On the Concrete Security of Goldreich's Pseudorandom Generator

Geoffroy Couteau[1], Aurélien Dupin[2,3], Pierrick Méaux[4], Mélissa Rossi[2,5,6], and Yann Rotella[6]

[1] Karlsruhe Institute of Technology, Karlsruhe, Germany
geoffroy.couteau@kit.edu
[2] Thales, Gennevilliers, France
[3] CentraleSupélec, Rennes, France and Irisa, Rennes, France
dupin.aurelien@gmail.com
[4] ICTEAM/ELEN/Crypto Group, Université catholique de Louvain, Belgium
pierrick.meaux@uclouvain.be
[5] École Normale Supérieure de Paris, Département d'informatique,
CNRS, PSL Research University, Paris, France
melissa.rossi@ens.fr
[6] Inria, Paris, France.
yann.rotella@inria.fr

Abstract. Local pseudorandom generators allow to expand a short random string into a long pseudorandom string, such that each output bit depends on a constant number d of input bits. Due to its extreme efficiency features, this intriguing primitive enjoys a wide variety of applications in cryptography and complexity. In the polynomial regime, where the seed is of size n and the output of size $n^{\mathtt{s}}$ for $\mathtt{s} > 1$, the only known solution, commonly known as *Goldreich's PRG*, proceeds by applying a simple d-ary predicate to public random size-d subsets of the bits of the seed.

While the security of Goldreich's PRG has been thoroughly investigated, with a variety of results deriving provable security guarantees against class of attacks in some parameter regimes and necessary criteria to be satisfied by the underlying predicate, little is known about its concrete security and efficiency. Motivated by its numerous theoretical applications and the hope of getting practical instantiations for some of them, we initiate a study of the concrete security of Goldreich's PRG, and evaluate its resistance to cryptanalytic attacks. Along the way, we develop a new guess-and-determine-style attack, and identify new criteria which refine existing criteria and capture the security guarantees of candidate local PRGs in a more fine-grained way.

Keywords: Pseudorandom generators, Algebraic attacks, Guess-and-Determine, Gröbner basis.

1 Introduction

One of the most fundamental problems in cryptography is the question of what makes an efficiently computable function hard to invert. The quest for the simplest design which leads to a primitive resisting all known attacks is at the heart of both symmetric and asymmetric cryptography: while we might be able to build seemingly secure primitives by relying on more and more complex designs to thwart cryptanalysis attempts, such a "security by obscurity" approach is unsatisfying. Instead, as advocated almost two decades ago by Goldreich [Gol00], we should seek to construct the simplest possible function that we do not know how to invert efficiently. Only this way, Goldreich argued, can we better understand what really underlies the security of cryptographic constructions.

Random Local Functions. In an attempt to tackle this fundamental problem, Goldreich suggested a very simple candidate one-way function as a promising target for cryptanalysis: let (n, m) be integers, and let $(\sigma^1, \ldots, \sigma^m)$ be a list of m subsets of $[n]$, such that each subset is of small size: for any $i \leq m$, $|\sigma^i| = d(n)$, where $d(n) \ll n$ (in actual instantiations, $d(n)$ can for example be logarithmic in n, or even constant). Fix a simple predicate $P : \{0,1\}^{d(n)} \mapsto \{0,1\}$, and define the function $f : \{0,1\}^n \mapsto \{0,1\}^m$ as follows: on input $x \in \{0,1\}^n$, for any subset S of $[n]$, let $x[\sigma]$ denote the subset of the bits of x indexed by σ. Compute $f(x)$ as $P(x[\sigma^1])||\cdots||P(x[\sigma^m])$ (that is, $f(x)$ is computed by applying the predicate P to all subsets of the bits of x indexed by the sets $\sigma^1, \ldots, \sigma^m$). We call *random local functions* the functions obtained by instantiating this template.

In his initial proposal, Goldreich advocated instantiating the above methodology with $m \approx n$ and $d(n) = O(\log n)$, and conjectured that if the subsets $(\sigma^1, \ldots, \sigma^m)$ form an expander graph[1], and for an appropriate choice of the predicate P, it should be infeasible to invert the above function f in polynomial time. While setting $d(n)$ to $O(\log n)$ offers stronger security guarantees, the more extreme design choice $d(n) = O(1)$ (also discussed in Goldreich's paper) enhances the above candidate with an appealing feature: it enjoys constant input locality (which puts it into the complexity class NC^0), hence it is highly parallelizable (it can be computed in constant parallel time). It appeared in subsequent works that a stronger variant of Goldreich's conjecture, which considers $m \gg n$ and claims that f is in fact a *pseudorandom generator*, was of particular interest; we will elaborate on this later on.

