i=1}^{n} \mathsf{E}_X \left[\left|\hat{f}_{X_{\leq i-1}}(\{i\})\right| - \varepsilon/n\right] \\ &\geq \mu + p \cdot (1 - |\hat{f}(\emptyset)| - \varepsilon).\end{aligned}$$

□

3.3 The General Case: Tampering with Bounded-Value Functions

We further consider the more general case of tampering non-Boolean, bounded-value functions. We present an efficient tampering algorithm that biases the expectation of the function by an amount linear in the variance of the function.

Theorem 1.6 (Restated)**.** (Main Technical Theorem: Biasing Bounded-Value Functions). *There exists an efficient oracle machine* Tam *such that for every* $n \in N$*, every bounded-value function* $f : \{0,1\}^n \to [-1,1]$*, and every* $p < 1$*,*

$$\mathsf{E}[f(U_n^{\mathsf{Tam}^f,p})] \geq \mathsf{E}[f(U_n)] + \frac{p \cdot \mathrm{Var}[f(U_n)]}{5}.$$

The following lemma follows similarly to lemma 3.3, but instead it relies on a squared version of Equation (1).

Lemma 3.5. *For every $x \in \{-1,1\}^n$,*

$$f(x)^2 = \hat{f}(\emptyset)^2 + \sum_{i=1}^{n} \left(\hat{f}_{x_{\leq i-1}}(\{i\})^2 + 2\hat{f}_{x_{\leq i-1}}(\emptyset) \cdot \hat{f}_{x_{\leq i-1}}(\{i\}) \cdot x_i \right).$$

Proof. Squaring Equation (1), and recalling that $x_i^2 = 1$ since $x_i \in \{-1,1\}$, for every f we obtain

$$\hat{f}_{x_1}(\emptyset)^2 = \hat{f}(\emptyset)^2 + \hat{f}(\{1\})^2 + 2\hat{f}(\emptyset) \cdot \hat{f}(\{1\}) \cdot x_1.$$

By expanding

$$\hat{f}_{x_{\leq j}}(\emptyset)^2 = \hat{f}_{x_{\leq j-1}}(\emptyset)^2 + \hat{f}_{x_{\leq j-1}}(\{j\})^2 + 2\hat{f}_{x_{\leq j-1}}(\emptyset) \cdot \hat{f}_{x_{\leq j-1}}(\{j\}) \cdot x_j$$

and using a simple induction over j it follows that

$$f(x)^2 = \hat{f}_{x_{\leq j}}(\emptyset)^2 + \sum_{i=j+1}^{n} \left(\hat{f}_{x_{\leq i-1}}(\{i\})^2 + 2\hat{f}_{x_{\leq i-1}}(\emptyset) \cdot \hat{f}_{x_{\leq i-1}}(\{i\}) \cdot x_i. \right),$$

which proves the lemma. □

To prove Theorem 1.2.1, we define a *Fourier tampering algorithm* FTam that on input $x_{\leq i-1}$, tampers with bit i in a way that makes $\mathsf{E}[\mathsf{FTam}(x_{\leq i-1})] = \hat{f}_{x_{\leq i-1}}(\{i\})$. In other words, FTam outputs $X_i = \pm 1$ with probability $(1 \pm \hat{f}_{x_{\leq i-1}}(\{i\}))/2$. Interestingly, such a FTam can be implemented extremely efficiently by making only a single query to f, as follows. On input $x_{\leq i-1} = (x_1, \ldots, x_{i-1})$:

- FTam samples uniformly random $(x'_i, \ldots, x'_n) \leftarrow U_{n-i+1}$, and computes $y = f(x_1, x_{i-1}, x'_i, \ldots, x'_n)$.
- FTam outputs a Boolean random bit X_i with expectation $y \cdot x'_i$ (i.e., $X_i = x'_i$ with probability $(1+y)/2$, and $X_i = -x'_i$ with probability $(1-y)/2$).

The following simple claim says that indeed, FTam satisfies $\mathsf{E}[\mathsf{FTam}(x_{\leq i-1})] = \hat{f}_{x_{\leq i-1}}(\{i\})$.

Claim 3.6. *For every $x_{\leq i-1} = (x_1, \ldots, x_{i-1}) \in \{-1,1\}^{i-1}$,*

$$\mathsf{P}[\mathsf{FTam}(x_{\leq i-1}) = 1] = \frac{1}{2} \cdot (1 + \hat{f}_{x_{\leq i-1}}(\{i\})),$$

or equivalently, $\mathsf{E}[\mathsf{FTam}(x_{\leq i-1})] = \hat{f}_{x_{\leq i-1}}(\{i\})$.

Proof. Let $X_i = \mathsf{FTam}(x_{\leq i-1})$. Therefore we have:

$$\mathsf{E}[X_i] = \mathsf{E}_{x \geq i \leftarrow U_{n-i+1}}[f(x) \cdot x_i] = \hat{f}_{x_{\leq i-1}}(\{i\}).$$

□

Let $X = U_n^{\mathsf{FTam}^f, p}$. Also, let the mean $\mathsf{E}[f(U_n)] = \mu$, the second moment $\mathsf{E}[f(U_n)^2] = \nu$, and the variance $\mathrm{Var}[f] = \sigma^2$ be denoted so. We lower bound $\mathsf{E}[f(X)]$ as follows.

Lemma 3.7.

$$\mathsf{E}[f(X)] - \mu \geq \frac{p}{1+2p} \cdot \left(\mathsf{E}[f(X)^2] - \nu + \sigma^2 \right).$$

Lemma 3.8.

$$\mathsf{E}[f(X)]+\frac{\mathsf{E}[f(X)^2]}{2}+\frac{\mathsf{E}[f(X)^2]^2}{4}\geq \mu+\frac{\nu}{2}+\frac{\nu^2}{4}.$$

We first use the above two lemmas to show that $\mathsf{E}[f(X)]-\mu\geq(p\sigma^2)/5$, which implies that $\mathsf{E}[f(X)]\geq\mu+p\cdot\mathrm{Var}[f]/5$, as desired. If $\mathsf{E}[f(X)^2]\geq\nu$, then Lemma 3.7 implies

$$\mathsf{E}[f(X)]-\mu\geq\frac{p}{1+2p}\cdot\sigma^2\geq\frac{1}{5}\cdot p\sigma^2.$$

For the case that $\mathsf{E}[f(X)^2]\leq\nu$, let $\alpha\triangleq\nu-\mathsf{E}[f(X)^2]\geq 0$. Lemma 3.8 implies that

$$\mathsf{E}[f(X)]-\mu\geq\frac{1}{2}(\nu-\mathsf{E}[f(X)^2])+\frac{1}{4}(\nu^2-\mathsf{E}[f(X)^2]^2)\geq\frac{\alpha}{2}$$

which together with Lemma 3.7 implies that

$$[f(X)]-\mu\geq\max\left\{\frac{p}{1+2p}\cdot\left(\sigma^2-\alpha\right),\frac{\alpha}{2}\right\}\geq\frac{p}{1+4p}\geq\frac{p\sigma^2}{5}.$$

Now we prove Lemmas 3.7 and 3.8. The proof of Lemma 3.7 is a generalization of the analysis for biasing Boolean functions.

Proof of Lemma 3.7. By applying Lemma 3.3 and the linearity of expectations, we have

$$\mathsf{E}[f(X)]=\hat{f}(\emptyset)+\sum_{i=1}^{n}\mathsf{E}_X\left[\hat{f}_{X_{\leq i-1}}(\{i\})\cdot X_i\right]=\hat{f}(\emptyset)+\sum_{i=1}^{n}\mathsf{E}_{X_{\leq i-1}}\left[\hat{f}_{X_{\leq i-1}}(\{i\})\cdot\mathsf{E}[X_i|X_{\leq i-1}]\right].$$

Since FTam gets to tamper with bit i with independent probability p and $\mathsf{E}[\mathsf{FTam}(x_{\leq i-1})]=\hat{f}_{x_{\leq i-1}}(\{i\})$, by Lemma 3.6, we have that $\mathsf{E}[X_i|X_{\leq i-1}]=p\cdot\hat{f}_{X_{\leq i-1}}(\{i\})$. Thus,

$$\mathsf{E}[f(X)]=\hat{f}(\emptyset)+p\cdot\sum_{i=1}^{n}\mathsf{E}_X\left[\hat{f}_{X_{\leq i-1}}(\{i\})^2\right].\tag{2}$$

Similarly, by applying Lemma 3.5 and the linearity of expectations, we have

$$\mathsf{E}[f(X)^2]=\hat{f}(\emptyset)^2+\sum_{i=1}^{n}\left(\mathsf{E}_X[\hat{f}_{X_{\leq i-1}}(\{i\})^2]\right)+\sum_{i=1}^{n}\left(2\mathsf{E}_{X_{\leq i-1}}[\hat{f}_{X_{\leq i-1}}(\emptyset)\cdot\hat{f}_{X_{\leq i-1}}(\{i\})\cdot\mathsf{E}[X_i|X_{\leq i-1}]]\right).$$

Simplifying using the trivial bound $|\hat{f}_{X_{\leq i-1}}(\emptyset)|\leq 1$ and $\mathsf{E}[X_i|X_{\leq i-1}]=p\cdot\hat{f}_{X_{\leq i-1}}(\{i\})$ gives

$$\mathsf{E}[f(X)^2]\leq\hat{f}(\emptyset)^2+(1+2p)\cdot\sum_{i=1}^{n}\mathsf{E}_X[\hat{f}_{X_{\leq i-1}}(\{i\})^2].\tag{3}$$

The lemma follows by combining Equations (2) and (3):

$$\begin{aligned}\mathsf{E}[f(X)]&\geq\hat{f}(\emptyset)+\frac{p}{1+2p}\left(\mathsf{E}[f(X)^2]-\hat{f}(\emptyset)^2\right)\\&=\mu+\frac{p}{1+2p}\left(\mathsf{E}[f(X)^2]-\nu+\sigma^2\right)\end{aligned}$$

where the last equality uses the fact that $\hat{f}(\emptyset)^2=\mu^2=\nu-\sigma^2$. □

The proof of Lemma 3.8 is less trivial. Our key observation is the following useful property of the Fourier tampering algorithm FTam: consider the function f together with an arbitrary function $g : \{-1,1\}^n \to [-1,1]$ (ultimately, we shall set $g(x) = f(x)^2$, but in the discussion that follows, g can be completely unrelated to f). While intuitively we expect the expectation of f to be increasing after tampering, it is clearly possible that the tampering causes the expectation of g to decrease. Nevertheless, we show that for a properly defined potential function combining the expectations of f and g, the potential is guaranteed to be non-decreasing after tampering. Namely we prove the following lemma.

Lemma 3.9. *Let $g : \{-1,1\}^n \to [-1,1]$ be an arbitrary function. For every prefix $x_{\le i} \in \{-1,1\}^i$, define a potential*

$$\Phi(x_{\le i}) := \hat{f}_{x_{\le i}}(\emptyset) + \frac{\hat{g}_{x_{\le i}}(\emptyset)}{2} + \frac{\hat{g}_{x_{\le i}}(\emptyset)^2}{4},$$

and let $\Phi := \Phi(x_{\le 0})$. Then it holds that $\mathsf{E}[\Phi(X)] \ge \Phi$.

Proof. We show that for every $x_{\le i-1} \in \{-1,1\}^{i-1}$,

$$\mathsf{E}_{x_i \leftarrow X_i | X_{\le i-1} = x_{\le i-1}}[\Phi(x_{\le i})] \ge \Phi(x_{\le i-1}).$$

To simplify the notation, let $A = \hat{f}_{x_{\le i-1}}(\emptyset), a = \hat{f}_{x_{\le i-1}}(\{i\}), B = \hat{g}_{x_{\le i-1}}(\emptyset)$, and $b = \hat{g}_{x_{\le i-1}}(\{i\})$. Using this notation we have,

$$\Phi(x_{\le i-1}) = A + B/2 + B^2/4.$$

Using Equation (1), we see that

$$\begin{aligned}\Phi(x_{\le i}) &= \hat{f}_{x_{\le i}}(\emptyset) + \frac{\hat{g}_{x_{\le i}}(\emptyset)}{2} + \frac{\hat{g}_{x_{\le i}}(\emptyset)^2}{4} \\ &= (A + a \cdot x_i) + \frac{B + b \cdot x_i}{2} + \frac{(B + b \cdot x_i)^2}{4}.\end{aligned}$$

Recall that in the tampering process, with probability $1-p$, X_i is uniformly random, and with probability p, $X_i = \mathsf{FTam}(x_{\le i-1})$ equals 1 with probability $(1 + \hat{f}_{x_{\le i-1}}(\{i\}))/2 = (1+a)/2$. Namely

$$\begin{aligned}&\mathsf{E}_{x_i \leftarrow X_i | X_{\le i-1} = x_{\le i-1}}[\Phi(x_{\le i})] \\ &= \frac{(1+pa)}{2}\left((A+a) + \frac{B+b}{2} + \frac{(B+b)^2}{4}\right) \\ &+ \frac{(1-pa)}{2}\left((A-a) + \frac{B-b}{2} + \frac{(B-b)^2}{4}\right).\end{aligned}$$

A calculation shows that

$$\begin{aligned}&\mathsf{E}_{x_i \leftarrow X_i | X_{\le i-1} = x_{\le i-1}}[\Phi(x_{\le i})] \\ &= \Phi(x_{\le i-1}) + p \cdot \left(a^2 + \frac{(1+B)ab}{2} + \frac{b^2}{4}\right) + (1-p)\frac{b^2}{4}.\end{aligned}$$

Note that since g is bounded, we have $|B| \le 1$ and thus

$$a^2 + \frac{(1+B)ab}{2} + \frac{b^2}{4} \ge |a|^2 - |a| \cdot |b| + \frac{|b|^2}{4} = (|a| - |b|/2)^2 \ge 0.$$

Therefore, $\mathsf{E}_{x_i \leftarrow X_i | X_{\le i-1} = x_{\le i-1}}[\Phi(x_{\le i})] \ge \Phi(x_{\le i-1})$. Applying the inequality iteratively shows that

$$\mathsf{E}[\Phi(X)] = \mathsf{E}_{x \leftarrow X}[\Phi(x)] \ge \mathsf{E}_{x \leftarrow X}[\Phi(x_{\le n-1})] \ge \cdots \ge \Phi.$$

□

Lemma 3.8 now follows easily.

Proof of Lemma 3.8. By applying Lemma 3.9 with $g = f^2$ and noting that $\hat{g}(\emptyset) = \nu$, we have

$$\mathsf{E}[f(X)] + \frac{\mathsf{E}[f(X)^2]}{2} + \frac{\mathsf{E}[f(X)^2]^2}{4} \ge \mu + \frac{\nu}{2} + \frac{\nu^2}{4}.$$

□

4 Impossibility of Tamper-Resilient Encryption

In this section we prove that no multi-message secure encryption scheme would remain secure in the presence of a tampering attack. At the end we show that an almost identical proof holds for private-key encryption schemes as well. We start by formally defining the notion of tamper resilient encryption.

Definition 4.1. *We call a public-key encryption scheme* p-tamper-resilient secure*, if for* every $\operatorname{poly}(n)$*-sized adversary* Adv *there exists a negligible function* $\operatorname{negl}(n)$ *such that for every sequence* $\{(x_0^n \neq x_1^n)\}_n$ *of pair of messages of equal (polynomial) length (i.e.,* $|x_0^n| = |x_1^n| = \operatorname{poly}(n)$*)* Adv *can win in the game below with probability at most* $1/2 + \operatorname{negl}(n)$ *where* n *is the security parameter.*

1. *A pair of keys* $(\mathsf{sk}, \mathsf{pk})$ *are generated and* Adv *receives the public-key* pk*.*
2. Adv *generates a* p*-tampering circuit* Tam *which can depend on* pk *and* $\{x_0^n, x_1^n\}$*.*
3. Tam *acts on* r_E*, the uniform randomness of the encryption, as a* p*-tampering circuit (see Definition 3.1) and transforms it into* r_E^{Tam}*.*
4. *The message* $x_b \in \{x_0^n, x_1^n\}$ *is chosen at random and* $c = \operatorname{Enc}_{\mathsf{pk}}(x_b, r_E^{\mathsf{Tam}})$ *is sent to* Adv*.*
5. Adv *receives* c*, outputs a guess* b'*, and wins if* $b = b'$*.*

$\mathsf{P}[$Adv *wins*$] - 1/2$ *is also called the* advantage *of* Adv*.*

Let us make a few remarks regarding Definition 4.1 follow.

