Electronic Colloquium on Computational Complexity, Report No. 76 (2011)

The Advanced Encryption Standard, Candidate Pseudorandom Functions, and Natural Proofs

Eric Miles* Emanuele Viola*

May 7, 2011

Abstract

We put forth several simple candidate pseudorandom functions $f_k : \{0,1\}^n \to \{0,1\}$ with security (a.k.a. hardness) 2^n that are inspired by the AES block-cipher by Daemen and Rijmen (2000). The functions are computable more efficiently, and use a shorter key (a.k.a. seed) than previous constructions. In particular, we have candidates computable by

(1) circuits of size $n \operatorname{poly} \lg n$ (thus using a seed of length $|k| \leq n \operatorname{poly} \lg n$);

(2) TC^0 circuits of size $n^{1+\epsilon}$, for any $\epsilon > 0$, using a seed of length $|k| = O(n)$;

(3) for each fixed seed k of length $|k| = O(n^2)$, a single-tape Turing machine with $O(n^2)$ states running in time $O(n^2)$.

Candidates (1) and (3) are natural asymptotic generalizations of AES with a specific setting of parameters; (2) deviates somewhat from AES, by relaxing a certain state permutation in AES to have larger range. We argue that the hardness of the candidates relies on similar considerations as those available for AES.

Assuming our candidates are secure, their improved efficiency brings the "Natural Proofs Barrier" by Razborov and Rudich (JCSS '97) closer to the frontier of circuit lower bounds. For example, the fact that standard pseudorandom function candidates could not be computed as efficiently as the one in (2) had given rise to a plan for TC^0 circuit lower bounds (Allender and Koucký; J. ACM 2010).

We also study the (asymptotic generalization of the) AES S-box. We exhibit a simple attack for the multi-bit output, while we show that outputting one, Goldreich-Levin bit results in a small-bias generator.

*Supported by NSF grant CCF-0845003. Email: {enmiles,viola}@ccs.neu.edu

ISSN 1433-8092

1 Introduction

Introduced by Goldreich, Goldwasser, and Micali [GGM86], pseudorandom functions (PRF) are functions $f_k : \{0,1\}^n \to \{0,1\}$, parameterized by a seed (or key) k of length s, that are indistinguishable from random by efficient adversaries (for background, see [Gol01]). This exponential-stretch extension of pseudorandom generators has many applications in cryptography. It also has bearings to our ability to prove circuit lower bounds, thanks to a connection by Razborov and Rudich [RR97], reviewed below.

Many applications would benefit from having PRF that are very efficient. A natural goal is to have PRF that are computable by circuits of linear size $O(n)$, in particular that use a seed of length $s = O(n)$ and have exponential hardness 2^n, i.e., circuits of size 2^n have advantage $\leq 1/2^n$ in distinguishing the PRF from random. (Note that in this setting of hardness 2^n pseudorandom "function" is a bit of a misnomer, since the distinguisher has time to look at the entire truth-table. The PRF is better seen as a pseudorandom generator.)

There are two reasons why we do not have such PRF, both stemming from the fact that PRF are typically constructed from more basic assumptions, such as one-way functions or pseudorandom generators. First, standard hardness assumptions such as those related to factoring and discrete-log have hardness (necessarily) of the form $2^{s^{1-\Omega(1)}}$, which is insufficient. Second, even if we had a size-$O(n)$ computable one-way function $f_k : \{0,1\}^n \to \{0,1\}^n$ with hardness 2^n, the known constructions of PRF blow-up both the seed length and the circuit size by a polynomial [GGM86, HILL99] (for optimizations, see [HRV10], especially Corollary 6.3).

The lack of candidates to reach the above natural goal stands in contrast with widespread *block-ciphers*, which can be thought of as PRF where both the input and seed lengths are fixed, such as the Advanced Encryption Standard (AES) by Daemen and Rijmen [DR02]. AES is designed to be very "efficient," and on seeds of length s (e.g. $s = 128$) it is believed to resist attacks running in "time" $2^{\alpha s}$ for a constant $\alpha \approx 1$.

In this paper we put forth several candidate PRF with hardness 2^n for seed lengths ranging from $O(n^2)$ to $O(n)$. The candidates are inspired by AES, and we show that they are computable more efficiently than previous candidates. For example one candidate is computable by circuits of quasi-linear size $n \operatorname{poly} \lg n$ (and in particular the seed length is $s = n \operatorname{poly} \lg n$ as well). We then argue that these candidates preserve the hardness considerations used in the design of AES (for example, jumping ahead, resistance to differential and linear cryptanalysis). So, we believe that even if an attack is found for some candidates, this should translate into new insights into the hardness of AES, for example by showing that certain tradeoffs between parameters are necessary.

Thus our work leverages block-ciphers to obtain new theoretical constructions. This is a somewhat different angle from other works that investigate efficient cryptographic primitives that could translate to efficient block-ciphers.

The search for efficient cryptographic primitives. There is a large body of work on proposing and analyzing efficient cryptography primitives such as one-way functions

and pseudorandom generators, and even public-key cryptosystems, see e.g. [Gol00, AIK06, AIK08, ACPS09, Pei09, ABW10, App11]. By contrast, there are few PRF candidates that are more efficient than what is given by the original construction [GGM86]. Some such candidates are due to Naor and Reingold [NR04] (cf. [NR99]), and later with Rosen [NRR02], based on factoring and discrete-log assumptions. They give PRF $f : \{0,1\}^n \to \{0,1\}$ with hardness 2^n computable by circuits of size n^b, where b is a constant that is smaller than what is implied by the generic construction [GGM86, HILL99] (a fact that is also noted in [NRR02]). These PRF also have the benefit of being parallelizable. For an alternative, fixed-input-length approach to PRF, see the work by Bellare, Canetti, and Krawczyk [BCK96].

We mention that an approach of Levin [Lev87] is useful to trade efficiency for hardness in PRF constructions. Given a PRF $f_k : \{0,1\}^n \to \{0,1\}$, the idea is to construct the PRF $f'_{k,h}(x) := f_k(h(x))$ that first hashes the input x to $h(x)$ with a hash function that is part of the seed, and then applies f. Thus we only compute f_k on inputs of length $h(x) \ll n$, resulting in increased efficiency and decreased hardness. Given a PRF with superpolynomial hardness, one can apply this idea to obtain a PRF with again superpolynomial hardness computable by circuits of size n^ϵ plus the size of hashing. Ishai, Kushilevitz, Ostrovsky, and Sahai [IKOS08] show how to compute hash functions by linear-size circuits, thus obtaining such PRF computable by linear-size circuits. However this approach is useless if the hardness is at premium and cannot be traded, for example for size-$O(n)$ PRF with hardness 2^n.

Natural proofs. The landscape of circuit lower bounds remains bleak, despite exciting recent results [Wil11]. Researchers however have been successful in explaining this lack of progress by pointing out several "barriers," i.e., establishing that certain proof techniques will not give new lower bounds [BGS75, RR97, AW08].

Of particular interest to us is the Natural Proofs work by Razborov and Rudich [RR97]. They make the following two observations. First, most lower-bound proofs that a certain function $f : \{0,1\}^n \to \{0,1\}$ cannot be computed by circuits C (e.g., C = circuits of size n^2) entail an algorithm that runs in time polynomial in $N := 2^n$ and can distinguish truth-tables of n-bit functions $g \in C$ from truth-tables of random functions (i.e., a random string of length N). (For example, the algorithm corresponding to the restriction-based proof that Parity is not in AC^0, given $f : \{0,1\}^n \to \{0,1\}$, checks if there is one of the $2^{O(n)} = N^{O(1)}$ restrictions of the n variables that makes f constant.) Informally, any proof that entails such an algorithm is called "natural."

The second observation is that, under standard hardness assumptions, no algorithm such as the above one exists when C is a sufficiently rich class. For example, under the assumption that computing a factor of a random s-bit Blum integer has hardness $2^{s^{\Omega(1)}}$, one can construct a PRF $f_k : \{0,1\}^n \to \{0,1\}$ with seed length $|k| = \mathrm{poly}(s)$ and also hardness $2^{s^{\Omega(1)}}$. Making $s = n^c$ for large enough $c = O(1)$, the hardness is $N^{\omega(1)}$ and in particular no algorithm as the above one exists.

The combination of the two observations is that no natural proof exists against circuits of size n^c, for a large enough $c = O(1)$. Moreover, the aforementioned PRF construction by Naor and Reingold [NR04] is implementable in the restricted circuit class TC^0 of unbounded

fan-in majority circuits of constant-depth and polynomial-size, pushing the above second observation "closer" to the frontier of known circuit lower bounds.

The gap between lower bounds and PRF. However, the natural proofs barrier still has a significant gap with known lower bounds, due to the lack of sufficiently strong PRF. For example, there is no explanation as to why one cannot prove superlinear-size circuit lower bounds. For this one would need to reach the natural goal mentioned at the beginning, i.e. have a PRF $f_k : \{0,1\}^n \to \{0,1\}$ that is computable by linear-size circuits (hence in particular with $|k| = O(n)$) and with exponential hardness 2^n. (So that, given n, if one had a distinguisher running in time $2^{O(n)}$, one could pick a PRF on inputs of length bn for a large enough constant b, to obtain a contradiction.)

