Bounds on the Efficiency of Generic Cryptographic Constructions*

Rosario Gennaro† Yael Gertner‡ Jonathan Katz§
Luca Trevisan¶

Abstract

A central focus of modern cryptography is the construction of efficient, "high-level" cryptographic tools (e.g., encryption schemes) from weaker, "low-level" cryptographic primitives (e.g., one-way functions). Of interest are both the *existence* of such constructions, and their *efficiency*.

Here, we show essentially-tight lower bounds on the best possible efficiency of any black-box construction of some fundamental cryptographic tools from the most basic and widely-used cryptographic primitives. Our results hold in an extension of the model introduced by Impagliazzo and Rudich, and improve and extend earlier results of Kim, Simon, and Tetali. We focus on constructions of pseudorandom generators, universal one-way hash functions, and digital signatures based on one-way permutations, as well as constructions of public- and private-key encryption schemes based on trapdoor permutations. In each case, we show that any black-box construction beating our efficiency bound would yield the unconditional existence of a one-way function and thus, in particular, prove $P \neq NP$.

1 Introduction

A central focus of modern cryptography is the construction of "high-level" cryptographic protocols and tools that are both provably secure and efficient. Generally speaking, work proceeds along two lines: (1) demonstrating the *feasibility* of a particular construction, based on the weakest possible primitive; and (2) improving the *efficiency* of such constructions, either based on the weakest primitive for which a construction is known or perhaps by assuming the existence of a stronger primitive. The first of these approaches has been immensely successful; for example, the existence of one-way functions is known to be sufficient for constructing pseudorandom generators [8, 42, 22, 27], pseudorandom functions [21],

*This work appeared in preliminary form as [15, 14].

†rosario@watson.ibm.com. IBM T.J. Watson Research Center.

‡ygertner@uiuc.edu. Department of Psychology, University of Illinois at Urbana-Champaign. Work done while in the Dept. of Computer and Information Science, University of Pennsylvania.

§jkatz@cs.umd.edu. Department of Computer Science, University of Maryland. Portions of this work were done while at DIMACS.

¶luca@eecs.berkeley.edu. Computer Science Division, UC Berkeley. Work done while at Columbia University. This research was supported by NSF grant CCR 9984703.

universal one-way hash functions and digital signature schemes [36, 38], private-key encryption schemes and message-authentication codes [20], and commitment schemes [35]. In each of these cases one-way functions are also known to be necessary [30, 38], thus exactly characterizing the feasibility of these constructs.

Unfortunately, progress on the second approach — i.e., improving the efficiency of these constructions — has been much less successful. Indeed, while the constructions referenced above are all important from a theoretical point of view, their practical impact has been limited due to their inefficiency. In practice, more efficient constructions based on stronger assumptions (or, even worse, heuristic solutions with no proofs of security) tend to be used. Furthermore, relying on stronger assumptions (or resorting to heuristic solutions) seems necessary to obtain improved efficiency; for each of the examples listed above, no provably-secure constructions based on general assumptions are known which improve upon the efficiency of the initial solutions.

This trade-off between the efficiency of a cryptographic construction and the strength of the complexity assumption on which it relies motivates the question: *How efficient can cryptographic constructions be when based on general assumptions?* We show in this paper that, in fact, the efficiency of many of the known constructions based on general assumptions *cannot be improved* without using non-black-box techniques, or finding an unconditional proof that one-way functions exist (and hence proving $P \neq NP$).

Our results hold in a generalization of the Impagliazzo-Rudich model [31], introduced by those authors in the context of proving impossibility results for the existence of certain black-box cryptographic constructions (see Section 1.3 for further discussion). Following their work, a number of additional black-box impossibility results have appeared [40, 41, 16, 32, 17, 13]. Kim, Simon, and Tetali [33] initiated work focused on bounding the *efficiency* of black-box cryptographic constructions (rather than their *existence*), and their work provided the original inspiration for our research. We compare our results with those of Kim, et al. in the following section.

1.1 Our Results

Informally, we say a permutation $\pi : \{0,1\}^n \to \{0,1\}^n$ is *one-way with security* S if any circuit of size[1] at most S inverts π with probability less than $1/S$ (one can think of S as a slightly super-polynomial function of n but our results hold for any choice of S). In this work, we consider two types of black-box constructions, described informally now and discussed in more detail in Section 1.3. Following the terminology introduced in [37], a *semi black-box construction* (based on a one-way permutation) is an oracle procedure $P^{(\cdot)}$ such that, for any one-way permutation f given as an oracle, (1) P^f has the desired functionality and (2) P^f is "secure" (in some appropriate sense) against every efficient (oracle) adversary $\mathcal{A}^f$ even when considering adversaries given oracle access to f. In contrast, a *weak black-box construction* is an oracle procedure $P^{(\cdot)}$ satisfying (1) as before but for which, for any one-way permutation f given as an oracle, the only guarantee is that (2) P^f is "secure" against efficient adversaries $\mathcal{A}$ that are *not* given oracle access to f. Both notions preclude using the code (or circuit) of the one-way permutation f in the construction; roughly speaking (see [37] for further elaboration), semi black-box constructions also rule out the use of

[1] We let the *size* of a circuit refer to the number of gates the circuit has.

the *adversary's* code in the *security reduction* (whereas weak black-box constructions do not). Clearly, any semi black-box construction is also a weak black-box construction and so impossibility results for the latter are stronger than impossibility results for the former.

Given these definitions, our results may be summarized informally as follows.

Pseudorandom generators (PRGs). Let U_ℓ denote the uniform distribution over ℓ-bit strings. A PRG is a deterministic, length-increasing function $G : \{0,1\}^\ell \to \{0,1\}^{\ell+k}$ such that $G(U_\ell)$ is computationally indistinguishable (by poly-time algorithms) from $U_{\ell+k}$. The notion of a PRG was introduced by Blum and Micali [8] and Yao [42], who showed that PRGs can be constructed from any one-way permutation. This was subsequently improved by Håstad, et al. [27], who show that a PRG can be constructed from any one-way function. The Blum-Micali-Yao construction, using a later improvement of Goldreich and Levin [22] (see also [18, Section 2.5.3]), requires $\Theta(k/\log S)$ invocations of a one-way permutation with security S in order to construct a PRG stretching its input by k bits. This is the best known efficiency for constructions based on arbitrary one-way permutations.

We show that this is essentially the best efficiency that can be obtained using black-box constructions. More formally, we show that any *weak* black-box construction of a PRG that stretches its input by k bits while making $o(k/\log S)$ invocations of a one-way permutation with security S implies the *unconditional* existence of a PRG (i.e., without any invocations of the one-way permutation). Put another way, the only way to design a more efficient construction of a PRG is to design a PRG from scratch! This would in particular imply the unconditional existence of a one-way function, as well as a proof that $P \neq NP$.

(Families of) universal one-way hash functions (UOWHFs). A UOWHF $\mathcal{H} = \{h_s\}$ is a family of length-decreasing functions (all defined over the same domain and range) such that for any input x and random choice of $h_i \in \mathcal{H}$ it is hard to find a *collision* (i.e., an $x' \neq x$ such that $h_i(x') = h_i(x)$). UOWHFs were introduced by Naor and Yung [36], who show that UOWHFs suffice to construct secure signature schemes and furthermore show how to construct the former from any one-way permutation. Rompel [38] later gave a construction of UOWHFs, and hence signature schemes, based on any one-way function. The Naor-Yung construction requires one invocation of the one-way permutation per bit of compression; that is, if $h_i : \{0,1\}^{\ell+k} \to \{0,1\}^\ell$ (for all $h_i \in \mathcal{H}$), then evaluating h_i requires k invocations of the one-way permutation. This can be improved easily to obtain a construction making $\Theta(k/\log S)$ invocations to compress by k bits.

We show that this, too, is essentially optimal. In particular, any *semi* black-box construction of a UOWHF whose hash functions compress their input by k bits yet can be evaluated using $o(k/\log S)$ invocations of a one-way permutation (with security S) implies the *unconditional* existence of a UOWHF. Since the existence of UOWHFs implies the existence of one-way functions, this consequence would again imply a proof of $P \neq NP$. This improves upon the work of Kim, Simon, and Tetali [33], who show a similar result but only for the case of constructions making $o(\sqrt{k/\log S})$ invocations of a one-way permutation.

Encryption schemes. PRGs and UOWHFs may be viewed as 1-party, or "stand-alone", cryptographic primitives for which there is no inherent notion of "interaction". We also explore the efficiency of 2-party protocols, including those used in a "public-key" setting.

A public-key encryption scheme for m-bit messages is semantically-secure [25] if for any two messages $M_0, M_1 \in \{0,1\}^m$ the distribution over encryptions of M_0 is computationally

indistinguishable from the distribution over encryptions of M_1, even when given the public key as input. A similar definition (but with no public key) holds for the case of private-key encryption. Public-key encryption schemes constructed using the hard-core bit paradigm [8, 7, 22, 42] require $\Theta(m/\log S)$ invocations of a trapdoor permutation to encrypt an m-bit message. Similarly, private-key encryption schemes constructed using this paradigm require $\Theta(\frac{m-k}{\log S})$ invocations of a one-way permutation, where k is the length of the shared key. The same bound holds in the private-key case even if one is willing to use the stronger assumption of a trapdoor permutation.

We show that the above constructions are essentially the best possible (at least, for the notions of security considered above). For the case of public-key encryption, we show that any *weak* black-box construction supporting encryption of m-bit messages requires $\Omega(m/\log S)$ invocations of the trapdoor permutation for the encryption algorithm alone (i.e., in addition to any invocations made during the key-generation phase). Using related techniques, we also show that any *weak* black-box construction of a private-key encryption scheme — even when based on trapdoor permutations — which securely encrypts m-bit messages using a k-bit key must query its permutation oracle $\Omega(\frac{m-k}{\log S})$ times. In each case, we show that any weak black-box construction beating our bound would imply the unconditional existence of a one-way function (from which a secure private-key encryption scheme, with no reliance on an oracle, can be derived), and hence a proof that $P \neq NP$.

Signature schemes. We say a one-time signature scheme is secure if no efficient adversary can forge a valid signature on a new message after seeing a signature on a single, random message (we remark that it is easy to covert any such scheme to one that is secure when an adversary sees a signature on a single, *chosen* message: run two of the "basic" schemes in parallel to obtain keys (PK_1, SK_1), (PK_2, SK_2) and set $PK = (PK_1, PK_2)$; to sign a message $M \in \{0,1\}^m$ choose random $r \in \{0,1\}^m$, sign r using SK_1, sign $r \oplus M$ using SK_2, and output both signatures). Of course, lower bounds on one-time schemes immediately extend to schemes satisfying stronger definitions of security [26]. We show that in any *semi* black-box construction of a one-time signature scheme for messages of length m based on a one-way permutation, the verification algorithm must evaluate the one-way permutation $\Omega(m/\log S)$ times. As before, any semi black-box construction beating our bound implies the unconditional existence of a one-way function (from which a secure signature scheme requiring no oracle access can be constructed).

We observe that there exist one-time signature schemes essentially meeting our lower bound; see Section 4.5 for further discussion.

1.2 Overview of Our Techniques

We prove our results in an extension of the model of Impagliazzo and Rudich [31, 39]. Among other things, Impagliazzo and Rudich prove that a semi black-box construction of a secure key-exchange protocol based on a one-way permutation would inherently yield a proof that $P \neq NP$, and hence it is presumably "hard" to come up with such a construction. (This was later strengthened by Reingold, et al. [37], who proved *unconditionally* that there exists no semi black-box construction of a key-exchange protocol based on one-way permutations.) Our results are in the same vein, but are in many respects even stronger. For one, some of our impossibility results concern the larger class of *weak* black-box constructions. Furthermore,

in all cases we show that constructions beating our bounds would imply the unconditional existence of a one-way function; this is stronger than the results of Impagliazzo-Rudich both because the existence of a one-way function is not known to be implied by $P \neq NP$, and also because (in all but one case) a one-way function suffices to give an *unconditional* construction of the object under consideration. Finally, we stress that Impagliazzo and Rudich were concerned with the question of *feasibility*, while we are concerned with questions of *efficiency*.

Each of our proofs hinges on a technical result that has not been stated or proved previously: a random permutation on t-bit strings is, with high probability, one-way with security $2^{\Omega(t)}$ even against non-uniform adversaries. For the related case of random functions, a similar result has been proven by Impagliazzo and Rudich [31] in the (much simpler) uniform case, and by Impagliazzo [29] in the non-uniform case.[2] We also show a similar result for the case of trapdoor permutations.

Using this result, we now describe the intuition behind our lower bound using the case of PRGs as an example. Given a secure construction G of a PRG having oracle access to a one-way permutation over n-bit strings, we run G with a permutation oracle that randomly permutes its first $t = \Theta(\log S)$ bits while leaving the remaining $n - t$ bits unchanged. It follows from the technical result above that with high probability such a permutation is one-way with security S, and hence G is secure when run with a permutation chosen from this distribution. Let q be the number of queries made by G to its oracle. The key point of our proof is to notice that the answers to these q oracle queries can be "simulated" by a deterministic function G' *itself* (i.e., without access to any oracle) if $q \cdot t$ random bits, representing the t-prefixes of the q answers to G's oracles queries, are included as part of the seed of G'. The distribution on the output of G' (over random choice of seed) is essentially identical to that of the output of G (over random seed, and random choice of oracle as above), and is thus indistinguishable from uniform. Finally, if q is "small" then the seed-length does not grow "too much" and the input of G' remains shorter than its output. But this means that G' is an *unconditional* PRG which does not require any oracle access (a corollary of which is a proof that $P \neq NP$).

Additional technical work is needed to prove our bounds on UOWHFs, public-key encryption schemes, and digital signature schemes. In the latter two cases in particular, which are in a "public-key" setting, there is no "seed" as part of which to include the necessary randomness for answering oracle queries, and thus no immediate way to apply the above technique. The proofs of our lower bounds in these cases follow a slightly different approach. In the case of public-key encryption, for example, we show that a scheme making fewer than the prescribed number of queries can be used to construct (unconditionally) a secure *private-key* encryption scheme in which the key is shorter than the message. Moving from the public-key to the private-key setting circumvents the issues above, and enables the necessary randomness to be included as part of the shared key without compromising the functionality or the security of the scheme. Unfortunately, our result in this case is somewhat weaker than what we obtain in all other cases; namely, we show that a *public*-key encryption scheme making "few" black-box oracle queries exists only if a secure *private*-key encryption scheme (or, equivalently, a one-way function) exists unconditionally. This

[2] Although one could derive our result from Impagliazzo's result and the fact that a random function is indistinguishable from a random permutation, our proof is quite different and a bit simpler.

is, of course, weaker than showing the unconditional existence of a secure public-key encryption scheme. We stress that for the case of PRGs, UOWHFs, private-key encryption schemes, and signature schemes, constructions beating our bounds imply the existence of an unconditional construction for each of these tasks.