- We do not allow the adversary to tamper with the randomness of the key-generation phase. One reason is that key-generation is run only once and is easier to protect than the encryption phase which might be executed many times. Note that this restriction only makes our negative result stronger.
- We require the adversary to generate the tampering circuit *without* the knowledge of the selected message x_b (even though this message finally will be present in the infected encrypting device). The reason is that the randomness r_E may be generated ahead of time and the message gets selected afterwards. Again, this restriction only makes our impossibility result stronger.

Theorem 4.2 (Impossibility of Encryption)**.** *Let* Π *be a CPA-secure public-key encryption scheme for message space* $\{0,1\}$ *and completeness* $1 - \operatorname{negl}(n)$*. For every* $p \ge 1/\operatorname{poly}(n)$*, there is an efficient* p*-tampering adversary that breaks* Π *(according to Definition 4.1) with advantage* $\Omega(p)$*.*

4.1 Proof of Theorem 4.2

In the following we prove Theorem 4.2 using Theorem 1.6.

Suppose the encryption randomness r_E has $m = \mathrm{poly}(n)$ bits. Recall that the adversary $\textsc{Adv}$ generates a tampering circuit Tam that tampers with the encryption randomness r_E of the challenger and changes its distribution from U_m into a p-tampering source $U_m^{\mathsf{Tam},p}$, receives the cipher text c, and has to guess the index b of the encrypted message with probability $1/2 + \Omega(p)$. To achieve this goal, we show how to sample an efficiently computable function $f \xleftarrow{\$} \mathcal{F}$ with range $\{0,1\}$ together with a p-tampering circuit $\mathsf{Tam}(f)$ for every f such that using Tam as the tampering circuit and applying f to the ciphertext breaks the security game of Definition 4.1. In the following we show how to find such $(\mathcal{F}, \mathsf{Tam})$. We first reduce the problem to biasing non-Boolean functions, and then will apply Theorem 1.6.

Definition 4.3. *For an implicitly fixed* pk *and a family of functions* $\mathcal{F}$ *with range* $\{0,1\}$ *and any* $f \in \mathcal{F}$*, let* $g_f(r) = \widetilde{f}(1,r) - \widetilde{f}(0,r)$ *where* $\widetilde{f}(b,r) = f(\mathrm{Enc}(x_b, r))$*. Also define* $\mathcal{G}_{\mathcal{F}} = \{g_f \mid f \in \mathcal{F}\}$*. We might skip the indexes* $f, \mathcal{F}$ *and simply write* g *and* $\mathcal{G}$ *when it is clear from the context. For such* $\mathcal{G}$ *and any oracle (tampering) algorithm* Tam *we call the pair* $(\mathcal{G}, \mathsf{Tam})$ p-detecting *iff* $\mathsf{E}_{g \xleftarrow{\$} \mathcal{G}, r \xleftarrow{\$} U_m^{\mathsf{Tam}^g,p}}[g(r)] \geq \Omega(p)$.

The following claim, formalizes the intuitive fact that finding p-detecting pairs $(\mathcal{G}, \mathsf{Tam})$ implies the type of adversaries we need for the proof of Theorem 4.2.

Claim 4.4. *For an implicitly fixed* pk *and a family of functions* $\mathcal{F}$ *with range* $\{0,1\}$*,* $\mathcal{G} = \mathcal{G}_{\mathcal{F}}$ *(as defined in Definition 4.3), and oracle algorithm* Tam*, if* $(\mathcal{G}, \mathsf{Tam})$ *is* p*-detecting, then the following adversary* $\textsc{Adv}$ *wins the security game of Definition 4.1 with advantage* $\Omega(p)$*:*

1. *Given* pk*,* $\textsc{Adv}$ *samples* $f \xleftarrow{\$} \mathcal{F}$ *and takes* $g = g_f$*.*
2. $\textsc{Adv}$ *uses the tampering circuit* $\mathsf{Tam} = \mathsf{Tam}^g$*.*
3. $\textsc{Adv}$ *outputs* $f(c)$ *where* c *is the received encryption.*

Proof. By linearity of expectation, it holds that

$$\begin{aligned}
&\mathsf{P}_{f \xleftarrow{\$} \mathcal{F}, r \xleftarrow{\$} U_m^{\mathsf{Tam}(f),p}}[\widetilde{f}(1,r) = 1] - \mathsf{P}_{f \xleftarrow{\$} \mathcal{F}, r \xleftarrow{\$} U_m^{\mathsf{Tam}(f),p}}[\widetilde{f}(0,r) = 1] \\
&= \mathsf{E}_{f \xleftarrow{\$} \mathcal{F}, r \xleftarrow{\$} U_m^{\mathsf{Tam}(f),p}}[\widetilde{f}(1,r)] - \mathsf{E}_{f \xleftarrow{\$} \mathcal{F}, r \xleftarrow{\$} U_m^{\mathsf{Tam}(f),p}}[\widetilde{f}(0,r)] \\
&= \mathsf{E}_{f \xleftarrow{\$} \mathcal{F}, r \xleftarrow{\$} U_m^{\mathsf{Tam}(f),p}}[f(\mathrm{Enc}(x_1, r)) - f(\mathrm{Enc}(x_0, r))] \\
&= \mathsf{E}_{g \xleftarrow{\$} \mathcal{G}, r \xleftarrow{\$} U_m^{\mathsf{Tam}^g,p}}[g(r)].
\end{aligned}$$

Therefore, if $(\mathcal{G}, \mathsf{Tam})$ is p-detecting, $\textsc{Adv}$ can use $(\mathcal{F}, \mathsf{Tam})$ to win the game of Definition 4.1 with advantage $\Omega(p)$. □

If the sampling of $f \xleftarrow{\$} \mathcal{F}$ is efficient and $(\mathcal{G}, \mathsf{Tam})$ is p-detecting, Claim 4.4 indeed would prove Theorem 4.2. In the following we show how to find such $\mathcal{F}$. The following claim provides a sufficient condition for obtaining a p-detecting pair $(\mathcal{G}, \mathsf{Tam})$ using the tampering algorithm of Theorem 1.6.

Claim 4.5. *Suppose* $\mathcal{G}$ *is a set of functions such that every* $g \in \mathcal{G}$ *is maps* $\{0,1\}^m$ *to* $[-1,+1]$*. If*

1. $\mathsf{E}_{g \xleftarrow{\$} \mathcal{G}} |\mathsf{E}_{r \xleftarrow{\$} U_m}[g(r)]| = o(p)$ *and*

2. $\mathsf{E}_{g \xleftarrow{\$} \mathcal{G}}[\mathrm{Var}[g(U_m)]] \geq \Omega(1)$

then there is a p-tampering oracle circuit Tam *such that* $(\mathcal{G}, \mathsf{Tam})$ *is p-detecting.*

Proof. Let Tam be the p-tampering circuit of Theorem 1.6 applied to the bounded function g. For every fixed $g \xleftarrow{\$} \mathcal{G}$ we have:

$$\mathsf{E}_{r \xleftarrow{\$} U_m^{\mathsf{Tam}^g, p}}[g(r)] \geq \Omega(p \cdot \mathrm{Var}[g(U_m]) + \mathsf{E}_{r \xleftarrow{\$} U_m}[g(r)] \geq \Omega(p \cdot \mathrm{Var}[g(U_m)]) - |\mathsf{E}_{r \xleftarrow{\$} U_m}[g(r)]|.$$

Therefore, by taking average over $g \xleftarrow{\$} \mathcal{G}$ we have:

$$\mathsf{E}_{g \xleftarrow{\$} \mathcal{G}, r \xleftarrow{\$} U_m^{\mathsf{Tam}(f)}}[g(r)] \geq \mathsf{E}_{g \xleftarrow{\$} \mathcal{G}}[\Omega(p \cdot \mathrm{Var}[g(U_m)])] - \mathsf{E}_{g \xleftarrow{\$} \mathcal{G}}|\mathsf{E}_{r \xleftarrow{\$} U_m}[g(r)]| \geq \Omega(p) - o(p) = \Omega(p).$$

□

The following claim shows that we can take $\mathcal{F}$ simply to be any family of pairwise independent functions mapping the encryptions to $\{0, 1\}$. This claim together with Claim 4.5 prove Theorem 4.2.

Claim 4.6. *Suppose $\mathcal{F}$ is a family of pairwise independent functions mapping the encryptions to $\{0, 1\}$, and let $\mathcal{G} = \mathcal{G}_f$ (as defined in Definition 4.3). Then it holds that:*

1. $\mathsf{E}_{g \xleftarrow{\$} \mathcal{G}}|\mathsf{E}_{r \xleftarrow{\$} U_m}[g(r)]| = \mathrm{negl}(n) \leq o(p)$.

2. $\mathsf{E}_{g \xleftarrow{\$} \mathcal{G}}[\mathrm{Var}[g(U_m)]] \geq \Omega(1)$.

Intuition. It is well-known that the CPA security of public key encryption schemes guarantees that the encryptions have "large" min-entropy (even for a fixed message), and therefore the pairwise independence of f will act as an extractor (see Lemma 4.7 below) implying the first condition of Claim 4.6. Moreover, the completeness of the scheme guarantees that the encryptions of any two messages $x_0 \neq x_1$ are almost always different and therefore, again by the pairwise independence of f, g will obtain $+1$ or -1 with large enough probability which guarantees that (on average) the variance is not too small. This would imply the second condition. The formal proofs follow.

We use the following variant of the leftover hash lemma.

Lemma 4.7. *Suppose X is a random variable defined over $\{0, 1\}^\ell$ and $\mathrm{H}_\infty(X) \geq k$. Let $\mathcal{F} = \{f_s \mid s \in \{0, 1\}^{2\ell}\}$ be a family of pairwise independent functions indexed by $s \in \{0, 1\}^{2\ell}$ that map $\{0, 1\}^\ell$ to $\{0, 1\}$. Then the statistical distance between $(s, f_s(X))_{s \xleftarrow{\$} U_{2\ell}}$ and $(U_{2\ell+1})$ is at most $O(1/2^k)$, which is $\mathrm{negl}(n)$ for $k = \omega(\log n)$.*

We also use the following well-known fact. See Section 4.2 for a proof (that handles also for private-key encryption schemes).

Lemma 4.8 (Large Entropy of Encryptions)**.** *Suppose Π is a CPA secure pubic-key bit-encryption scheme with completeness $1 - \mathrm{negl}(n)$. Then for any message x, with probability $1 - \mathrm{negl}(n)$ over the generation of the keys, it holds that $\mathrm{H}_\infty(\mathrm{Enc}(x)) \geq \omega(\log n)$ where the min-entropy is only over the randomness of the encryption.*

Now we prove Claim 4.6 using Lemmas 4.8 and 4.8.

First Condition of Claim 4.6: Expectation of g. By Lemmas 4.8 and 4.7 we conclude that, with overwhelming probability over the choice of the keys and over the choice of f from a family of pairwise independent functions mapping the encryptions to $\{0,1\}$, it holds that:

$$\mathsf{P}_{r \xleftarrow{\$} U_m}[f(\mathrm{Enc}(x, r_E) = 1] \in 1/2 \pm \mathrm{negl}(n)$$

in which case for $g = g_r$ (according to Definition 4.3)

$$\mathsf{E}_{r \xleftarrow{\$} U_m}[g(r)]| \leq |(1/2 \pm \mathrm{negl}(n)) - (1/2 \pm \mathrm{negl}(n))| \leq \mathrm{negl}(n)$$

which implies the first condition of Claim 4.6.

Before proving the second condition of Claim 4.6 we briefly note that, just by achieving the first condition of Claim 4.6, we would be able to use the *Boolean* biasing attack of Theorem 1.2.1 to prove Theorem 4.2 *if* the tampering circuit Tam had access to the message x_b as auxiliary input. In this case, the tampering circuit could simply bias the average of f toward b. By biasing the function g (as a non-Boolean function), however, we can obtain almost the same successful attack *without* the knowledge of the message x_b.

Second Condition of Claim 4.6: Variance of g. By using $\varepsilon = 1 - \mathrm{negl}(n)$ in the following lemma we conclude that with probability $1 - \mathrm{negl}(n)$ over the generation of the keys, the encryptions of 0 and 1 are almost always different.

Claim 4.9. *If the generated keys* $(\mathsf{sk}, \mathsf{pk})$ *are such that the scheme has completeness* $(1+\varepsilon)/2$ *and* $x_0 \neq x_1$*, then it holds that*

$$\mathsf{P}_{r_E \xleftarrow{\$} U_m}[\mathrm{Enc}(x_0, r_E) \neq \mathrm{Enc}(x_1, r_E)] \geq \varepsilon.$$

Proof. For every fixed r, if $\mathrm{Enc}(x_0, r) = \mathrm{Enc}(x_1, r)$, then the decryption algorithm can retrieve the encrypted message x_b only with probability at most $1/2$. Therefore if $\mathsf{P}_{r \xleftarrow{\$} U_m}[\mathrm{Enc}(x_0, r) = \mathrm{Enc}(x_1, r)] > 1 - \varepsilon$, then by a union bound, the decryption algorithm can only retrieve the encrypted message $x_b \xleftarrow{\$} \{x_0, x_1\}$ with probability less than $(1-\varepsilon)/2 + \varepsilon = (1+\varepsilon)/2$. This contradicts the $(1+\varepsilon)/2$ completeness of the scheme. □

For fixed keys and function $\widetilde{f}(b, r) = f(\mathrm{Enc}(x_b, r))$ let $val(\widetilde{f}) = \mathsf{P}_{r \xleftarrow{\$} U_m}[\widetilde{f}(0, r) \neq \widetilde{f}(1, r)]$.

Claim 4.10. *Let* $\mathcal{F}$ *be a family of pairwise independent functions mapping the encryptions to* $\{0,1\}$*. Then it holds that* $\mathsf{E}_{f \xleftarrow{\$} \mathcal{F}}[val(\widetilde{f})] \geq \varepsilon/2$.

Proof. We have:

$$\mathsf{E}_{f \xleftarrow{\$} \mathcal{F}}[val(\widetilde{f})] = \mathsf{E}_{f \xleftarrow{\$} \mathcal{F}}[\mathsf{P}_{r \xleftarrow{\$} U_m}[\widetilde{f}(0, r) \neq \widetilde{f}(1, r)]] =$$
$$\mathsf{E}_{r \xleftarrow{\$} U_m}[\mathsf{P}_{f \xleftarrow{\$} \mathcal{F}}[\widetilde{f}(0, r) \neq \widetilde{f}(1, r)]].$$

But note that whenever $\mathrm{Enc}(x_0, r) = \mathrm{Enc}(x_1, r)$ it holds that $\mathsf{P}_{f \xleftarrow{\$} \mathcal{F}}[\widetilde{f}(0, r) \neq \widetilde{f}(1, r)] = 0$, and whenever $\mathrm{Enc}(x_0, r) \neq \mathrm{Enc}(x_1, r)$ it holds that $\mathsf{P}_{f \xleftarrow{\$} \mathcal{F}}[\widetilde{f}(0, r) \neq \widetilde{f}(1, r)] = 1/2$ (due to the pairwise independence of $f \xleftarrow{\$} \mathcal{F}$). Therefore

$$\mathsf{E}_{r \xleftarrow{\$} U_m}[\mathsf{P}_{f \xleftarrow{\$} \mathcal{F}}[\widetilde{f}(0, r) \neq \widetilde{f}(1, r)]] =$$
$$\mathsf{P}_{r \xleftarrow{\$} U_m}[\mathrm{Enc}(x_0, r) \neq \mathrm{Enc}(x_1, r)] \cdot (1/2)$$

which by Claim 4.9 is at least $\varepsilon/2$. □

Finally we prove the second property of g for Claim 4.6.

Claim 4.11. *Let $\mathcal{F}$ be a family of pairwise independent functions mapping the encryptions to $\{0,1\}$ and let $\mathcal{G} = \mathcal{G}_{\mathcal{F}}$ (see Definition 4.3). Then $\mathsf{E}_{g \xleftarrow{\$} \mathcal{G}}[\mathrm{Var}[g(U_m)]] = \Omega(1)$.*

Proof. Claim 4.10, together with $1 - \mathrm{negl}(n)$ completeness of the encryption scheme show that the average of $val(g)$ is at least $1/2 - \mathrm{negl}(n) \geq \Omega(1)$. In addition, with overwhelming probability over the choice of f (which determines g) the average of $g(U_m)$ is within $\pm\,\mathrm{negl}(n)$. By the definition of variance, it holds that $\mathrm{Var}[g(U_m)] \geq val(g)(1 - |\mathsf{E}[g]|)^2$. Since $\mathsf{E}_f[val(g)] \geq \Omega(1)$ and $|\mathsf{E}[g]| \leq \mathrm{negl}(n) = o(1)$, therefore it holds that $\mathsf{E}_f[\mathrm{Var}[g(U_m)]] \geq \Omega(1)$. □

4.2 Private-Key Schemes

In this subsection we extend Theorem 4.2 to private key encryption schemes.