Local Pseudorandom Generators. The question of whether cryptographic primitives can exist in weak complexity classes such as NC^0 has attracted a lot of attention in the cryptographic community. A primitive of particular interest, which has been the focus of most works on the subject, is the notion of pseudorandom generators (PRGs), which are functions $G : \{0,1\}^n \mapsto \{0,1\}^m$ extending a short random seed into a longer, pseudorandom string. The existence of PRGs in NC^0 was first considered by Cryan and Miltersen in [CM01]. Remarkably, it was shown by Applebaum, Ishai, and Kushilevitz [AIK04, AIK08] that cryptographically secure pseudorandom generators (with linear stretch $m = O(n)$) exist in a complexity class as low as NC^0_4 (the class of constant depth, polysize circuits where each output bit depends on at most 4 input bits), under widely believed standard assumption for the case of PRG with sublinear stretch (such as factorization, or discrete logarithm), and under a specific intractability assumption related to the hardness of decoding "sparsely generated" linear codes, for the case of PRG with linear stretch. While this essentially settled the question of the existence of linear stretch PRGs in NC^0, an intriguing open question remained: could PRGs in NC^0 have *polynomial* stretch, $m = \mathsf{poly}(n)$?

Some early negative results were given by Cryan and Miltersen [CM01] (who ruled out the existence of PRGs in NC^0_3 with stretch $m > 4n$) and Mossel, Shpilka, and Trevisan [MST03] (who ruled out the existence of PRGs in NC^0_4 with stretch $m > 24n$). The authors of [CM01] also conjectured that any candidate PRG with superlinear stretch in NC^0 would be broken by simple, linear distinguishing tests[2]; this conjecture was refuted in [MST03], who gave a concrete candidate PRG in NC^0, by instantiating a random local function with $d = 5$, and the predicate

$$P_5 : (x_1, x_2, x_3, x_4, x_5) \mapsto x_1 + x_2 + x_3 + x_4 x_5 \ .$$

where the $+$ denotes the addition in $\mathbb{F}_2$ *i.e.* the xor.

They proved that this PRG fools linear tests, even when m is a (sufficiently small) polynomial in n. By the previously mentioned negative result on PRGs in NC^0_4, this candidate PRG, which has locality 5, achieves the best possible locality. Recently, there has been a renewed interest in the study of this local PRG, now commonly known as Goldreich's PRG, and its generalizations [BQ09, App12, OW14, CEMT14, App15, ABR16, AL16, IPS08, LV17, BCG+17].

1.1 Implications of Polynomial-Stretch Local Pseudorandom Generators

The original motivation for the study of local pseudorandom generators was the intriguing possibility of designing cryptographic primitives that can be evaluated in *constant time*, using polynomially many cores. While this is already a strong motivation in itself, it was observed in several works that the existence of (poly-stretch) local PRGs had a number of non-trivial implications, and is at the heart of feasibility results for several high-end cryptographic primitives. We provide below a brief overview.

- *Secure computation with constant computational overhead.* In the recent work [IKOS08], the authors explored the possibility of computing cryptographic primitives with essentially optimal efficiency, namely, constant overhead over a naive insecure implementation of the same task. One

[1] The subsets form an expander graph if for some k, every k subsets cover $k + \Omega(n)$ elements of $[n]$. In practice, it suffices to pick once for all the subsets $(\sigma^1, \ldots, \sigma^m)$ at random to guarantee that they will be expanding except with $o(1)$ probability.

[2] A linear test attempts to distinguish a string from random by checking whether the xor of a subset of the bits of the string is biased toward either 0 or 1.

of their main results establishes the existence of constant-overhead two-party computation protocols for any boolean circuit, assuming the existence of poly-stretch local PRGs (and oblivious transfers). In a recent work [ADI+17a], this result was extended to arithmetic circuits, using an arithmetic generalization of local PRGs.