1.3 Black-Box Lower Bounds and Impossibility Results

We provide here a brief discussion regarding the notion of "black-box constructions". Our presentation adapts the recent definitional work of Reingold, et al. [37], simplifying their definitions when appropriate for the present context. The discussion here is relatively informal, and we have chosen to provide definitions specific to each primitive in the relevant sub-sections of Section 4 rather than providing a single, generic definition as in [37].

At a high level, a construction P of a primitive based on, say, a one-way permutation is a procedure which takes as "input" a permutation f and outputs a (description of a) circuit[3] having some desired functionality. Informally, a construction P *uses f as a black-box* if P relies only on the input/output characteristics of the provided function f and not on any internal structure of the circuit computing f; formally, this means the construction can be described as an oracle procedure $P^{(\cdot)}$ such that P^f itself realizes the desired functionality.

In addition to correctness (i.e., the requirement that P^f has the desired functionality for any permutation f), a construction also provides some guarantee with regard to the security of the resulting implementation P^f. We say P is a *semi black-box construction* if the following holds:

> For any hard-to-invert permutation f, the implementation P^f is "secure" (in some sense appropriate for the primitive being constructed) against all efficient adversaries, *even those given oracle access to f*.

In contrast, P is a *weak black-box construction* if the following relaxed definition holds:

> For any hard-to-invert permutation f, the implementation P^f is "secure" against all efficient adversaries (who are *not* given oracle access to f).

The distinction between whether an adversary is given oracle access to f or not is important since the above are required to hold *even when f is not efficiently computable* (and so the only way for an efficient adversary to evaluate f, in general, may be via oracle access to f). We hasten to point out, however, that even weak black-box constructions suffice to give implementations with meaningful security guarantees in the real world: in this case, f *will* be efficiently-computable and furthermore an explicit circuit for f will be known; hence, it is irrelevant whether an adversary is given oracle access to f or not.

Clearly, any semi black-box construction is also a weak black-box construction and hence impossibility results for the latter are stronger. Indeed, Reingold, et al. [37] show that (1) a semi black-box construction of key-exchange from one-way functions is unconditionally impossible, yet (2) (informally) if the statement "one-way functions imply key exchange" is true, then there does exist a weak black-box construction of key exchange (for a single-bit key) from one-way functions. Roughly speaking, a difference between semi black-box

[3] For simplicity, the present discussion is phrased in terms of circuits rather than Turing machines although similar definitions could be made in the latter case as well.

and weak black-box constructions is with regard to whether the circuits of *adversaries* attacking P^f can be used in the security *reduction* (i.e., the proof that P is secure). In more detail: typical security proofs for a construction P proceed by showing via *reduction* how any efficient adversary $\mathcal{A}_P$ "breaking" the security of P^f can be used to construct an efficient adversary $\mathcal{A}_f$ inverting f. Since the latter is impossible by assumption on f, this implies that no $\mathcal{A}_P$ with the claimed properties exists. If the reduction relies only on the input/output characteristics of $\mathcal{A}_P$, we refer to this as a reduction which *uses the adversary as a black-box*. In contrast, a reduction which relies on knowledge of the circuit for $\mathcal{A}_P$ is said to *make non-black-box use of the adversary*. A security reduction for a *weak* black-box construction may potentially make non-black-box use of the adversary. As argued in [37], however, security reductions for *semi* black-box constructions are essentially restricted to using the adversary as a black-box (see [37] for further discussion).

Non-black-box constructions. Black-box constructions form an important subclass, since most cryptographic constructions are black-box (indeed, most known constructions are *fully black-box* [37], a notion which is even more restrictive than semi black-box). We stress, however, that a number of non-black-box cryptographic constructions are known.[4] The examples of which we are aware occur in two ways: due to the use of generic zero-knowledge proofs (of knowledge) [23, 12, 4] or due to the use of generic protocols for secure computation [43, 24]. As an illustrative example: let L_f denote the image of a function f; i.e., $L_f \stackrel{\text{def}}{=} \{y | y = f(x)\}$. A cryptographic protocol which utilizes a zero-knowledge proof that $y \in L_f$ (for f a one-way function, say) requires the parties to agree on a circuit computing f and is thus inherently non-black-box.[5] Examples of non-black-box constructions where zero-knowledge proofs of this sort are used include a construction of an identification protocol based on one-way functions [12], a signature scheme based on non-interactive zero-knowledge [5], and *all* known constructions of chosen-ciphertext-secure encryption schemes from trapdoor permutations (e.g., [11]). Furthermore, protocols for distributed computation (without honest majority) tolerating computationally-bounded, malicious adversaries [24, 43] are *themselves* non-black-box[6] (this is in contrast to the case of zero-knowledge proofs; cf. footnote 5).

Knowledge of the circuit computing f is also necessary if one wants to evaluate f in a secure, distributed fashion (e.g., if two parties with respective inputs x_1, x_2 want to evaluate $y = f(x_1 \oplus x_2)$ without revealing any more information about their inputs than what is revealed by y itself). Thus, a generic construction of a threshold cryptosystem [10] based on a family F of trapdoor permutations (in which the parties share the trapdoor for inverting a single member of this family) would inherently make non-black-box use of the underlying circuit(s) for F. Another example is a result of Beaver [3] which makes non-black-box use

[4] We focus here on constructions making non-black-box use of an underlying function f, rather than on constructions whose security analysis makes non-black-box use of the adversary (as in [1, 2]).

[5] Note that constructions of zero-knowledge proofs for NP (e.g., [23]) are *themselves* black-box in their usage of primitives such as one-way functions. The issue is that a proof for the language of interest — e.g., L_f in the example in the text — cannot be given unless a (poly-size) circuit computing f is available.

[6] For example: although the well-known protocol for oblivious transfer secure against *semi-honest* adversaries [19, Sec. 7.3.2] makes black-box use of trapdoor permutations, adapting the protocol (using zero-knowledge proofs) to ensure security against *malicious* adversaries involves *non*-black-box use of the circuit for the trapdoor permutation.

of a one-way function f to extend "few" oblivious transfers into "many".

Given the above, a black-box impossibility result cannot be said to rule out the feasibility of a particular construction. Yet, it is unclear how non-black-box techniques can help outside the domains mentioned above (i.e., generic use of zero-knowledge proofs or secure computation). Furthermore, a black-box impossibility result is useful insofar as it indicates the type of techniques that will be necessary to achieve a desired result, or, conversely, the type of techniques that are ruled out. Finally, it is fair to say that non-black-box constructions are much less efficient than black-box ones (this is certainly the case for all the examples given above, and we are aware of no exceptions), and thus a black-box impossibility result does seem to rule out constructions likely to be *practical*.

1.4 Future Work and Open Problems

This work suggests a number of intriguing research directions. The results given here suggest that assuming only the existence of one-way permutations (or, in some cases, even trapdoor permutations) may be too weak of a computational hypothesis to obtain *efficient* cryptographic constructions. Thus, stronger assumptions may be needed to build practical schemes. It is important to explore the "minimal" such assumptions necessary to achieve greater efficiency, as well as to bound the maximum achievable efficiency even when such stronger assumptions are made. For example: what additional efficiency is possible if *homomorphic* one-way permutations (i.e., permutations over a group G satisfying $f(ab) = f(a)f(b)$ for all $a, b \in G$) are assumed?

In a related vein, it will be interesting to explore more efficient constructions based on specific number-theoretic assumptions. As will be evident from our proof techniques, the efficiency limitations of constructions based on arbitrary (trapdoor) one-way permutations stem from the fact that a one-way permutation may have security S even if it has only $\Theta(\log S)$ "hard-core" bits. (Actually, we use "pathological" functions of this form to prove our lower bounds.) But specific one-way permutations and trapdoor permutations with $\Theta(n)$ hard-core bits are known under suitable number-theoretic assumptions (e.g., [28, 9]). Given such functions, we know how to construct PRGs and semantically-secure private- and public-key encryption schemes with improved efficiency. It remains open, however, whether such functions can also be used to improve the efficiency of digital signature schemes or (say) public-key encryption schemes achieving chosen-ciphertext security.

The present work also leaves some more concrete open questions. First, can bounds on the efficiency of other cryptographic constructions (e.g., commitment schemes) also be given? Additionally, our lower bounds essentially match known upper bounds only for schemes achieving relatively weak notions of security: namely, semantic security for encryption of a *single* message in the case of encryption and one-time security for the case of signatures. What can be said about schemes achieving stronger notions of security? Examples of interest include private-key encryption schemes secure when polynomially-many messages are encrypted (this seems related to the efficiency of pseudorandom *functions*, for which a gap remains between known upper and lower bounds), public-key encryption schemes satisfying various flavors of non-malleability/security against chosen-ciphertext attacks, and signature schemes secure when polynomially-many messages are signed.

Finally, our bound for signatures pertains to the efficiency of signature verification. It

would be nice to have corresponding bounds for the efficiency of key-generation/signing.

2 Definitions and Preliminaries

In this section, we review some notation and definitions for the various standard cryptographic primitives considered in this work. Appropriate definitions of "black-box" constructions are deferred to the relevant sections containing our lower bounds.

In what follows, we consider a number of definitions of "(S, ε)-security" having the form "no circuit of size S can have advantage better than ε", where "advantage" is defined in some appropriate way. In each of the cases considered here, the existence of an (S, ε)-secure scheme for any $\varepsilon < 1$ implies the existence of an (S', ε')-secure scheme for an arbitrarily small $\varepsilon' > 0$. For this reason, in all our lower bounds we content ourselves with showing the existence of an (S, ε)-secure construction for an arbitrary $\varepsilon < 1$.

All our results are stated and proved in the non-uniform setting for simplicity only; we stress that they extend immediately to the uniform setting as well. Indeed, it is for this reason that we claim that constructions beating our efficiency bounds imply the existence of one-way functions and $P \neq NP$. For example: Theorem 4.2 states only that the existence of a certain PRG construction $G^{(\cdot)} : \{0,1\}^{\ell} \to \{0,1\}^{\ell+k}$ based on a permutation $\pi : \{0,1\}^n \to \{0,1\}^n$ implies the unconditional existence of a PRG $G' : \{0,1\}^{\ell'} \to \{0,1\}^{\ell+k}$. Essentially the same proof, however, also shows that a certain construction of a PRG *family* $\{G_i^{(\cdot)} : \{0,1\}^{\ell(i)} \to \{0,1\}^{\ell(i)+k(i)}\}$ based on a one-way permutation family $\{\pi_i : \{0,1\}^{n(i)} \to \{0,1\}^{n(i)}\}$, where $G_i^{(\cdot)}$ can be evaluated by a uniform algorithm in polynomial time (in i), implies the unconditional existence of a PRG family $\{G'_i : \{0,1\}^{\ell'(i)} \to \{0,1\}^{\ell(i)+k(i)}\}$, where again each G'_i can be evaluated by a uniform algorithm in polynomial time.

2.1 One-Way Permutations and Trapdoor Permutations

One-way functions/permutations. We say that a function $\pi : \{0,1\}^n \to \{0,1\}^n$ is (S, ε)-*one way* if for every circuit A of size $\leq S$ we have $\Pr_x[A(f(x)) \in f^{-1}(f(x))] \leq \varepsilon$. When f is given as an oracle, we provide A with access to f and write this as A^f. To reduce the number of parameters, we will call a function S-*hard* if it is $(S, 1/S)$-one way.

Let Π_t denote the set of all permutations over $\{0,1\}^t$. In Section 3.1 we prove:

Theorem 2.1 *For all sufficiently large t, a random $\pi \in \Pi_t$ is $2^{t/5}$-hard with probability at least $1 - 2^{-2^{t/2}}$.*

For $t \leq n$, let $\Pi_{t,n}$ denote the subset of Π_n such that $\pi \in \Pi_{t,n}$ iff $\pi(a, b) = (\hat{\pi}(a), b)$ for some $\hat{\pi} \in \Pi_t$ (that is, π permutes the first t bits of its input, while leaving the remaining $n - t$ bits fixed). An immediate corollary of the above theorem is that if $t = 5 \log S$, then for any $n \geq t$ a random $\pi \in \Pi_{t,n}$ is S-hard with very high probability.

Corollary 2.2 *For all sufficiently large t and any $n \geq t$, a random $\pi \in \Pi_{t,n}$ is $2^{t/5}$-hard with probability greater than $1 - 2^{-2^{t/2}}$.*

(One-way) trapdoor permutations. Our model for *trapdoor* permutations is somewhat more involved. We represent a family of trapdoor permutations as a tri-partite oracle

$\tau = (G, F, F^{-1})$. Informally, G corresponds to the *key generation* oracle which when queried on a string td (intended as a "trapdoor") produces the corresponding public key k. The oracle F is the actual trapdoor permutation, which will be queried on key k and an input x. The oracle F^{-1} allows inversion of F; i.e., if $G(td) = k$ and $F(k, x) = y$, then $F^{-1}(td, y) = x$.

More formally, consider the class $T_n = \{\tau \mid \tau = (G, F, F^{-1})\}$ where:

- $G \in \Pi_n$ is a permutation over $\{0,1\}^n$. (Having G map trapdoors to keys rather than having G map "seeds" to a (trapdoor, key) pair does not affect our results.)
- $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is an oracle such that, for each $k \in \{0,1\}^n$, $F(k, \cdot)$ is a permutation on $\{0,1\}^n$.
- $F^{-1} : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ is an oracle defined as follows: $F^{-1}(td, y)$ returns the unique x such that $G(td) = k$ and $F(k, x) = y$.

A uniformly random $\tau = (G, F, F^{-1}) \in T_n$ is chosen in the natural way: G is chosen at random from Π_n and, for each $k \in \{0,1\}^n$, the permutation $F(k, \cdot)$ is chosen independently at random from Π_n. We say that trapdoor permutation family $\tau = (G, F, F^{-1})$ is (S, ε)-trapdoor one way if for every circuit A of size $\leq S$ we have

$$\mathop{\mathbf{Pr}}_{x,td}[k := G(td) \ : \ A^{\tau}(k, F(k, x)) = x] \leq \varepsilon.$$

We say that τ is *S-trapdoor one way* if it is $(S, 1/S)$-trapdoor one way. When clear from the context, we will also say that τ is *S-hard*. Although technically one must always speak of families of trapdoor permutations, we will often abuse terminology and simply refer to a $\tau \in T_n$ as a trapdoor permutation.