Definition 4.12. *A* p-tamper-resilient secure *private-key encryption scheme is defined similarly to Definition 4.1 with the only difference that the tampering circuit* Tam *gets to know the secret key* key *as auxiliary input.*

Note that in Definition 4.1 we did not allow Tam to access the private key, while in Definition 4.12 we do allow Tam to access the private key. This difference is justified since in a private-key encryption scheme the private-key should be stored and used in the encrypting device, and thus could be accessed by the tampering circuit, while in a public-key encryption scheme this is not necessarily the case. Moreover, there are private-key encryption schemes that remain tamper-resilient (against tampering with encryption's randomness) if the tampering circuit does not access the private key. E.g., consider the following scheme in which the message space is $\mathcal{M} = \{0,1\}^m$, the key length is k, and R is a PRG with a k-bit key, domain size n and image size m. To encrypt any message x, the encrypting algorithm chooses a random seed $r \xleftarrow{\$} \{0,1\}^n$ and takes $\mathrm{Enc}(\mathsf{key}, x, r) = (r, R_{\mathsf{key}}(r) \oplus x)$ where $\oplus$ is the bit-wise "exclusive or" operation. It is easy to see that **(1)** this scheme is multi-message secure, and **(2)** as long as $p < 1 - \omega(\log n)/n$ the p-tampering source $U_n^{\mathsf{Tam},p}$ (as a result of p-tampering attack by Tam) still has $\omega(\log n)$ min-entropy and that suffices for multi-message security of the scheme under a p-tampering attack (to encryption).

Theorem 4.13 (Impossibility of Private-Key Encryption)**.** *Let Π be a multi-message secure private-key encryption scheme for message space $\{0,1\}$ and completeness $1 - \mathrm{negl}(n)$. For every $p \geq 1/\mathrm{poly}(n)$, there is an efficient p-tampering adversary that breaks Π (according to Definition 4.12) with advantage $\Omega(p)$.*

The proof of Theorem 4.13 is indeed identical to that of Theorem 4.2 by changing the public-key to the encryption key all along. The only remaining thing is to prove the lower-bounds on the entropy of the messages in public-key and private-key encryption schemes.

Lemma 4.14 (Large Entropy of Private-Key Encryptions)**.** *The same lower-bound of Lemma 4.8 holds for any multi-message secure private-key encryption scheme.*

Proof of Lemmas 4.8 and 4.14. We first prove Lemma 4.14 for the case Π is a multi-message secure private-key scheme. We will extend the proof to public-key encryption (note that CPA security of public-key schemes imply their multi-message security as well).

Suppose on the contrary that with probability $q' = 1/\mathrm{poly}(n)$ over the choice of key, there is a message $x \in \mathcal{M}$ and a ciphertext c such that $\mathsf{P}[\mathrm{Enc}(x, \mathsf{key}, r_E) = c] = q \geq 1/\mathrm{poly}(n)$. We show

how to break the 2-message security of Π with advantage $\Omega(q' \cdot q^2) \geq 1/\operatorname{poly}(n)$ as follows. Let $x_0 = (0,0)$ and $x_1 = (0,1)$ be two tuples, each consisting of two (bit) messages. The distinguisher D, given the vector of encryptions $y = (c_0, c_1)$, outputs 0 iff $c_0 = c_1$.

If the vector y consists of the encryptions of $(0,1)$, then by the $1 - \operatorname{negl}(n)$ completeness of Π, the probability of $c_0 = \operatorname{Enc}(x_0) = \operatorname{Enc}(x_1) = c_1$ is only $\operatorname{negl}(n)$, and therefore the probability of D outputting 0 is $\operatorname{negl}(n)$. On the other hand, if y consists of the encryptions of $(0,1)$, with probability at least q^2 the encryption of two zeros becomes equal to c. Therefore with probability $q' \cdot q^2$ the distinguisher outputs 0 if y has the encryptions of $(0,0)$.

One can employ a similar and in fact simpler argument for public-key schemes (based on the fact that the encryption-key is known to the adversary). Suppose for the generated keys sk, pk there is some c such that $\mathsf{P}[\operatorname{Enc}(x_0) = c] = q \geq 1/\operatorname{poly}(n)$. Then, by the $1 - \operatorname{negl}(n)$ completeness we have $\mathsf{P}[\operatorname{Enc}(x_1) = c] = \operatorname{negl}(n)$. Now suppose in a single-message security game the attacker outputs 0 iff it is given c. It is easy to see that it can distinguish between $\operatorname{Enc}(x_0)$ and $\operatorname{Enc}(x_1)$ by advantage least q. □

5 Impossibility of Tamper-Resilient Zero-Knowledge for NP

In this section we show that no interactive proof system (with an efficient prover) for languages in $\mathsf{NP} \setminus \mathsf{BPP}$ can be zero-knowledge against a tampering verifier. For simplicity, in the following we assume that the prover's randomness and the common input are both of length n (otherwise we pad the shorter one). First we give an explicit definition for tamper-resilient zero-knowledge.

Definition 5.1 (Tamper-Resilient Zero-Knowledge)**.** *Suppose (P, V) is an interactive proof system for language L where x (of length n) is the common input and $y \in \{0,1\}^{\operatorname{poly}(n)}$ is prover's private input. A p-tampering verifier V^* at the beginning of the interaction generates a tampering circuit Tam that gets y as auxiliary input and transforms the uniform source of private randomness U_n of the prover into a p-tampering source $U_n^{\mathsf{Tam},p}$. The view of V^* includes its private randomness as well as the messages exchanged. We call such proof system α-tamper-resilient zero-knowledge against p-tampering, if for every p-tampering PPT verifier V^*, there exists a simulator* $\textsc{Sim}$ *such for every sequence of triples (x, y, z) of: the common input x, prover's private input y, and verifier's auxiliary input z the following two ensembles are $\alpha(|x|)$-indistinguishable:*

$$\{\mathsf{View}_{V^*}\langle V^*(z), P(y)\rangle(x)\}_{x \in L, z} \text{ and } \{\textsc{Sim}(x, z)\}_{x \in L, z}.$$

Theorem 5.2 (Impossibility of Zero-Knowledge for NP)**.** *Suppose there exists an efficient prover zero-knowledge proof system Π for $L \in \mathsf{NP}$ with negligible completeness and soundless errors. Then Π cannot be $o(p)$-tamper-resilient zero-knowledge against p-tampering verifiers for $p > 1/\operatorname{poly}(n)$ unless $L \in \mathsf{BPP}$.*

5.1 Proof of Theorem 5.2

In this subsection we prove Theorem 5.2.

Intuition. At a high level, the proof goes through the following two steps.

1. First we show that there exists an efficiently computable Boolean function f over the transcript τ of the interaction such that: for $1 - \operatorname{negl}(n)$ fraction of random seeds of the (honest) verifier r_V, $f(\tau(r_P))$ has little bias as a function of prover's randomness r_P. We show the

existence of such “unbiased” function f by proving that any zero-knowledge proof system needs to have min-entropy at least $\omega(\log n)$ in the messages coming from the prover, even conditioned on (almost all of) verifier random seeds. Having the lower bound on the min-entropy of the messages, we can take f easily to be a strong extractor (e.g., chosen from a family of pairwise independent functions). This step of our proof improves upon a result by Goldreich and Oren [GO94] who showed that the min-entropy of τ is positive.

2. Now suppose we do have such efficient unbiased function f as described above. Since every unbiased Boolean function has $\Omega(1)$ variance, we can then apply the tampering attack of Theorem 1.2.1 and get a malicious verifier V^* who tampers with the randomness of the prover through a tampering circuit Tam in a way that $\mathsf{P}_{r_P^{\mathsf{Tam}}}[f(\tau) = y_i] \geq 1/2 + \Omega(p)$ where i is the index of the bit of y that V^* wishes to learn. Since the view of such tampering verifier should be simulated by the efficient simulator $\textsc{Sim}(\cdot)$, by running $\textsc{Sim}(x)$ enough number of times we can learn y_i for every i and reconstruct y completely.

We start by formally describing our extension of the result of Goldreich and Oren [GO94], which corresponds to the first step above. Note that here we only rely on the zero-knowledge property of the proof system (regardless of the tampering).

Theorem 5.3 (Message-Entropy of Zero-Knowledge Proofs)**.** *Suppose (P, V) is an interactive proof system for a language L with negligible completeness and soundness errors. Then there is a PPT algorithm A such that the following properties hold.*

- *A takes as input x and 1^K and runs in time $\mathrm{poly}(|x|, K)$.*
- *If A accepts x, it implies $x \in L$ up to $\mathrm{negl}(n)$ error: $\mathsf{P}[A(x) = 1$ and $x \notin L] \leq \mathrm{negl}(n)$.*
- *Suppose in addition that (P, V) is zero-knowledge, then either of the following holds for $x \in L$:*
  1. *$A(1^K, x)$ accepts x with probability at least $1/\mathrm{poly}(K, n)$, or*
  2. *with probability $1 - 1/K$ over the choice of r_A, it holds that the min-entropy of the transcript $\tau = \langle V, P\rangle(x)$ conditioned on r_A is at least: $\mathrm{H}_\infty(\tau \mid r_A) \geq \log(K)$.*

Interpretation. Theorem 5.3 implies that if the entropy of the messages coming from the prover in a zero-knowledge prover is $O(\log n)$, then by taking $K = \mathrm{poly}(n)$ large enough $A(1^k, x)$ can decide $x \in L$ with $1/\mathrm{poly}(n)$ advantage (i.e., it accepts $x \in L$ with $1/\mathrm{poly}(n)$ probability and accepts any $x \notin L$ with $\mathrm{negl}(n)$ probability). This improves over the result of Goldreich and Oren [GO94] that obtained the same conclusion based on assumption that the prover is *deterministic*.

The following theorem corresponds to the second step of the proof of Theorem 5.2.

Theorem 5.4 (Signaling the Witness Bits)**.** *Suppose (P, V) is an efficient-prover $o(p)$-tamper-resilient zero-knowledge proof system against p-tampering adversaries for a language L and $p = 1/\mathrm{poly}(n)$. Also suppose for some $K = \omega(1/p^2)$ the following holds: with probability $1 - 1/K$ over the choice of r_A, the min-entropy of the transcript $\tau = \langle V, P\rangle(x)$ conditioned on r_A is at least: $\mathrm{H}_\infty(\tau \mid r_A) \geq \log K$. Then there is an efficient algorithm $B_p(x)$ that for every $x \in L$, with probability $1 - \mathrm{negl}(n)$ outputs the private input y of P (i.e., the witness of $x \in L$).*

Proof of Theorem 5.2. We first prove Theorem 5.2 using Theorems 5.3 and 5.4. We present an efficient algorithm C that decides membership of $x \in L$ for the language $L \in \mathsf{NP}$ in probabilistic polynomial time by relying on the existence of the efficient algorithms A and B of Theorems 5.3 and 5.4.

Algorithm $C_p(x)$.

1. Take $K = \omega(1/p^2)$ to be some $\mathrm{poly}(n)$ (which is possible since $p = 1/\mathrm{poly}(n)$).
2. Run algorithm $B_p(x)$ to get some NP-witness y. Output $x \in L$ if y is an acceptable witness.
3. Otherwise, run the algorithm $A(1^K, x)$ and output whatever A decides about $x \in L$.

Soundness of C. We first show that if $x \notin L$, C accepts x with $\mathrm{negl}(n)$ probability. The reason is that since we verify the extracted "witness" y. Therefore if $x \notin L$, no such witness can exist and pass the verification and so we would not accept x in Step 2. On the other hand, Theorem 5.3 asserts that the probability that if $x \notin L$, the algorithm A accept x with negligible probability.

Completeness of C. There are two possibilities.

1. First suppose that with probability $1 - 1/K$ over the choice of r_A, the min-entropy of the transcript $\tau = \langle V, P\rangle(x)$ conditioned on r_A is at least: $\mathrm{H}_\infty(\tau \mid r_A) \geq \log K$. In this case, the algorithm B will extract the witness with probability $1 - \mathrm{negl}(n)$ and so C accepts x with probability $1 - \mathrm{negl}(n)$.
2. Otherwise, by the properties of the algorithm A specified in Theorem 5.3, x will be accepted with probability at least $1/\mathrm{poly}(n, K) \geq 1/\mathrm{poly}(n)$.

□

In the following we will prove Theorems 5.3 and 5.4.

5.1.1 Proof of Theorem 5.3

Our proof, at a high level, follows the approach of Goldreich and Oren [GO94], who showed that zero-knowledge with *deterministic* provers is impossible unless $L \in \mathsf{BPP}$. Namely, one starts by assuming that the prover is deterministic for *all* $x \in L$ and derive $L \in \mathsf{BPP}$. In our Theorem 5.3, however, we conclude an instance-dependent statement; namely, for every $x \in L$, *either* there is "sufficient" entropy in the messages, or that we can decide the membership of x in L efficiently.

Below, first, we give a comparison between the approach of [GO94] and our approach for the weaker claim (than Theorem 5.3) that the entropy of the prover messages in case of $x \in L$ cannot *always* be $O(\log n)$. Then will then prove Theorem 5.3 formally.

The Approach of [GO94]. Suppose the prover is deterministic. This means that for every prefix of the transcript of the interaction between the prover and the verifier $p_1, v_1, \ldots, p_{i-1}, v_{i-1}$, the next message p_i of the prover is determined. This fact can be used, together with the existence of the simulator Sim, to efficiently generate an accepting transcript τ whenever $x \in L$, in a way that the same procedure does *not* generate an accepting transcript whenever $x \notin L$. The main ideas of [GO94] to obtain both properties simultaneously are as follows:

1. At a high level, in the process of generating τ we are executing the verifier V against some fixed (simulated) prover strategy P^*. This way, the soundness condition guarantees that this will not lead to an accept for $x \notin L$ unless with negligible probability.

2. The (simulated) prover P^* needs to behave close to the honest prover if $x \in L$ to generate an accepting τ. For this, we use the simulator Sim to get the prover messages p_i *one by one*. To get the i'th message p_i, we use the simulator over a verifier who knows the previously generated partial transcript $p_1, v_1, \ldots, p_{i-1}$ as auxiliary input, and sends v_{i-1} as the next answer according to the algorithm of the honest verifier. This way the value p'_i will be indeed the same as the actual p_i (that the honest prover would return) with $1 - \text{negl}(n)$ probability, or otherwise these two answers would be distinguishable which contradicts the zero-knowledge property (note that p_i is fixed can be known to the distinguishing circuit).

The Extensions. When the prover is randomized, we cannot conclude that the generated message p'_i is necessarily the same as p_i with high probability because the prover's messages (even conditioned on the previous messages $p_1, p_2, \ldots, p_{i-1}$) could be different in every new execution of the protocol against the same fixed honest verifier. Thus, instead we follow the following approach.

1. **How to Generate Messages:** Instead of executing the simulator once for getting every message p'_{i+1}, we repeatedly execute the simulator Sim up to some $\text{poly}(n)$ times till we get the *same* the partial transcript $(p_1, v_1 \ldots, p_{i-1}, v_i)$ as generated previously, and only then we look at the simulated value p'_i (which we hope to be the same as p_i with good probability).

2. **Analysis:** The challenging part is to show that if there is at least one transcript $\tau = (p_1, v_1, \ldots, p_m, v_m)$ that appears with $\alpha \geq 1/\text{poly}(n)$ probability, then we will generate τ through the process above with non-negligible probability. A naive analysis would use the fact that in every step, the provability of obtaining p_i is at least $\beta_i > \alpha > 1/\text{poly}(n)$. This simple analysis does not work because $(1/\text{poly}(n))^{\text{poly}(n)}$ could be negligible (but note that this analysis in fact works for constant number of rounds because $(1/\text{poly}(n))^{O(1)} \geq 1/\text{poly}(n)$). For arbitrary polynomial number of rounds we need a sharper analysis to show that if we obtain p_i with probability β_i, the product $\prod \beta_i$ is also non-negligible. We do so by decomposing α into $\alpha = \alpha_1 \cdot \alpha_2 \ldots$ where α_i is the probability of p_i being the honest message conditioned on the previously generated transcript. We show that for every i, $\beta_i/\alpha_i \approx 1 + \varepsilon$ for arbitrary small $\varepsilon = 1/\text{poly}(n)$ and thus $\beta = \prod_i \beta_i \approx \prod_i \alpha_i = \alpha$.