A recent work by Allender and Koucký [AK10] brings to the forefront another setting where the Natural Proofs barrier does not apply: proving lower bounds on TC^0 circuits of size $n^{1+\epsilon}$ and depth d, for any $\epsilon > 0$ and a large enough $d = d(\epsilon)$. Naor and Reingold's PRF in TC^0 requires larger size. The setting is especially interesting because [AK10] shows that such a lower bound for certain functions implies a "full-fledged" lower bound for TC^0 circuits of polynomial-size. Moreover even if the first lower bound were natural, the latter would not be, thus circumventing Naor and Reingold's PRF in TC^0. Even more, a lower bound where $\epsilon = \epsilon(d)$ (as opposed to $d = d(\epsilon)$) is known [IPS97].

Another long-standing problem is to exhibit a candidate PRF in ACC^0, which would explain the slow progress on lower bounds, cf. [Wil11].

Of course, circuit models such as the above ones are only some of the models in which the gap between candidate PRF and lower bounds is disturbing. Other such models include various types of Turing machines, and small-space branching programs. For example, there is no explanation as to why the lower bounds for single-tape Turing machines stop at quadratic time, cf. [KN97, §12.2].

1.1 Background on AES

To define our candidates, we begin with a review of AES; the reader is referred to [DR02] for a thorough treatment. AES is a block cipher operating on ($n = 128$)-bit messages, parameterized by a seed K of length $s = 128$ (other seed lengths are supported as well). On an input message x, the computation proceeds in $r = 10$ *rounds*. Each round $i = 1, \ldots, r$ uses a different n-bit string $K^{(i)}$, referred to as the ith round key, and each round key is derived from K via a process known as the AES key schedule. In addition there is an n-bit "whitening" key $K^{(0)}$, also derived from K.

Each round i maps the current n-bit state to a new n-bit state, where the initial state is $x \oplus K^{(0)}$. Each round is computed as follows (see Figure 1). First, the state is divided into *bundles* of $b = 8$ bits, and to each bundle a permutation γ is applied, which has the form

$$\gamma(x) := L\left(x^{2^b-2}\right),$$

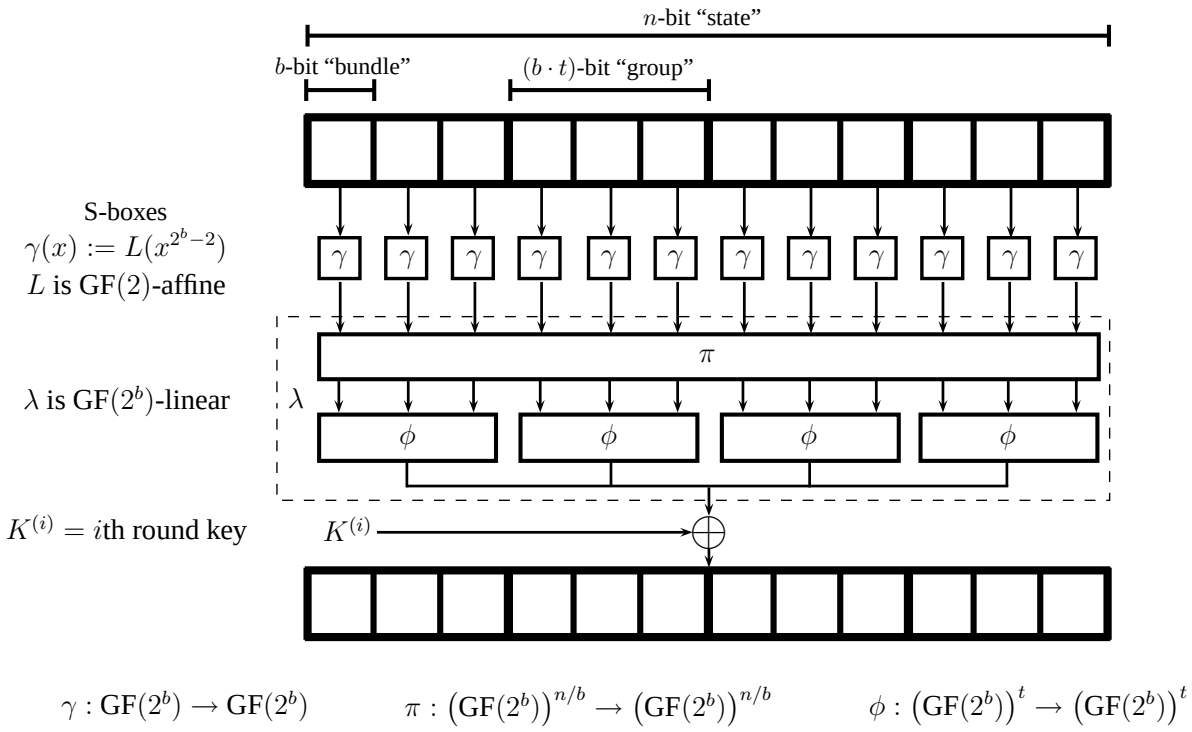

Figure 1: One round of AES

where L is an affine transformation over $\mathrm{GF}(2)^b$; note that x^{2^b-2} is field inversion x^{-1} over $\mathrm{GF}(2^b)$ if $x \neq 0$. Each instance of γ is called an *S-box*.

Second, a linear permutation λ is applied to the entire state; specifically, each bundle is seen as an element in $\mathrm{GF}(2^b)$, and λ is seen as a $n/b \times n/b$ matrix over this field. Up to a permutation π of the positions of the input bundles, λ is a block-diagonal matrix where each block is the same $t \times t$ matrix ϕ ($t = 4$).

Finally, the state is XORed with the current round key $K^{(i)}$.

Design criteria. AES is designed to resist linear and differential cryptanalysis ([Mat94, BS91]; see also Chapters 6-9 of [DR02]). This resistance is not proved formally, but is believed to follow from the combination of some heuristics and two theorems. The first theorem gives an upper bound ($\star$) of $2^{-(b/2-1)}$ on certain input-output correlations of the S-box γ (this will be made precise in a later section; see Theorem 2.2).

The second theorem gives a lower bound on a certain quantity related to the computation of AES on non-zero inputs, a quantity known as the *number of active S-boxes*. The number $\mathcal{A}$ of active S-boxes can be defined as follows. Let AES' be the function that is identical to AES, except that $K^{(i)} := 0^n$ for $i = 0, \ldots, r$ and the S-box γ is the identity map. Let $w_b : \left(\{0,1\}^b\right)^{n/b} \to \mathbb{N}$ be the function which counts the number of non-zero b-bit bundles in

its input. Then,

$$\mathcal{A} := \min_{0^n \neq x \in \{0,1\}^n} \sum_{i=1}^{r} w_b(\text{state of AES'}(x) \text{ at the beginning of round } i).$$

Given this definition, the second theorem is the following.

Theorem 1.1 ([DR02], Thm. 9.5.1)**.** $\mathcal{A} \geq \lfloor r/4 \rfloor \cdot (t+1)^2$.

With respect to linear and differential cryptanalysis, the hardness of AES is measured as the time needed to recover the round keys (or some portion thereof). The dominating factor of the workload for these attacks is the number of plaintext/ciphertext pairs needed, and thus lower bounding this number is the route taken to show hardness. To do this, an upper bound similar to ($\star$) is used, though now what is bounded are the correlations of the entire cipher, and not just the S-box. For AES, this bound is believed to be $2^{-\mathcal{A}\cdot(b/2-1)}$, which is the previous bound ($\star$) raised to $\mathcal{A}$. Intuitively, this should hold because the aforementioned correlations should multiply across active S-boxes. The number of plaintext/ciphertext pairs needed for the attacks is inversely proportional to this correlation[1], and thus the hardness of AES is believed to be

$$\geq 2^{\mathcal{A}\cdot(b/2-1)}.$$

(See Definition 2.1 and Theorem 2.2 for more precise details.)

In AES, $b = 8$ and Theorem 1.1 guarantees $\mathcal{A} \geq 50$ because $t = 4$ and $r = 10$, so the hardness is at least 2^{150}. The number of rounds r is chosen so as to achieve hardness which is greater than the work factor 2^s of an exhaustive search for the seed K. Accordingly, r increases for seeds larger than $s = 128$. Extra rounds are also added to provide additional "diffusion", which in this context means propagating (possibly small) changes in the state bits throughout the entire state. See [DR02, §3.5] for a more detailed discussion on the number of rounds.

While resistance to linear and differential cryptanalysis is "the most important criterion in the design" of AES [DR02, p. 81], considerations have also been taken to prevent attacks that would exploit algebraic structure in the cipher (e.g. the interpolation attacks of Jakobsen and Knudsen [JK01]). For instance, the affine portion L of the S-box γ is included to "allow γ to have a complex algebraic expression" when considered as a univariate polynomial over $\mathrm{GF}(2^b)$ [DR02, p. 36]. Choices also appear to have been made to ensure that the computation has high degree when considered as a multivariate polynomial over $\mathrm{GF}(2)$. For example, the use of $x \mapsto x^{2^b-2}$ results in each of γ's output bits having (near-maximum) degree $\geq b-1$. Using instead $x \mapsto x^3$ would not diminish AES's resistance to linear and differential cryptanalysis, but it would result in degree (only) 2 [Pie91, Nyb93] (cf. [Kop11, Lemma 9]). We refer the reader to [CMR06] for an overview of the algebraic structure of AES and potential means of exploiting it. In general, algebraic cryptanalysis appears less understood than linear and differential cryptanalysis.