In Section 3.2, we prove the following analogue of Theorem 2.1 for trapdoor permutations:

Theorem 2.3 *For all sufficiently large t, a random $\tau \in T_t$ is $2^{t/5}$-hard with probability greater than $1 - 2^{-2^{t/2}}$.*

For $t \leq n$, we let $T_{t,n}$ be the subset of T_n defined as follows: $\tau = (G, F, F^{-1}) \in T_{t,n}$ iff:

- $G \in \Pi_{t,n}$, and thus $G(td_a, td_b) = (\hat{G}(td_a), td_b)$ for some $\hat{G} \in \Pi_t$.
- $F((k_a, k_b), (x_a, x_b)) = (\hat{F}(k_a, x_a), x_b)$, where $\hat{F}(k_a, \cdot) \in \Pi_t$. Equivalently, $F(k, \cdot) \in \Pi_{t,n}$ and furthermore this permutation is determined by the first t bits of k.
- As before, $F^{-1}(td, y)$ returns the unique x such that $G(td) = k$ and $F(k, x) = y$.

An immediate corollary of Theorem 2.3 is that if $t = 5 \log S$, then for any $n \geq t$ a random $\tau \in T_{t,n}$ is S-hard with very high probability.

Corollary 2.4 *For all sufficiently large t and any $n \geq t$, a random $\tau \in T_{t,n}$ is $2^{t/5}$-hard with probability greater than $1 - 2^{-2^{t/2}}$.*

2.2 Pseudorandom Generators

Two distributions X, Y are (S, ε)-indistinguishable if for every distinguishing circuit Dist of size at most S we have

$$\left| \Pr_{x \in X}[\mathsf{Dist}(x) = 1] - \Pr_{y \in Y}[\mathsf{Dist}(y) = 1] \right| \leq \varepsilon.$$

We also write this as $X \overset{(S,\varepsilon)}{\approx} Y$. We say a function $G : \{0,1\}^\ell \to \{0,1\}^{\ell+k}$ is an (S_g, ε)-secure PRG if $G(U_\ell)$ is (S_g, ε)-indistinguishable from $U_{\ell+k}$, where U_n denotes the uniform distribution over $\{0,1\}^n$.

2.3 Universal One-Way Hash Functions

As discussed in the Introduction, a family of universal one-way hash functions (UOWHFs) is a family $\mathcal{H}$ of length-decreasing functions such that, for a random function $h \in \mathcal{H}$ and a random point x in the domain, it is hard (given h, x) to find $x' \neq x$ such that $h(x') = h(x)$. More formally, a family $\mathcal{H} = \{h_s : \{0,1\}^{\ell+k} \to \{0,1\}^\ell\}_{s \in \{0,1\}^r}$ of functions is an (S, ε)-UOWHF if for every circuit A of size at most S we have

$$\Pr_{s,x}[A(s, x, h_s(x)) = x' : x' \neq x \bigwedge h_s(x') = h_s(x)] \leq \varepsilon.$$

We will represent such a family as a single function $H : \{0,1\}^r \times \{0,1\}^{\ell+k} \to \{0,1\}^\ell$ where $H(s, x) = h_s(x)$.

2.4 Public- and Private-Key Encryption

Public-key encryption. A public-key encryption scheme for m-bit messages is a tuple of algorithms $\mathcal{PKE} = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ having the following functionality:

- The *key generation algorithm* Gen is a probabilistic algorithm which generates a key pair (pk, sk). We say pk is the public key and sk is the secret key.
- The *encryption algorithm* Enc is a probabilistic algorithm which, on input a public key pk and a message $M \in \{0,1\}^m$, outputs a ciphertext C.
- The *decryption algorithm* Dec is a deterministic algorithm which, on input a secret key sk and a ciphertext C, outputs either a message $M \in \{0,1\}^m$ or $\perp$.

We also require perfect correctness; that is, for all (pk, sk) output by Gen, all $M \in \{0,1\}^m$, and all C output by $\mathsf{Enc}(pk, M)$, we have $\mathsf{Dec}(sk, C) = M$. (Our results can be modified appropriately for the case of decryption schemes with error; see the remark following Lemma 4.6.)

For $M \in \{0,1\}^m$, let $\mathcal{PKE}(M)$ denote the distribution on the view of an adversary eavesdropping on the encryption of message M; i.e.,

$$\mathcal{PKE}(M) \overset{\text{def}}{=} \{(pk, sk) \leftarrow \mathsf{Gen}; C \leftarrow \mathsf{Enc}(pk, M) : (pk, C)\}.$$

We say that $\mathcal{PKE}$ is (S_e, ε)-secure if for all $M_0, M_1 \in \{0,1\}^m$ we have

$$\mathcal{PKE}(M_0) \stackrel{(S_e,\varepsilon)}{\approx} \mathcal{PKE}(M_1).$$

Note that this corresponds to a definition of indistinguishability, or semantic security [25].

The above can be extended in the natural way to allow for interactive encryption schemes. However, since we do not explicitly consider interactive public-key encryption in this work and since a formal definition of interactive encryption in the private-key setting is given below, we omit the details.

Private-key encryption. The model for private-key encryption is an appropriate modification of the above. A private-key encryption scheme for m-bit messages using k-bit keys is a pair of algorithms $\mathcal{SKE} = (\mathsf{Enc}, \mathsf{Dec})$, where:

- The encryption algorithm Enc is a probabilistic algorithm which, on input a key $sk \in \{0,1\}^k$ and a message $M \in \{0,1\}^m$, outputs a ciphertext C.
- The decryption algorithm Dec is a deterministic algorithm which, on input a key $sk \in \{0,1\}^k$ and a ciphertext C, outputs either a message $M \in \{0,1\}^m$ or $\perp$.

As in the public-key case, we require perfect correctness: i.e., for all $sk \in \{0,1\}^k$, all $M \in \{0,1\}^m$, and all C output by $\mathsf{Enc}(sk, M)$, we have $\mathsf{Dec}(sk, C) = M$. (Our results can be modified appropriately for the case of decryption schemes with error; see the remark following Lemma 4.6.)

For $M \in \{0,1\}^m$, denote by $\mathcal{SKE}(M)$ the probability distribution over the view of an adversary eavesdropping on the encryption of message M (where the shared key sk is chosen uniformly at random); i.e.,

$$\mathcal{SKE}(M) \stackrel{\text{def}}{=} \left\{ sk \leftarrow \{0,1\}^k; C \leftarrow \mathsf{Enc}(sk, M) : C \right\}.$$

We say that $\mathcal{SKE}$ is (S_e, ε)-secure if for all $M_0, M_1 \in \{0,1\}^m$ we have:

$$\mathcal{SKE}(M_0) \stackrel{(S_e,\varepsilon)}{\approx} \mathcal{SKE}(M_1).$$

The above can be extended to allow for interactive encryption in the natural way. In this case, Enc and Dec represent interactive Turing machines operating in a sequence of rounds. For notational convenience, we let $T \leftarrow \widehat{\mathsf{Enc}}(sk, M)$ denote the experiment in which random coins ω_1, ω_2 are chosen for Enc and Dec, respectively, and T is the transcript resulting from the interaction of $\mathsf{Enc}(sk, M; \omega_1)$ with $\mathsf{Dec}(sk; \omega_2)$. We also let $M' \leftarrow \widehat{\mathsf{Dec}}(sk, M)$ denote the final output of Dec at the conclusion of the above experiment. Perfect correctness requires that $\widehat{\mathsf{Dec}}(sk, M) = M$ with probability 1. In the interactive setting, $\mathcal{SKE}(M)$ denotes the distribution

$$\mathcal{SKE}(M) \stackrel{\text{def}}{=} \left\{ sk \leftarrow \{0,1\}^k; T \leftarrow \widehat{\mathsf{Enc}}(sk, M) : T \right\};$$

definitions for (S_e, ε)-security follow in the obvious way.

2.5 Signature Schemes

A signature scheme for m-bit messages is a tuple of algorithms $\mathcal{SIG} = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ having the following functionality:

- The *key generation algorithm* Gen is a probabilistic algorithm which generates a key pair (PK, SK). We say that PK is the public key and SK is the secret key.
- The *signing algorithm* Sign is a probabilistic algorithm which, on input SK and a message $M \in \{0,1\}^m$, outputs a signature σ.
- The *verification algorithm* Vrfy is a deterministic algorithm which, on input PK, a message M, and a signature σ, outputs a single bit.

We also require that for all (PK, SK) output by Gen, all $M \in \{0,1\}^m$, and all σ output by $\mathsf{Sign}(SK, M)$ we have $\mathsf{Vrfy}(PK, M, \sigma) = 1$.

Our definition of security for signature schemes is extremely weak: we only require security against existential forgery for an adversary who gets a signature on a single, random message (i.e., we consider a *one-time signature scheme* secure under *random-message attack*). Our lower bounds apply even to "weakly-secure" schemes of this type. Since any signature scheme secure against adaptive chosen-message attack (cf. [26]) trivially achieves this "weak" level of security, our results immediately imply a bound for the more general case. Formally, signature scheme $\mathcal{SIG}$ is (S_Σ, ε)-secure if for all circuits A of size at most S_Σ we have

$$\Pr\left[\begin{array}{l}(PK, SK) \leftarrow \mathsf{Gen}; M \leftarrow \{0,1\}^m; \\ \sigma \leftarrow \mathsf{Sign}(SK, M); (M', \sigma') := A(PK, M, \sigma) : \\ \mathsf{Vrfy}(PK, M', \sigma') = 1 \bigwedge M' \neq M\end{array}\right] \leq \varepsilon.$$

3 "Hardness" of Random (Trapdoor) Permutations

In this section, we state and prove two key technical theorems which show that a random permutation $\pi \in \Pi_t$ and a random trapdoor permutation $\tau \in T_t$ are exponentially "hard" with all but negligible probability for t large enough. Before doing so, we first state the following bound on the number of oracle circuits of a given size S:

Lemma 3.1 *The number of circuits of size S having input/output length n and oracle access to a function $f : \{0,1\}^n \to \{0,1\}^n$ is at most $2^{2S+(S+1)n(\log(Sn+n)+1)}$.*

Proof A circuit of the form considered here consists of three types of gates: "and" gates, "or" gates, and "oracle" gates; the first two have in-degree 2 and out-degree 1, while "oracle" gates have in-degree and out-degree n. A circuit may be specified by listing for each gate: (1) its type; and (2) for each of this gate's at most n input wires, the source of the wire (which may be the output wire of another gate or one of the input wires of the circuit) and whether its value is complemented. Finally, for each of the n output wires of the circuit one must also specify the source of this wire and whether its value is complemented. The information associated with each gate can be specified using at most $2 + n(\log(Sn+n)+1)$ bits (per gate), while the information associated with each output wire can be specified using at most an additional $\log(Sn+n)+1$ bits. The lemma follows. ■

3.1 "Hardness" of Random Permutations

We now prove Theorem 2.1, restated for convenience.

Theorem *For all sufficiently large t, a random $\pi \in \Pi_t$ is $2^{t/5}$-hard with probability at least $1 - 2^{-2^{t/2}}$.*

Proof We begin by showing that given any (π, A) such that A inverts π with "high" probability, the permutation π has a "short" description (given A).

Claim *Let A be a circuit that makes q queries to a permutation $\pi : \{0,1\}^t \to \{0,1\}^t$, and for which $\Pr_y[A^\pi(y) = \pi^{-1}(y)] \geq \varepsilon$. Then π can be described using at most*

$$2\log\binom{2^t}{a} + \log\left((2^t - a)!\right)$$

bits (given A), where $a = \varepsilon 2^t/(q+1)$.

Proof (of claim) Let $N = 2^t$. Consider the set I of at least εN points on which A is able to invert π, after making q queries to π. We argue that there exists a set $Y \subseteq I$ such that $|Y| \geq a$ and such that the value of π^{-1} is completely determined by the circuit A, the sets Y and $X \stackrel{\text{def}}{=} \pi^{-1}(Y)$, and the value of π^{-1} on all points in $\{0,1\}^t \setminus Y$.

Define Y via the following process: initially Y is empty, and all elements in I are candidates for inclusion in Y. Take the lexicographically first element y from I, and place it in Y. Next, simulate the computation of $A^\pi(y)$ and let $x_1, \ldots, x_q$ be the queries made by A to π (we assume without loss of generality that they are different), and $y_1, \ldots, y_q$ be the corresponding answers (i.e., $y_i = \pi(x_i)$). If $y \notin \{y_i\}_{i=1}^q$, then remove $y_1, \ldots, y_q$ from I. If $y = y_i$ for some i, then remove $y_1, \ldots, y_{i-1}$ from I. Then take the lexicographically smallest of the remaining elements of I, put it into Y, etc. At any step of the construction, one element is added to Y and at most q are removed from I. Since I initially contains at least εN elements, in the end we have $|Y| \geq \lceil \varepsilon N/q \rceil > \varepsilon N/(q+1)$.

We claim that given descriptions of the sets Y and $X = \pi^{-1}(Y)$, the values of π on $\{0,1\}^t \setminus X$, and the circuit A, it is possible to invert (or, equivalently, compute) π everywhere. For $y \notin Y$, the value of $\pi^{-1}(y)$ is explicitly given. The values of π^{-1} on Y can be reconstructed sequentially for all $y \in Y$, taken in lexicographic order, as follows: Simulate the computation of $A^\pi(y)$. By construction of Y, during its computation $A^\pi(y)$ will query π either on points not in X, on points $x \in X$ for which $\pi(x) <_{lex} y$, or on the point $x \in X$ for which $\pi(x) = y$. In the first two cases, we have enough information to continue the simulation. In the last case, the query itself gives the desired answer $\pi^{-1}(y)$. In all possible cases, we have enough information to reconstruct $\pi^{-1}(y)$.

Describing Y, X, and the values of π on $\{0,1\}^t \setminus X$ requires $2\log\binom{N}{|Y|} + \log\left((N - |Y|)!\right)$ bits, which is at most the number of bits claimed. □

Given the above claim, we may now easily prove the theorem. Let A be an oracle circuit of size at most $S = 2^{t/5}$. Note that A will make at most $q = 2^{t/5}$ queries to π. Let $N = 2^t$. From the claim, we see that the fraction of permutations $\pi \in \Pi_t$ such that

$$\Pr_x[A^\pi(\pi(x)) = x] \geq 2^{-t/5} \tag{1}$$

is at most

$$\frac{\binom{N}{a}^2 (N-a)!}{N!} = \frac{\binom{N}{a}}{a!},$$

where $a = 2^{-t/5}2^t/(2^{t/5}+1) \geq N^{3/5}/2$. Using the inequalities $a! \geq (a/e)^a$ and $\binom{N}{a} \leq (eN/a)^a$, we may derive the following upper bound on the above expression:

$$\frac{\binom{N}{a}}{a!} \leq \left(\frac{e^2 N}{a^2}\right)^a < \left(\frac{4e^2}{N^{1/5}}\right)^a < 2^{-a} < 2^{-N^{3/5}/2}$$

for all sufficiently large N.