The Formal Proof. Let $\tau = Q(x, r_V)$ be the random variable denoting the transcript of the protocol when the common input is x, and the verifier has random coins fixed to r_V. Since the verifier acts deterministically once x and r_V are fixed, the distribution of Q is a deterministic function of the random coins of the prover. For an implicitly fixed r_V, we further let $Q_i(x, \tau_{i-1})$ denote the distribution of the message sent by the prover in the i^{th} round, given that $\tau_{i-1} = p_1, v_1, \ldots, p_{i-1}, v_{i-1}$ is the set of messages exchanged in the previous rounds.

In the following we assume that the proof system (P, V) has m rounds.

Observe that the message sent by the verifier in the i^{th} round is a (poly-time computable) deterministic function f of x, r_V, and the messages sent by the prover in the previous rounds. We can use this fact to write Q in terms of Q_i, as follows. Let $\tau = p_1, v_1, \ldots, p_m, v_m$ be a transcript of the protocol consistent with x and r_V. Then, assuming that τ_i is the partial transcript up to and including round i, it holds that:

$$\mathsf{P}[Q(x, r_V) = \tau] = \mathsf{P}[Q_1(x, \tau_0) = p_1] \cdot \cdots \cdot \mathsf{P}[Q_m(x, \tau_{m-1}) = p_m].$$

Our algorithm A will be such that for $K = \text{poly}(n)$ and $x \in L$, if for a $1/K$ fraction of r_V, the min-entropy of $Q(x, r_V)$ is $\leq \log K$, then $A(1^K, x)$ will output "accept" with probability $\geq {}^1\!/_{16K^2}$.

Suppose for a $1/K$ fraction of r_V, there is some corresponding "heavy" message transcript $\tau^* = \tau^*(x, r_V)$ that occurs with probability $\geq 1/K$; we call such r_V *special.*

Our approach will be to produce an efficient prover strategy P^* that, when interacting with V on x with such a special r_V, will produce the corresponding τ^* as its transcript with probability $\geq 1/8K$ (without knowing r_V and τ^* in advance).

First, assuming that we have such a prover strategy P^*, we claim that we can use P^* to create the required A. We observe that for fixed $x \in S$ (of large enough length $n = |x|$), not more than $1/2$ fraction of the special r_V can have their corresponding τ^* to be a rejecting transcript, because otherwise the honest verifier will reject on common input x with probability $1/2K > \text{negl}(n)$, violating the $1 - \text{negl}(n)$ completeness. Thus a uniformly chosen r_V will be both special and have an accepting τ^* with probability $> 1/2K$.

Now we examine what will happen when we simulate (P^*, V) on common input $x \in S$, with a uniformly chosen r_V. With probability $> 1/2K$, the random coins of V will have a corresponding τ^* that is an accepting transcript, and conditioned on getting such an r_V, the output of (P^*, V) will be τ^* with probability $\geq 1/8K$. Thus, with probability $\geq 1/16K^2$, (P^*, V) will be accepting. Further, notice that when the common input is $x \notin L$, then the output of (P^*, V) can be accepting with probability at most $\text{negl}(n)$, or otherwise the $1 - \text{negl}(n)$ soundness would be violated. Therefore, we can simply define A to be the machine simulating (P^*, V) on common input x.

It remains to show how to construct P^*. Define V^* to be the verifier strategy that takes a partial message transcript as its auxiliary input z, sends its first messages according to z, and then aborts. By the auxiliary input zero-knowledge, there exists an efficient simulator S for V^*.

The malicious prover P^* (defined based on S), over the common input x, works as follows.

```
1. Let $\hat{\tau} = \emptyset$.
2. For $i = 1$ to $k$ :
   (a) Run $S(x, \hat{\tau})$ repeatedly, until you find a transcript $\tau$ whose first $i - 1$ messages match
       with $\hat{\tau}$. If no such message was found in $K(\log K)$ runs, abort.
   (b) Send $p_i$ according to $\tau$, and receive $v_i$ from the verifier.
   (c) $\hat{p_i} = p_i$, $\hat{v_i} = v_i$
```

Assuming that the randomness of the verifier, r_V, is special, ideally we want that in each iteration i, P^* sends the i^{th} prover message such that it matches τ^* for x and r_V of the verifier it is interacting with.

To actually analyze the behavior of P^*, let assume that P^* and V have each sent all messages correctly corresponding to τ^* for all iterations up to the t^{th} iteration (i.e., $\hat{\tau} = \tau^*_{t-1}$). We will calculate the probability that P^* then sends p^*_t on the t^{th} iteration. Then, since V acts deterministically given x, r_V and the prover messages, V will also send v^*_t on this iteration.

Our analysis will initially assume that we have access to an oracle simulator $O(x, z)$, that produces transcripts identically distributed to a real interaction between (P, V^*). We will estimate the success probability for each iteration when we are using this oracle. Then, we will argue that replacing the oracle O with our computationally indistinguishable simulator S gives at most a negligible loss.

Using the Ideal Oracle. We now give the analysis when we use O. First we analyze the probability that our oracle simulator finds a transcript matching the messages sent so far. Since the verifier V^* will send messages according to its auxiliary input, the verifier messages up to step t will always match v^*_i (after this, V^* will abort, but it turns out we don't care about the $(t+1)^{\text{th}}$ and later messages in this transcript). So we only need to find the probability that the transcript has matching prover messages. Since O is a perfect simulator oracle, the first $t - 1$ prover messages of

$O(x, \tau^*_{t-1})$ will be distributed exactly as the random variables Q_i for $i \leq t-1$. Thus the probability that a transcript produced by $O(x, \tau^*_{t-1})$ matches the first $t-1$ messages of τ^* is given by

$$\prod_{i=1}^{t-1} \mathsf{P}[Q_i(x, \tau^*_{i-1}) = p^*_i] \geq \prod_{i=1}^{k} \mathsf{P}[Q_i(x, \tau^*_{i-1}) = p^*_i] \geq \frac{1}{K}.$$

Thus, repeating $K(\log K)$ times will give us a probability $\geq 1-(1-1/K)^{K(\log K)}$ of producing a matching transcript, which is $\geq 1 - 1/K$. Given that we have produced a matching transcript, the probability that the t^{th} message of this transcript is p^*_t is given by $\geq \mathsf{P}[Q_t(x, \tau^*_{t-i}) = p^*_t]$. Therefore the overall probability that P^* using O sends p^*_t in the t^{th} iteration, given that it sent the correct messages in the previous iterations, is $(1 - 1/K) \cdot \mathsf{P}[Q_t(x, \tau^*_{t-i}) = p^*_t]$.

Using the Simulator. Now we replace the ideal oracle O by using the simulator S. We argue that P^* using S must succeed with probability at least $(1 - 1/K) \cdot (1 - 1/K) \cdot \mathsf{P}[Q_t(x, \tau^*_{t-i}) = p^*_t]$. Otherwise, we can create a distinguisher $\mathcal{D}$ to distinguish between $\mathsf{View}_{V^*}\langle P, V^*(\tau^*_{t-1})\rangle(x)$ and $S(x, \tau^*_{t-1})$ as follows: Simulate the t^{th} iteration of P^* and outputs 1 whenever the message sent is m^*_i (where 1 corresponds to guessing that the distribution is $\mathsf{View}_{V^*}\langle P, V^*(\tau^*_{t-1})\rangle(x)$). Then $\mathcal{D}$ distinguishes between $\mathsf{View}_{V^*}\langle P, V^*(\tau^*_{t-1})\rangle(x)$ and $S(x, \tau^*_{t-1})$ with advantage at least

$$\frac{1}{K} \cdot (1 - \frac{1}{K}) \cdot \mathsf{P}[Q_t(x, \tau^*_{t-i}) = p^*_t] \geq \frac{1}{K} \cdot (1 - \frac{1}{K}) \cdot \frac{1}{K}$$

contradicting the fact that the protocol is zero-knowledge.

Finally note that the probability that every message sent by P^* using S matches τ^* is at least:

$$\prod_{t=1}^{k} (1 - \frac{1}{K})^2 \cdot \mathsf{P}[Q_t(x, \tau^*_{t-i}) = p^*_t] = (1 - \frac{1}{K})^{2K} \cdot \mathsf{P}[Q(x, r_V) = \tau^*] > \frac{1}{8K}.$$

The last inequality holds for large enough K because $(1 - 1/K)^{2K} \approx 1/e^2$ for large enough K.

5.1.2 Proof of Theorem 5.4

In the following we use $\{-1, +1\}$ (instead of $\{0, 1\}$) to represent the bits of the witness y.

The Description of the Verifier V^*. The p-tampering malicious verifier V^* receives auxiliary input $z = (i, f)$ where $i \in [|y|]$ and f is a $2 \cdot |\tau|$ bit string representing a function mapping $\{0,1\}^{|\tau|}$ to $\{-1, +1\}$ (supposedly chosen from a family of pairwise independent functions). The verifier V^* designs the following tampering circuit Tam. The goal of Tam is to signal the i^{th} bit of the prover's private input y (i.e. witness). The tampering circuit $\mathsf{Tam}_{(r_V, f, i)}$ simply *assumes* that for the sampled $r_A \xleftarrow{\$} \{0,1\}^{\text{poly}(n)}$ the function f has small bias $|\mathsf{E}_{r_P}[f(\tau)]| = o(p)$ (which implies that $\text{Var}(f(r_P)) > 1 - o(1)$), uses the biasing algorithm of Theorem 1.2.1, and with the knowledge of y as an auxiliary input, the p-tampering circuit Tam tries to bias $\mathsf{E}_{r_P}[f(\tau)]$ toward y_i by $\Omega(p)$

The assumptions $|\mathsf{E}_{r_P}[f(\tau)]| = o(p)$ made in the design of Tam could simply be false, but as we will see shortly, this condition holds almost always and that turns out to be sufficient for us.

The Analysis of the Verifier V^*. Here we show that the p-tampering verifier V^* is indeed able to guess the bit y_i with probability $1/2 + \Omega(p)$ by looking at $f(\tau)$ where τ is the transcript of the interaction of V^* with the the prover whose randomness is changed into the p-tampering source $U_n^{\mathsf{Tam},p}$. More formally, we prove the following.

Claim 5.5. *Suppose $\mathcal{F}$ is a family of pairwise independent Boolean functions mapping $\{0,1\}^{|\tau|}$ to $\{-1,+1\}$. It holds that $\mathsf{P}_{f\xleftarrow{\$}\mathcal{F},r_P\xleftarrow{\$}U_n^{\mathsf{Tam},p},r_V}[f(\tau)=y_i] \geq 1/2+\Omega(p)$ where the transcript τ is the result of the interaction between the p-tampering verifier $V^*(i,f)$ who uses the randomness r_V and the prover $P(y)$ who uses the p-tampering randomness $U_n^{\mathsf{Tam},p}$, over the common input x.*

Before proving Claim 5.5 we show how to prove Theorem 5.4 using Claim 5.5.

Proof of Theorem 5.4. The existence of the simulator $\textsc{Sim}$ for V^* shows that by running $\textsc{Sim}(x,f,i)$, for a fixed i and a *random* $f\xleftarrow{\$}\mathcal{F}$, we shall be able to obtain an output which is $o(p)$-indistinguishable from the distribution of the transcript $\tau=\langle V^*(i,f),P(y,U_n^{\mathsf{Tam},p})\rangle(x)$, even when the distinguisher is given the auxiliary input $z=(i,f)$. In particular they should be $o(p)$-indistinguishable against a distinguisher who simply computes the function f over τ and outputs this bit. The main point here is that the output bit b, when computed using the simulator's simulated transcript τ', should satisfy $\mathsf{P}[b=y_i]\geq 1/2+\Omega(p)-o(p)=1/2+\Omega(p)$, or otherwise the distinguisher can $\Omega(p)$-distinguish the value of $f(\tau)$ from $f(\tau')$.

If we execution $\textsc{Sim}(x,f,i)$ using $\Theta(1/p^2)$ random choices of f and take the majority, by Chernoff bound it holds that with probability $1-\operatorname{negl}(n)$ the output bit will be equal to y_i. By doing the same for every $i\in[|y|]$ we can learn all the bits of y with probability $1-\operatorname{poly}(n)\cdot\operatorname{negl}(n)=1-\operatorname{negl}(n)$. □

Finally, we prove Claim 5.5.

Proof of Claim 5.5. First note that by the assumption of Theorem 5.4, with probability $1-1/K$ over the choice of r_V, the min-entropy of the transcript τ (over the randomness of the prover) is at least $(\log K)$. By our choice of the parameter, we can safely ignore events of probability at most $1/K=o(p^2)\leq o(p)$, because they cannot affect our claim that $\mathsf{P}_{f\xleftarrow{\$}\mathcal{F},r_P\xleftarrow{\$}U_n^{\mathsf{Tam},p},r_V}[f(\tau)=y_i]\geq 1/2+\Omega(p)$. Thus in the following we will assume that r_V is such that $\mathrm{H}_\infty(\tau\mid r_V)\geq\log K$.

By using Lemma 4.7 over $k=\log K$ and an averaging argument, it holds that with probability at least $1-O(K^{-1/2})$ the selected f is a good extractor in the sense that the statistical distance between $(f,f(\tau))$ and (f,U_1) is at most $1-O(K^{-1/2})$. Since $O(K^{-1/2})=o(p)$ again we can safely ignore the event that the selected f is not a good extractor.

Thus, it only remains to study the case that r_V and f both have the desired properties specified above. In this case, we would have $\mathsf{E}_{r_P}[f(\tau)]=o(p)$ and therefore, we can conclude that $\operatorname{Var}_{r_P}(f(\tau))>0.9$ and thus we can apply Theorem 1.6 (or even its simpler version of Theorem 1.2.1) to bias f toward y_i by at least $\Omega(p)$. Since we already had $\mathsf{E}_{r_P\xleftarrow{\$}U_n}[f(\tau)]=o(p)$, after the biasing attack, we would get $\mathsf{E}_{r_P\xleftarrow{\$}U_n^{\mathsf{Tam},p}}[f(\tau)]=y_i\cdot\Omega(p)+o(p)=y_i\cdot\Omega(p)$. □

6 Impossibility of Tamper-Resilient Commitments

In this section we prove Theorem 1.3. We start by formalizing the notion of tamper-resilient commitments. Since we present a negative result in which only the receiver is a tampering adversary[5], we restrict our definition for tamper-resilient hiding.

Definition 6.1 (Tamper-Resilient Hiding Bit-Commitments)**.** *A bit commitment scheme (S,R) is tamper-resilient hiding if the hiding property holds in the following stronger form.*

[5] Note that this is a stronger than giving two attacks by sender and receiver such that one of them works.

- *Suppose a malicious receive $\widehat{R}$ is able to run a p-tampering attack in the commitment phase by generating a circuit* Tam *in the beginning of the commitment phase and having* Tam *run a p-tampering attack over the randomness r_S of the sender S. We call (S, R) p-tamper-resilient $(1 - \varepsilon)$-hiding if any such $\widehat{R}$, by the end of the commitment phase, can guess the randomly chosen committed bit $b \overset{\$}{\gets} \{0, 1\}$ with probability at most $1/2 + \varepsilon$.*

The formal statement of our result follows.

Theorem 6.2 (No Tamper-Resilient Commitments)**.** *If (S, R) is a commitment scheme which is δ-binding, for any $\delta > 1/\operatorname{poly}(n)$, then a p-tampering receiver can break the hiding with advantage $\Omega(p \cdot \delta)$.*

Theorem 6.2 implies Theorem 1.3 by using $\delta = 1/2$.

Proof of Theorem 6.2. First we show that for any δ-binding commitment scheme, with probability at least δ over the choice of the randomness of the parties, the commitments to 0 and 1 will have different transcripts. More formally, let $\tau(b, r_S, r_R)$ be the transcript of the commitment phase of (S, R) when the committed bit is b, and r_S, r_R are in order the randomness of the sender and the receiver.

Claim 6.3. *If (S, R) is δ-binding, then it holds that $\mathsf{P}_{r_S, r_R}[\tau(0, r_S, r_R) \neq \tau(1, r_S, r_R)] \geq \delta$.*

Proof. For any pair of random seeds r_S, r_R such that $\tau(0, r_S, r_R) = \tau(1, r_S, r_R)$, the sender can use the very same r_S to decommit to both of 0 and 1 successfully. Therefore, the δ-binding of the scheme implies that this event happens with probability at most δ. □

Our malicious receiver $\widehat{R}$ will execute the commitment phase honestly, but at the same time performs a p-tampering attack over the randomness of the sender.