[1]For linear cryptanalysis, the number is actually inversely proportional to the square of the correlation.

Finally, although AES's hardness is often measured against *key-recovery* attacks, we share many researchers' viewpoint that *distinguishing* attacks are the correct model. We also point out that the linear and differential cryptanalysis techniques can be seen as falling in the latter type of attacks, as they do construct a distinguishing algorithm (which is then used to select the correct round key from a set of potential keys).

1.2 Our candidates

We propose several ways to generalize AES to obtain candidate PRF defined on infinitely many input lengths. In each of our candidates, we do away with the AES key schedule and instead use uniform and independent round keys; as a result, the seed length is always $s = n(r+1)$. Using independent round keys, as opposed to generating them from a single n-bit seed, should only increase hardness.

Inspired by AES, we use the value $2^{\mathcal{A}\cdot(b/2-1)}$ as a measure of hardness for each of our candidates; namely, for n-bit functions we require that $\mathcal{A}\cdot(b/2-1) \geq n$.

We observe another restriction on the candidates, originating when each output bit is viewed as a polynomial over GF(2) in the input bits. To resist attacks that exploit the degree of this polynomial, we need the degree of each of our candidates to be $\geq \epsilon n$ for a certain constant ϵ. (For completeness we present such an attack in Appendix A, showing that a PRF which has degree $o(n)$ cannot have hardness 2^n.) The only non-linear operation in the entire cipher is the S-box γ, which operates on b bits and has degree $b-1$, and hence the maximum possible degree of each output bit is at most $(b-1)^r$. We require $b^r \geq n$ in each of our candidates. (The distinction between $(b-1)^r \geq \epsilon n$ and $b^r \geq n$ is unimportant, as in our candidates we can always increase r by a constant factor, except in Candidate 4 where we have $r = 1$ and $b = n$.)

Another obvious obstruction to an output bit having large degree is that this bit depends on few input bits. Towards avoiding this obstruction, we make sure the following syntactic requirement is satisfied. Consider the graph with r layers and degree t corresponding to π, in which nodes on layer h represent bundles at round h, and there is an edge $i \to j$ between two bundles i, j if they are in the same group, i.e., if bundle i is an input to the occurrence of ϕ outputting bundle j. We make sure that each output bundle is connected to $\Omega(n/b)$ input bundles in this graph (in particular, $t^r \geq \Omega(n/b)$). We note that this holds for AES after even 2 rounds [DR02, §3.5].

We now briefly describe each of our candidates. Each is candidate to having hardness 2^n, and the seed length will range from $s = O(n^2)$ to $s = O(n)$. Candidates 1-2 output n bits, Candidate 3 outputs $O(n)$ bits, and Candidate 4 outputs 1 bit.

Candidate 1. This candidate is our most straightforward generalization of AES, and may be folklore. We keep $b = 8$ and $t = 4$, leave γ unchanged, and we leave λ unchanged except for the necessary increase in the number of input/output bundles. To get $\mathcal{A}\cdot(b/2-1) \geq n$, we take $r = n$. (Theorem 1.1 is actually overkill for this candidate, because the trivial lower bound $\mathcal{A} \geq r$ suffices when $r = n$.)

Candidate 1 is computable by size $O(n^2)$, depth $O(n)$ Boolean circuits. Note this is already better than previous candidate PRF. For each fixed seed K, Candidate 1 is also computable in time $O(n^2)$ by a single-tape Turing machine with $O(n^2)$ states (note that the fixed-seed setting is sufficient for the natural proofs connection).

It is an open problem to get a candidate computable in time $O(n)$ on a 2-tape Turing machine.

Candidate 2. In this candidate, we set $b = \log n$, $t = \sqrt{n/\log n}$ and $r = \Theta(\log n/\log\log n)$. γ and λ are again the same as in AES except for an increase in the input/output sizes.

We prove that Candidate 2 is computable by Boolean circuits of quasilinear-size $\widetilde{O}(n) := n \operatorname{poly}\log n$. To show this, note that since r is logarithmic, it is enough to show how to compute each round with those resources. Moreover, since b is logarithmic, computing the S-boxes comes at little cost.

Our main technical contribution in this candidate is to show how to efficiently compute each $t \times t$ block of AES's block-diagonal linear transformation λ; specifically, we show that each block can be computed with size $\widetilde{O}(t) = t \operatorname{poly}\lg t$, for a total size of

$$\widetilde{O}\left(\frac{n}{t} \cdot t\right) = \widetilde{O}(n).$$

In AES (and in Candidates 1-2), each block of λ is constructed from the generator matrix G of a $t \to 2t$ maximum distance separable (MDS) code; specifically, if $G^T = [I \mid M]$, then each block of λ is M. Our method for computing λ efficiently has two parts. First, we use a result by Roth and Seroussi [RS85] that if G generates a Reed-Solomon code (which is well-known to be MDS), then M forms a $t \times t$ *Cauchy matrix* (a certain type of matrix specified by $O(t)$ elements). We then use a result by Gerasoulis [Ger88] to compute the product of a length-t vector (consisting of bundles of the state) and a Cauchy matrix in quasilinear time; this requires a simple adaptation of the algorithm in [Ger88] to fields of characteristic 2.

It is an open problem to get a candidate computable by circuits of size $O(n)$.

Candidate 3. In the previous two candidates, the components γ and λ remain essentially unchanged from AES. In Candidate 3, we also keep γ the same (aside from the increase in input/output size), but we modify the linear permutation λ.

Our observation is that the important property of λ is just that it allows one to bound the number $\mathcal{A}$ of active bundles. With a simple modification to λ, we obtain a lower bound of the form $\mathcal{A} = \Omega(t^{r-1})$, rather than the weaker $\mathcal{A} = \Omega(r \cdot t^2)$ (see Theorem 3.7). (Note: we do not know that the transformation λ in AES would not give a bound of the form $t^{\Omega(r)}$, but because of technical complications we are not able to prove it for their construction, which relies on a somewhat subtle propagation of the active S-boxes (cf. §3.3).) The strengthened lower bound allows us to take $b, t = n^\epsilon$ and $r = O(1/\epsilon)$ for arbitrarily small $\epsilon > 0$, and so each round is computable in size

$$\frac{n}{b}\operatorname{poly}(b) + \frac{n}{tb}\operatorname{poly}(b, t) = n^{1+O(\epsilon)},$$

and the whole circuit also in size $n^{1+O(\epsilon)}$.

We mention that our modified λ retains much of the same features of the original λ. It is still a linear, invertible map, corresponding to a block-diagonal matrix. However it will not be a permutation anymore, and instead we double the size of the state at each round (so the blocks are not square). As a result, Candidate 3 maps n bits to $2^r n$ bits. However, because we take $r = O(1)$, the growing state size poses no problem for the analysis.

We further show that Candidate 3 is computable even by TC^0 circuits of size $n^{1+O(\epsilon)}$ for any $\epsilon > 0$ (with depth depending on ϵ), cf. §"The gap between lower bounds and PRF" above. The technical difficulty in implementing this candidate with the required resources is that the S-box function γ requires computing *inversion* in a field of size 2^b (recall $b = n^{\Omega(1)}$). To implement this in TC^0 we note (cf. [HV06]) that inverting the field element $\alpha(x)$ can be accomplished as:

$$\alpha(x)^{2^b-2} = \alpha(x)^{\sum_{i=1}^{b-1} 2^i} = \prod_{i=1}^{b-1} \alpha(x)^{2^i} = \prod_{i=1}^{b-1} \alpha\left(x^{2^i}\right)$$

where the last equality follows from the fact that we are working in characteristic 2. By hard-wiring the $\leq b$ powers $x, x^2, \ldots, x^{2^{b-1}}$ of x in the circuit, and using the fact that the iterated product of $\mathrm{poly}(n)$ field elements is computable by $\mathrm{poly}(n)$-size TC^0 circuits (see e.g. [HAB02, Corollary 6.5] and cf. [HV06]), we obtain a TC^0 circuit.

A more radical candidate is a variant of Candidate 3 in which we apply an error-correcting code to the whole state. After even one round this gives a good enough bound on $\mathcal{A}$, but the details of computing the code efficiently by TC^0 circuits of size $n^{1+\epsilon}$ are more involved (details omitted).

Finally, we mention that implicit in the assumption that Candidate 3 is indeed hard is the assumption that field inversion cannot be computed by unbounded fan-in constant depth circuits with parity gates $\mathrm{AC}^0[\oplus]$. For otherwise, it can be shown that the whole candidate would be in that class, in contradiction with an algorithm in [RR97, §3.2.1] which distinguishes truth tables of $\mathrm{AC}^0[\oplus]$ functions from random ones. (λ can be seen to be a linear operation over GF(2), hence it can be computed easily with parity gates.) Some evidence that field inversion is not in $\mathrm{AC}^0[\oplus]$ comes from the fact that related functions can indeed be shown not to be in $\mathrm{AC}^0[\oplus]$; see [HV06] and the recent work by Kopparty [Kop11].

Candidate 4. In our final candidate, we use the extreme setting of parameters $b = n$ and $r = t = 1$. In other words, Candidate 4 consists of one round, and this round contains only a single S-box (and in particular no linear permutation). While this setting does indeed preserve resistance to linear and differential cryptanalysis, we exhibit a simple attack, inspired by Jakobsen and Knudsen [JK01], in which we exploit the algebraic structure to recover the seed with just 4 queries.