By Lemma 3.1 there are at most $2^{2S+(S+1)t(\log(St+t)+1)} = 2^{\widetilde{O}(N^{1/5})}$ circuits of size S (where the $\widetilde{O}$-notation suppresses polylogarithmic factors). A union bound thus shows that the probability over a random choice of $\pi \in \Pi_t$ that there *exists* a circuit of size S for which Equation (1) holds is at most

$$2^{\widetilde{O}(N^{1/5})} \cdot 2^{-N^{3/5}/2} < 2^{-N^{1/2}}$$

for all sufficiently large N. ■

3.2 "Hardness" of Random Trapdoor Permutations

We now prove an analogue of the above theorem for trapdoor permutations. (This is Theorem 2.3, restated for convenience.)

Theorem *For all sufficiently large t, a random $\tau \in T_t$ is $2^{t/5}$-hard with probability greater than $1 - 2^{-2^{t/2}}$.*

Proof The proof is substantially similar to the proof of Theorem 2.1. We first prove:

Claim *Let A be a circuit making q queries to a trapdoor permutation $\tau \in T_t$ and for which $\Pr_{k,y}[A^\tau(k,y) = x \wedge F(k,x) = y] \geq \varepsilon$. Then τ can be described using at most*

$$1 + 2^t \log(2^t!) + t + 2\log\binom{2^t}{a} + \log\left((2^t - a)!\right)$$

bits (given A), where $a = \varepsilon 2^t/(2q+1)$.

Proof (of claim) Let $N = 2^t$, and let $Q(k,y)$ denote the event that $A^\tau(k,y)$ queries either $G(td)$ or $F^{-1}(td, y')$ where y' is arbitrary and $G(td) = k$. Also, we say "A^τ inverts (k,y)" if $A^\tau(k,y) = x$ such that $F(k,x) = y$. There are two possibilities: either

$$\Pr_{k,y}[A^\tau \text{ inverts } (k,y) \wedge Q(k,y)] \geq \varepsilon/2 \quad \text{or} \quad \Pr_{k,y}[A^\tau \text{ inverts } (k,y) \wedge \overline{Q(k,y)}] \geq \varepsilon/2. \tag{2}$$

Consider the first case. Here, there certainly exists a $\hat{y}$ for which it is the case that $\Pr_k[A^\tau \text{ inverts } (k,\hat{y}) \wedge Q(k,\hat{y})] \geq \varepsilon/2$. Proceeding as in the proof of Theorem 2.1, we will specify G using a "small" number of bits. Let I be the set of at least $\varepsilon N/2$ points $(k,\hat{y})$ on which A^τ inverts $(k,\hat{y})$ and $Q(k,\hat{y})$ occurs. Define a set $K \subseteq I$ via the following process: initially K is empty, and all elements in I are candidates for inclusion in K. Take

the lexicographically first element $(k, \hat{y})$ from I, and place it in K. Next, simulate the computation of $A^\tau(k, \hat{y})$. Since $Q(k, \hat{y})$ occurs, we know that there is a query i of the form $G(td)$ or $F^{-1}(td, y')$, where $G(td) = k$. Looking at each of the preceding $i - 1$ queries, for each query of the form $G(td')$ with answer k' remove $(k', \hat{y})$ from I. For each query of the form $F^{-1}(td', y')$ let $G(td') = k'$ and remove $(k', \hat{y})$ from I. Then take the lexicographically smallest of the remaining elements of I, put it into K, etc. At any step of the construction, one element is added to K and at most q are removed from I. Since I initially contains at least $\varepsilon N/2$ elements, in the end we have $|K| \geq \lceil \varepsilon N/2q \rceil > \varepsilon N/(2q+1)$.

Exactly as in the proof of Theorem 2.1 (and so we omit the details), the permutation G is completely specified given A, $\hat{y}$, descriptions of K and $G^{-1}(K)$, the values of G^{-1} on $\{0,1\}^t \setminus K$, and the values of F on all points. This requires $2\log\binom{N}{|K|} + \log((N - |K|)!)$ bits plus an additional $2^t \log(2^t!)$ bits to specify F. Using an additional bit to specify that we are in the first case, we can completely specify τ in at most the number of bits claimed.

Consider next the second case of Equation (2). Here, there must exist a $\hat{k}$ for which $\Pr_y[A^\tau \text{ inverts } (\hat{k}, y) \wedge \overline{Q(\hat{k}, y)}] \geq \varepsilon/2$. Now, we specify $F(\hat{k}, \cdot)$ using a "small" number of bits. Let I be the set of at least $\varepsilon N/2$ points $(\hat{k}, y)$ on which A^τ inverts $(\hat{k}, y)$ and $Q(\hat{k}, y)$ does not occur. Define a set $Y \subseteq I$ via the following process: initially Y is empty, and all elements in I are candidates for inclusion in Y. Take the lexicographically first element $(\hat{k}, y)$ from I, and place it in Y. Next, simulate the computation of $A^\tau(\hat{k}, y)$. Consider the $\ell \leq q$ queries made by A of the form $\{F(\hat{k}, x_i)\}$ with corresponding answers $\{y_i\}$. If $y \notin \{y_i\}_{i=1}^{\ell}$ then remove $\{(\hat{k}, y_i)\}_{i=1}^{\ell}$ from Y. If $y = y_j$ for some j, then remove $\{(\hat{k}, y_i)\}_{i=1}^{j-1}$ from Y. Then take the lexicographically smallest of the remaining elements of I, put it into Y, etc. At any step of the construction, one element is added to Y and at most q are removed from I. Since I initially contains at least $\varepsilon N/2$ elements, in the end we have $|Y| \geq \lceil \varepsilon N/2q \rceil > \varepsilon N/(2q+1)$.

Following the proof of Theorem 2.1, the permutation $F(\hat{k}, \cdot)$ is completely specified given A, descriptions of Y and the set X such that $F(\hat{k}, X) = Y$, the value of $F(\hat{k}, \cdot)$ on $\{0,1\}^t \setminus X$, the values of G at all points, and the values of $F(k, \cdot)$ at all points for all $k \neq \hat{k}$. (We omit the details, but remark that we crucially use the fact that in the computation of $A^\tau(\hat{k}, y)$ with $y \in Y$ we are guaranteed that $Q(\hat{k}, y)$ does not occur and, in particular, A does not make a query of the form $F^{-1}(td, y')$ with $G(td) = \hat{k}$.) This requires $2\log\binom{N}{|Y|} + \log((N - |Y|)!)$ bits plus $\log(2^t!)$ bits to specify G and $(2^t - 1)\log(2^t!)$ bits to specify $F(k, \cdot)$ for $k \neq \hat{k}$. Using an additional t bits to specify $\hat{k}$, as well as a bit to specify that we are in the second case, we have that τ is specified in at most the number of bits claimed. □

Proceeding as in Theorem 2.1, let A be an oracle circuit of size at most $S = 2^{t/5}$ and note that A makes at most $q = 2^{t/5}$ queries to τ. Let $N = 2^t$. From the claim, we see that the fraction of $\tau \in T_t$ such that $\Pr_{k,y}[A^\tau \text{ inverts } (k, y)] \geq 2^{-t/5}$ is at most

$$\frac{2^{t+1}\binom{N}{a}}{a!} \leq 2^{-N^{3/5}/4}$$

for sufficiently large N, where $a = 2^{-t/5}2^t/(2^{t/5+1} + 1)$ and using the fact that $a \geq N^{3/5}/4$.

Applying Lemma 3.1 (and taking into account that A here has input length $2t$, output length t, and access to three oracles some of which are functions from $\{0,1\}^{2t}$ to $\{0,1\}^t$), there are at most $2^{\widetilde{O}(N^{1/5})}$ circuits of size S. A union bound thus shows that the probability

over a random choice of $\pi \in \Pi_t$ that there *exists* a circuit of size S for which Equation (1) holds is at most

$$2^{\widetilde{O}(N^{1/5})} \cdot 2^{-N^{3/5}/4} < 2^{-N^{1/2}}$$

for all sufficiently large N. ■

4 Lower Bounds

4.1 Pseudorandom Generators

In this section we show our lower bound for PRG constructions, defined formally as follows.

Definition 4.1 *A PRG construction from a one-way permutation is an oracle procedure $G^{(\cdot)} : \{0,1\}^\ell \to \{0,1\}^{\ell+k}$ that expects as an oracle a permutation $\pi \in \Pi_n$. We refer to k as the* stretch *of G.*

We say $G^{(\cdot)}$ is an (S_p, S_g, ε)-OWP-to-PRG weak black-box construction if for every permutation π that is S_p-hard, G^π is an (S_g, ε)-secure PRG.

For any (S_p, S_g, ε)-OWP-to-PRG weak black-box construction with stretch k, we prove that unless G queries π on at least $\Omega(k/\log S_p)$ points, it is possible to derive an *unconditional* construction of a pseudorandom generator. Before giving the proof, we provide some intuition.

The basic idea of the proof is as follows: let $t = \Theta(\log S_p)$. First, note that if G uses as an oracle $\pi \in \Pi_{t,n}$ chosen uniformly at random, then G is an (S_g, ε)-secure PRG with all but negligible probability (since a random permutation from $\Pi_{t,n}$ is S_p-hard with all but negligible probability). Now, if G queries the oracle at only a few (say, q) points, we can encode the answers to these queries in the seed of the PRG itself. Furthermore, this encoding is "short" since only t bits are needed to answer a query to $\pi \in \Pi_{t,n}$. We thus obtain a PRG $G' : \{0,1\}^{\ell'} \to \{0,1\}^{\ell+k}$ which uses no oracle at all, but which is able to "simulate" the computation of G when using a random permutation oracle. The desired bound comes from the fact that G' still stretches its input provided that ℓ' is smaller than $\ell + k$. But $\ell' = \ell + qt$ since qt bounds the number of bits needed to encode the t-bit answers to G's q queries.

Theorem 4.2 *Let $G^{(\cdot)} : \{0,1\}^\ell \to \{0,1\}^{\ell+k}$ be an (S_p, S_g, ε)-OWP-to-PRG weak black-box construction that makes q queries to an oracle $\pi \in \Pi_n$. If $q < k/(5 \log S_p)$ then there exists an $(S_g, \varepsilon + 2^{-S_p^2} + q^2/S_p^5)$-secure PRG $G' : \{0,1\}^{\ell'} \to \{0,1\}^{\ell+k}$ with $\ell' < \ell + k$.*

Proof Since $G^{(\cdot)}$ is an (S_p, S_g, ε)-OWP-to-PRG construction, this means that if $\pi : \{0,1\}^n \to \{0,1\}^n$ is S_p-hard then for any distinguisher T of size at most S_g we have

$$\left| \Pr_{x \in U_{\ell+k}} [T(x) = 1] - \Pr_{s \in U_\ell} [T(G^\pi(s)) = 1] \right| \leq \varepsilon.$$

Let $t = 5 \log S_p$. From Corollary 2.2 we know that a random permutation $\pi \in \Pi_{t,n}$ is S_p-hard with probability greater than $1 - 2^{-2^{t/2}}$. An averaging argument thus shows that

for any circuit T of size at most S_g we have

$$\left| \Pr_{x \in U_{\ell+k}} [T(x) = 1] - \Pr_{\substack{\pi \in \Pi_{t,n} \\ s \in U_\ell}} [T(G^\pi(s)) = 1] \right| < \varepsilon + 2^{-2^{t/2}}. \tag{3}$$

Recall that any $\pi \in \Pi_{t,n}$ operates only on its first t input bits; i.e., any such π satisfies $\pi(a,b) = (\hat{\pi}(a), b)$ where $\hat{\pi}$ is a permutation over $\{0,1\}^t$. Without loss of generality, we now assume that G always queries π with strings having distinct t-prefixes. Indeed, for any G asking arbitrary queries, one can construct a $\widehat{G}$ with essentially the same running time, such that if G asks (a,b) with a equal to the t-prefix of a previous query, $\widehat{G}$ simulates the answer without querying π using the previously-obtained value of $\hat{\pi}(a)$. In general the behavior of $\widehat{G}$ is different than that of G, but when we restrict to $\pi \in \Pi_{t,n}$ they are equivalent.

By assumption, G queries π at most $q < k/t$ times. We now construct G' which takes as input a seed s' of length $\ell' \stackrel{\text{def}}{=} \ell + qt < \ell + k$. Let s denote the first ℓ bits of s', and let $y_1, \ldots, y_q \in \{0,1\}^t$ denote the remaining qt bits. We then define

$$G'(s') = G'(s, y_1, \ldots, y_q) \stackrel{\text{def}}{=} G^{y_1,\ldots,y_q}(s),$$

where the notation $G^{y_1,\ldots,y_q}(s)$ denotes the computation of $G^{(\cdot)}(s)$ when its i^{th} oracle query $x_i = (a_i, b_i)$ (with $|a_i| = t$) is answered with (y_i, b). (Here we use the fact that the t-prefixes of G's queries are distinct.)

G' stretches its input by at least one bit, and requires no oracle access. Furthermore,

$$\left| \Pr_{\substack{\pi \in \Pi_{t,n} \\ s \in U_\ell}} [T(G^\pi(s)) = 1] - \Pr_{s' \in U_{\ell'}} [T(G'(s')) = 1] \right| \leq 2 \cdot \Pr_{\vec{y} \in \{0,1\}^{qt}} [\mathsf{Coll}],$$

where we let $\vec{y} = y_1, \ldots, y_q$ and Coll denotes the event that these q values are not distinct. An easy "birthday problem" calculation shows that $\Pr_{\vec{y} \in \{0,1\}^{qt}}[\mathsf{Coll}] \leq q^2/2^{t+1}$, which together with Equation (3) implies the statement of the theorem. ■

4.2 Universal One-Way Hash Functions

In this section we prove lower bounds for constructions of universal one-way hash functions based on one-way permutations. The formal definition of such constructions follows.