Tampering Algorithm of Cheating Receiver $\widehat{R}$—Informally Described. Roughly speaking the strategy of $\widehat{R}$ to break the hiding of (S, R) with advantage $\Omega(p\delta)$ is as follows.

1. Choose r_R such that $\mathsf{P}_{r_S}[\tau(0, r_S, r_R) \neq \tau(1, r_S, r_R)] \geq \delta/2$.

2. Choose $f \overset{\$}{\gets} \mathcal{F}$ at random from a family of pairwise independent functions mapping the transcript $\tau(b, r_S, r_R)$ to $\{0, 1\}$. If f already reveals the bit b (this is a feature that can be "tested"), keep f and use it in an interaction with R.

3. If the sampled f does not reveal b, use the tampering algorithm of Theorem 1.6 to make f a good detecting function w.r.t. the tampered randomness.

In the following we describe each step formally.

The first step can be done easily by sampling $\operatorname{poly}(n/\delta)$ many instances of r_R and choosing one such that $\mathsf{P}_{r_S}[\tau(0, r_S, r_R) \neq \tau(1, r_S, r_R)] \geq \delta/2$. By Chernoff bound such r_R can be found with probability $1 - \operatorname{negl}(n)$. In the following let assume that the length of τ is m.

In the second step, $\widehat{R}$ chooses a Boolean function $f \overset{\$}{\gets} \mathcal{F}$ at random from a family of pairwise independent functions mapping the transcript of the commitment phase τ to $\{0, 1\}$. Similar to the tampering attacks against encryption, the goal here is to design a tampering circuit Tam such that:

$$\mathsf{P}_{f \overset{\$}{\gets} \mathcal{F}, r \overset{\$}{\gets} U_m^{\mathsf{Tam}, p}}[f(\tau(1, r, r_R)) = 1] - \mathsf{P}_{f \overset{\$}{\gets} \mathcal{F}, r \overset{\$}{\gets} U_m^{\mathsf{Tam}, p}}[f(\tau(0, r, r_R)) = 1] > \Omega(p\delta).$$

The above condition is equivalent to $\mathsf{E}_{f \xleftarrow{\$} \mathcal{F}, r \xleftarrow{\$} U_m^{\mathsf{Tam},p}}[g(r)] \geq \Omega(p\delta)$ where $g(r) = \widetilde{f}(1,r) - \widetilde{f}(0,r)$ and $\widetilde{f}(b,r) = f(\tau(b,r,r_R))$. By using $\tau(\cdot)$ instead of $\mathrm{Enc}(\cdot)$ and $\delta/2$ instead of ε, the very same proof of Claim 4.10 implies the following due to the pairwise independence of $f \xleftarrow{\$} \mathcal{F}$.

Claim 6.4. *If $val(\widetilde{f}) = \mathsf{P}_{r_E}[\widetilde{f}(0,r_E) \neq \widetilde{f}(1,r_E)]$, then $\mathsf{E}_{f \xleftarrow{\$} \mathcal{F}}[val(\widetilde{f})] \geq \delta/4$.*

Claim 6.4 and an averaging argument show that with probability at least $\delta/8$ over $f \xleftarrow{\$} \mathcal{F}$, it holds that $val(\widetilde{f}) \geq \delta/8$. The malicious receiver $\widehat{R}$ in its second step tries to find such r_R as follows: sample n/δ many samples $f \xleftarrow{\$} \mathcal{F}$, and for each sampled f sample n/δ^2 many r's and output the sampled f if for at least $\delta/9$ fraction of the sampled r's it holds that $f(\tau(0,r,r_R) \neq f(\tau(0,r,r_R)$. Using Chernoff bound, it can be shown that with probability $1 - \mathrm{negl}(n)$ this procedure finds a function $f \in \mathcal{F}$ such that $val(\widetilde{f}) \geq \delta/10$.

In the second step, the malicious receiver $\widehat{R}$ tests the sampled f to see if it is already a good detecting function *without any tampering* or not.[6] Namely, let c be the universal constant in Theorem 1.6. If it holds that

$$\left| \mathsf{P}_{r \xleftarrow{\$} U_m}[f(\tau(1,r,r_R)) = 1] - \mathsf{P}_{r \xleftarrow{\$} U_m}[f(\tau(0,r,r_R)) = 1] \right| > p \cdot \delta \cdot c/1000 \tag{4}$$

then either of the functions f or $1-f$ can be used as a detecting function w.r.t. the uniform distribution U_m (without tampering) to break the binding with advantage $\Omega(p\delta)$. By repeated sampling, the receiver $\widehat{R}$ can approximate the quantity $\mathsf{P}_{r \xleftarrow{\$} U_m}[f(\tau(1,r,r_R)) = 1] - \mathsf{P}_{r \xleftarrow{\$} U_m}[f(\tau(0,r,r_R)) = 1]$ up to additive error $p\delta c/100$ and conclude that with probability $1 - \mathrm{negl}(n)$ either the Inequality (4) holds, or that:

$$\left| \mathsf{P}_{r \xleftarrow{\$} U_m^{\mathsf{Tam},p}}[f(\tau(1,r,r_R)) = 1] - \mathsf{P}_{r \xleftarrow{\$} U_m^{\mathsf{Tam},p}}[f(\tau(0,r,r_R)) = 1] \right| < p \cdot \delta \cdot c/100. \tag{5}$$

In this case we show how to tamper with the randomness of the sender so that f is detecting w.r.t. $U_m^{\mathsf{Tam},p}$. This corresponds to the third step of the malicious receiver $\widehat{R}$.

Claim 6.5. *Suppose both of the following holds:*

1. $val(\widetilde{f}) \geq \delta/10$.
2. $\left| \mathsf{P}_{f \xleftarrow{\$} \mathcal{F}, r \xleftarrow{\$} U_m}[f(\tau(1,r,r_R)) = 1] - \mathsf{P}_{f \xleftarrow{\$} \mathcal{F}, r \xleftarrow{\$} U_m}[f(\tau(0,r,r_R)) = 1] \right| < 1/2$.

Then it holds that $\mathrm{Var}[g(\cdot)] \geq \delta/40$.

Proof. Recall that $\mathrm{Var}[X] = \mathsf{E}[(X - \mathsf{E}[X])^2]$. By the first property, with probability at least $\delta/10$ over the choice of r it holds that $g(r) \in \{-1, 1\}$. For such r, by the second property, the value of $|g(r) - \mathsf{E}[g]|$ is at least $1/2$. Therefore, it holds that $\mathrm{Var}_r[g(\cdot)] \geq \delta/10 \cdot (1/2)^2 = \delta/40$. □

Note that Inequality (5) is equivalent to $|\mathsf{E}_{r \xleftarrow{\$} U_m^{\mathsf{Tam},p}}[g(r)]| < p\delta c/100$. Also by Claim 6.5 we know that $\mathrm{Var}_r[g(\cdot)] \geq \delta/40$. Therefore, if the receiver $\widehat{R}$ uses the p-tampering biasing attack of Theorem 1.6 to bias the function g toward $+1$ it can make the average of the function g to be at least $pc\delta/40 - pc\delta/100 = \Omega_c(p\delta)$ which in turn implies that now the Boolean function f is indeed a $\Omega(pc\delta)$ detecting function to guess the bit b with respect to the tampered randomness $U_m^{\mathsf{Tam},p}$. This finishes the proof of Theorem 6.2. □

[6] Note that we could *not* do the same "testing" procedure to find a "good" f in the case of tampering attacks against private-key encryptions, since to test the quality of f as a detecting function one needs the private key to run the encryption. That is the reason why we relied on the large entropy of the encrypted messages, because in this case we would know the actual bias of the function f and would also know its average after the biasing attack.

7 Impossibility of Tamper-Resilient Secure Computation

Suppose f is a two-input finite function, Alice holds an input x, Bob holds an input y and they want to jointly compute $f(x, y)$ in a way that *only Bob* receives $f(x, y)$. In this section we show that any such finite function is either *trivial* and has a deterministic secure protocol, or that any efficient protocol to compute $f(x, y)$ is vulnerable to tampering attacks. We restrict ourselves to the case that only one party receives the output of f, and extending our result to the setting that both parties get outputs remains as an interesting open question.

We use the following semi-honest (weak) definition for the security of computing functions. This makes our negative result only stronger.

Definition 7.1 (Semi-Honest SFE)**.** *Suppose Π is a two-party protocol in which Alice and Bob receive 1^n as common input, Alice gets input $x \in \mathcal{X}$ and Bob gets input $y \in \mathcal{Y}$. We call Π a secure function to compute a two-input function f with input sets $\mathcal{X}$ and $\mathcal{Y}$ if the following hold:*

- *At the end of the interaction Bob receives $f(x, y)$ with probability $1 - \operatorname{negl}(n)$.*
- *For any pair of inputs for Bob $y_1 \neq y_2 \in \mathcal{Y}$ and input x for Alice, if Bob uses y_b for a random $b \in \{0, 1\}$ to interact with Alice who uses x, after an honest execution, Alice cannot guess b with probability more than $1/2 + \operatorname{negl}(n)$.*
- *For any pair of inputs $x_1 \neq x_2$ for Alice and y for Bob, if $f(x_1, y) = f(x_2, y)$, at the end of the interaction, an honest execution of Bob using y cannot guess with probability more than $1/2 + \operatorname{negl}(n)$ the randomly chosen input of Alice x_b.*

Definition 7.2 (Tamper-Resilient (Semi-Honest) SFE)**.** *We call Π a secure protocol for f against p-tampering adversaries, if the conditions stated in Definition 7.1 hold even if the party (e.g., Alice) who tries to guess the randomly chosen input of the other party (e.g., Bob's input) is allowed to perform a p-tampering attack along the execution of the protocol. Namely, if Alice generates a tampering circuit Tam which transforms the uniform randomness of Bob into $U^{\mathsf{Tam}}_{\operatorname{poly}(n)}$ and then interacts with Bob, she still should not be able to guess Bob's input which is chosen as $y \xleftarrow{\$} \{y_1, y_2\}$ with probability more than $1/2 + \operatorname{negl}(n)$. The same holds if Bob tries to guess Alice's input $x \xleftarrow{\$} \{x_1, x_2\}$ when he uses an input y such that $f(x_1, y) = f(x_2, y)$ even if he gets to perform a p-tampering attack over Alice's randomness.*

Following [BMM99] we say that f has an *insecure minor* if there are $x_1 \neq x_2 \in \mathcal{X}, y_1 \neq y_2 \in \mathcal{Y}$ such that $f(x_1, y_1) \neq f(x_2, y_1)$ but $f(x_1, y_2) = f(x_2, y_2)$. It was shown in [BMM99] that if f does not have an insecure minor, then there is a deterministic single message protocol to compute f which is secure even against malicious parties. Here we show that if f has an insecure minor, then it cannot have a tamper-resilient secure protocol.

Theorem 7.3 (Impossibility of Tamper-Resilient SFE)**.** *Suppose f is a two-input function with an insecure minor. Then for every $p > 1/\operatorname{poly}(n)$, there is no protocol to compute f securely against p-tampering adversaries (according to Definition .*

To prove Theorem 7.3, roughly speaking, we show that computing f can be interpreted as a form of commitment scheme which was already ruled out (to be tamper resilient) by Theorem 6.2.

Definition 7.4 (Semi-Honest Commitments)**.** *We call (S, R) a semi-honest bit-commitment protocol, if the parties receive inputs and interact as in Definition A.7, and the following hold.*

- **Completeness.** *Similar to Definition A.7.*
- **Semi-Honest Hiding.** *We call (S, R) $(1-\varepsilon)$ semi-honest hiding if an honest executing of R cannot guess a randomly chosen committed bit b with probability more than ε by the end of the commitment phase. We call (S, R) simply semi-honest hiding if it is $(1-\operatorname{negl}(n))$ semi-honest hiding.*
- **Semi-Honest Binding.** *We call (S, R) $(1-\varepsilon)$ semi-honest binding, if the probability that (in an honest execution), the randomness r_S is accepted as decommitment to* both *of $\{0,1\}$ is at most ε. Namely, with probability at least $1-\varepsilon$ over the choice of (r_S, r_R) it holds that $\tau(0, r_S, r_R) \neq \tau(1, r_S, r_R)$ where $\tau(b, r_S, r_R)$ is the transcript of the commitment phase. We call (S, R) simply semi-honest binding if it is $(1-\operatorname{negl}(n))$ semi-honest binding.*

The following construction shows how to get a commitment scheme from any protocol that computes a function f with an insecure minor. We will show that if the protocol to compute f if semi-honest secure, then so is the commitment scheme.

Construction 7.5. *Suppose $\Pi = (A, B)$ is a two party protocol for computing a function f with an insecure minor: $f(x_1, y_1) \neq f(x_2, y_1)$ but $f(x_1, y_2) = f(x_2, y_2)$. We get a bit commitment scheme $\Sigma_\Pi = (S, R)$ as follows.*

- *For an input $b \in \{0,1\}$ given to the sender, the commitment phase consists of an execution of Π to compute $f(x_b, y_2)$ where S emulates $A(x_b)$ and R emulates $B(y_2)$.*
- *In the decommitment phase, S sends b and the randomness she used to emulate $A(x_b)$ to R.*

Theorem 7.6. *If Π is a semi-honest secure computation for f (with insecure minor), then the commitment scheme Σ_Π of Construction 7.5 is semi-honest secure.*

Proof. The completeness of Σ_Π is clear.

The hiding of Σ_Π also follows immediately from the security of Π for Alice (and the fact that the receiver is honestly executing the protocol using the input y_2).

In the following we prove the semi-honest binding of Σ_Π. Suppose for sake of contradiction that with probability $\varepsilon > 1/\operatorname{poly}(n)$ over the choice of r_S, r_R, it holds that using both inputs $b \in \{0,1\}$ leads to the *same* transcript: $\tau(0, r_S, r_R) = \tau(1, r_S, r_R)$. This event which we denote by E can be efficiently verified to hold by the sender (who is emulating the execution of Alice).

Now suppose, in a different game, the receiver emulates the execution of $B(y_2)$ instead of $B(y_1)$. In this case, by the security of the protocol Π for Bob, the event E should happen with probability at least $\varepsilon' = \varepsilon - \operatorname{negl}(n) > 1/\operatorname{poly}(n)$, or otherwise Alice can distinguish Bob's inputs by using either of the inputs $\{x_1, x_2\}$, executing the protocol with Bob, and checking wether the event E holds or not. This contradicts the completeness of the protocol Π, because with non-negligible probability ε', both inputs x_1, x_2 lead to the same exact transcripts and therefore the same outputs for Bob, but with $1-\operatorname{negl}(n)$ probability the outputs should be different. □

Finally we prove Theorem 7.3.