We put forth a related, clean candidate where we only output the Goldreich-Levin bit [GL89]: $f_{k,k'}(x) := \langle (x+k)^{2^b-2}, k' \rangle$. We prove that this candidate is a small-bias generator [NN93, AGHP92], and using Braverman's result [Bra09] (cf. [Baz09, Raz09]) we obtain that this candidate also fools small-depth AC^0 circuits of any size $w = 2^{n^{o(1)}}$ (that look at only w fixed output bits of the candidate).

Using the same ideas for Candidate 3, this candidate is also computable by poly-size TC^0 circuits. For unbounded-depth circuits, a more refined size bound $\widetilde{O}(n^2)$ follows from the exponentiation algorithm in [GvzGPS00].

Organization. In §2 we give some additional details on AES, in particular on the features of its component functions that provide hardness. In §3 we construct each of our PRF candidates described above. We conclude in §4. Appendix A sketches an attack on low-degree PRF.

2 The Advanced Encryption Standard

We now give a more thorough description of AES, building on the overview in §1.1. Recall that AES is computed over r rounds, each of which maps the current n-bit state to a new n-bit state, and that we view the state as being divided into n/b bundles of b bits (see Figure 1). Each round is computed using the three functions γ, π and ϕ; there are specific restrictions put on these functions in order to provide resistance to differential and linear cryptanalysis, which we describe now.

The nonlinear S-box γ. The permutation $\gamma : \mathrm{GF}(2^b) \to \mathrm{GF}(2^b)$ is defined by $\gamma(x) := L(x^{2^b-2})$, where L is a transformation that is affine over $\mathrm{GF}(2)^b$ (but not over $\mathrm{GF}(2^b)$). Note that x^{2^b-2} is simply inversion in $\mathrm{GF}(2^b)$ with $0^{-1} := 0$, and we will refer to this as "patched inversion". To describe the desirable features of γ, we need the following definition.

Definition 2.1. Let $f : \{0,1\}^b \to \{0,1\}^b$ be a function. The *maximum differential propagation probability* of f is $\mathrm{MaxDPP}(f) := \max_{\Delta_x \neq 0^b, \Delta_y} \left(\Pr_x \left[f(x) \oplus f(x \oplus \Delta_x) = \Delta_y\right]\right)$, and the *maximum linear correlation* of f is $\mathrm{MaxCor}(f) := \max_{u,w \neq 0^b} \left|2 \cdot \Pr_x[\langle u, x\rangle = \langle w, f(x)\rangle] - 1\right|$.

The results of [Nyb93] and the references therein establish the following theorem.

Theorem 2.2. *Let $\gamma : \{0,1\}^b \to \{0,1\}^b$ be defined as above. Then,*

$$1.\ \mathrm{MaxDPP}(\gamma) \leq 2^{-(b-2)}. \qquad 2.\ \mathrm{MaxCor}(\gamma) \leq 2^{-(b/2-1)}.$$ [2]

Theorem 2.2 is the first theorem which was alluded to in §1.1, and it is primarily the combination of this and Theorem 1.1 which supports the belief that AES has hardness 2^{150} and 2^{300} against linear and differential cryptanalysis, respectively. More specifically, it is believed that MaxDPP and MaxCor multiply across active S-boxes, and so it is believed that $\mathrm{MaxDPP}(\mathrm{AES}) \leq 2^{-\mathcal{A}(b-2)}$ and $\mathrm{MaxCor}(\mathrm{AES}) \leq 2^{-\mathcal{A}(b/2-1)}$. We refer the reader to [DR02, Ch. 7-8] for further details on this argument.

We also note that the bound MaxCor(AES) incorporates the same bound on each fourier coefficient of each output bit of AES. In turn, this implies that each output bit depends on

[2] [Nyb93] actually bounds a related quantity known as the *non-linearity* of γ, but it translates directly to the stated result.

$\Omega(n)$ input bits. (For else it can be verified that it would have too large a fourier coefficient by Parseval's identity.)

The affine portion of γ is defined by $L(x) := M_\gamma x + a_\gamma$. M_γ is a circulant matrix, in which the first row consists of $\lceil b/2 \rceil + 1$ ones followed by $\lfloor b/2 \rfloor - 1$ zeros, and each subsequent row is equal to the row above it shifted circularly one place to the right. a_γ was chosen so that there are no fixed points and no opposite fixed points, i.e. values $x \in \{0,1\}^b$ such that $\gamma(x) = x$ or $\gamma(x) = \overline{x}$.

We stress that we do not modify the function γ in any of our candidates. We do allow the bundle size b to vary, but γ is always computed as described here.

The bundle permutation π. The combination of π and ϕ provides inter-bundle diffusion throughout the computation of AES. Recall that we view the n/b bundles of the state as being grouped consecutively into groups of size t. The criterion that π should satisfy is referred to as diffusion optimality by [DR02, Def. 9.4.1].

Definition 2.3. A bundle permutation π is *diffusion optimal* if for every group i, and every two bundles $j \neq k$ in group i, $\pi(j)$ and $\pi(k)$ are in different groups.

Note that it is only possible to have a diffusion optimal bundle permutation if there are at least as many groups as there are bundles per group, i.e. $n/(bt) \geq t \iff n \geq bt^2$.

In AES, the bundle permutation π is computed as follows. First, the bundles are placed column-wise into a $t \times (n/bt)$ matrix (each group is completely contained in one column). Then row i of the matrix ($0 \leq i < t$) is shifted circularly to the left by i places, and finally the bundles are extracted column-wise from the new matrix.

Recall that in our candidates we impose the additional restriction that in the layered connection graph specified by π, each output bundle is connected to $\Omega(n/b)$ input bundles. It can be verified that for $r \geq n/(bt)$, this holds for the above construction of π. In the special case when the matrix is square, i.e. when $t = \sqrt{n/b}$ (which is the setting of both AES and our Candidate 2), it holds for $r \geq 2$.

The linear function ϕ. ϕ is specified by choosing a matrix in $\mathrm{GF}(2^b)^{t \times t}$ (we let ϕ denote both the matrix and the function defined by left-multiplication with this matrix). The crucial design criterion here is that ϕ has maximal branch number:

Definition 2.4. Let $\phi : \mathbb{F}^t \to \mathbb{F}^t$ be a linear transformation acting on vectors over a field $\mathbb{F}$. The *branch number* of ϕ is

$$\mathrm{Br}(\phi) = \min_{\alpha \neq 0^t} \left(w(\alpha) + w(\phi(\alpha)) \right)$$

where $w(\cdot)$ denotes the number of non-zero elements.

The maximum possible branch number for any transformation operating on vectors of length t is $t+1$. In AES, ϕ is constructed from the generator matrix of a maximum distance separable (MDS) code as follows. Let G be the generator matrix for any MDS code mapping

$\mathbb{F}^t \to \mathbb{F}^{2t}$, and take G to be in reduced echelon form, i.e. $G^T = [I \mid M]$ where I is the $t \times t$ identity matrix. Then, it is easy to see that $\mathrm{Br}(M) = G$'s minimum distance, namely $t+1$.

Finally, we briefly revisit Theorem 1.1, the lower bound on the number $\mathcal{A}$ of active S-boxes in AES. The theorem holds for any cipher with the AES structure in which π and ϕ satisfy the above criteria.

Theorem 2.5 ([DR02], Thm. 9.5.1)**.** *Let $\mathcal{F} : \{0,1\}^n \to \{0,1\}^n$ be any block cipher computed over r rounds, in which each round consists of: (1) n/b parallel S-boxes; (2) a diffusion optimal bundle permutation; (3) n/bt parallel linear transformations each with branch number $t+1$; (4)* XOR *with the current round key. Then, $\mathcal{A} \geq \lfloor r/4 \rfloor \cdot (t+1)^2$.*

3 Candidate pseudorandom functions

In this section we give the details of the candidate PRFs described in §1.2. Notationally, we will use $\mathcal{F}_i$ to refer to Candidate i.

3.1 Candidate 1

We keep the choices $b = 8$ and $t = 4$, and let the number of rounds r grow with n as needed to preserve the hardness. Let $\mathcal{F}_1 : \{0,1\}^n \to \{0,1\}^n$ be this function, computed as described in §2, and note that this is well-defined for any n that is a multiple of 32.

Efficiency: small circuits. In each round, the $O(n)$ instances of γ and ϕ each perform computations on a constant number of bits; because permuting the bundles and adding the round key can also be done with $O(n)$ wires, each round of $\mathcal{F}_1$ can be computed by a circuit of depth $d = O(1)$ and size $w = O(n)$. Thus the entire circuit for $\mathcal{F}_1$ has depth $d = O(r)$ and size $w = O(rn)$. We will take $r = n$ to get hardness 2^n below, so $\mathcal{F}_1$ is computable by circuits of size $w = O(n^2)$ and depth $d = O(n)$.

Efficiency: fast Turing machines. Similarly, for any fixed seed $K \in \{0,1\}^s$, each round of $\mathcal{F}_1$ can be computed in time $O(n)$ on a single-tape Turing machine with $O(n^2)$ states. To do so, we encode the bundles on the tape so that the matrix used by π is written column-wise. As before, the $O(n)$ instances of γ and ϕ in a single round can be done in time $O(n)$. To see that π can also be computed in time $O(n)$, note that due to the column-wise representation each bundle needs to move ≤ 3 places away, except for the 6 bundles which are shifted circularly to the other end of the tape. Finally, encoding the $(s = O(n^2))$-bit seed in the TM's state transitions, the addition of each round key also takes time $O(n)$. Therefore, the $r = n$ rounds of $\mathcal{F}_1$ can be computed in time $O(n^2)$.