- *Indistinguishability obfuscation (iO).* Introduced in the seminal paper of Barak et al. [BGI+01], iO is a primitive that has received a considerable attention from the crypto community in the past years, as a long sequence of works starting with [SW14] has demonstrated that iO had tremendous theoretical implications, to the point that it is often referred to as being a "crypto-complete" primitive. All known candidate constructions of iO rely, directly or indirectly, on a primitive called k-linear map, for some degree k. Recently, a sequence of papers (culminating with [LT17]) has attempted to find out the minimal k for which a k-linear map would imply the existence of iO (with the ultimate goal of reaching $k = 2$, as bilinear maps are well understood objects). These works have established a close relation between this value k and the existence of pseudorandom generators with poly-stretch, and locality k.[1]
- *MPC-friendly primitives.* Historically, the design of symmetric cryptographic primitives (such as block ciphers, pseudorandom generators, and pseudorandom functions) has been motivated by efficiency considerations (memory consumption, hardware compatibility, ease of implementation,...). The field of multiparty computation (MPC), where parties want to jointly evaluate a function on secret inputs, has led to the emergence of new efficiency considerations: the efficiency of secure evaluation of symmetric primitives is strongly related to parameters such as the circuit depth of the primitive, and the number of its AND gates. This observation has motivated the design of MPC-friendly symmetric primitives in several recent works (*e.g.* [ARS+15, CCF+16, MJSC16, GRR+16]). Local pseudorandom generators make very promising candidate MPC-friendly PRGs (and lead, through the GGM transform [GGM84], to promising candidates for MPC-friendly pseudorandom functions). Secure evaluation of such symmetric primitives enjoys a wide variety of applications.
- *Cryptographic capsules.* In [BCG+17], Boyle et al. studied the recently introduced primitive of homomorphic secret sharing (HSS). An important implication of HSS is that, assuming the existence of a local PRG with poly-stretch, one can obtain multiparty computation protocols in the preprocessing model[2] where the amount of communication between the parties is considerably smaller than the circuit size of the function, by constructing a primitive called cryptographic capsule which, informally, allows to compress correlated (pseudo-)random coins. MPC protocols with low-communication preprocessing have numerous appealing applications; however, the efficiency of the constructions of cryptographic capsule strongly depends on the locality and seed size of the underlying local PRG (both should be as small as possible to get a reasonably efficient instantiation).

In addition to the above (non-exhaustive) overview, we note that the existence of poly-stretch local pseudorandom generators also enjoys interesting complexity-theoretic implications. For example, they have been shown in [AIK08] to imply strong (tight) bounds on the average-case inapproximability of constraints satisfactions problems such as Max3SAT.

1.2 On the Security of Goldreich's PRG

In this section, we provide a brief overview of the state-of-the-art regarding the security of local pseudorandom generators. For a more detailed and well-written overview dating from 2015, we refer the reader to [App15].

Positive Results: Security against Class of Attacks. The seminal paper of Goldreich [Gol00] made some preliminary observations on necessary properties for a local one-way function. Namely, the

[1] The locality requirement can in fact be weakened to a related notion of *block locality*.

[2] In this model, n parties securely compute a function f on private inputs $(x_1, \ldots, x_n)$; in the preprocessing phase, the parties have access to f (but not to the input), and generate some preprocessing material. Then, in the online phase, the parties execute an *information-theoretically secure* protocol to compute $f(x)$, using the preprocessed material. MPC protocols in the preprocessing model are among the most promising candidates for getting practical solutions to the multiparty computation problem.

predicate P must satisfy some non-degeneracy properties, such as being non-linear (otherwise, one could inverse the function using Gaussian elimination). It also noted that to avoid a large class of natural "backtracking" attacks, which make a guess on the values of bit inputs based on local observations and attempt to combine many local solutions into a global solution, the subsets $(S_1, \ldots, S_m)$ should be sufficiently *expanding*: for some k, every k subsets should cover $k + \Omega(n)$ elements of $[n]$. The security of Goldreich's candidate one-way function against a large class of backtracking algorithm was formally analyzed in [AHI05, CEMT14], where it was proven that two restricted types of backtracking algorithms (called "drunk" and "myopic" backtracking algorithms) take exponential time to invert the function (with high probability). They also ran experiments to heuristically evaluate its security against SAT solvers (and observed experimentally an exponential increase in running time as a function of the input length).