Definition 4.3 *A construction of a UOWHF from a one-way permutation is an oracle procedure $H^{(\cdot)}(\cdot,\cdot)$ that expects as an oracle a permutation $\pi \in \Pi_n$ and is given inputs $s \in \{0,1\}^r$ and $x \in \{0,1\}^{\ell+k}$. The output is $H^\pi(s,x) \in \{0,1\}^\ell$. We refer to k as the* compression *of H.*

We say H is an (S_p, S_h, ε)-OWP-to-UOWHF semi black-box construction if for every π that is S_p-hard, H^π is an (S_h, ε)-UOWHF even for circuits given oracle access to π.

We show that if there exists an (S_p, S_h, ε)-OWP-to-UOWHF semi black-box construction with compression k making $q < k/(5 \log S_p)$ queries to its oracle π, then it is possible to derive an unconditionally-secure construction of a UOWHF (i.e., without any access to π).

As in the case of PRGs, we first observe that H^π is secure when π is chosen uniformly at random from $\Pi_{t,n}$, for $t = 5 \log S$. We then show that the computation of H^π for random $\pi \in \Pi_{t,n}$ can be simulated by an H' (without any oracle access) by including as part of the key for H' the t-prefixes of the answers for the q queries H makes to π. Furthermore, we include in the output of H' the t-prefixes of the q queries themselves. The crux of the proof is to show that H' is a UOWHF. As some intuition toward this, note that since the t-prefixes of the queries (resp., answers) are included with the output (resp., key), an adversary finding a collision in H' is bound to these particular values. Hence, any collision in H' is also a collision in H^π. Since $\ell + qt < \ell + k$ (and hence H' is length-decreasing) whenever $q < k/(5 \log S_p)$, this yields the desired bound.

Theorem 4.4 *Let $H^{(\cdot)} : \{0,1\}^r \times \{0,1\}^{\ell+k} \to \{0,1\}^\ell$ be an (S_p, S_h, ε)-OWP-to-UOWHF semi black-box construction that makes q queries to an oracle $\pi \in \Pi_n$. If $q < k/(5 \log S_p)$ then there exists an $(S_h - S_H, \varepsilon + 2^{-S_p^2} + q^2/2S_p^5)$-secure UOWHF $H' : \{0,1\}^{r'} \times \{0,1\}^{\ell+k} \to \{0,1\}^{\ell'}$ with $\ell' < \ell + k$, where S_H is the size of the circuit computing H.*

Proof Since $H^{(\cdot)}$ is an (S_p, S_h, ε)-OWP-to-UOWHF construction, this means that if $\pi \in \Pi_n$ is S_p-hard then any circuit A of size $\leq S_h$ finds a collision with probability at most ε; that is

$$\Pr_{\substack{s \in U_r \\ z \in U_{\ell+k}}} \left[A^\pi(s, z, H^\pi(s,z)) = z' : z' \neq z \bigwedge H^\pi(s,z') = H^\pi(s,z) \right] \leq \varepsilon.$$

We say that z' as above is a *collision* exactly when $z' \neq z$ but $H^\pi(s,z') = H^\pi(s,z)$.

We now restrict the class of adversaries A under consideration: we consider adversaries that do not access π arbitrarily, but are instead simply given the t-prefixes of the queries and answers made during the computation of $H^\pi(s,z)$. Since such restricted adversaries can be simulated by general adversaries with overhead S_H (recall that this is the size of the circuit computing H), we have that if π is S_p-hard and A is a circuit of size $\leq S_h - S_H$ then

$$\Pr_{\substack{s \in U_r \\ z \in U_{\ell+k}}} \left[A(s, y_1, \ldots, y_q, z, H^\pi(s,z), x_1, \ldots, x_q) = z' : z' \text{ is a collision} \right] \leq \varepsilon,$$

where $x_1, \ldots, x_q$ are the t-prefixes of the q queries made to π during computation of $H^\pi(s,z)$, and $y_1, \ldots, y_q$ are the t-prefixes of the corresponding answers.

Let $t = 5 \log S_p$. From Corollary 2.2 we know that a random permutation $\pi \in \Pi_{t,n}$ is S_p-hard with probability greater than $1 - 2^{-2^{t/2}}$. An averaging argument thus shows that for any circuit A of size $\leq S_h - S_H$ we have

$$\Pr_{\substack{\pi \in \Pi_{t,n} \\ s \in U_r, z \in U_{\ell+k}}} \left[A(s, y_1, \ldots, y_q, z, H^\pi(s,z), x_1, \ldots, x_q) = z' : z' \text{ is a collision} \right] < \varepsilon + 2^{-2^{t/2}}. \quad (4)$$

As in the proof of Theorem 4.2, we may assume without loss of generality that H queries π on points with distinct t-prefixes. Consider the function $H' : \{0,1\}^{r'} \times \{0,1\}^{\ell+k} \to \{0,1\}^{\ell'}$ defined as follows, where $r' = r + qt$ and $\ell' = \ell + qt$:

$$H'(s', z) = H'\left((s, y_1, \ldots, y_q), z\right) \stackrel{\text{def}}{=} H^{y_1, \ldots, y_q}(s, z), x_1, \ldots, x_q,$$

where by $H^{y_1,\ldots,y_q}(s,z)$ we mean the computation of $H^{(\cdot)}(s,z)$ when its i^{th} oracle query (a_i,b_i) (with $|a_i|=t$) is answered with (y_i,b_i), and where $x_i,\ldots,x_q$ denote the t-prefixes of the q oracle queries made (i.e., $x_i=a_i$). Note that if $q<k/t$ then $\ell'<\ell+k$, so $H'(s',\cdot)$ is a length-decreasing function.

By definition of H', if $H'((s,y_1,\ldots,y_q),z) = H'((s,y_1,\ldots,y_q),z')$ then $H^\pi(s,z) = H^\pi(s,z')$ for any π satisfying $\pi(x_1)=y_1,\ldots,\pi(x_q)=y_q$. Thus,

$$\begin{aligned}
&\Pr_{\substack{s'\in U_{r'}\\ z\in U_{\ell+k}}}\left[A(s',z,H'(s',z))=z' : z'\neq z \bigwedge H'(s',z')=H'(s',z)\right]\\
&\quad\leq \Pr_{\substack{\pi\in\Pi_{t,n}\\ s\in U_r,\, z\in U_{\ell+k}}}\left[A(s,y_1,\ldots,y_q,z,H^\pi(s,z),x_1,\ldots,x_q)=z' : z' \text{ is a collision}\right]\\
&\qquad + \Pr_{\vec{y}\in\{0,1\}^{qt}}[\mathsf{Coll}],
\end{aligned}$$

where Coll denotes the event that the q values $y_1,\ldots,y_q$ are not distinct.[7] Equation (4) and a simple "birthday problem" calculation give the result stated in the theorem. ■

4.3 Public-Key Encryption

Definition 4.5 *A construction of a public-key encryption scheme based on trapdoor permutations is a tuple of oracle procedures $\mathcal{PKE}^{(\cdot)} = (\mathsf{Gen}^{(\cdot)},\mathsf{Enc}^{(\cdot)},\mathsf{Dec}^{(\cdot)})$ such that, for all $\tau\in T_n$, the resulting $\mathcal{PKE}^\tau$ satisfies the functional definition of a public-key encryption scheme (the construction may be for either an interactive or a non-interactive encryption scheme, as it does not affect our results).*

We say $\mathcal{PKE}^{(\cdot)}$ is an (S_p,S_e,ε)-TDP-to-PKE weak black-box construction if for every oracle $\tau\in T_n$ that is S_p-trapdoor one way, $\mathcal{PKE}^\tau$ is (S_e,ε)-secure.

We prove that for any such construction which encrypts messages of length m, unless Enc^τ queries τ at least $\Omega(m/\log S_p)$ times there exists a one-way function which does not require any oracle access. Our proof proceeds by showing that unless Enc^τ makes at least $\Omega(m/\log S_p)$ queries to τ, we can explicitly construct an interactive, *private*-key encryption scheme $(\mathsf{Enc}',\mathsf{Dec}')$ requiring no access to the oracle and in which the encrypted message is longer than the shared key. Using a previous result of Impagliazzo and Luby [30] (see also Lemma 4.6), this implies the existence of a one-way function.

As in the previous proofs, we first observe that a random $\tau\in T_{t,n}$ is S_p-hard with all but negligible probability when $t=5\log S_p$ (cf. Corollary 2.4). To construct an interactive, private-key encryption scheme without access to an oracle, we have the parties "simulate" a random τ by appropriately choosing random t-prefixes for the answers to their queries, as needed. The bits to simulate τ cannot be included in the private key, since the encryption and decryption algorithms may make their queries in different order and, indeed, may make different queries altogether. However, we must somehow ensure consistency between the

[7] In fact — in contrast to the seemingly-similar situation arising in the proof of Theorem 4.2 — there is no need to conserve random bits here since the values $y_1,\ldots,y_q$ are included as part of the *key* and not the *output* (and the length of the key is irrelevant for our purposes). A tighter security reduction is possible by lowering $\Pr[\mathsf{Coll}]$; for example, by including random values $y_1,\ldots,y_{2q}$ in the key and using the first q distinct values (when they exist) to answer the queries of H.

oracle answers of the sender and receiver. A possibility that comes to mind is to have each party include, along with each protocol message it sends to the other party, a list of t-prefixes for the queries and answers generated thus far in accessing τ. In this case, however, privacy is no longer guaranteed as the queries may reveal information about the plaintext message. But this is easily be remedied in the private-key setting: the parties simply share a sufficiently-long one-time pad in advance and then "encrypt" their queries and answers using this pad.

Let q_g be the number of queries made to τ by Gen, and let q_e be the number of queries made by Enc. The private-key encryption scheme outlined above requires a shared key of size roughly $O(t) \cdot (q_g + q_e)$ to encrypt an m-bit message. Recalling the result of Impagliazzo and Luby [30], if $O(t) \cdot (q_g + q_e) < m$ then the key is shorter than the message and a one-way function exists. This already gives a "weak" lower bound. To obtain the better lower bound $q_e < m/O(t)$ (so that we bound the efficiency of encryption alone), additional work is needed; details are given in the proof of Theorem 4.7.

We begin by showing that the existence of a private-key encryption scheme $(\mathsf{Enc}, \mathsf{Dec})$ which securely encrypts messages longer than the shared key implies the existence of a one-way function. Although this result is already known [30] (without the concrete bounds given below), we give a much simpler and more direct proof. We stress that the result applies even in the case of interactive encryption.

Lemma 4.6 *Let $(\mathsf{Enc}, \mathsf{Dec})$ be an (S, ε)-secure private-key encryption scheme for messages of length m using a shared key of length $k < m$. Let S_{Enc} be the size of the circuit needed to run the encryption protocol (i.e., the size of the circuit for Enc in the non-interactive case, or the combined sizes of the circuits for Enc and Dec in the interactive setting). Then for any $\ell \in \mathbb{N}$ there exists a function f which is $(S - \ell S_{\mathsf{Enc}}, \ell\varepsilon + 2^{-\ell(m-k)})$-one way.*

Proof We prove the lemma for the case of interactive encryption, which implies the same result for the degenerate, non-interactive case as well. First note that, via standard hybrid argument, running ℓ parallel copies of $(\mathsf{Enc}, \mathsf{Dec})$ using ℓ independent keys yields an $(S, \ell\varepsilon)$-secure private-key encryption scheme for messages of length ℓm in which the shared key has length ℓk. Let $\mathcal{SKE}_\ell = (\mathsf{Enc}_\ell, \mathsf{Dec}_\ell)$ denote this modified scheme.

Define f by $f(sk, M, \omega_1, \omega_2) = \widehat{\mathsf{Enc}}_\ell(sk, M; \omega_1, \omega_1) \| M$, where ω_1, ω_2 represent the random coins used by Enc_ℓ and Dec_ℓ, respectively. We claim that this function is (S', ε')-one way, where $S' = S - \ell S_{\mathsf{Enc}}$ and $\varepsilon' = \ell\varepsilon + 2^{-\ell(m-k)}$. If not, then there is an algorithm B of size at most S' for which $\mathsf{Succ}_{B,f} > \varepsilon'$, where

$$\mathsf{Succ}_{B,f} \stackrel{\text{def}}{=} \Pr\left[sk \leftarrow \{0,1\}^{\ell k}; M \leftarrow \{0,1\}^{\ell m}; T \leftarrow \widehat{\mathsf{Enc}}_\ell(sk, M) : B(T\|M) \in f^{-1}(T\|M)\right].$$

We show how such a B can be used to construct an algorithm A of size at most S for which $\mathsf{Succ}_{A,\mathcal{SKE}_\ell} > \ell\varepsilon$, where

$$\mathsf{Succ}_{A,\mathcal{SKE}_\ell} \stackrel{\text{def}}{=} \left| \Pr_{\substack{M_0,M_1 \in \{0,1\}^{\ell m} \\ T \in \mathcal{SKE}_\ell(M_0)}}[A(M_0, M_1, T) = 1] - \Pr_{\substack{M_0,M_1 \in \{0,1\}^{\ell m} \\ T \in \mathcal{SKE}_\ell(M_1)}}[A(M_0, M_1, T) = 1] \right|.$$

This implies that there exist two messages M_0, M_1 for which A can distinguish encryptions of M_0 from encryptions of M_1 with probability better than $\ell\varepsilon$, contradicting the assumed security of $(\mathsf{Enc}_\ell, \mathsf{Dec}_\ell)$. Thus, we are done once we have demonstrated such an A.

Define A as follows: on input (M_0, M_1, T), algorithm A runs $B(T\|M_0)$ to obtain the result $sk'\|M'\|\omega_1'\|\omega_2'$. It then checks whether $f(sk', M', \omega_1', \omega_2') \stackrel{?}{=} T\|M_0$. If so (i.e., B has succeeded in inverting f), then A outputs 0. Otherwise, A outputs 1. Note that $|A| = |B| + \ell S_{\mathsf{Enc}} \leq S$, as required.