Alternatively, consider the Turing machine variant with two tapes, in which the first tape is read-only and contains the n-bit input followed by the $n(r+1)$-seed, the second tape is read/write, and the TM has $O(1)$ states. Then $\mathcal{F}_1$ can again be computed in time $O(n^2)$ exactly as described above, because in round i only bits $in+1, \ldots, in+n$ of the seed are used.

Hardness. We let the number of rounds $r = n$. This guarantees that $b^r \geq n$, and furthermore (by Theorem 2.5) that $\mathcal{A} \cdot (b/2 - 1) > n$.

3.2 Candidate 2

In this section we give a candidate, $\mathcal{F}_2$, which is computable by Boolean circuits of size $n \log^{O(1)} n$. γ and π remain unchanged from AES other than the increase in the input/output size. We choose ϕ from the generator matrix of an MDS code as is done in AES, but we do this in a way so that multiplication by ϕ can be done with few operations over $GF(2^b)$.

We now specify the parameters for this candidate. Let $\ell \in \mathbb{N}$ be sufficiently large, and set $b = 2^{2\ell}$, $n = 2^b$, $t = 2^{(b-2\ell)/2}$, and $r = \lceil b/2\ell \rceil$; this gives $b = \log n$, $t = \sqrt{n/\log n}$ and $r = \lceil \log n / \log\log n \rceil$. Observe that we have $bt^2 \leq n$ which is necessary for π to be well-defined (cf. Definition 2.3), and also that we have $2^b \geq 2t$ which is necessary for the construction of a $t \to 2t$ Reed-Solomon code over $GF(2^b)$ below.

Construction of ϕ. Let G be the $2t \times t$ generator matrix of a Reed-Solomon code over $GF(2^b)$. [RS85, Theorem 1] shows that when G is put into reduced echelon form, i.e. when $G^T = [I \,|\, M]$ where I is the $t \times t$ identity matrix, then M is a $t \times t$ generalized Cauchy matrix.

Definition 3.1. Let $\mathbb{F}$ be any field of characteristic 2. A matrix $C \in \mathbb{F}^{t \times t}$ is a *Cauchy matrix* if there exist $2t$ distinct values $\alpha_1, \ldots, \alpha_t, \beta_1, \ldots, \beta_t \in \mathbb{F}$ such that $C_{i,j} = (\alpha_i + \beta_j)^{-1}$. Furthermore, a matrix $M \in \mathbb{F}^{t \times t}$ is a *generalized Cauchy matrix* if it can be written as $M = BCD$, where C is a Cauchy matrix and $B, D \in \mathbb{F}^{t \times t}$ only have non-zero values on the diagonal.

Recall that because G generates an MDS code, the operation defined by left multiplication with M has (maximal) branch number $t + 1$. So, we choose $\phi = M$.

[Ger88] shows that multiplication of a vector by a $t \times t$ Cauchy matrix can be done with $\widetilde{O}(t)$ operations when the underlying field is $\mathbb{C}$. (Multiplication with B and D in the above definition can be done with $O(t)$ operations, so we will focus on multiplication by C.) This algorithm can also be made to work over $GF(2^b)$, as we now show. We stress that we are using the algorithm from [Ger88] without modification; the purpose here is to show that it works over $\mathrm{GF}(2^b)$.

Theorem 3.2. *Let $C \in GF(2^b)^{t \times t}$ be a Cauchy matrix defined by the (distinct) elements $\{\alpha_j, \beta_j\}_{j \in [t]}$. Then, given any vector $z \in GF(2^b)^t$, the product $C \cdot z$ can be computed with $O(t \cdot \log^2 t \cdot \log\log t)$ operations over $GF(2^b)$.*

Proof. Define the following polynomial.

$$f(x) := \sum_{j=1}^{t} z_j (x + \beta_j)^{2^b - 2}$$

Then we have $C \cdot z = (f(\alpha_1), \ldots, f(\alpha_t))$, and so it suffices to evaluate f at the points $\{\alpha_i\}_i$. Now define the following three polynomials.

$$g(x) := \prod_{j=1}^{t}(x + \beta_j)$$
$$h(x) := \sum_{i=1}^{t}\left[z_i(x + \beta_i)^{2^b-1} \cdot \prod_{j\neq i}(x + \beta_j)\right]$$
$$h_*(x) := \sum_{i=1}^{t} z_i \prod_{j\neq i}(x + \beta_j)$$

Then we have $f(x) = h(x)/g(x)$ as formal polynomials. Furthermore, for any $y \notin \{\beta_j\}_j$ we have $h(y) = h_*(y)$, using the identity $y^{2^b-2} = 1$ valid for any $y \neq 0$. Since our goal is to evaluate $f(\alpha_i)$ for all i, this is now seen to be equivalent to evaluating $h_*(\alpha_i)/g(\alpha_i)$ because $\alpha_i \neq \beta_j$ for all i, j.

Notice that, for each β_j, we have $h_*(\beta_j) = z_j \cdot g'(\beta_j)$, where $g'(x) = \sum_{i\in[t]} \prod_{j\neq i}(x + \beta_j)$ is the derivative of g. So, another way to view $h_*(x)$ is that it is the unique degree $\leq t - 1$ polynomial interpolating the points $\{(\beta_j,\ z_j \cdot g'(\beta_j)\}_{j\in[t]}$. The algorithm is now the following:

1. Compute $g(x)$ and $g'(x)$ in coefficient form.
2. Evaluate $g'(\beta_j)$ for each β_j.
3. Compute all values of $z_j \cdot g'(\beta_j)$.
4. Interpolate the points $\{(\beta_j, z_j \cdot g'(\beta_j)\}$ to obtain $h_*(x)$ in coefficient form.
5. Evaluate both $g(\alpha_j)$ and $h_*(\alpha_j)$ for each α_j.
6. Compute each value of $f(\alpha_j) = h_*(\alpha_j)/g(\alpha_j)$.

We note that steps 1 and 2 do not involve the vector z and thus can be pre-processed, and that steps 3 and 6 can easily be done with t operations over $GF(2^b)$ each. We thus focus on the remaining steps. The following results can be found in (for example) [vzGG03, Ch. 10], and they hold for any commutative ring with unity R.

Theorem 3.3 ([vzGG03], Corollary 10.8)**.** *Evaluation of a polynomial in $R[x]$ of degree $< t$ at t points can be done with $O(t \cdot \log^2 t \cdot \log\log t)$ operations in R.*

Theorem 3.4 ([vzGG03], Corollary 10.12)**.** *Given t distinct values $u_1, \ldots, u_t \in R$ and t arbitrary values $v_1, \ldots, v_t \in R$, the unique polynomial in $R[x]$ of degree $< t$ which interpolates $\{(u_i, v_i)\}_i$ can be computed in coefficient form with $O(t \cdot \log^2 t \cdot \log\log t)$ operations in R.*

As a result, steps 4 and 5, and thus the entire multiplication by C, can be performed with the stated number of operations in $GF(2^b)$. □

Efficiency. We now show that $\mathcal{F}_2$ can be computed by Boolean circuits of quasilinear size. We first consider the circuit size necessary to compute one round of $\mathcal{F}_2$ for any given values of n, t and b. One round consists of the following:

1. n/b parallel instances of exponentiation in $GF(2^b)$ (i.e. $x \to x^{2^b-2}$).
2. n/b parallel instances of an affine transformation $L : \{0,1\}^b \to \{0,1\}^b$.
3. One instance of the bundle permutation π.
4. $n/(bt)$ parallel instances of multiplication by $\phi \in GF(2^b)^{t \times t}$.
5. One instance of the round key addition.

Because finite field arithmetic and affine transformations are computable by polynomial size circuits, steps 1 and 2 can be computed by a circuit with at most $(n/b) \cdot b^{O(1)} = n \log^{O(1)} n$ wires. Steps 3 and 5 can each clearly be done with $O(n)$ wires. For step 4, we have size at most $(n/(bt)) \cdot t \log^3 t \cdot b^{O(1)} = n \log^{O(1)} n$. Note that the factor of t is crucially cancelled out in step 4, and indeed this is the reason for using a ϕ that is a generalized Cauchy matrix.

We conclude that the $r = \Theta(\log n / \log \log n)$ rounds of $\mathcal{F}_2$ are computable by a circuit of size at most $w = n \log^{O(1)} n$.

Hardness. The parameter choices ensure that $b^r \geq n$. Theorem 2.5 guarantees that $\mathcal{A} \geq (r/4) \cdot (t+1)^2 = \Omega(n/\log \log n)$, so $\mathcal{A} \cdot (b/2 - 1) \geq n$ for sufficiently large n.

3.3 Candidate 3

In this section we give a candidate, $\mathcal{F}_3$, that can be computed by TC^0 circuits of size $O(n^{1+\epsilon})$ for any fixed $\epsilon > 0$. This variant differs from the previous ones in that the size of the state doubles each time, and thus $\mathcal{F}_3$ maps n bits to $2^r n$ bits. The parameter choices will be $b = t = n^{\Theta(\epsilon)}$ and $r = \Theta(1/\epsilon)$. The generalization of γ is the same as in the previous candidates, but the choices of ϕ and π will differ, as described below.