The pseudorandomness of random local functions was originally analyzed in [MST03]. They proved (among other results) that the random local function instantiated with the predicate P_5 fools $\mathbb{F}_2$-linear distinguishers for a stretch up to $m(n) = n^{1.25-\varepsilon}$ (for an arbitrary small constant ε). This result was later extended to a larger stretch $n^{1.5-\varepsilon}$ in [OW14]. In the same paper, the authors proved that this candidate PRG is also secure against a powerful class of attacks, the Lasserre/Parrilo semidefinite programming (SDP) hierarchy, up to the same stretch. Regarding security against $\mathbb{F}_2$-linear attacks, a general dichotomy theorem was proven in [ABR12], which identified a class of *non-degenerate* predicates and showed that for most graphs, a local PRG instantiated with a non-degenerate predicate is secure against linear attacks, and for most graphs, a local PRG instantiated with a degenerate predicate is insecure against linear distinguishers. In general, to fool $\mathbb{F}_2$-linear distinguishers, the predicate should have high *algebraic degree* (in particular, a random local function instantiated with a degree-ℓ predicate cannot be pseudorandom for a stretch ℓ ($m \equiv n^\ell$), as it is broken by a straightforward Gaussian elimination attack).

Being pseudorandom seems to be a much stronger security property than being one-way. Nevertheless, in the case of random local functions, it was shown in [App12] that the existence of local pseudorandom generators follows from the existence of *one-way* random local functions (with sufficiently large output size).

Negative Results. The result of O'Donnell and Witmer [OW14] regarding security against SDP attacks is almost optimal, as attacks from this class are known to break the candidate for a stretch $\Theta(n^{1.5} \log n)$. More generally, optimizing SDP attacks leads to a polytime inversion algorithm for any predicate P which is (even slightly) correlated with some number c of its inputs, as soon as the output size exceeds $m \in \Omega(n^{c/2} + n \log n)$ [OW14, App15]. Therefore, a good predicate should have high *resiliency* (*i.e.* it should be k-wise independent, for a k as large as possible). This result shows, in particular, that a random local function with a constant locality d and with an output size $m > \mathsf{poly}(d) \cdot n$ is insecure when instantiated with a uniformly random predicate P. Combining this observation with the result of Siegenthaler [Sie84], which studied the correlation of d-ary predicates, gives a polytime inversion algorithm for any random local function implemented with a d-ary predicate, and with an output size $m \in \Omega(n^{1/2\lfloor 2d/3 \rfloor} \log n)$.

Bogdanov and Qiao [BQ09] studied the security of random local functions when the output is sufficiently larger than the input (*i.e.*, $m \geq Dn$, for a large constant D). They proved that for sufficiently large D, inverting a random local function could be reduced to finding an *approximate inverse* (*i.e.* finding any x' which is close to the inverse x in Hamming distance), by showing how to invert the function with high probability given an advice x' close to x. For random local function with an output size polynomial in n, $m = n^{\mathsf{s}}$ for some s, this leads to a subexponential-time attack [App15]: fix a parameter ε, assign random values to the $(1-2\varepsilon)n$ first inputs, and create a list that enumerates over all possible $2\varepsilon n$ assignments for the remaining variables. Then the list is guaranteed to contain a value x' that agree with the preimage x on a $(1/2 + \varepsilon)n$ fraction of the coordinates with good probability. By applying the reduction of [BQ09], using each element of the list as an advice string, one recovers the preimage in time $\mathsf{poly}(n) \cdot 2^{2\varepsilon n}$ provided that $m = \Omega(n/\varepsilon^{2d})$ (d is the arity of the predicate P). In the case of the 5-ary predicate P_5, this leads to an attack in subexponential-time $2^{O(n^{1-(\mathsf{s}-1)/2d})}$ (*e.g.* using $\mathsf{s} = 1.45$ gives an attack in time $2^{O(n^{0.955})}$).

By the previous observations, we know that the predicate of a random local function must have high resiliency and high algebraic degree to lead to a pseudorandom function. A natural question is whether this characterization is also sufficient; this question was answered negatively in [AL16], who proved that a predicate must also have high *bit-fixing degree* to fool linear attacks.[1] In particular, this observation disproved a previous conjecture of Applebaum that XOR-AND predicates (which are natural generalizations of the predicate P_5) could lead to local PRGs with stretch greater than 2 that fools all linear tests (see [AL16, Corollary 1.3]).