Proof of Theorem 7.3. We observe that the very same proof of Theorem 6.2 also rules out the possibility of tamper-resilient *semi-honest* commitment schemes. Namely, the p-tampering attack of Theorem 7.3 only relied on the semi-honest security of the underlying commitment scheme. By Theorem 7.6 the very same same attack can be used against the two party protocol Π when restricted to the inputs x_1, x_2, y_1, y_2 establishing Theorem 7.3. □

References

[AGV09] Adi Akavia, Shafi Goldwasser, and Vinod Vaikuntanathan. Simultaneous hardcore bits and cryptography against memory attacks. In Omer Reingold, editor, *Theory of Cryptography, 6th Theory of Cryptography Conference, TCC 2009, San Francisco, CA, USA, March 15-17, 2009. Proceedings*, volume 5444 of *Lecture Notes in Computer Science*, pages 474–495. Springer, 2009. 1

[AK96] Ross Anderson and Markus Kuhn. Tamper resistance – a cautionary note. In *Proceedings of the Second USENIX Workshop on Electronic Commerce*, pages 1–11, November 1996. 1

[Ale96] Aleph One. Smashing the stack for fun and profit. *Phrack Magazine*, 7(49):File 14, 1996. 1

[BDL97] Boneh, DeMillo, and Lipton. On the importance of checking cryptographic protocols for faults. In *EUROCRYPT: Advances in Cryptology: Proceedings of EUROCRYPT*, 1997. 1, 44

[BKKV10] Zvika Brakerski, Yael Tauman Kalai, Jonathan Katz, and Vinod Vaikuntanathan. Overcoming the hole in the bucket: Public-key cryptography resilient to continual memory leakage. In *FOCS*, pages 501–510. IEEE Computer Society, 2010. 1

[BMM99] Amos Beimel, Tal Malkin, and Silvio Micali. The all-or-nothing nature of two-party secure computation. In *CRYPTO*, pages 80–97, 1999. 28

[BS97] Biham and Shamir. Differential fault analysis of secret key cryptosystems. In *CRYPTO: Proceedings of Crypto*, 1997. 1

[CGGM00] Ran Canetti, Oded Goldreich, Shafi Goldwasser, and Silvio Micali. Resettable zero-knowledge (extended abstract). In *STOC*, pages 235–244, 2000. 2

[CKM11] Seung Geol Choi, Aggelos Kiayias, and Tal Malkin. BiTR: Built-in tamper resilience. In Dong Hoon Lee and Xiaoyun Wang, editors, *Advances in Cryptology - ASIACRYPT 2011 - 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings*, volume 7073 of *Lecture Notes in Computer Science*, pages 740–758. Springer, 2011. 1, 44

[DGK$^+$10] Yevgeniy Dodis, Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In Daniele Micciancio, editor, *Theory of Cryptography, 7th Theory of Cryptography Conference, TCC 2010, Zurich, Switzerland, February 9-11, 2010. Proceedings*, volume 5978 of *Lecture Notes in Computer Science*, pages 361–381. Springer, 2010. 1

[DHLAW10] Yevgeniy Dodis, Kristiyan Haralambiev, Adriana López-Alt, and Daniel Wichs. Cryptography against continuous memory attacks. In *FOCS*, pages 511–520. IEEE Computer Society, 2010. 1

[DO03] Dodis and Oliveira. On extracting private randomness over a public channel. In *RANDOM: International Workshop on Randomization and Approximation Techniques in Computer Science*. LNCS, 2003. 35, 42

[DOPS04] Dodis, Ong, Prabhakaran, and Sahai. On the (im)possibility of cryptography with imperfect randomness. In *FOCS: IEEE Symposium on Foundations of Computer Science (FOCS)*, 2004. 3, 5

[DP08] Stefan Dziembowski and Krzysztof Pietrzak. Leakage-resilient cryptography. In *FOCS*, pages 293–302. IEEE Computer Society, 2008. 1

[DPW10] Stefan Dziembowski, Krzysztof Pietrzak, and Daniel Wichs. Non-malleable codes. In Andrew Chi-Chih Yao, editor, *ICS*, pages 434–452. Tsinghua University Press, 2010. 1, 44

[DSK12] Dana Dachman-Soled and Yael Tauman Kalai. Securing circuits against constant-rate tampering. *IACR Cryptology ePrint Archive*, 2012:366, 2012. informal publication. 1

[FPV11] Sebastian Faust, Krzysztof Pietrzak, and Daniele Venturi. Tamper-proof circuits: How to trade leakage for tamper-resilience. In *ICALP (1)*, pages 391–402, 2011. 1, 44

[Fry00] Niklas Frykholm. Countermeasures against buffer overflow attacks. Technical report, RSA Data Security, Inc., pub-RSA:adr, November 2000. 1

[GLM$^+$04] Rosario Gennaro, Anna Lysyanskaya, Tal Malkin, Silvio Micali, and Tal Rabin. Algorithmic tamper-proof (atp) security: Theoretical foundations for security against hardware tampering. In Moni Naor, editor, *TCC*, volume 2951 of *Lecture Notes in Computer Science*, pages 258–277. Springer, 2004. 1, 3, 43, 44

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984. 40

[GO94] Oded Goldreich and Yair Oren. Definitions and properties of zero-knowledge proof systems. *Journal of Cryptology*, 7(1):1–32, 1994. 6, 20, 21

[GR12] Shafi Goldwasser and Guy Rothblum. How to compute in the presence of leakage. 2012. 1

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999. 34, 38, 40

[IL89] Russell Impagliazzo and Michael Luby. One-way functions are essential for complexity based cryptography. In *Proceedings of the 30th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 230–235, 1989. 3, 36, 37

[IPSW06] Yuval Ishai, Manoj Prabhakaran, Amit Sahai, and David Wagner. Private circuits II: Keeping secrets in tamperable circuits. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 308–327. Springer, 2006. 1, 44

[KKS11] Yael Tauman Kalai, Bhavana Kanukurthi, and Amit Sahai. Cryptography with tamperable and leaky memory. In *CRYPTO*, pages 373–390, 2011. 1, 44

[KLR09] Yael Tauman Kalai, Xin Li, and Anup Rao. 2-source extractors under computational assumptions and cryptography with defective randomness. In *FOCS*, pages 617–626. IEEE Computer Society, 2009. 35, 41, 42

[KLR12] Yael Kalai, Allison Lewko, and Anup Rao. Formulas resilient to short-circuit errors. 2012. 1

[LL10] Feng-Hao Liu and Anna Lysyanskaya. Algorithmic tamper-proof security under probing attacks. In *SCN*, pages 106–120, 2010. 1

[LL12] Feng-Hao Liu and Anna Lysyanskaya. Tamper and leakage resilience in the split-state model. In *Crypto*, 2012. 1, 44

[MR04] Micali and Reyzin. Physically observable cryptography (extended abstract). In *Theory of Cryptography Conference (TCC), LNCS*, volume 1, 2004. 1

[PB04] Jonathan D. Pincus and Brandon Baker. Beyond stack smashing: Recent advances in exploiting buffer overruns. *IEEE Security & Privacy*, 2(4):20–27, 2004. 1

[Rot12] Guy N. Rothblum. How to compute under $\dashv\rfloor^{\mathsf{0}}$ leakage without secure hardware. In Reihaneh Safavi-Naini and Ran Canetti, editors, *CRYPTO*, volume 7417 of *Lecture Notes in Computer Science*, pages 552–569. Springer, 2012. 1

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, Feb 1978. 1

[SV86] Miklos Santha and Umesh V. Vazirani. Generating quasi-random sequences from semi-random sources. *J. Comput. Syst. Sci.*, 33(1):75–87, 1986. 2, 8, 9

A Standard Definitions

Definition A.1 (Encryption Schemes)**.** *A* public-key *encryption scheme with security parameter n for message space $\mathcal{M}$ is composed of three PPT algorithms* $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *such that:*

- Gen *takes as input $r \in \{0,1\}^{\mathrm{poly}(n)}$ and outputs a secret key* sk *and a public key* pk*.*
- Enc *is a PPT that takes $x \in \mathcal{M}$ and* pk *and generates a cipher text c.*
- Dec *is a PPT that takes c and* sk *and outputs some $x' \in \mathcal{M}$.*

We say that $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *has completeness ρ if it holds that $\mathsf{P}[x = x'] \geq \rho$ where x, x' are generated as follows:*

$$(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathrm{Gen}(1^n), x \xleftarrow{\$} \mathcal{M},\ r_E, r_D \xleftarrow{\$} \{0,1\}^{\mathrm{poly}(n)},\ c = \mathrm{Enc}(\mathsf{pk}, r_E, x), x' = \mathrm{Dec}(\mathsf{sk}, r_D, c).$$

A private-key *encryption scheme is defined similarly with the difference that* $\mathsf{pk} = \mathsf{sk}$ *which we typically denote by* key*.*

Definition A.2 (CPA Security)**.** *A public-key encryption scheme* $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ *with security parameter n is called* CPA-secure *if for any* $\mathrm{poly}(n)$*-sized adversary* $\textsc{Adv}$ *the probability of winning in the following game is* $1/2 + \mathrm{negl}(n)$*.*

1. *They keys* $(\mathsf{pk}, \mathsf{sk})$ *are generated and the adversary receives* pk*.*

2. $\textsc{Adv}$ *chooses two messages* $x_0 \neq x_1$ *of the same length* $|x_0| = |x_1| = \mathrm{poly}(n)$*.*

3. $b \xleftarrow{\$} \{0,1\}$ *is chosen and* $c = \mathrm{Enc}_{\mathsf{pk}}(m_b)$ *is returned to* $\textsc{Adv}$*.*

4. $\textsc{Adv}$ *outputs* b' *and wins if* $b' = b$*.*

Definition A.3 (Multi-Message Security)**.** *For security parameter* n *and* $1 \leq k(n) \leq \mathrm{poly}(n)$*, we call a private-key encryption scheme for message space* k-message secure*, if for every two sequences of vectors of messages* $\overline{x}^i = (x^i_1, \ldots, x^i_{k(n)}), \overline{x}'^i = (x'^i_1, \ldots, x'^i_{k(n)})$*, where* $|x^i_j| = |x'^i_j| = \mathrm{poly}(i)$ *for all* i, j*, the following two ensembles of random variables are computationally indistinguishable for* $\mathrm{poly}(n)$*-sized circuits*

$$\{\overline{y}^i = (\mathrm{Enc}(x^i_1), \ldots, \mathrm{Enc}(x^i_{k(n)}))\},$$
$$\{\overline{y}'^i = (\mathrm{Enc}(x'^i_1), \ldots, \mathrm{Enc}(x'^i_{k(n)}))\}$$

where the encryptions are computed under the same key and independent randomness. The k*-message security for public-key encryption schemes is defined similarly where the distinguisher is also given the public key* pk*.*

It is easy to see that any CPA-secure public-key encryption scheme is also k-message secure for any $k \in [1, \mathrm{poly}(n)]$.

Definition A.4 (Interactive Proofs)**.** *An interactive proof* (P, V) *for membership in a language* L *is a pair of interactive Turing machines: a prover* P *and a verifier* V*, that each receive common input* x *and exchange* $m = \mathrm{poly}(|x|)$ *messages,* $(p_1, v_1, \ldots, p_m, v_m)$*. At the end,* V *either accepts or rejects and we have:*

- ***Efficiency****: The verifier* V *is a PPT interactive algorithm. Also, in case the prover* P *is also efficient, it receives a secret input* y *(which would be a witness for* $x \in L$ *when* $L \in \mathsf{NP}$*).*

- ***Completeness****: If* $x \in L$*,* V *accepts with probability* $1 - \mathrm{negl}(|x|)$*.*

- ***Soundness****: If* $x \notin L$*, for every prover strategy* P^**,* V *rejects with probability* $1 - \mathrm{negl}(|x|)$*.*

Definition A.5 (Auxiliary-Input Zero-Knowledge)**.** *An interactive proof* (P, V) *for language* L *is (auxiliary-input)* α-zero-knowledge *if for every PPT (malicious) verifier* V^**, there exists a* simulator $\textsc{Sim}$ *running in polynomial time over its first input such that for every sequence of triples* (x, y, z) *of: common input* $x \in L$*, prover's private input* $y \in \{0,1\}^{\mathrm{poly}(|x|)}$*, and verifier's auxiliary input* $z \in \{0,1\}^{\mathrm{poly}(|x|)}$*, the two ensembles*

$$\{\mathsf{View}_{V^*}\langle V^*(z), P(y)\rangle(x)\}_{x \in L, z}\{\textsc{Sim}(x, z)\}_{x \in L, z}$$

are $\alpha(|x|)$*-indistinguishable. We simply call* (P, V) *zero-knowledge if it is* $\mathrm{negl}(n)$*-zero-knowledge.*

Remark A.6. *Note that in Definition A.5 the distinguisher is also given the auxiliary input* z*. Since the auxiliary-input variant of zero-knowledge is the definition employed in this work, from now on we just call it zero-knowledge.*

Definition A.7 (Bit Commitments). *A* bit commitment scheme (S, R) *consists of two interactive PPTs (sender and receiver) engaging in a two-phase interaction called the* commitment *and* decommitment *phases. Both parties receive* 1^n *as input where* n *is the security parameter. The sender* S *also gets a privet input* $b \in \{0,1\}$*, and the decommitment phase consists of senders randomness* r_S *and* b *sent from* S *to* R*. We also require the following properties:*

- **Completeness.** *For both of* $b \in \{0,1\}$*, the receiver* R *accepts the interaction with probability* $1 - \mathrm{negl}(n)$*.*

- **Hiding.** *We call* (S, R) $(1-\varepsilon)$*-hiding for any malicious receive* $\widehat{R}$*, when* $b \xleftarrow{\$} \{0,1\}$ *is chosen at random, the probability that* $\widehat{R}$ *can correctly guess the bit* b *by the end of the commitment phase is at most* $1/2 + \varepsilon$*. We call* (S, R) *simply hiding if it is* $(1 - \mathrm{negl}(n))$*-hiding.*

- **Binding.** *We call* (S, R) $(1-\varepsilon)$*-binding, if for any malicious sender* $\widehat{S}$*, who outputs two decommitments* r_S^0 *and* r_S^1 *at the end of the commitment phase, the probability that both of* $(0, r_S^0)$ *and* $(1, r_S^1)$ *are accepted by* R *is* ε*. We call* (S, R) *simply binding if it is* $(1 - \mathrm{negl}(n))$*-binding.*

B Achieving Tamper Resilience Using Pseudorandomness

We present our positive results on tamper-resilient cryptography in this section. We show that assuming the existence of pseudorandom generators (PRGs), which is implied by the existence of one-way functions [HILL99], a wide range of cryptographic primitives, including signatures, identification schemes, witness-hiding protocols, as well as encryption schemes with a "weak" notion of semantic security (see Definition B.9) can be made resilient to p-tampering for $p = n^{-\alpha}$, where n is the security parameter and $\alpha > 0$ is an arbitrary constant. Our construction can be extended in a straightforward way to achieve resilience to p-tampering with $p = \log^{-c}(n)$ for some constant c, assuming the existence of PRGs with sub-exponential security.

Using Pseudorandomness with Short Seeds. We obtain our positive results by a generic transformation that converts a secure implementation P of a primitive $\mathcal{P}$ to one that is secure even in the presence of p-tampering attacks. Our transformation is in fact very simple: given an implementation P with standard security, we convert it to $\overline{P}$ that generates a short random seed x with length $s \leq 1/p$, and then uses a PRG $G : \{0,1\}^s \to \{0,1\}^{\mathrm{poly}(n)}$ to generate a pseudorandom string $G(x)$. Finally, it emulates P with $G(x)$ as the randomness. Note that by doing so, P only needs to use s random bits to generate a seed x for G, and so, on average, only 1 bit of the randomness gets tampered with during the p-tampering attack.

First note that when we use the PRG G over a seed s of length $1/p$ and use $G(s)$ instead of the truly random bits, the scheme $\overline{P}$ (typically) remains secure if P was originally secure due to the pseudorandomness of $G(s)$.[7] Also, since the min-entropy of the tampered s is at least $\lg[((1+p)/2)^{1/p}]$, any event E involving a system that executes P, would happen with a tampered s with probability at most $\varepsilon \cdot ((1+p)/2)^{1/p}/2^{1/p} < \varepsilon \cdot e$ where ε is the probability of E happening in the original (un-tampered) game. Therefore, if originally we had $\varepsilon = \mathrm{negl}(n)$, the probability of adversary winning remains negligible $e\varepsilon = \mathrm{negl}(n)$.

Our transformation applies to natural primitives with a security game that can be captured by a "threshold-t falsifiable game" with $t = 0$. A threshold-t falsifiable game Π is simply a game

[7] More formally, for this statement to be true, we need the event E that the security is broken to be efficiently recognizable.

between an efficient challenger C and an adversary A such that C outputs accept or reject after the interaction with A, and the game Π is secure if for every efficient adversary A, C outputs accept with probability at most $t(n) + \mathrm{negl}(n)$. Threshold-0 falsifiable games, in general, capture primitives with security defined as hardness of searching secrets. As mentioned, this includes signature schemes, witness-hiding protocols, and identification schemes.

Our result about tamper-resilient identification schemes might seem skeptical at first because most known identification schemes are based on zero-knowledge protocols, and we have demonstrated earlier that zero-knowledge is impossible in the presence of tampering. However, tamper-resilient identification schemes are still possible to obtain because identification schemes only rely on the weaker property of *witness hiding* (as opposed to zero-knowledge property), and in this section we show that the witness hiding property can be preserved under tampering.

Beyond Threshold-0 Primitives. The above mentioned idea of using pseudorandomness can only be applied to threshold-0 primitives since it relies on the fact that adversary's original winning probability is at most negligible. We also show how to obtain tamper-resilient implementations of cryptographic primitives even when the security game is not threshold-0, but here we assume the number of honest parties to be at least 2. This result, on a high level, is obtained by one honest party Alice helping the other honest party Bob extract pure randomness from its tampered randomness $\widehat{X_B}$. To do this, Alice sends random string $\widehat{X_A}$ (which might also be a tampered string) to Bob, and Bob applies a two-source extractor to get pure randomness $R_B = \mathsf{Ext}(\widehat{X_A}, \widehat{X_B})$. In case $\widehat{X_A}$ is sent over a public channel observed by the adversary, we would need a *strong* two-source extractor [DO03] such that R_B remains uniformly random even conditioned on the value of $\widehat{X_A}$. This idea can be applied to obtain private truly random seeds where we have two conditions: **(1)** $p < \sqrt{2} - 1$; this is required to guarantee that each of $\widehat{X_A}, \widehat{X_B}$ have sufficient min-entropy required by the known two-source extractors, and that **(2)** the honest parties know the set of honest parties. For the more general case of any constant $p < 1$, and where the (at least two) honest parties do not know each other, we rely on the network extractor paradigm of [KLR09] to obtain private *pseudorandom* seeds under (non-standard) computational assumptions.