Construction of ϕ. To construct the linear transformation ϕ, we again start from the $(2t \times t)$ generator matrix G of an MDS code, but we no longer take only the parity-check portion. Instead, we set $\phi = G$, which in particular means that ϕ now maps t bundles to $2t$ bundles. (This is the reason for the growing state size.) In fact, we will allow ϕ to differ between rounds, and specifically $\phi^{(j)}$ in round j will be the $(2^j t \times 2^{j-1} t)$ generator matrix of an MDS code. It is important to note that while the *size* of the groups on which ϕ acts grows at each round, the *number* of groups remains the same, namely $n/(bt)$.

To see the advantage that this has over the choice of ϕ in AES, consider an input vector to $\phi^{(1)}$ in which all bundles are non-zero. If we use only the fact that $\phi^{(1)}$ has maximal branch number, then we are only guaranteed that $\phi^{(1)}$'s output will have *one* non-zero bundle. However if we instead take $\phi^{(1)} = G$ as just described, then we are guaranteed that at least

$t+1$ out of the $2t$ output bundles will be non-zero, even if all input bundles were non-zero. Since these non-zero bundles are the source of the active S-boxes, this guarantee allows us to show that the number of active S-boxes increases exponentially with the number of rounds.

Construction of π. Recall that the requirement for the bundle permutation π in AES is that all bundles in each *single* group (of size t) are moved into t distinct groups by π. We instead view the groups as being collected into consecutive sets, and generalize the requirement by requiring that π moves all bundles in each *set* of groups into distinct groups (the size of these sets will grow with each round). As with ϕ, we necessarily allow π to differ between rounds and we let $\pi^{(j)}$ denote the instance in round j. We will require that each $\pi^{(j)}$ acts as a collection of parallel *transpose permutations*, defined as follows.

Definition 3.5. Let $c, d \in \mathbb{N}$ such that $c \mid d$. A permutation $\tau : \{0, \ldots, d-1\} \to \{0, \ldots, d-1\}$ is a *c-transpose permutation* if it is computed as follows:

1. The elements of $\{0, \ldots, d-1\}$ are placed row-wise into a $c \times (d/c)$ matrix T; i.e., $T_{i,j} = i \cdot (d/c) + j$, where $i \in [0, c-1]$ and $j \in [0, (d/c)-1]$.
2. The elements of $\{0, \ldots, d-1\}$ are extracted column-wise from T; i.e., $\tau(k) := T_{i,j}$ for $i = k \pmod c$ and $j = \lfloor k/c \rfloor$.

More concisely, τ is a c-transpose permutation if $\tau(k) = (k \pmod c) \cdot (d/c) + \lfloor k/c \rfloor$.

Because of the growing state size, $\pi^{(j)}$ acts on a set of $2^{j-1} \cdot n/b$ bundles which are divided into $n/(bt)$ groups of size $2^{j-1}t$. Let $g := n/(bt)$ denote the number of groups (which does not vary across rounds), and let $P^{(j)} = (P_1^{(j)}, \ldots, P_{g/t^{j-1}}^{(j)})$ be the canonical partition[3] of the g groups into sets of size t^{j-1}. Note that each $P_i^{(j)}$ contains t^{j-1} groups, and each group contains $2^{j-1}t$ bundles. Furthermore, for $P^{(j)}$ to be well-defined we must (and will) ensure that t^{j-1} always divides g. We say that a bundle is *encompassed by* $P_i^{(j)}$ if it is contained within a group in $P_i^{(j)}$.

Our requirement on $\pi^{(j)}$ can now be formally stated as follows.

Requirement 3.6. When restricted to the $2^{j-1}t^j$ bundles encompassed by any $P_i^{(j)} \in P^{(j)}$, $\pi^{(j)}$ is a $(2^{j-1}t)$-transpose permutation. In particular, $\pi^{(j)}$ acts separately on each $P_i^{(j)}$.

We actually only require that this holds for rounds $j \in \{2, \ldots, r-1\}$, and we let $\pi^{(1)}$ and $\pi^{(r)}$ each be the identity permutation. Note that with the combination of Definition 3.5 and Requirement 3.6, $\pi^{(j)}$ is now completely specified for all rounds j.

Before discussing the hardness of this candidate, we note that this construction of π satisfies the requirement that each output bundle be connected to $\Omega(n/b)$ input bundles in the layered connection graph specified by π (cf. discussion below Definition 2.3). Indeed, the edges between layers $2, \ldots, r-1$ form a *butterfly network* (cf. [MU05, §4.5.2]), and thus every output bundle is connected to every input bundle.

[3]Meaning that $P_1^{(j)}$ contains the first t^{j-1} groups, $P_2^{(j)}$ contains the next t^{j-1} groups, and so on.

Putting it together. We now define our parameters and prove the lower bound on $\mathcal{A}$. Let $b, r \in \mathbb{N}$ be arbitrary for now, and set $t := b$ and $n := t^r$. (This indeed ensures that t^{j-1} divides $g = n/(bt)$ for $j \in \{2, \ldots, r-1\}$, which is necessary for our construction of π.) Let $\mathcal{F}_3 : \{0,1\}^n \to \{0,1\}^{2^r n}$ be defined by the above choices of ϕ and π.

Theorem 3.7. *For the function $\mathcal{F}_3$, the number $\mathcal{A}$ of active S-boxes is bounded by*

$$\mathcal{A} \;\geq\; \sum_{j=1}^{r-1} \left(t^j \cdot \prod_{i=1}^{j-1} 2^i \right).$$

In particular, $\mathcal{A} > 4t^{r-1}$.

Proof. Let $x \in \{0,1\}^n$ be any non-zero input to $\mathcal{F}_3$. x must contain at least one active (non-zero) bundle; let $i \leq g$ be the value such that this active bundle is encompassed by $P_i^{(1)}$. Then, after the application of $\phi^{(1)}$, $P_i^{(1)}$ will encompass at least t active bundles. In the remainder of the proof, for readability we will use $\sum[\ell]$ to denote the value $\sum_{i=1}^{\ell} i$.

We argue inductively that for $1 \leq j \leq r-1$, at the end of round j some set in $P^{(j)}$ encompasses at least $2^{\sum[j-1]} t^j$ active bundles. The base case $j = 1$ is established by the preceding paragraph. Assume the claim holds up to $j-1$, and let $i \leq g/t^{j-2}$ be the value such that $P_i^{(j-1)}$ encompasses at least $2^{\sum[j-2]} t^{j-1}$ active bundles. The relationship between sets in $P^{(j-1)}$ and sets in $P^{(j)}$ can be expressed as follows:

$$\forall k \leq \frac{g}{t^{j-1}} : \quad P_k^{(j)} = \bigcup_{\ell=(k-1)t+1}^{kt} P_\ell^{(j-1)}.$$

Therefore, let k be the value such that $P_i^{(j-1)} \subset P_k^{(j)}$. Then, when $\pi^{(j)}$ acts on $P_k^{(j)}$, it will move each of the active bundles encompassed by $P_i^{(j-1)}$ into a distinct group of size $2^{j-1}t$ (this follows directly from Definition 3.5 and the fact that $\pi^{(j)}$ satisfies Requirement 3.6). Finally, after the application of $\phi^{(j)}$, each group that contained one of these active bundles will contain at least $2^{j-1}t$ active bundles, and so $P_k^{(j)}$ encompasses at least $2^{j-1}t \cdot 2^{\sum[j-2]} t^{j-1} = 2^{\sum[j-1]} t^j$ active bundles as claimed.

Summing the number of active bundles at the end of rounds 1 through $r-1$ gives the statement of the theorem. □

Efficiency. We now show that $\mathcal{F}_3$ can be computed by TC^0 circuits of size $O(n^{1+\epsilon})$. We first observe that each component of $\mathcal{F}_3$ is computable by polynomial-size TC^0 circuits, and thus there exists a constant C such that round j can be computed by a TC^0 circuit with at most $n \cdot (2^j bt)^C$ wires. Indeed, the only "tricky" component is exponentiation in $\mathrm{GF}(2^b)$, but as mentioned in §1.2 this can be done by hardwiring $x, x^2, \ldots, x^{2^{b-1}}$ into the circuit. Therefore, $\mathcal{F}_3$ is computable by a threshold circuit of depth $d = O(r)$ and total size $w \leq (r2^{rC}) \cdot n \cdot (bt)^C$.

Now fix any $\epsilon > 0$. Let $r := \lceil 2C/\epsilon \rceil$. Let $b \in \mathbb{N}$ be sufficiently large, and set $t := b$ and $n := b^r$. Then, we have $w \leq (r2^{rC}) \cdot n^{1+\epsilon} = O(n^{1+\epsilon})$ as claimed. Furthermore, this is indeed a TC^0 circuit because $r = O(1)$.

Hardness. We have $b^r = n$ by construction. By Theorem 3.7 and the fact that $b = t$, we have $\mathcal{A} \cdot (b/2 - 1) > 2t^r - 4t^{r-1} = 2n - 4n^{1-1/r} \geq n$ for sufficiently large n.

Finally, we wish to note that an arguably simpler construction for $\mathcal{F}_3$ would be to let ϕ compute an $n \to O(n)$ error correcting code with minimum distance $\Omega(n)$, apply ϕ to the entire state at each round, and eliminate π all together. As such codes can be computed by TC^0 circuits of size $O(n^{1+\epsilon})$ (details omitted), this would preserve both the circuit size and the hardness properties of the function presented here. However, we feel that the version detailed above better preserves the "spirit" of AES, and in particular Requirement 3.6 is a natural generalization of AES's concept of diffusion optimality (Definition 2.3).