In the same work, Applebaum and Lovett considered the class of algebraic attacks on local pseudorandom function, which are incomparable to linear attacks. An algebraic attack against a function $f : \{0,1\}^n \mapsto \{0,1\}^m$ starts with an output y and uses it to initialize a system of polynomial equations over the input variables $x = (x_1, \ldots, x_n)$. The system is further manipulated and extended until a solution is found or until the system is refuted. Applebaum and Lovett proved that a predicate must also have high *rational degree* to fool algebraic attacks (a predicate P has rational degree e if it is the smallest integer for which there exist degree e polynomials Q and R, not both zero, such that $PQ = R$). Indeed, if $e < \mathsf{s}$ then P is not s-pseudorandom against algebraic attacks (see [AL16], Theorem 1.4). In the symmetric cryptography community, the rational degree denotes the well-known *algebraic immunity* criterion on Boolean function that underlies the so-called *algebraic attacks* on stream ciphers [CM03, Cou03]. An algebraic immunity of e implies an r-bit fixing degree greater than or equal to $e - r$ ([DGM05], Proposition 1), giving that an high algebraic immunity guarantees both high rational degree and high bit fixing degree. The algebraic degree is equivalent to the 0-bit fixing degree, then it leads to the following characterization: a predicate of a random local function must have high resiliency and high algebraic immunity. In light of this characterization, the authors of [AL16] suggested the XOR-MAJ predicate as a promising candidate for building high-stretch local PRGs, the majority function having optimal algebraic immunity [DMS05].

Security against Subexponential Attacks. While there is a large body of work that studied the security of random local functions, leading to a detailed characterization of the parameters and predicates that lead to insecure instantiations, relatively little is known on the *exact* security of local PRGs instantiated with non-degenerated parameters. In particular, most papers only prove that some classes of polytime attacks provably fail to break candidates local PRGs; however, these results do not preclude the possible existence of non-trivial subexponential attacks (specifically, these polytime attacks do not "degrade gracefully" into subexponential attacks when appropriate parameters are chosen for the PRG; instead, they do always and provably not succeed). To our knowledge, the only results in this regard are the proof from [AHI05, CEMT14] that many backtracking-type attacks require exponential time to invert a random local function, and the subexponential-time attack arising from the work of Bogdanov and Qiao [BQ09]. However, as we saw above, the latter attack only gives a slightly-subexponential algorithm, in time $2^{O(n^{1-(\mathsf{s}-1)/2d})}$ for a d-ary predicate, and an n^{s}-stretch local PRG.

1.3 Our Goals and Results

In this work, we continue the study of the most common candidate local pseudorandom generators. However, we significantly depart from the approach of previous works, in that we wish to analyze the *concrete* security of local PRGs. To our knowledge, all previous works were only concerned about establishing asymptotic security guarantees for candidate local PRGs, without providing any insight on, *e.g.*, which parameters can be conjectured to lead to a primitive with a given bit-security. Our motivations for conducting this study are twofold.

– Several recent results, which we briefly overviewed in Section 1.1, indicate that (poly-stretch) local PRGs enjoy important theoretical applications. However, the possibility of instantiating these applications with concrete PRG candidates remains unclear, as their efficiency quickly deteriorates with the parameters of the underlying PRG. For example, the iO scheme of [LT17], which

[1] A predicate P has r-bit fixing degree e if the minimal degree of the restriction of P obtained by fixing r inputs is e

requires low-degree multilinear maps and therefore might be a viable approach to obtain efficiency improvements in iO constructions (as candidate high-degree multilinear maps are prohibitively expensive); however, it has a cost cubic in the seed size of a poly-stretch local PRG, which renders it practical only if we can safely use local PRGs with reasonably small seeds. Overall, we believe that there is a growing need for a better understanding of the exact efficiency of candidate local PRGs, and providing concrete estimations can prove helpful for researchers willing to understand which efficiency could potentially be obtained for local-PRG-based primitives.
– At a more theoretical level, previous works on (variants of) Goldreich's PRG have identified criteria which characterize the predicates susceptible to lead to secure local PRGs. Identifying such criteria is particularly relevant to the initial goal set up by Goldreich in [Gol00], which is to understand what characteristics of a function is the source of its cryptographic hardness, by designing the simplest possible candidate that resists all attacks we know of. However, existing criteria only distinguish predicates leading to insecure instances from those leading to instances for which no polynomial-time attack is known. We believe that it is also of particular relevance to this fundamental question to find criteria which capture in a more fine-grained way the cryptographic hardness of random local functions.