In the following subsections we present our positive results in more details.

B.1 Tamper Resilient Signatures

As a concrete example, we will first show how to achieve tampering resilient signatures. The arguments for the other primitives will be indeed similar.

Definition B.1 (Many-Time Signatures). *A many-time signature scheme $P = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Ver})$ is* secure *if every PPT adversary A wins in the following game $\Pi = (A, C)$ with negligible probability:*

- *C executes $\mathrm{Gen}(1^n)$ to generate signing key sk and verification key vk, and sends vk to A.*
- *For $i \in [\mathrm{poly}(n)]$ where $\mathrm{poly}(n)$ is bounded by the running time of A, A sends a message m_i to C and receives $\sigma_i = \mathrm{Sign}_{\mathsf{sk}}(m_i)$ from C.*
- *A generates and sends (m, σ) to C.*
- *C accepts (i.e., A wins) if $m \neq m_i$ for every i, and $\mathrm{Ver}_{\mathsf{vk}}(m, \sigma) = 1$.*

We say that any A breaks the security of P if he wins with non-negligible probability.

Recall that, assuming the existence of OWFs, we know how to create many-time secure signature schemes that are deterministic in their signing and verification phases. In fact, any many-time signature scheme that uses randomness in the signing phase can be converted into one that is deterministic, by using PRFs. This could be done by generating the seed for the PRF in the key-generation phase, and adding it as part of the secret key. Then, whenever a message m is to be signed, we apply the PRF to m, and use the result as the randomness needed to sign.

The definition of p-tamper-resilient signatures follows from Definition B.1 and asserting that the security holds against p-tampering adversaries.

Definition B.2 (Tamper Resilient Signature)**.** *A many-time signature scheme is p-tamper-resilient if it remains secure according to Definition B.1, even if the adversary, at the beginning of the game, generates a p-tampering circuit Tam that transforms the uniform randomness $U_{\mathrm{poly}(n)}$ of the challenger (i.e., the randomness for key-generation and signing) into a $U_{\mathrm{poly}(n)}^{\mathsf{Tam},p}$ and the challenger uses $U_{\mathrm{poly}(n)}^{\mathsf{Tam},p}$ in its interaction.*

Theorem B.3. *Let $\alpha \in (0,1)$ be a constant. If there exists a many-time secure signature scheme, then there exists a many-time $(n^{-\alpha})$-tamper-resilient signature scheme.*

Proof. Let $P = (\mathrm{Gen}, \mathrm{Sign}, \mathrm{Ver})$ be a secure signature scheme. W.l.o.g., we assume that P is deterministic except in Gen. Suppose further that $\mathrm{Gen}(1^n)$ uses $m(n)$ bits of randomness. Let $G : \{0,1\}^s \to \{0,1\}^{m(n)}$ be a pseudorandom generator with $s = O(n^\alpha)$. (Recall that secure signature schemes imply OWFs [IL89], which imply PRGs with arbitrary polynomial length output.) We define $\overline{P} = (\overline{\mathrm{Gen}}, \mathrm{Sign}, \mathrm{Ver})$, where $\overline{\mathrm{Gen}}(1^n)$ works as follows: $\overline{\mathrm{Gen}}$ generates a random seed X of length $s = n^\alpha$, and then computes $r = G(X)$. It then emulates Gen using r as random coins. In the following by $\overline{\Pi}$ we refer to the new scheme and by $\overline{C}$ we refer to the new challenger of the multi-message security game of $\overline{\Pi}$.

Roughly speaking, Theorem B.3 is proved by observing that **(1)** P remains secure even if the randomness of P is generated from the pseudorandom source $G(X)$, and **(2)** after tampering, the seed X used by $\overline{P}$ still has $s - O(1)$ bits of min-entropy.

The Formal Proof. Suppose that there exists an efficient adversary $\overline{A}$ that breaks the scheme $\overline{\Pi}$ by winning the game against $\overline{C}$ with non-negligible probability. Namely, there exists a non-negligible ε such that for infinitely many n,

$$\mathsf{P}[\langle \overline{A}, \overline{C} \rangle(1^n) = 1] \geq \varepsilon(n).$$

By an averaging argument and fixing coins of A, we can assume w.l.o.g. that $\overline{A}$ is deterministic and thus the tampering circuit Tam is fixed. Observe that in this case, the interaction $\langle \overline{A}, \overline{C} \rangle$ in $\overline{\Pi}$ is equivalent to the interaction $\langle A, C[G(U_s^{\mathsf{Tam}})] \rangle$ in Π, where the notation $\langle A, C[D] \rangle$ means the output of simulating $\langle A, C \rangle$ with C's coins drawn from distribution D, and A is simply $\overline{A}$ without sending the tampering circuit. Thus, we have

$$\mathsf{P}[\langle A, C[G(U_s^{\mathsf{Tam}})] \rangle = 1] \geq \varepsilon.$$

Note that since U_s^{Tam} is a p-tampering source over $s = 1/p$ bits, it has min-entropy $s - \log e$. This can be seen by observing that for every $r \in \{0,1\}^s$, $\mathsf{P}[U_s^{\mathsf{Tam}} = r] \leq ((1+p)/2)^s \leq e^{p \cdot s}/2^s = e/2^s$.

This means that if we use the randomness U_s rather than U_s^{Tam} the probability of any event would decrease at most by a factor of e. Namely, A is a successful attacker breaking the scheme with probability $\varepsilon(n)/e > 1/\mathrm{poly}(n)$. But, this contradicts the security of the scheme P and completes the proof of Theorem B.3. □

Our proofs of security for identification schemes and witness hiding protocols follow the same lines as those of our result about signature schemes. Thus we will only discuss the specific modifications we make in each case.

B.2 Identification Schemes

An identification scheme is a protocol that allows Alice to prove her identity to Charlie, and a malicious Bob cannot impersonate Alice even if he gets to see Alice proving her identity polynomially many times. Each identity can be represented by some $\alpha \in \{0,1\}^n$ and there exists a "public-file" that couples every identity α by some secretly generated public information $I(s,\alpha)$. The secret s (which is chosen at random) is given to the person with identity α. The public record $(\alpha, I(s,\alpha))$ will be accessed by both the "prover" (of the identity) and the "verifier" during the identification process.

Definition B.4. *An identification scheme $\Pi = (I, P, V)$ has three efficient components: The algorithm I takes as input $(\alpha \in \{0,1\}^n, s \in \{0,1\}^{\mathrm{poly}(n)})$ and outputs $v \in \{0,1\}^{\mathrm{poly}(n)}$ and (P, V) form an interactive proof system. We also demand the following properties.*

- **Completeness:** $\mathsf{P}[\langle P(s), V\rangle(\alpha, I(s,\alpha)) = 1] = 1$ *holds for every* $\alpha \in \{0,1\}^n, s \in \{0,1\}^{\mathrm{poly}(n)}$.
- **Soundness:** *For every $\alpha \in \{0,1\}^n$, every* $\mathrm{poly}(n)$*-sized (non-uniform) interactive adversary A wins in the following game with* $\mathrm{negl}(n)$ *probability.*
  1. *For $s \xleftarrow{\$} \{0,1\}^{\mathrm{poly}(n)}$, A interacts with $P(s)$ polynomially many times (bounded by the running time of A)*
  2. *Then A (while keeping its internal state) interacts with V on common input $(\alpha, I(s,\alpha))$.*
  3. *A wins if V accepts the interaction.*

Remark B.5 (Identification from Signatures)**.** *Many-times secure signature schemes can be used to obtain identification schemes as follows. The information generation algorithm I, for every identity α generates a pair of signing key s and verification key v. (Note that the signing key can always be assumed to be a uniformly sampled string.) The verifier V, given the public record (α, v) sends a uniformly chosen random message m to P who signs m using the signing key s, and sends the signature σ back to V. Finally V accepts if σ is a valid signature of m. It is easy to see that the many-time security of the signature scheme implies the soundness of the constructed identification scheme. Further, if the signature scheme has a deterministic* Sign *algorithm, then the identification scheme has a deterministic prover P (but the verifier is randomized).*

Although secure signature schemes imply secure identification schemes according to Remark B.5, this transformation does not preserve the tamper resilience by definition. The reason is that, even though it is known that there are signature schemes with deterministic verification, this identification scheme uses some extra randomness which could also be tampered with by the adversary.

Theorem B.6. *If there exists a secure identification scheme, then there exists a p-tamper-resilient identification scheme, with $p = n^{-\alpha}$, for any constant $\alpha > 0$.*

Proof. The existence of a secure identification scheme implies the existence of OWFs [IL89], which in turn implies the existence of a secure signature scheme Σ with a deterministic signing algorithm. The signature scheme Σ can be used to obtain an identification scheme Π with a deterministic prover P according to the construction described in Remark B.5.

Since the prover P in the obtained identification scheme is deterministic, the adversary A can only usefully tamper with the randomness of the information generation algorithm I and the verifier V. I uses its randomness in the initial *generation* phase, and V uses its randomness in the final *verification* phase. Since in the security game of Π there is only a *single* verification performed, the number of random bits needed for the information generation and the final verification is an a priori fixed polynomial. Therefore, we can apply the same idea that we used to make signature schemes tamper resilient. Namely, we modify I and V *both* so that each of them uses only a short random seed of length $O(1/p)$ and applies a PRG to expand it to the right size $\text{poly}(n)$. The proof of tamper-resilience of the new scheme is identical to that of Theorem B.3.

□

B.3 Witness Hiding Protocols

Definition B.7 (Witness Hiding Protocols)**.** *Suppose R is an* NP *relation. Namely, there is an efficient algorithm that accepts (x, w) when $|w| = \text{poly}(|x|)$ iff $(x, w) \in R$. Suppose* Gen *is a randomized sampling procedure that given 1^n (and enough random bits) runs in time* $\text{poly}(n)$ *and outputs some (x, w) of length $|x| = n$. A proof system (P, V) for the relation R is* one-time witness hiding *w.r.t. the sampling procedure* Gen *if, for all PPT adversaries A, A wins in the following game with negligible probability:*

1. $\text{Gen}(1^n)$ *generates* $(x, w) \in R$.
2. *A interacts in* $\langle P(w), A\rangle(x)$.
3. *A outputs w', and wins if* $(x, w') \in R$.

(P, V) is called q-time witness hiding (resp. witness hiding) if A is allowed to participate in $q =$ $\text{poly}(n)$ *(resp. any polynomial) number of interactions for the same (x, w) before outputting w'.*

All variants of witness hiding defined in Definition B.7 can be defined under tampering attacks.

Theorem B.8. *Any (q-times) witness hiding protocol (P, V) for some relation R can be converted into a stateless (q-times) p-tamper-resilient witness-hiding protocol, with $p = n^{-\alpha}$, for any constant $\alpha > 0$. Achieving p-tamper resilience witness-hiding (with unbounded number of repetitions) is also possible assuming that the prover is allowed to keep internal state between repetitions.*

Proof. First note that any witness hiding protocol implies the existence of one-way functions as follows: Given input r, use r as randomness and using $\text{Gen}(\cdot)$ sample $(x, w) \in R$ and output $x = f(r)$. Inverting $f(r)$ for a random r implies breaking the witness hiding of the corresponding protocol even without participating in the interactive phase. Again, we can again use the existence of OWFs to get PRGs [HILL99].

To achieve tamper resilience under a single execution of the interactive proof we can use the same exact trick as we did for signature schemes: $\overline{\text{Gen}}$ uses a "short" random seed s of length $1/p$, and applies a PRG to make it of sufficiently long to run Gen. The prover also does the same thing; i.e., it tosses $1/p$ many coins and expands them to the right length using a PRG. In the following we focus on the case of sequential repetition.

A Priori Bounded Number of Repetitions. If the number of sequential repetitions is an a priori bounded polynomial $q = \text{poly}(n)$, then we can still take the same approach as that of the single-repetition case. Namely, one can think of the security game with a fixed number $q = \text{poly}(n)$

of repetitions as a game with a fixed time-complexity that needs a fixed $\ell = \mathrm{poly}(n)$ number of random bits used by the challenger $C = (\mathrm{Gen}, P_1, P_2, \ldots, P_q)$. Thus, by using a PRG we can expand an initial $1/p$ number of true random bits to ℓ bits and apply the same analysis.

Unbounded Number of Repetitions. At first sight, it seems that our general technique of using PRGs does not work if we go through an unbounded number of repetitions (i.e., a polynomial that is chosen by the adversary). But, if the prover can keep internal state, we can use a PRF as a way to obtain a PRG with an unbounded number of output bits. (Note that if g is a PRG, for any polynomial ℓ, the function $G(s) = [g_s(1), \ldots, g_s(\ell)]$ is a PRG.) Thus, again, we start by using $1/p$ truly random bits s, and use a PRG g with key s and compute it over $g_s(1), g_s(2), \ldots$ to obtain enough number of pseudorandom bits that is needed for the execution of the repetitions.

To analyze the tamper resilience of the scheme above, we apply the same analysis presented for the previous cases. All we have to do additionally is to first *fix* the adversary, which fixes the number of its sequential repetitions, and then apply the same analysis as before. This was not previously needed because, e.g., in case of signatures, even though the adversary's complexity was not fixed before the analysis, we had an upper-bound on the number of random bits needed by the challenger, but here we get this upper-bound after fixing the adversary's running time. □

B.4 Weak Semantic Security

Here we present the following new definition as a relaxation of the semantic security and prove our positive result about the possibility of tamper resilient encryption under this relaxed definition.

Definition B.9. *A public-key or private-key encryption scheme* (Gen, Enc, Dec) *is called* $(1+\delta)$-weakly semantically-secure *if for every* $\mathrm{poly}(n)$*-sized (non-uniform) adversary* Adv, *arbitrary distribution* $\mathcal{X}$ *over* $\mathcal{M}$ *and arbitrary functions* $I, f \colon \mathcal{M} \mapsto \{0,1\}^{\mathrm{poly}(n)}$ *there exists a PPT simulator* Sim *such that:*

$$\mathsf{P}_{m \xleftarrow{\$} \mathcal{X}}[\textsc{Adv}(I(m), \mathrm{Enc}(m)) = f(m)] \leq (1+\delta) \cdot \mathsf{P}_{m \xleftarrow{\$} \mathcal{X}}[\textsc{Sim}(I(m)) = f(m)] + \mathrm{negl}(n).$$

The probabilities above are also over the generation of the encryption key. Note that (Gen, Enc, Dec) *is* semantically-secure *if it is* 1*-weakly semantically secure.*

Definition B.10 (Tamper Resilient Weakly Semantically Secure Encryption)**.** *We call an encryption scheme* Π tamper-resilient weakly semantically secure *if for every* $p = 1/poly(n)$ *(where* n *is the security parameter)* Π *is secure under Definition B.9 even when the adversary* Adv *is able to generate a* p*-tampering circuit* Tam *that modifies the randomness of the key generation and encryption algorithms into* p*-tampering sources.*

We prove the following positive result about the possibility of achieving weakly semantically secure encryptions under Definition B.9. Our result implies that when we consider the weak semantic security according to Definition B.9 for functions $f(m)$ which cannot be computed by more than $\mathrm{negl}(n)$ probability given the encryption of m, one can always preserve this property (of f remaining "hard" to compute given the encryptions) even in the presence of p-tampering attacks for any $p = 1/\mathrm{poly}(n)$.

Theorem B.11. *Suppose there exits a semantically secure (public-key or private-key) encryption scheme, and* $p = n^{-\alpha}$ *for a constant* $\alpha > 0$*. Then for every* $0 < \beta < \alpha$*, there exists a* $(1 + O(n^{-\beta}))$ *weakly semantically-secure encryption scheme that is secure against* p*-tampering adversaries.*

Proof. We follow the same paradigm of previous sections but with even smaller PRG seeds. The existence of any semantically secure encryption scheme implies the existence of OWFs, which in turn implies the existence of PRGs with arbitrary polynomial stretch [HILL99]. Thus, for every constant $\gamma > 0$ we can use two seeds of length $n^\gamma/2$ to get enough number of pseudorandom bits to use in Gen and Enc. To get the $(1 + O(n^{-\beta}))$ weakly semantically secure scheme $(\mathrm{Gen}', \mathrm{Enc}', \mathrm{Dec})$ we set $\gamma = \alpha - \beta$ and then modify the Gen and Enc algorithms (and call them $\mathrm{Gen}', \mathrm{Enc}'$) to use a truly random seed of length $n^\gamma/2$ and stretch it to the necessary number of bits.