3.4 Candidate 4

As our final generalization of AES, we choose the extreme setting of $b = n$ and $t = r = 1$, which means that the function is computed over one round and essentially consists of just a single S-box. More specifically, the function is indexed by a seed $(k_0, k_1) \in \{0,1\}^{2n}$, and is computed as

$$\mathcal{F}_4(x) := \gamma(x \oplus k_0) \oplus k_1.$$

Recall that γ consists of patched inversion in $GF(2^n)$ followed by an invertible affine transformation over $GF(2)^n$. So, letting $L(\cdot)$ denote the affine transformation, we can write

$$\mathcal{F}_4(x) = L\left((x + k_0)^{2^n - 2}\right) + k_1.$$

Though $\mathcal{F}_4$ does indeed preserve resistance to differential and linear cryptanalysis, we note that the seed can be recovered with four known plaintext/ciphertext pairs, using an attack similar in spirit to the so-called interpolation attack of [JK01].

Claim. *Let $\mathcal{F}_4$ be the above function indexed by $k_0, k_1 \in \{0,1\}^n$. Let $\{(p_i, c_i)\}_{1 \leq i \leq 4}$ be any set such that $c_i = \mathcal{F}_4(p_i)$ for all i and $p_i \neq p_j$ for $i \neq j$. Then, with probability $(1 - 1/2^{n-2})$ over k_0, the values of k_0 and k_1 can be recovered from $\{(p_i, c_i)\}_i$.*

Proof. The attack is performed by using the four pairs to create two equations over $GF(2^n)$ that are linear in the seed, as follows. Assume that $k_0 \notin \{p_i\}_i$, which happens with probability $(1 - 1/2^{n-2})$. Then the equation

$$L^{-1}(c_i + k_1) \cdot (p_i + k_0) = 1$$

holds for $1 \leq i \leq 4$. Let $\overline{c_i} := L^{-1}(c_i)$ and $\overline{k_1} := L^{-1}(k_1)$. We can rewrite the equations as

$$k_0\overline{k_1} + \overline{c_i}k_0 + p_i\overline{k_1} + \overline{c_i}p_i = 1. \tag{1}$$

If we sum (1) for $i = 1, 2$, the quadratic terms cancel and we obtain

$$(\overline{c_1} + \overline{c_2})k_0 + (p_1 + p_2)\overline{k_1} + (\overline{c_1}p_1 + \overline{c_2}p_2) = 0.$$

Summing (1) for $i = 3, 4$ gives another linear equation in $k_0, \overline{k_1}$. The attack concludes by solving the two linear equations, and then applying L to $\overline{k_1}$ to obtain the desired seed. □

We now consider a slight modification to $\mathcal{F}_4$ that is not susceptible to this simple attack, and furthermore is a small-bias generator. The new function $\mathcal{F}_5 : \{0,1\}^n \to \{0,1\}$ is defined as follows:

$$\mathcal{F}_5(x) := \left\langle (x + k_0)^{2^n-2}, k_1 \right\rangle .$$

In other words, we combine the AES S-box (minus the affine transformation) with the Goldreich-Levin hardcore predicate [GL89]. Note that we now output only a single bit.

The next theorem shows that $\mathcal{F}_5$ is a small-bias generator. This result is reminiscent of the "exponentiation" small-bias generator in [AGHP92], where the x-th output bit is $\langle k_0^x, k_1 \rangle$. Indeed, our proof is inspired by theirs. However we face the extra difficulty that the polynomials we work with are not of low degree.

Theorem 3.8. *For any choice of $d \leq 2^n$, $\mathcal{F}_5$ is a d-wise small-bias generator with error $d/2^n$: for any distinct $a_1, \ldots, a_d \in \{0,1\}^n$,*

$$\left| \Pr_{k_0,k_1} \left[\sum_{i=1}^{d} \mathcal{F}_5(a_i) = 0 \right] - \frac{1}{2} \right| < \frac{d}{2^n}.$$

Proof. Fix any distinct choices of $a_1, \ldots, a_d$. Then, identifying elements of $GF(2^n)$ with elements of $\{0,1\}^n$, we have

$$\begin{aligned} \sum_{i \leq d} \mathcal{F}_5(a_i) &= \sum_{i \leq d} \left\langle (a_i + k_0)^{2^n-2}, k_1 \right\rangle \\ &= \left\langle p(x) := \sum_{i \leq d} (a_i + k_0)^{2^n-2},\ k_1 \right\rangle . \end{aligned}$$

We now show that the polynomial $p(x) = \sum_{i \leq d} (a_i + x)^{2^n-2}$ has at most $2d - 1$ distinct roots. This will conclude the proof because when k_0 is not a root of $p(x)$, we have $\Pr_{k_1}[\langle p(k_0), k_1 \rangle = 0] = 1/2$. Therefore,

$$\left| \Pr_{k_0,k_1} \left[\sum_{i=1}^{d} \mathcal{F}_5(a_i) = 0 \right] - \frac{1}{2} \right| \leq \frac{1}{2} \Pr_{k_0}[p(k_0) = 0] < \frac{d}{2^n}.$$

To show the bound on the number of roots, define the following polynomials:

$$\begin{aligned} \overline{p}(x) &:= p(x) \cdot \prod_{i \leq d} (a_i + x) = \sum_{i \leq d} \left[(a_i + x)^{2^n-1} \prod_{j \neq i} (a_j + x) \right], \\ \overline{p}_*(x) &:= \sum_{i \leq d} \prod_{j \neq i} (a_j + x). \end{aligned}$$

Observe that any root y of $p(x)$ is also a root of $\overline{p}(x)$. Moreover, note for any $y \notin \{a_j : j \leq d\}$, $\overline{p}(y) = \overline{p}_*(y)$, using the identity $y^{2^b-2} = 1$ valid for any $y \neq 0$.

Also observe that $\overline{p}_*(x)$ is not identically zero. Indeed, by inspection, the constant term of the polynomial $\overline{p}_*(x+a_1)$ is $\prod_{j\neq 1}(a_j+a_1)$, which is non-zero because the a_j are distinct; therefore $\overline{p}_*(x+a_1)$ is not identically zero, and so neither is $\overline{p}_*(x)$. Since $\overline{p}_*(x)$ is a non-zero polynomial of degree $d-1$, it has at most $d-1$ distinct roots.

So, if $p(x)$ has r roots, also $\overline{p}$ has r roots. At least $r-d$ of these do not belong to $\{a_j : j \leq d\}$, and so they are also roots of $\overline{p}_*(x)$. Therefore, $r-d \leq d-1$, or $r \leq 2d-1$. □

By Braverman's result [Bra09] (cf. [Baz09, Raz09]), we obtain that this candidate also fools small-depth AC^0 circuits of any size $w = 2^{n^{o(1)}}$ (that look at only w fixed output bits of the candidate).

Indeed, fix any function $w = 2^{n^{o(1)}}$ and any constant $d = O(1)$; let $N := 2^n$. By Theorem 3.8, any w output bits have bias $< w/N$. By [AGM03], for any $k \leq w$, the output distribution on those w bits is $w^k w/N$-close to a k-wise independent distribution. By [Bra09], $k = \lg^{O(d^2)} w \leq n^{o(1)}$ is sufficient to fool circuits of depth d with error $1/w$. Hence the overall error will be $1/w = 1/2^{n^{o(1)}}$ plus

$$\frac{w^k w}{N} = \frac{\left(2^{n^{o(1)}}\right)^{n^{o(1)}}}{N} \leq \frac{1}{\sqrt{N}},$$

for a total of $1/w + 1/\sqrt{N} = O(1/w)$.

Efficiency. As noted in §1.2, $\mathcal{F}_5$ is computable by Boolean circuits of size $\widetilde{O}(n^2)$ and TC^0 circuits of size $n^{O(1)}$.

4 Conclusion

We believe a good candidate PRF should be the *simplest* candidate that resists known attacks. As noted in [DR02], some of the choices in the design of AES are not motivated by any known attack, but are there as a safeguard (for example, one can reduce the number of rounds and still no attack is known). While this is comprehensible when having to choose a standard that is difficult to change, one can argue that a better way to proceed is to put forth the *simplest* candidate PRF, possibly break it, and iterate until hopefully converging to a PRF. We view this paper as a step in this direction.

Abstracting from the design of AES, one may arrive to the following paradigm for constructing PRF: alternate the application of (1) an error-correcting code and (2) a bundle-wise application of any local map that has high degree over GF(2) and resists attacks corresponding to linear and differential cryptanalysis.

This viewpoint may lead to a PRF candidate computable in ACC^0, since for (1) one just needs parity gates, while, say, taking parities of suitable mod 3 maps one should get a map that satisfies (2). However a good choice for this latter map is not clear to us at this moment.

Another direction is to obtain candidate PRF starting from other block-ciphers. We focus in this work on AES because it is widespread, and so its hardness seems more accepted than for other block-ciphers.

Acknowledgments. We thank Guevara Noubir for helpful discussions, and Salil Vadhan for mentioning AES.

References

[ABW10] Benny Applebaum, Boaz Barak, and Avi Wigderson. Public-key cryptography from different assumptions. In *Symposium on the Theory of Computing (STOC)*, pages 171–180, 2010.