First, note that

$$\Pr_{\substack{M_0,M_1\in\{0,1\}^{\ell m}\\ T\in\mathcal{SKE}_\ell(M_0)}}[A(M_0,M_1,T)=0] = \mathsf{Succ}_{B,f} > \varepsilon'.$$

For a transcript T, we say (sk, M) is *consistent with* T if there exist ω_1, ω_2 such that $T = \widehat{\mathsf{Enc}}_\ell(sk, M; \omega_1, \omega_2)$. We have:

$$\begin{aligned}
&\Pr_{\substack{M_0,M_1\in\{0,1\}^{\ell m}\\ T\in\mathcal{SKE}_\ell(M_1)}}[A(M_0,M_1,T)=1] \\
&\quad\leq \Pr[sk\leftarrow\{0,1\}^{\ell k}; M_0,M_1\leftarrow\{0,1\}^{\ell m}; \\
&\qquad T\leftarrow\widehat{\mathsf{Enc}}_\ell(sk,M_1) : \exists sk' \text{ s.t. } (sk', M_0) \text{ is consistent with } T] \\
&\quad\leq \sum_{sk'\in\{0,1\}^{\ell k}} \Pr[sk\leftarrow\{0,1\}^{\ell k}; M_0,M_1\leftarrow\{0,1\}^{\ell m}; \\
&\qquad\qquad T\leftarrow\widehat{\mathsf{Enc}}_\ell(sk,M_1) : (sk', M_0) \text{ is consistent with } T].
\end{aligned}$$

Perfect correctness of the encryption scheme implies that for any sk, T there is at most one value of $M \in \{0,1\}^{\ell m}$ such that (sk, M) is consistent with T. Using this and the fact that M_0 is chosen at random independent of anything else gives:

$$\begin{aligned}
\Pr_{\substack{M_0,M_1\in\{0,1\}^{\ell m}\\ T\in\mathcal{SKE}_\ell(M_1)}}[A(M_0,M_1,T)=1] &\leq \sum_{sk'\in\{0,1\}^{\ell k}} 2^{-\ell m} \\
&= 2^{\ell(k-m)}.
\end{aligned}$$

Putting everything together shows that $\mathsf{Succ}_{A,\mathcal{SKE}_\ell} > \varepsilon' - 2^{\ell(k-m)} \geq \ell\varepsilon$, giving the desired contradiction. ■

We remark that an analog of the above lemma is known to hold also for the case of (interactive) private-key encryption schemes *with error* [30]. This, in turn, implies results analogous to those of Theorems 4.7 and 4.9 for black-box constructions of encryption schemes with error. However, as we were unable to simplify the proof of [30] in this setting (and as the concrete bounds on the resulting one-way function are rather unwieldy) we do not explicitly focus on the case of encryption schemes with error here.

Our main result of this section follows. The theorem is stated for the case of non-interactive public-key encryption, but the proof immediately extends to the case of *interactive* public-key encryption as well (where q_e will in this case refer to the total number of queries made by sender and receiver during the encryption protocol, and S_{Enc} will refer to the sizes of Enc and Dec jointly).

Theorem 4.7 *Let $\mathcal{PKE}^{(\cdot)} = (\mathsf{Gen}^{(\cdot)}, \mathsf{Enc}^{(\cdot)}, \mathsf{Dec}^{(\cdot)})$ be an (S_p, S_e, ε)-TDP-to-PKE weak black-box construction for messages of length m, and let $t = 5 \log S_p$. Assume Gen makes q_g queries to an oracle $\tau \in T_n$ and Enc makes q_e queries to τ; set $\ell = 2 \cdot \lceil 5tq_g / (m - 5tq_e) \rceil$. Assume further that $\varepsilon < (1/4 - 2^{-S_p^2})/\ell$. If $q_e < m/5t$, then there exists an $(S_e - 3\ell S_{\mathsf{Enc}}, 3/4)$-one-way function (without access to any oracle), where S_{Enc} is the size of the circuit for Enc.*

Proof Note that we did not try to optimize the constants in the proof or the required bound on ε. In applications of cryptographic interest, S_p and S_e are typically super-polynomial, S_{Enc}, q_g, q_e, and m are (small) polynomials, and ε is negligible; thus, $\varepsilon \ll (1/4 - 2^{-S_p^2})/\ell$ and $S_e \gg 3\ell S_{\mathsf{Enc}}$ anyway.

Let $\mathcal{PKE}^{(\cdot)} = (\mathsf{Gen}^{(\cdot)}, \mathsf{Enc}^{(\cdot)}, \mathsf{Dec}^{(\cdot)})$. As in the proof of Lemma 4.6, for any $\ell \in \mathbb{N}$ we may construct a public-key encryption scheme $\mathcal{PKE}_\ell^{(\cdot)} = (\mathsf{Gen}_\ell^{(\cdot)}, \mathsf{Enc}_\ell^{(\cdot)}, \mathsf{Dec}_\ell^{(\cdot)})$ for ℓm-bit messages in the natural way; furthermore, we may set $\mathsf{Gen}_\ell = \mathsf{Gen}$ since we are now in the public-key setting and so key generation need only be done once. It is easy to see (via hybrid argument) that $\mathcal{PKE}_\ell^{(\cdot)}$ is an $(S_p, S_e - \ell S_{\mathsf{Enc}}, \ell\varepsilon)$-TDP-to-PKE construction, where the number of queries made by Gen_ℓ is q_g and the number of queries made by Enc_ℓ is at most ℓq_e.

Set $\ell = 2 \cdot \lceil 5t\, q_g/(m - 5t\, q_e) \rceil$ as in the statement of the theorem, and let $S' = S_e - \ell S_{\mathsf{Enc}}$ and $\varepsilon' = \ell\varepsilon + 2^{-S_p^2}$. We use $\mathcal{PKE}_\ell^{(\cdot)}$ to construct an (S', ε')-secure interactive, private-key encryption scheme $\mathcal{SKE} = (\mathsf{Enc}', \mathsf{Dec}')$ for ℓm-bit messages in which the shared key will have length $5t \cdot (q_g + \ell q_e)$. Furthermore, $\mathcal{SKE}$ will require no access to the trapdoor permutation oracle. Finally, we have $5t \cdot (q_g + \ell q_e) < \ell m$ (in fact, $\ell m - 5t \cdot (q_g + \ell q_e) \geq 1$) and $\varepsilon' < 1/4$; thus, application of Lemma 4.6 (with $\ell = 1$ there) yields the desired result.

Security of $\mathcal{PKE}_\ell^{(\cdot)}$ implies that if τ is S_p-hard then, for any circuit B of size $\leq S'$ and for any messages $M_0, M_1 \in \{0,1\}^{\ell m}$ we have

$$\left| \mathop{\mathbf{Pr}}_{v \in \mathcal{PKE}_\ell^\tau(M_0)} [B(v) = 1] - \mathop{\mathbf{Pr}}_{v \in \mathcal{PKE}_\ell^\tau(M_1)} [B(v) = 1] \right| \leq \ell\varepsilon.$$

Corollary 2.4 shows that a random $\tau \in T_{t,n}$ is S_p-hard except with probability less than $2^{-S_p^2}$. A straightforward averaging argument thus shows that for any circuit B of size $\leq S'$ and for any messages $M_0, M_1 \in \{0,1\}^{\ell m}$ we have:

$$\left| \mathop{\mathbf{Pr}}_{\substack{\tau \in T_{t,n} \\ v \in \mathcal{PKE}_\ell^\tau(M_0)}} [B(v) = 1] - \mathop{\mathbf{Pr}}_{\substack{\tau \in T_{t,n} \\ v \in \mathcal{PKE}_\ell^\tau(M_1)}} [B(v) = 1] \right| < \ell\varepsilon + 2^{-S_p^2} = \varepsilon'. \tag{5}$$

As mentioned in the discussion at the beginning of this section, our private-key encryption scheme $\mathcal{SKE}$ will "simulate" a random $\tau \in T_{t,n}$ for algorithms $\mathsf{Gen}, \mathsf{Enc}_\ell$, and Dec_ℓ. We achieve this simulation using a "simulation procedure" $\mathcal{SIM}$ which ensures consistency of the answers to all oracle queries. This procedure takes as input a list L of (appropriate prefixes of) previous oracle queries and answers; before answering any query, $\mathcal{SIM}$ examines L and ensures that the answer it gives will not generate any inconsistencies. After answering a query, L is updated accordingly. As an example, if query $td \| b$ to G was answered by $k \| b$

(where $|td| = |k| = t$), then subsequent query $td'\|b'$ to G must be answered by $k'\|b'$ where $k' = k$ iff $td' = td$. A more involved procedure is needed to answer queries to F and F^{-1}. We now describe the details of this simulation.

$\mathcal{SIM}(L)$:

- On query $G(td\|b)$ (where $|td| = t$): if $\exists k$ s.t. $(td, k) \in L$, return $k\|b$. Otherwise pick random $k \in \{0,1\}^t$ such that $\forall td' : (td', k) \notin L$, return $k\|b$, and store (td, k) in L.
- On query $F(k\|b, x\|b')$ (where $|k| = |x| = t$):
  1. if $\exists y$ s.t. $(k, x, y) \in L$, return $y\|b'$.
  2. Otherwise, if $\exists td$ s.t. $(td, k) \in L$, choose random $y \in \{0,1\}^t$ such that $\forall x' : (k, x', y) \notin L$, return $y\|b'$, and store (k, x, y) in L.
  3. Otherwise, choose random $td \in \{0,1\}^t$ such that $\forall k' : (td, k') \notin L$, choose random $y \in \{0,1\}^t$, return $y\|b'$, and store (k, td) and (k, x, y) in L.
- On query $F^{-1}(td\|b, y\|b')$ (where $|tk| = |y| = t$):
  1. if $\exists k, x$ s.t. $(td, k), (k, x, y) \in L$, return $x\|b'$
  2. Otherwise, if $\exists k$ s.t. $(td, k) \in L$, choose random $x \in \{0,1\}^t$ such that $\forall y' : (k, x, y') \notin L$, return $x\|b'$, and store (k, x, y) in L
  3. Otherwise, choose random $k \in \{0,1\}^t$ such that $\forall td' : (td', k) \notin L$, choose random $x \in \{0,1\}^t$, return $x\|b'$, and store (td, k) and (k, x, y) in L

Note that each time a query is answered, at most $5t$ bits are stored in L.

Construct $\mathcal{SKE}$ as follows. Parse the shared key s as (s_1, s_2) where $|s_1| = 5tq_g$ and $|s_2| = 5t\ell q_e$. To encrypt message M, the receiver Dec' begins by initializing list $L := \emptyset$. The receiver then computes $(pk, sk) \leftarrow \mathsf{Gen}^{\mathcal{SIM}(L)}$ (updating L in the process) and sends $pk, s_1 \oplus L$ to the sender. The receiver stores sk, L for later use. Upon receiving the first message $pk, \hat{s}_1$, the sender computes $L_1 := s_1 \oplus \hat{s}_1$ and sets $L := L_1$. The sender then computes $C \leftarrow \mathsf{Enc}_\ell^{\mathcal{SIM}(L)}(pk, M)$ and sets $L_2 := L \backslash L_1$. Finally, Enc' sends $C, s_2 \oplus L_2$ to the receiver. Upon receiving message $C, \hat{s}_2$, the receiver decrypts by setting $L_2 := s_2 \oplus \hat{s}_2$ and $L := L_0 \cup L_2$ (here, L_0 is the list stored by the receiver from the first stage). The receiver can then compute $M := \mathsf{Dec}_\ell^{\mathcal{SIM}(L)}(sk, C)$.

It is clear that $\mathcal{SKE}$ has correct decryption. We now show that the scheme is (S', ε')-secure. Assume toward a contradiction that there exists a circuit A of size $\leq S'$ and messages $M_0, M_1 \in \{0,1\}^{\ell m}$ such that

$$\left| \Pr_{v \in \mathcal{SKE}(M_0)}[A(v) = 1] - \Pr_{v \in \mathcal{SKE}(M_1)}[A(v) = 1] \right| > \varepsilon'.$$

We construct circuit B attacking $\mathcal{PKE}_\ell$ as follows. On input pk, C, circuit B picks random strings $\hat{s}_1, \hat{s}_2$ where $|\hat{s}_1| = 5tq_g$ and $|\hat{s}_2| = 5t\ell q_e$. Then, B outputs $A(pk, \hat{s}_1, C, \hat{s}_2)$. Because the keys s_1, s_2 of $\mathcal{SKE}$ are used as a one-time pad, it is easy to see that, for $b \in \{0,1\}$:

$$\Pr_{\substack{\tau \in T_{t,n} \\ (pk,C) \in \mathcal{PKE}_\ell^\tau(M_b)}}[B(pk, C) = 1] = \Pr_{v \in \mathcal{SKE}(M_b)}[A(v) = 1].$$

Thus, the advantage of B is equal to the advantage of A (which is greater than ε'), contradicting Equation (5). ■

4.4 Private-Key Encryption

The techniques of the previous section can be adapted to show a similar lower bound for private-key encryption schemes based on trapdoor permutations; note that trapdoor permutations generalize one-way permutations (which are sufficient for private-key encryption), and therefore our result shows that improved efficiency cannot be obtained in this case even by assuming a stronger primitive.

Definition 4.8 *A construction of a private-key encryption scheme based on trapdoor permutations is a pair of oracle procedures $\mathcal{SKE}^{(\cdot)} = (\mathsf{Enc}^{(\cdot)}, \mathsf{Dec}^{(\cdot)})$ such that, for all $\tau \in T_n$, the resulting $\mathcal{SKE}^{\tau}$ satisfies the functional definition of a private-key encryption scheme given earlier (again, the construction may yield either an interactive or a non-interactive scheme).*

We say $\mathcal{SKE}^{(\cdot)}$ is an (S_p, S_e, ε)-TDP-to-SKE weak black-box construction if for every $\tau \in T_n$ that is S_p-hard, $\mathcal{SKE}^{\tau}$ is (S_e, ε)-secure.

Suppose a construction of this type exists which encrypts m-bit messages using a shared key of length k. We show that unless Enc^{τ} queries τ at least $q = \Omega(\frac{m-k}{\log S_p})$ times, then an unconditional one-way function exists. This matches the known upper bound, even for schemes constructed using one-way permutations.

The proof is similar to that of Theorem 4.7, in that we convert $\mathcal{SKE}^{(\cdot)}$ to a private-key encryption scheme $\mathcal{SKE}'$ that does not access an oracle at all. The only difference between the proof here and the proof of Theorem 4.7 is that here the parties need to share a k-bit key in addition to the one-time pad used to encrypt their "simulated" queries and answers to the oracle. Set $t = 5 \log S_p$. The resulting $\mathcal{SKE}'$ requires a shared key of length $k + 5tq$ and encrypts an m-bit message. As before, then, if $k + 5tq < m$ we obtain a private-key encryption scheme (making no oracle queries) in which the message is longer than the key. By Lemma 4.6, this implies the existence of a one-way function.

The following theorem is stated for the case of non-interactive private-key encryption, but the proof immediately extends to the case of interactive private-key encryption as well.