First we claim that if $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ was semantically secure, $(\mathrm{Gen}', \mathrm{Enc}', \mathrm{Dec})$ remains semantically secure. This is nontrivial because the distribution $\mathcal{X}$ is not necessarily efficiently samplable and f, I are not necessarily efficiently computable. We use the classical result of [GM84] that a public key scheme is semantically secure iff it is CPA secure. A CPA secure is defined based on a security game in which the adversary, given the public key, chooses two messages m_0, m_1, receives the encryption of m_b for $b \xleftarrow{\$} \{0,1\}$ and wins if he guesses b correctly with probability $1/2 + 1/\operatorname{poly}(n)$. It is easy to see that the CPA property is preserved when we use pseudorandom bits in Gen, because otherwise the security game itself can be turned into a distinguisher against the PRG that is used to expand the pseudorandom bits used in Gen.

Now we study the security of $(\mathrm{Gen}', \mathrm{Enc}', \mathrm{Dec})$ in the presence of $n^{-\alpha}$-tampering attacks. During a $n^{-\alpha}$-tampering attack, every bits of the PRG seed of length n^γ (applied to two seeds of length $n^\gamma/2$) is tampered with by the adversary only with probability $n^{-\alpha}$. Therefore, the probability of any seed being the result of the tampering is at most $(1 + n^{-\alpha}/2)^{n^\gamma} \leq e^{n^{-\beta}}/2^{n^\gamma}$. Similarly to the argument in the analysis of the tamper-resilient signatures, we conclude that any event E which an efficient adversary could make it happen without the tampering, would happen now (with the tampering) with probability at most

$$e^{(n^{-\beta})} \cdot \mathsf{P}[E \text{ without tampering}] \leq (1 + O(n^{-\beta})) \cdot \mathsf{P}[E \text{ without tampering}].$$

We conclude Theorem B.11 by considering the event E to be the event that the adversary is computing $f(m)$ correctly. In the original semantically secure scheme it holds that

$$\begin{aligned} &\mathsf{P}_{m \xleftarrow{\$} \mathcal{X}}[\textsc{Adv}(I(m), \mathrm{Enc}(m)) = f(m)] \\ \leq\ &\mathsf{P}_{m \xleftarrow{\$} \mathcal{X}}[\textsc{Sim}(I(m)) = f(m)] + \operatorname{negl}(n) \end{aligned}$$

which implies that in the new scheme $(\mathrm{Gen}', \mathrm{Enc}', \mathrm{Dec})$ it holds that

$$\begin{aligned} &\mathsf{P}_{m \xleftarrow{\$} \mathcal{X}}[\textsc{Adv}(I(m), \mathrm{Enc}(m)) = f(m)] \\ \leq\ &(1 + O(n^{-\beta})) \cdot (\mathsf{P}_{m \xleftarrow{\$} \mathcal{X}}[\textsc{Sim}(I(m)) = f(m)] + \operatorname{negl}(n)) \\ \leq\ &\operatorname{negl}(n) + (1 + O(n^{-\beta})) \cdot \mathsf{P}_{m \xleftarrow{\$} \mathcal{X}}[\textsc{Sim}(I(m)) = f(m)]. \end{aligned}$$

□

B.5 Generalization to Threshold-0 Primitives

Our tamper-resilient constructions for Signatures, Identification Schemes, Weakly-Semantically Secure Encryptions, etc, above clearly suggest a general approach that can be applied to a wide range of primitives where the adversary's job is to make the challenger accept with non-negligible probability. Here we describe the abstract properties of the primitives that makes this approach applicable.

Any natural cryptographic primitive $\mathcal{P}$ can be viewed as a set of interactive algorithms with some completeness properties imposed on them. For example an encryption scheme has three components of $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$. In the security game of $\mathcal{P}$ some of these components are executed on the "challenger" side and their randomness is prone to online tampering. There might be several "instantiations" of each of these components by the challenger during its interaction with the adversary. For example in the security game of multi-message signatures, the signing algorithm might be executed an unbounded polynomial number of times depending on adversary's choice. If an executed component can keep internal state across different instantiations, we still consider this a single instantiation. For example, when a tampering verifier interacts with a prover sequentially in $\mathrm{poly}(n)$ repetitions while the prover keeps an internal state (as it was the case in witness-hiding protocols in Section B.3), we consider this a single instantiation.

In order to apply the pseudorandomness-based approach of previous sections to a primitive and make its implementation resilient to p-tampering for $p = 1/\mathrm{poly}(n)$, it is sufficient to have both of the following two conditions:

1. The threshold of the security game is 0 (i.e., adversary needs to with with $1/\mathrm{poly}(n)$).

2. The number of randomized components of the primitive $\mathcal{P}$ which are instantiated during the security game are at most $n^{-\beta}$ for a constant $\beta < \alpha$ where $p = n^{-\alpha}$.

When we have the above two conditions, all we have to do is to use pseudorandomness to execute the components of $\mathcal{P}$ as follows: each component uses a seed s of length n^{γ} as the key to a PRF g where $\gamma = \alpha - \beta$ and uses $g_s(1), g_s(2), \dots$ to get its "random" bits during its execution. Note that, for any fixed polynomial t, the sequence $[g_s(1), g_s(2), \dots, g_s(t)]$ is pseudorandom. So after fixing the complexity of the adversary $\textsc{Adv}$ who interacts with the challenger, each instantiated component of $\mathcal{P}$ is using a PRG that stretches a seed s of length n^{γ} to the needed number of random bits. Each of these seeds could be p-tampered with by the adversary in an instantiation during the security game, and so their joint min-entropy loss (compared to a truly random seed) is at most $O(n^{\beta} \cdot n^{\gamma} \cdot n^{-\alpha}) = O(1)$. Therefore, any event E that the original (non-tampering) adversary could make it happen with probability ρ, will happen in the security game with the tampering attack with probability at most $2^{O(1)}\rho$, which remains $\mathrm{negl}(n)$ assuming that ρ was already negligible.

B.6 Beyond Threshold-0 Primitives

Here we describe a general method that allows us to make a cryptographic primitive tamper-resilient to p-tampering attacks for any constant $p < 1$, as long as there are two honest parties involved in the primitive (that know each other). We start by describing a solution for the task of key agreement (which is a primitive in which both parties are honest and the adversary is a passive eavesdropper), and then describe how it can be generalized to more settings by relying the work of [KLR09] at the cost of non-standard computational assumptions.

B.6.1 Tamper-Resilient Key Agreement

A tampering adversary attacking a key agreement protocol $\widehat{\Pi}$ between Alice and Bob sends tampering circuits Tam_A and Tam_B to Alice and Bob where each tampering circuit tampers with the randomness of the corresponding party. In this section we prove the following theorem.

Theorem B.12. *Suppose Π is a secure key agreement protocol. Then for every constant $p < \sqrt{2}-1$ there is another key agreement protocol $\widehat{\Pi}$ which is secure against p-tampering adversaries.*

Proof. We show how Alice and Bob can start from $2n$ bits of tampered random bits and obtain $\Omega(n)$ bits X_A, X_B each in a way that (X_A, X_B) are (jointly) statistically close to uniform.

Alice and Bob divide their original $2n$-bit random strings into two equal parts and call them $(X_A^1, X_A^2), (X_B^1, X_B^2)$ where the bits in X_A^1 (resp. X_B^1) are tossed before the bits in X_A^2 (resp. X_B^2). Suppose the tampered values of these random seeds are $(\widehat{X}_A^2, \widehat{X}_A^2), (\widehat{X}_B^2, \widehat{X}_B^2)$. We use the following lemma which is implied by Theorem 1 of [DO03].

Lemma B.13 (Strong Two-Source Extractors)**.** *Suppose X and Y are two random variables defined over $\{0,1\}^n$ with min-entropy $\alpha \cdot n$ and $\alpha > 1/2$, then there is an efficient function $\mathsf{Ext}\colon \{0,1\}^{2n} \mapsto \{0,1\}^m$ (independent of X, Y) for $m = \Omega(n)$ such that $(\mathsf{Ext}(X,Y), Y)$ is* $\operatorname{negl}(n)$*-close to (U_m, Y).*

Description of $\widehat{\Pi}$: Alice (resp. Bob) will send $\widehat{X}_A^1$ (resp. $\widehat{X}_B^1$) to the other party. Then Alice (resp. Bob) will apply the strong extractor of Lemma B.13 and gets $X_A = \mathsf{Ext}(\widehat{X}_A^2, \widehat{X}_B^1)$ (resp. $X_B = \mathsf{Ext}(\widehat{X}_B^2, \widehat{X}_A^1)$). Then they use (X_A, X_B) as their randomness and execute Π. Note that By taking n large enough, X_A, X_B would be long enough to execute Π.

Since the tampering circuits Tam_A and Tam_B cannot communicate, it follows that the random variables $\widehat{X}_A^2$ and $\widehat{X}_B^1$ are independent. Since, $p < \sqrt{2} - 1$, the min-entropy of the "source" $\widehat{X}_A^2$ is at least α for $\alpha > 1/2$ even conditioned on the publicly revealed message $\widehat{X}_A^1$, simply because $\widehat{X}_A^2$ was tampered with *after* $\widehat{X}_A^1$. Therefore, the extracted X_A will remain statistically close to a secret U_m for $m = \Omega(n)$ in eyes of the adversary, even conditioned on the information sent over the public channel. The same argument holds for the uniformity of X_B. Thus, Alice and Bob can use the (statistically close to) pure random seeds X_A, X_B to run the protocol Π securely.

□

Remark B.14 (Synchronization Issue)**.** *A subtle point here is that Alice and Bob should toss coin and obtain $(\widehat{X}_A^2, \widehat{X}_A^1)$ and $(\widehat{X}_B^2, \widehat{X}_B^1)$* before *sending $\widehat{X}_A^1$ and $\widehat{X}_B^1$ to each other. Otherwise, if say Alice tosses coin after receiving $\widehat{X}_B^1$, then the tampering circuit Tam_A could tamper X_A^2 into $\widehat{X}_A^2$* with the knowledge *of $\widehat{X}_B^1$ which makes $\widehat{X}_B^1$ and $\widehat{X}_A^2$ dependent and thus we cannot apply Lemma B.13.*

B.6.2 Generalization

The basic idea behind Theorem B.12 can be generalizes to make primitives with at least two honest parties (where the honest parties know each other) p-tamper-resilient for constant $p < \sqrt{2}-1$. Using a result of [KLR09] we can obtain tamper-resilient primitives for *all* constants $p < 1$ when there are at least two honest parties, *even without knowing them.* This comes, however, at the cost of making a non-standard computational assumption.

Definition B.15. *We call a sequence of permutations $f\colon \{0,1\}^n \mapsto \{0,1\}^n$ a k-one-way permutation if $f(X)$ is $n^{\log n}$-hard to invert for every distribution X of min-entropy $\geq k$.*

Theorem B.16 (Network Extractors – [KLR09])**.** *Suppose $P_1, \ldots, P_n$ are n parties where at least two of them are honest, and suppose the party P_i has access to a random source $\overline{X}_i$ with min-entropy $\geq \alpha \cdot n$. Assuming the existence of $\alpha n/4$-one-way permutations, there is a protocol between $P_1, \ldots, P_n$ at the end of which each honest party P_i receives a private pseudorandom seed Y_i of length $\Omega(n)$.*

To make any primitive with at least two honest parties tamper-resilient for an arbitrary constant $p < 1$, we can use Theorem B.16 over $\alpha = \lg({}^{2}/{}_{1+p})$ where the defective random source of each party is a Santha-Vazirani source which in turn is the result of a p-tampering attack. Similarly to the compiler of Theorem B.12, here the parties first toss coins to obtain their samples form the defective sources $\overline{X}_i$'s. Then, they will engage in a network extraction protocol of Theorem B.16 to obtain their pseudorandom seeds X_i's, and then will run any protocol which is proven to be secure without tampering.

Similarly to the case of Theorem B.16 (as mentioned in Remark B.14), to apply Theorem B.16 to cases where the source of imperfect randomness is due to tampering, we require the parties to be synchronized and perform the multiparty task in two phases (where the first phase should be finished before any party starts the second phase). In the first phase the parties toss their coins, which through the tampering attack is transformed into imperfect sources with high min-entropy. Then, they will engage in the network extraction protocol of Theorem B.16 to purify this randomness to be used later on.

C Related Work on Tampering

Some of the previous works on tamper-resilient cryptography focused on compiling a circuit C holding a secret into a new circuit C' that hides the secret even if the attacker gets to tamper with the computation of C'.

Main Difference: Achieving Security. In contrast, in this work we focus on whether one can preserve the security of cryptographic schemes under tampering attacks. The simulation property of tamper-resilient compilers do not necessarily guarantee that if the sender algorithm is compiled into a "tamper-resilient" version, then the encryption scheme is tamper-resilient. This is due to the fact that the simulation property of those compilers only guarantee that an attacker cannot learn more from tampering with the sender strategy than it could have with black-box access to it. But in the case of encryption schemes, it is actually the *input* to the algorithm (i.e., the message to be encrypted) that we wish to hide (as opposed to some secret held by the algorithm)

For completeness, below we briefly outline some of these works which can be divided into two groups: achieving resilience against tampering with memory and against tampering with the computation.

Tampering with Memory. The work of Gennaro et al. [GLM$^+$04] was the first to formally study tamper resilience from an algorithmic (as opposed to a hardware-based) point of view. In their model they deal with a cryptographic algorithm $G(s, x)$ holding a secret s that given an input x outputs some y (e.g., G could be a signing or a decryption algorithm). The adversary gets hold to a box running G and is allowed to perform tampering attacks over the "memory" of G that holds the secret state of G (which in particular includes the secret s). The adversary's goal is to break the security of G with respect to the *original* value of s (e.g., forge a signature). Gennaro et al. showed that any cryptographic protocol that can be "tested" for malfunctioning (e.g., a signature box can be publicly tested by verifying the signatures it produces) can be broken by an adversary that only performs "resetting" attacks over the bits of the secret state. The idea is to test the bits of s one by one. In i^{th} step the adversary sets the bit s_i of s to zero. Then if the newly tampered value of s passes the malfunctioning test (e.g., can be used to sign messages successfully), it means that an acceptable value for the first i bit of s is discovered; otherwise we set s_i to one.

Gennaro et al. also present a positive result based on the assumption that there is a tamper proof component (e.g., some "circuitry") available. This way, a trusted party who generates the circuit of G would sign the original secret state of G and hardwire this signature together with the corresponding verification key into the tamper proof part of the code of G. The execution of G

will always start by first testing the signature of its own secret state. If the signature verification passes it will use this state to run its algorithm, and otherwise it will "self-destruct".

The work of Dziembowski et al. [DPW10] extended the positive result of [GLM$^+$04] and achieved information theoretic security for a more restricted class of tampering functions through introducing and constructing "non-malleable" codes. These codes, then, are used to encode and decode the internal secret state of G before and after using it (instead of self-destruct). The works of [KKS11, LL12] go beyond only tamper resilience by achieving also *leakage resilience* when both of tampering and leakage are performed only over the memory. Finally the work of Choiet al. [CKM11] studies the tamper resilience in the context of universal composability and studies specific cryptographic algorithms against *affine* tampering functions.

Tampering with Computation. Boneh, DeMillo and Lipton [BDL97] showed that introducing minor random errors during the computation of some implementations of certain cryptographic schemes can be exploited by the adversary to a large extent and break the scheme completely. The result was rather shocking, since some natural implementations would completely break down by a *single* call along with a random tampering performed.

Ishai et al. [IPSW06] took on the positive side and showed how to make a circuit that is already accessible by the adversary as an input-output functionality secure against being tampered with up to t wires in every input-output call. The security here means that the view of any such adversary can be simulated by a simulator who does *not* tamper with the circuit and only uses it as a black-box. Thus the compiler of [IPSW06] shows how to keep a key inside a circuit in a secure way against tampering (e.g., a decryption circuit, or a signing circuit). The tampering functions here are restricted in the following sense: they only choose a set of t wires, and for each of them decide whether to set them to zero or one, or to flip their values. Our positive results (see Section B), however, apply even to the case that the adversary can observe the whole internal state of the tampered algorithm, and choose the value of the tampered "random" bits based on that.

The subsequent work of Faust et al. [FPV11] followed the framework of [IPSW06] and extended their work to the setting that there is no fixed upper bound t on the number of tampered wires in each round of executing the tampered algorithm, but there is a constant probability $\delta > 0$ that each wire that is chosen to be tampered with remains untampered.