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *CRYPTO*, pages 595–618, 2009.

[AGHP92] Noga Alon, Oded Goldreich, Johan Håstad, and René Peralta. Simple constructions of almost k-wise independent random variables. *Random Structures & Algorithms*, 3(3):289–304, 1992.

[AGM03] Noga Alon, Oded Goldreich, and Yishay Mansour. Almost k-wise independence versus k-wise independence. *Inf. Process. Lett.*, 88(3):107–110, 2003.

[AIK06] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptography in NC^0. *SIAM J. Comput.*, 36(4):845–888, 2006.

[AIK08] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. On pseudorandom generators with linear stretch in NC^0. *Computational Complexity*, 17(1):38–69, 2008.

[AK10] Eric Allender and Michal Koucký. Amplifying lower bounds by means of self-reducibility. *J. ACM*, 57(3), 2010.

[App11] Benny Applebaum. Pseudorandom generators with long stretch and low locality from random local one-way functions. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:7, 2011.

[AW08] Scott Aaronson and Avi Wigderson. Algebrization: a new barrier in complexity theory. In *40th ACM Symposium on the Theory of Computing (STOC)*, pages 731–740, 2008.

[Baz09] Louay M. J. Bazzi. Polylogarithmic independence can fool dnf formulas. *SIAM J. Comput.*, 38(6):2220–2272, 2009.

[BCK96] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Pseudorandom functions revisited: The cascade construction and its concrete security. In *Symposium on Foundations of Computer Science (FOCS)*, pages 514–523, 1996.

[BGS75] Theodore Baker, John Gill, and Robert Solovay. Relativizations of the P=?NP question. *SIAM J. Comput.*, 4(4):431–442, 1975.

[Bra09] Mark Braverman. Poly-logarithmic independence fools AC^0 circuits. In *24th Conference on Computational Complexity (CCC)*. IEEE, 2009.

[BS91] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4(1):3–72, 1991.

[CMR06] Carlos Cid, Sean Murphy, and Matthew Robshaw. *Algebraic Aspects of the Advanced Encryption Standard*. Springer, 2006.

[DR02] Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Springer Verlag, 2002.

[Ger88] A. Gerasoulis. A fast algorithm for the multiplication of generalized Hilbert matrices with vectors. *Mathematics of Computation*, 50:179–188, 1988.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, October 1986.

[GL89] Oded Goldreich and Leonid Levin. A hard-core predicate for all one-way functions. In *21st ACM Symposium on the Theory of Computing (STOC)*, pages 25–32, 1989.

[Gol00] Oded Goldreich. Candidate one-way functions based on expander graphs. Technical report, Electronic Colloquium on Computational Complexity, 2000.

[Gol01] Oded Goldreich. *Foundations of Cryptography: Volume 1, Basic Tools*. Cambridge University Press, 2001.

[GvzGPS00] Shuhong Gao, Joachim von zur Gathen, Daniel Panario, and Victor Shoup. Algorithms for exponentiation in finite fields. *J. Symb. Comput.*, 29(6):879–889, 2000.

[HAB02] William Hesse, Eric Allender, and David A. Mix Barrington. Uniform constant-depth threshold circuits for division and iterated multiplication. *J. Comput. System Sci.*, 65(4):695–716, 2002. Special issue on complexity, 2001 (Chicago, IL).

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

[HRV10] Iftach Haitner, Omer Reingold, and Salil P. Vadhan. Efficiency improvements in constructing pseudorandom generators from one-way functions. In *42nd ACM Symposium on the Theory of Computing (STOC)*, pages 437–446, 2010.

[HV06] Alexander Healy and Emanuele Viola. Constant-depth circuits for arithmetic in finite fields of characteristic two. In *23rd Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 672–683. Springer, 2006.

[IKOS08] Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptography with constant computational overhead. In *40th ACM Symposium on the Theory of Computing (STOC)*, pages 433–442, 2008.

[IPS97] Russell Impagliazzo, Ramamohan Paturi, and Michael E. Saks. Size-depth tradeoffs for threshold circuits. *SIAM J. Comput.*, 26(3):693–707, 1997.

[JK01] T. Jakobsen and L.R. Knudsen. Attacks on block ciphers of low algebraic degree. *Journal of Cryptology*, 14:197–210, 2001.

[KN97] Eyal Kushilevitz and Noam Nisan. *Communication complexity*. Cambridge University Press, 1997.

[Kop11] Swastik Kopparty. On the complexity of powering in finite fields. In *Symposium on the Theory of Computing (STOC)*, 2011.

[Lev87] Leonid A. Levin. One way functions and pseudorandom generators. *Combinatorica*, 7(4):357–363, 1987.

[Mat94] M. Matsui. Linear cryptanalysis method for DES cipher. In T. Helleseth, editor, *Advances in Cryptology, Proc. Eurocrypt'93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer-Verlag, 1994.

[MU05] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algo-*

rithms and Probabilistic Analysis. Cambridge University Press, 2005.

[NN93] Joseph Naor and Moni Naor. Small-bias probability spaces: efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

[NR99] Moni Naor and Omer Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *J. Comput. Syst. Sci.*, 58(2):336–375, 1999.

[NR04] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004.

[NRR02] Moni Naor, Omer Reingold, and Alon Rosen. Pseudorandom functions and factoring. *SIAM J. Comput.*, 31(5):1383–1404, 2002.

[Nyb93] Kaisa Nyberg. Differentially uniform mappings for cryptography. In *EUROCRYPT*, pages 55–64, 1993.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *Symposium on the Theory of Computing (STOC)*, pages 333–342, 2009.

[Pie91] Josef Pieprzyk. On bent permutations. In *Proceedings of the International Conference on Finite Fields, Coding Theory, and Advances in Communications and Computing, Las Vegas*, August 1991.

[Raz09] Alexander A. Razborov. A simple proof of Bazzi's theorem. *ACM Transactions on Computation Theory (TOCT)*, 1(1), 2009.

[RR97] Alexander Razborov and Steven Rudich. Natural proofs. *Journal of Computer and System Sciences*, 55(1):24–35, August 1997.

[RS85] Ron M. Roth and Gadiel Seroussi. On generator matrices of MDS codes. *IEEE Transactions on Information Theory*, 31:826–830, 1985.

[vzGG03] Joachim von zur Gathen and Jürgen Gerhard. *Modern Computer Algebra*. Cambridge University Press, 2003.

[Wil11] Ryan Williams. Non-uniform ACC lower bounds. In *Conference on Computational Complexity (CCC)*, 2011.

A Distinguishing $o(n)$-degree PRFs

In this section, we show that any PRF $f_k : \{0,1\}^n \to \{0,1\}$ which is computable by an $o(n)$-degree polynomial over GF(2) cannot have hardness 2^n. This just follows from the fact that in time 2^n one can write down the polynomial representation of f restricted to $\Omega(n)$ input bits. Details follow.

For simplicity, we instead show that any such PRF can be broken in time $2^{O(n)}$. This implies the desired goal, for if we had a PRF $f_k : \{0,1\}^n \to \{0,1\}$ with hardness 2^n we could consider it over bn input bits, note that the degree would still be $o(n) = o(bn)$, and obtain a contradiction.

To start, let $f : \{0,1\}^n \to \{0,1\}$ be any function, and define the following three values:

- $T_f \in \{0,1\}^{2^n}$ is the truth table of f; i.e. $(T_f)_i := f(i)$, identifying a natural number with its binary representation.
- $C_f \in \{0,1\}^{2^n}$ is the coefficient vector of f, defined as follows. Fix some ordering on the

2^n possible multilinear monomials in n variables. Then, $(C_f)_i = 1$ iff the ith monomial appears in the polynomial representation of f over GF(2).

- $A \in \{0,1\}^{2^n \times 2^n}$ is the matrix with rows indexed by the set $\{0,1\}^n$ and columns indexed by the set of degree $\leq n$ multilinear monomials (as with C_f), defined by $A_{ij} := 1$ iff monomial j has value 1 under input i.

Note that A is independent of the function f. Furthermore, A is invertible because it has full rank, which follows from the fact that any two linear combinations of A's columns give the truth tables of two distinct polynomials. We now show how to distinguish a low-degree PRF using the fact that $A \cdot C_f = T_f$ for all f.

Theorem A.1. *Let $\{f_k : \{0,1\}^n \to \{0,1\}\}_k$ be a PRF such that, for each key k, the polynomial representation of f_k over GF(2) has degree $o(n)$. Then, there is an adversary that runs in time $\leq 2^{O(n)}$ and distinguishes f_k from random with advantage $\geq 1 - 2^{-2^{\Omega(n)}}$.*

Proof. For any function $f : \{0,1\}^n \to \{0,1\}$, we can use C_f to check if the polynomial representation of f contains a monomial of degree $\geq n/2$. Clearly this will be false for any f_k drawn from the PRF, and for a uniformly random function F we have

$$\Pr_F[F \text{ has a monomial of degree } \geq n/2] \;\geq\; 1 - 2^{-\binom{n}{n/2}} \;\geq\; 1 - 2^{-2^{\Omega(n)}}$$

which can be seen by viewing F as being randomly chosen by including each possible monomial independently with probability 1/2. Finally, note that C_f can be computed from the truth table of f in time $2^{O(n)}$ as $C_f = A^{-1} \cdot T_f$. □

ECCC ISSN 1433-8092

http://eccc.hpi-web.de