Theorem 4.9 *Let $\mathcal{SKE}^{(\cdot)} = (\mathsf{Enc}^{(\cdot)}, \mathsf{Dec}^{(\cdot)})$ be an (S_p, S_e, ε)-TDP-to-SKE weak black-box construction for messages of length m using a key of length k in which Enc makes q queries to an oracle $\tau \in T_n$. Let $t = 5 \log S_p$. If $q < \frac{m-k}{5t}$ then there exists an $(S_e, \varepsilon + 2^{-S_p^2})$-secure private-key encryption scheme in which the message is longer than the key, without access to any oracle.*

Proof The proof is substantially similar to the proof of Theorem 4.7, so the discussion here will be somewhat terse. Let $t = 5 \log S_p$. As in the previous proof, security of the given construction and a straightforward averaging argument imply that for any circuit B of size $\leq S_e$ and for any messages $M_0, M_1 \in \{0,1\}^m$ we have:

$$\left| \Pr_{\substack{\tau \in T_{t,n} \\ v \in \mathcal{SKE}^{\tau}(M_0)}} [B(v) = 1] - \Pr_{\substack{\tau \in T_{t,n} \\ v \in \mathcal{SKE}^{\tau}(M_1)}} [B(v) = 1] \right| < \varepsilon + 2^{-S_p^2}.$$

We now construct an $(S_e, \varepsilon + 2^{-S_p^2})$-secure private-key encryption scheme $\mathcal{SKE}' = (\mathsf{Enc}', \mathsf{Dec}')$ for m-bit messages in which the shared key has length $k' \stackrel{\text{def}}{=} k + 5t \cdot q$. Furthermore, $\mathcal{SKE}'$ requires no oracle access. If $q < (m-k)/5t$, then $k' < m$ and we obtain the desired result.

The approach for constructing $\mathcal{SKE}'$ is exactly as in Theorem 4.7 (and as discussed earlier in this section), and in particular we use the same simulation procedure $\mathcal{SIM}$ as there. $\mathcal{SKE}'$ is constructed as follows: The shared key s of length k' is parsed as (s_1, s_2) where $|s_1| = k$ and $|s_2| = 5t \cdot q$. To encrypt message M, the sender initializes $L := \emptyset$, computes $C \leftarrow \mathsf{Enc}^{\mathcal{SIM}(L)}(s_1, M)$ (updating L in the process), and then sends $C, s_2 \oplus L$ to the receiver. The receiver obtains ciphertext $C, \hat{L}$, recovers $L = \hat{L} \oplus s_2$, and then computes $M = \mathsf{Dec}^{\mathcal{SIM}(L)}(s_1, C)$.

It is not hard to see that $\mathcal{SKE}'$ has correct decryption. Arguing exactly as in Theorem 4.7 we see that $\mathcal{SKE}'$ is $(S_e, \varepsilon + 2^{-S_p^2})$-secure, completing the proof of the theorem. ■

4.5 Signature Schemes

We now demonstrate a lower bound on the efficiency of signature verification for any signature scheme based on one-way permutations.

Definition 4.10 *A construction of a digital signature scheme for m-bit messages (based on one-way permutations) is a tuple of procedures $\mathcal{SIG}^{(\cdot)} = (\mathsf{Gen}^{(\cdot)}, \mathsf{Sign}^{(\cdot)}, \mathsf{Vrfy}^{(\cdot)})$ such that, for all $\pi \in \Pi_n$, the resulting $\mathcal{SIG}^{\pi}$ satisfies the functional definition of a signature scheme given earlier. We say $\mathcal{SIG}^{(\cdot)}$ is an $(S_p, S_\Sigma, \varepsilon)$-OWP-to-signature semi black-box construction if for every oracle $\pi \in \Pi_n$ that is S_p-hard, $\mathcal{SIG}^{\pi}$ is (S_Σ, ε)-secure, where this must hold even for circuits given access to π.*

Given a construction of this sort, we prove that unless Vrfy queries π at least $\Omega(m/\log S_p)$ times, then it is possible to construct from $\mathcal{SIG}$ a one-way function which does not access any oracle. Note that this one-way function could then be used to construct an secure signature scheme (which requires no oracle access) [38].

We start with an informal overview of our proof technique. As a first attempt to construct a one-way function from the verification algorithm, one might define

$$F_1(PK, M, \sigma) = PK \| \mathsf{Vrfy}^{(\cdot)}(PK, M, \sigma).$$

Intuitively, this function is difficult to invert on elements of the form $PK\|1$ if PK is a valid and randomly-generated public key, since inverting the function on points of this form is equivalent to signature forgery. As presently defined, however, evaluating F_1 requires calls to π; however, our goal is to construct a function which does not require access to any oracle. As in the previous section, though, one may observe that π is S_p-hard (and thus $\mathcal{SIG}$ is secure) when π is uniformly chosen from $\Pi_{t,n}$ for $t = 5 \log S_p$ (cf. Corollary 2.2). So, if Vrfy makes q queries to π, then specifying qt bits as the answers to these queries removes any need to query the oracle. Based on this, one might consider the function

$$F_2(PK, y_1, \ldots, y_q, M, \sigma) = PK \| y_1 \| \cdots \| y_q \| x_1 \| \cdots \| x_q \| \mathsf{Vrfy}^{y_1, \ldots, y_q}(PK, M, \sigma), \tag{6}$$

where $|y_1| = \cdots = |y_q| = |x_1| = \cdots = |x_q| = t$ and the i^{th} query $x_i \| b_i$ of Vrfy is answered with $y_i \| b_i$ (we assume without loss of generality that Vrfy queries π with strings having distinct t-prefixes). The intuition, as before, is that F_2 is difficult to invert on elements of the form $PK \| \vec{y} \| \vec{x} \| 1$ if $PK, \vec{y}$, and $\vec{x}$ are chosen appropriately.

Now, however, another problem arises. In order for F_2 to qualify as a one-way function, it must be hard to invert $F_2(PK, \vec{y}, M, \sigma)$ when $PK, \vec{y}, M, \sigma$ are sampled from an *efficiently sampleable* distribution (cf. Lemma 4.12, below). More specifically, a proof of one-wayness will need to show how to efficiently sample $PK, \vec{y}, M, \sigma$ such that inverting the value $F_2(PK, \vec{y}, M, \sigma)$ results in a signature forgery and hence a contradiction. A necessary condition for inversion to result in signature forgery is that $\mathsf{Vrfy}^{\vec{y}}(PK, M, \sigma) = 1$. Generating $PK, \vec{y}, M, \sigma$ such that this holds is easy if we have the secret key SK; but then inverting F_2 and forging a signature does not yield the desired contradiction!

Instead, in the proof we will obtain M, σ from the signer. In this case, however, inverting $F_2(PK, \vec{y}, M, \sigma)$ does not result in a forgery if it results in the same message/signature pair M, σ. We come now to the crux of our proof. If the number of queries is "small", we show that inversion of F_2 yields a different message M' (and thus a successful forgery) with noticeable probability. More precisely, for randomly-generated $PK, \vec{y}, \vec{x}, \sigma$, let

$$Y = PK \| \vec{y} \| \vec{x} \| 1 = F_2(PK, \vec{y}, M, \sigma).$$

If $|\vec{x}| = \sum_{i=1}^{q} |x_i| < |M|$, then (on average) there exists an element $PK \| \vec{y} \| M' \| \sigma' \in F_2^{-1}(Y)$ with $M' \neq M$; hence inverting Y results in a forgery with noticeable probability. This idea is formalized in the proof of the following theorem.

Theorem 4.11 *Let $\mathcal{SIG}^{(\cdot)}$ be an $(S_p, S_\Sigma, \varepsilon)$-OWP-to-signature semi black-box construction for messages of length m in which Vrfy makes q queries to an oracle $\pi \in \Pi_n$, algorithms Gen, Sign, and Vrfy jointly make $\hat{q}$ queries to π, and $\varepsilon < 1/4 - 2^{-S_p^2}$. If $q < m/(5 \log S_p)$ then there exists an $(S_\Sigma - S_{\mathsf{Gen}} - S_{\mathsf{Sign}} - 2S_{\mathsf{Vrfy}}, 3/4 + \hat{q}^2/2S_p^5)$-one-way function (without access to a permutation oracle), where S_{Gen}, S_{Sign}, and S_{Vrfy} are the sizes of the circuits for Gen, Sign, and Vrfy, respectively.*

Proof As in Theorem 4.7, we did not try to optimize the constants in the proof or the required bound on ε. In applications of cryptographic interest, one will anyway have $S_\Sigma \gg S_{\mathsf{Gen}} + S_{\mathsf{Sign}} + 2S_{\mathsf{Vrfy}}$ and $\varepsilon \ll 1/4 - 2^{-S_p^2}$.

We first present a technical lemma showing that the existence of a function which is one-way over an efficiently sampleable domain implies the existence of a function which is one-way under the definition of Section 2.1.

Lemma 4.12 *Let $\mathcal{D}$ be a distribution sampleable by a circuit of size $S_\mathcal{D}$ and let f be a function such that for every circuit A of size $\leq S$ we have:*

$$\Pr[x \leftarrow \mathcal{D} : A(f(x)) \in f^{-1}(f(x))] \leq \delta.$$

Then there exists a function $\hat{f}$ that is $(S - S_\mathcal{D}, \delta)$-one way.

Proof (of lemma) We equate the distribution $\mathcal{D}$ with the circuit of size $S_\mathcal{D}$ which samples it; i.e., $\{\mathcal{D}(r)\} \equiv \mathcal{D}$ (where r is a string of the appropriate length chosen uniformly at

random). Define $\hat{f}(r) \stackrel{\text{def}}{=} f(\mathcal{D}(r))$. We claim that $\hat{f}$ is $(S - S_{\mathcal{D}}, \delta)$-one way. Assume the contrary. Then there exists a circuit $\hat{A}$ of size at most $S - S_{\mathcal{D}}$ for which

$$\Pr_r[\hat{A}(\hat{f}(r)) \in \hat{f}^{-1}(\hat{f}(r))] > \delta.$$

Toward a contradiction, define a circuit A as follows: $A(y) \stackrel{\text{def}}{=} \mathcal{D}(\hat{A}(y))$. Notice that $|A| \leq S$. Furthermore,

$$\begin{aligned}
&\Pr[x \leftarrow \mathcal{D} : A(f(x)) \in f^{-1}(f(x))] \\
&\quad = \Pr_r[x := \mathcal{D}(r) : f(A(f(x))) = f(x)] \\
&\quad = \Pr_r[f(A(\hat{f}(r))) = \hat{f}(r)] \\
&\quad = \Pr_r[\hat{f}(\hat{A}(\hat{f}(r))) = \hat{f}(r)] \\
&\quad = \Pr_r[\hat{A}(\hat{f}(r)) \in \hat{f}^{-1}(\hat{f}(r))] \ > \ \delta.
\end{aligned}$$

□

The proof of the theorem proceeds by using $\mathcal{SIG}$ to construct a function F along with a distribution $\mathcal{D}$ such that for every circuit A of size $\leq S_\Sigma - S_{\mathsf{Vrfy}}$ we have:

$$\begin{aligned}
\Pr[X \leftarrow \mathcal{D} : A(F(X)) \in F^{-1}(F(X))] &\leq \varepsilon + 2^{-S_p} + 1/2 + \hat{q}^2/2S_p^5 \\
&< 3/4 + \hat{q}^2/2S_p^5. \qquad (7)
\end{aligned}$$

Furthermore, $\mathcal{D}$ will be computable by a circuit of size $S_{\mathsf{Gen}} + S_{\mathsf{Sign}} + S_{\mathsf{Vrfy}}$. Applying Lemma 4.12 then yields the desired result.

Since $\mathcal{SIG}^{(\cdot)}$ is an $(S_p, S_\Sigma, \varepsilon)$-OWP-to-signature construction, if π is S_p-hard then, for any circuit B of size $\leq S_\Sigma$ we have $\mathsf{Succ}_{\pi,B} \leq \varepsilon$ where

$$\begin{aligned}
&\mathsf{Succ}_{\pi,B} \stackrel{\text{def}}{=} \\
&\quad \Pr\left[(PK, SK) \leftarrow \mathsf{Gen}^\pi; M \leftarrow \{0,1\}^m; \ \sigma \leftarrow \mathsf{Sign}^\pi(SK, M); (M', \sigma') := B^\pi(PK, M, \sigma) : \right. \\
&\qquad \left. \mathsf{Vrfy}^\pi(PK, M', \sigma') = 1 \wedge M' \neq M\right].
\end{aligned}$$

Let $t = 5 \log S_p$. Corollary 2.2 shows that a random $\pi \in \Pi_{t,n}$ is S_p-hard except with probability less than $2^{-S_p^2}$. An averaging argument then implies that for any circuit B of size $\leq S_\Sigma$ we have $\mathsf{Succ}^*_B < \varepsilon + 2^{-S_p^2}$ where Succ^*_B is defined analogously to $\mathsf{Succ}_{\pi,B}$ except that the probability is now taken over random choice of $\pi \in \Pi_{t,n}$ as well.

Define a function F as in Equation (6), repeated here for convenience:

$$F(PK, \vec{y}, M, \sigma) = PK \| \vec{y} \| \vec{x} \| \mathsf{Vrfy}^{\vec{y}}(PK, M, \sigma),$$

where $\vec{y} = (y_1, \ldots, y_q)$, $\vec{x} = (x_1, \ldots, x_q)$, $|y_1| = \cdots = |y_q| = |x_1| = \cdots = |x_q| = t$, and the i^{th} query $x_i \| b_i$ of Vrfy is answered with $y_i \| b_i$. (As in the proof of Theorem 4.2, we assume Vrfy queries its oracle on points having distinct t-prefixes.) We also define distribution $\mathcal{D}$ by the following experiment which depends on uniformly distributed coins r_g, r_s (of some appropriate length), $r_y \in \{0,1\}^{\hat{q}t}$ (parsed as a sequence of t-bit strings $\hat{y}_1, \ldots, \hat{y}_{\hat{q}} \in \{0,1\}^t$), and $M \in \{0,1\}^m$:

$$\left\{ \begin{array}{c} (PK, SK) := \mathsf{Gen}(r_g); \\ \sigma := \mathsf{Sign}(SK, M; r_s); \mathsf{Vrfy}(PK, M, \sigma) \end{array} \quad : \quad PK \| \vec{y} \| M \| \sigma \right\}.$$

In the above experiment, the coins $r_y = \hat{y}_1, \ldots, \hat{y}_{\hat{q}}$ are used to give a simulation[8] of a random $\pi \in \Pi_{t,n}$ which is consistent across the executions of Gen, Sign, and Vrfy. The component y_i of $\vec{y}$ is the t-prefix of the answer given in response to the i^{th} query of Vrfy. Note that $\mathcal{D}$ is computable by a circuit of size essentially $S_{\mathsf{Gen}} + S_{\mathsf{Sign}} + S_{\mathsf{Vrfy}}$.

We claim that F satisfies the requirement expressed in Equation (7). Assume toward a contradiction that there exists a circuit A of size $\leq S_\Sigma - S_{\mathsf{Vrfy}}$ for which Equation (7) does not hold. We use A to construct an algorithm B which — given PK, a random message M, and a signature σ on M — forges a valid signature on a new message M' with "high" probability. $B^\pi(PK, M, \sigma)$ first runs $\mathsf{Vrfy}(PK, M, \sigma)$, answering the queries of Vrfy by forwarding them to π. Let $\vec{x}$ be the t-prefixes of the queries made by Vrfy, and let $\vec{y}$ be the t-prefixes of the corresponding answers. Define $X = PK \| \vec{y} \| M \| \sigma$ and $Y = PK \| \vec{y} \| \vec{x} \| 1$; note that $Y = F(X)$. Finally, algorithm B computes $PK' \| \vec{y}' \| M' \| \sigma' = A(Y)$ and outputs (M', σ'). We clearly have $|B| \leq S_\Sigma$.

B outputs a successful forgery if both $F(PK' \| \vec{y}' \| M' \| \sigma') = Y$ and $M' \neq M$ hold. To see this, note that the first condition implies $\vec{y} = \vec{y}'$, and hence $\mathsf{Vrfy}^{\vec{y}}(PK, M', \sigma') = 1$ and furthermore Vrfy makes queries with t-prefixes $\vec{x}$. Thus, $\mathsf{Vrfy}^\pi(PK, M', \sigma') = 1$. If furthermore $M' \neq M$, then (M', σ') is a successful forgery. Finally, the distribution on X — over random choice of $\pi \in \Pi_{t,n}$ — is statistically close to distribution $\mathcal{D}$, where the difference is due to the fact that the $\hat{y}_1, \ldots, \hat{y}_{\hat{q}}$ used in the experiment defining $\mathcal{D}$ may not be distinct. This accounts for the term $\hat{q}^2/2S_p^2$ in the analysis below (obtained using a simple "birthday problem" calculation), but see footnote 7.

Let Eq be the event that $M' = M$. Then

$$\begin{aligned}
\mathsf{Succ}^*_B &\geq \Pr_{X \leftarrow \mathcal{D}}[Y = F(X); X' = A(Y) : X' \in F^{-1}(Y) \wedge \overline{\mathsf{Eq}}] - \hat{q}^2/2S_p^5 \\
&= \Pr_{X \leftarrow \mathcal{D}}[Y = F(X); X' = A(Y) : X' \in F^{-1}(Y)] \\
&\quad - \Pr_{X \leftarrow \mathcal{D}}[Y = F(X); X' = A(Y) : X' \in F^{-1}(Y) \wedge \mathsf{Eq}] - \hat{q}^2/2S_p^5 \\
&> \varepsilon + 2^{-S_p} + 1/2 \\
&\quad - \Pr[X \leftarrow \mathcal{D};\ PK \| \vec{y} \| \vec{x} \| 1 = F(X); \\
&\qquad PK' \| \vec{y}' \| M' \| \sigma' = A(PK \| \vec{y} \| \vec{x} \| 1) : M' = M] \\
&= \varepsilon + 2^{-S_p} + 1/2 \\
&\quad - \sum_{\vec{z}} \Pr[X \leftarrow \mathcal{D};\ PK \| \vec{y} \| \vec{x} \| 1 = F(X); \\
&\qquad PK' \| \vec{y}' \| M' \| \sigma' = A(PK \| \vec{y} \| \vec{x} \| 1) : M' = M \wedge \vec{x} = \vec{z}],
\end{aligned}$$

where the sum is over $\vec{z}$ consisting of q distinct t-bit strings. Substituting $\vec{z}$ for $\vec{x}$ in part of the final equation above gives

$$\begin{aligned}
\mathsf{Succ}^*_B &\geq \varepsilon + 2^{-S_p} + 1/2 \\
&\quad - \sum_{\vec{z}} \Pr[X \leftarrow \mathcal{D};\ PK \| \vec{y} \| \vec{x} \| 1 = F(X);
\end{aligned}$$

[8] Simulating a random $\pi \in \Pi_{t,n}$ is done as expected: the oracle query $x_j \| b$ with the j^{th} distinct t-prefix *across the executions of* Gen, Sign, *and* Vrfy is answered with $\hat{y}_j \| b$, and an oracle query $x \| b$ with a previously-used t-prefix is answered in a consistent manner in the obvious way.

$$
\begin{aligned}
& PK' \| \vec{y}' \| M' \| \sigma' = A(PK \| \vec{y} \| \vec{z} \| 1) : M' = M \wedge \vec{x} = \vec{z}] \\
\geq \quad & \varepsilon + 2^{-S_p} + 1/2 \\
& - \sum_{\vec{z}} \Pr[X \leftarrow \mathcal{D};\ PK \| \vec{y} \| \vec{x} \| 1 = F(X); \\
& \qquad PK' \| \vec{y}' \| M' \| \sigma' = A(PK \| \vec{y} \| \vec{z} \| 1) : M' = M].
\end{aligned}
$$

In this last equation we may note that A has no information about M, which is chosen at random from $\{0,1\}^m$ independently of PK, $\vec{y}$, and $\vec{z}$. Therefore:

$$
\mathsf{Succ}_B^* \quad \geq \quad \varepsilon + 2^{-S_p} + 1/2 - \sum_{\vec{z}} 2^{-m}. \tag{8}
$$

Noting that there are $\binom{2^t}{q} < 2^{qt} \leq 2^{m-1}$ terms in the sum of Equation (8), we derive the contradiction $\mathsf{Succ}_B^* > \varepsilon + 2^{-S_p}$. ■

Upper bounds on the efficiency of signature schemes. As mentioned in the Introduction, our lower bounds focus on the efficiency of signature verification. We briefly observe some upper bounds on the efficiency of verification for one-time signatures (satisfying the notion of security considered here) on m-bit messages. The Lamport scheme [34] requires m invocations of a one-way permutation to verify a signature. Instead of signing bit-by-bit, the scheme can be modified to sign block-by-block. When basing the construction on an S-hard one-way permutation, it is possible to obtain provable security using blocks of length $\Theta(\log(S/m))$; this gives a signature scheme requiring only $\Theta(m/\log(S/m))$ invocations for verification. When S is polynomial, this is essentially optimal as far as verification is concerned (although the key-generation time and public-key size are prohibitive); however, the resulting scheme does not even run in polynomial time when S is super-polynomial. An alternate approach is to include a universal one-way hash function h_s as part of the public key, and to use the (basic) Lamport scheme to sign $h_s(M)$. Verification now requires evaluation of h_s followed by a verification in the underlying Lamport scheme. Since h_s can be used to compress an arbitrary-length message to an n-bit string (when using an S-hard permutation on n bits) [36], we obtain a verification complexity of $\Theta(n + (m-n)/\log S)$ when $m \geq n$.

References

[1] B. Barak. How to Go Beyond the Black-Box Simulation Barrier. *42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 106–115, 2001.

[2] B. Barak. Constant-Round Coin-Tossing with a Man in the Middle, or Realizing the Shared Random String Model. *43rd IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 345–355, 2002.

[3] D. Beaver. Correlated Pseudorandomness and the Complexity of Private Computations. *28th ACM Symposium on Theory of Computing (STOC)*, ACM, pp. 479–488, 1996.

[4] M. Bellare and O. Goldreich. On Defining Proofs of Knowledge. *Adv. in Cryptology — Crypto 1992*, LNCS vol. 740, Springer-Verlag, pp. 390–420, 1993.

[5] M. Bellare and S. Goldwasser. New Paradigms for Digital Signatures and Message Authentication Based on Non-Interactive Zero-Knowledge Proofs. *Adv. in Cryptology — Crypto 1989*, LNCS vol. 435, Springer-Verlag, pp. 194–211, 1990.

[6] M. Bellare, S. Halevi, A. Sahai, and S. Vadhan. Many-to-one Trapdoor Functions and their Relation to Public-Key Cryptosystems. *Adv. in Cryptology — Crypto 1998*, LNCS vol. 1462, Springer-Verlag, pp. 283–298, 1998.

[7] M. Blum and S. Goldwasser. An Efficient Probabilistic Encryption Scheme Which Hides All Partial Information. *Adv. in Cryptology — Crypto '84*, LNCS vol. 263, Springer-Verlag, pp. 289–302, 1985.

[8] M. Blum and S. Micali. How to Generate Cryptographically Strong Sequences of Pseudo-Random Bits. *SIAM J. Computing* 13(4): 850–864, 1984.

[9] D. Catalano, R. Gennaro, and N. Howgrave-Graham. Paillier's Trapdoor Function Hides up to $O(n)$ Bits. *J. Cryptology* 15(4): 251–269, 2002.

[10] Y. Desmedt and Y. Frankel. Threshold Cryptosystems. *Adv. in Cryptology — Crypto 1989*, LNCS vol. 435, Springer-Verlag, pp. 307–315, 1990.

[11] D. Dolev, C. Dwork, and M. Naor. Non-Malleable Cryptography. *SIAM J. Computing* 30(2): 391–437, 2000.

[12] U. Feige, A. Fiat, and A. Shamir. Zero-Knowledge Proofs of Identity. *J. Cryptology* 1(2): 77–94, 1988.

[13] M. Fischlin. On the Impossibility of Constructing Non-Interactive Statistically-Secret Protocols From any Trapdoor One-Way Function. *Cryptographers' Track — RSA 2002*, LNCS vol. 2271, pp. 79–95, 2002.

[14] R. Gennaro, Y. Gertner, and J. Katz. Lower Bounds on the Efficiency of Encryption and Digital Signature Schemes. *35th ACM Symposium on Theory of Computing (STOC)*, ACM, pp. 417–425, 2003.

[15] R. Gennaro and L. Trevisan. Lower bounds on the Efficiency of Generic Cryptographic Constructions. *41st IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 305–313, 2000.

[16] Y. Gertner, S. Kannan, T. Malkin, O. Reingold, and M. Viswanathan. The Relationship between Public-Key Encryption and Oblivious Transfer. *41st IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 325–335, 2000.

[17] Y. Gertner, T. Malkin, and O. Reingold. On the Impossibility of Basing Trapdoor Functions on Trapdoor Predicates. *42nd IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 126–135, 2001.

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

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

[20] O. Goldreich, S. Goldwasser, and S. Micali. On the Cryptographic Applications of Random Functions. *Adv. in Cryptology — Crypto '84*, LNCS vol. 263, Springer-Verlag, pp. 276–288, 1985.

[21] O. Goldreich, S. Goldwasser, and S. Micali. How to Construct Random Functions. *J. ACM* 33(4): 792–807, 1986.

[22] O. Goldreich and L. Levin. Hard-Core Predicates for any One-Way Function. *21st ACM Symposium on Theory of Computing (STOC)*, ACM, pp. 25–32, 1989.

[23] O. Goldreich, S. Micali, and A. Wigderson. Proofs that Yield Nothing but Their Validity or All Languages in NP Have Zero-Knowledge Proof Systems. *J. ACM* 38(3): 691–729, 1991.

[24] O. Goldreich, S. Micali, and A. Wigderson. How to Play Any Mental Game or A Completeness Theorem for Protocols with Honest Majority. *19th ACM Symposium on Theory of Computing (STOC)*, ACM, pp. 218–229, 1987.

[25] S. Goldwasser and S. Micali. Probabilistic Encryption. *J. Computer and System Sciences* 28(2): 270–299, 1984.

[26] S. Goldwasser, S. Micali, and R. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen Message Attacks. *SIAM J. Computing* 17(2): 281–308, 1988.

[27] J. Håstad, R. Impagliazzo, L. Levin, and M. Luby. A Pseudorandom Generator From any One-Way Function. *SIAM J. Computing* 28(4): 1364–1396, 1999.

[28] J. Håstad, A. Schrift, and A. Shamir. The Discrete Logarithm Modulo a Composite Hides $O(n)$ Bits. *J. Computer and System Sciences* 47(3): 376–404, 1993.

[29] R. Impagliazzo. Very Strong One-Way Functions and Pseudo-random Generators Exist Relative to a Random Oracle. Manuscript, 1996.

[30] R. Impagliazzo and M. Luby. One-Way Functions are Essential for Complexity-Based Cryptography. *30th IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 230–235, 1989.

[31] R. Impagliazzo and S. Rudich. Limits on the Provable Consequences of One-Way Permutations. *21st ACM Symposium on Theory of Computing (STOC)*, ACM, pp. 44–61, 1989.

[32] J. Kahn, M.E. Saks, and C.D. Smyth. A Dual Version of Reimer's Inequality and a Proof of Rudich's Conjecture. *15th IEEE Conference on Computational Complexity*, IEEE, pp. 98–103, 2000.

[33] J.H. Kim, D.R. Simon, and P. Tetali. Limits on the Efficiency of One-Way Permutation-Based Hash Functions. *40th IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 535–542, 1999.

[34] L. Lamport. Constructing Digital Signatures From a One-Way Function. Technical Report CSL-98, SRI International, 1979.

[35] M. Naor. Bit Commitment Using Pseudorandom Generators. *J. Cryptology* 4(2): 151–158, 1991.

[36] M. Naor and M. Yung. Universal One-Way Hash Functions and Their Cryptographic Applications. *21st ACM Symposium on Theory of Computing (STOC)*, ACM, pp. 33–43, 1989.

[37] O. Reingold, L. Trevisan, and S. Vadhan. Notions of Reducibility Between Cryptographic Primitives. *1st Theory of Cryptography Conference*, LNCS vol. 2951, Springer-Verlag, pp. 1–20, 2004.

[38] J. Rompel. One-Way Functions are Necessary and Sufficient for Secure Signatures. *22nd ACM Symposium on Theory of Computing (STOC)*, ACM, pp. 387–394, 1990.

[39] S. Rudich. *Limits on the Provable Consequences of One-Way Functions*. PhD thesis, University of California at Berkeley, 1988.

[40] S. Rudich. The Use of Interaction in Public Cryptosystems. *Adv. in Cryptology — Crypto 1991*, LNCS vol. 576, Springer-Verlag, pp. 242–251, 1992.

[41] D.R. Simon. Finding Collisions on a One-Way Street: Can Secure Hash Functions be Based on General Assumptions? *Adv. in Cryptology — Eurocrypt 1998*, LNCS vol. 1403, Springer-Verlag, pp. 334–345, 1998.

[42] A. C.-C. Yao. Theory and Application of Trapdoor Functions. *23rd IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 80–91, 1982.

[43] A. C.-C. Yao. How to Generate and Exchange Secrets. *27th IEEE Symposium on Foundations of Computer Science (FOCS)*, IEEE, pp. 162–167, 1986.