Our Results. We provide new cryptanalytic insights on the security of Goldreich's pseudorandom generator.

– *A new subexponential attack on Goldreich's PRG.* We start by devising a new attack on Goldreich's PRG. Our attack relies on a *guess-and-determine* technique, in the spirit of the recent attack [DLR16] on the FLIP family of stream ciphers [MJSC16]. The complexity of our attack is $2^{O(n^{2-\mathsf{s}})}$ where s is the stretch and n is the seed size. This complements O'Donnel and Witmer's result [OW14] showing that Goldreich's PRG is likely to be secure for stretch up to 1.5, with a more fine-grained complexity estimation. We implemented our attack[1] and provide experimental results regarding its concrete efficiency, for various seed size and stretch parameters.
– *Generalization.* We generalize the previous attack to a large class of predicates, which are divided into two parts, a linear part and a non-linear part, XORed together. This captures all known candidate generalizations of Goldreich's PRG. Our attack takes subexponential time as soon as the stretch of the PRG is strictly above one. Importantly, our attack does not depend on the locality of the predicate, but only on the number of variables involved in the non-linear part. In a recent work [AL16], Applebaum and Lovett put forth an explicit candidate local PRG (of the form XOR-MAJ), as a concrete target for cryptanalytic effort. Our attack gives a new subexponential algorithm for attacking this candidate.
– *Extending the Applebaum-Lovett polynomial-time algebraic attack.* Applebaum and Lovett recently established that local pseudorandom generators can be broken in polynomial time, as long as the stretch s of the PRG is greater than the *rational degree* e of its predicate. We extend this result as follows: we show that the seed of a large class of local PRGs (which include all existing candidates) can be recovered in polynomial time whenever $\mathsf{s} \geq e - \log N_e / \log n$, where e is the rational degree, n is the seed size, and N_e is the number of independent annihilators of the predicate (or of its conjugate) [2] of degree at most e.
– *Linearization and Gröbner attack.* We complement our study with an analysis of the efficiency of algebraic attacks *à la* Gröbner on Goldreich's PRG. While it is known that Goldreich's PRG (and its variants) provably resists such attacks for appropriate choices of (asymptotic) parameters [AL16], little is known about its exact security against such attacks for concrete choices of parameters. We evaluated the concrete security of Goldreich's PRG against a degree-two linearization attack. The existence of such an attack allows to derive bounds on Gröbner basis performance. Using an implemented proof of concept, we introduce heuristic bounds for vulnerable parameters.

As illustrated by our attacks, both the number of annihilators of the predicate and the r bit fixing algebraic immunity play an important role in the security of Golreich's PRG. These criteria were

[1] Our proof of concept can be found at https://github.com/LuMopY/SecurityGoldreichPRG.
[2] An annihilator of a predicate P is a non-zero polynomials Q such that $Q \cdot P = 0$, the conjugate of a predicate P is the predicate $P + 1$

overlooked in all previous works on local PRGs. Last but not least, our concrete analysis indicates that Gröbner basis attacks, although provably "ruled out" asymptotically, matters when studying the vulnerabilities of Goldreich's PRG, and the security of concrete instances.

1.4 Organization of the Paper

Section 2 introduces necessary preliminaries on predicates and local pseudorandom generators. Section 3 describes a guess-and-determine attack on Goldreich's PRG instantiated with the predicate P_5 and analyzes it. Section 4 investigates algebraic cryptanalysis of Goldreich's PRG with P_5, presenting a degree 2 linearization attack, and an attack using Gröbner basis approach. In Section 5 are developed attacks relatively to other predicates, with a particular focus on all predicates of the form XOR-MAJ. Section 6 improves the theorem of [AL16], by taking into account the number of annihilators of the predicate. Finally, Appendix B considers the case of using Goldreich's PRG with ordered subset (as was initially advocated in [Gol00]) and provides indications that this weakens its concrete security.

2 Preliminaries

Throughout this paper, n denotes the size of the seed of the PRGs considered. A probabilistic polynomial time algorithm (PPT, also denoted *efficient* algorithm) runs in time polynomial in the parameter n. A positive function f is *negligible* if for any polynomial p there exists a bound $B > 0$ such that, for any integer $k \geq B$, $f(k) \leq 1/|p(k)|$. An event depending on n occurs with *overwhelming probability* when its probability is at least $1 - \mathsf{negl}(n)$ for a negligible function negl. Given an integer k, we write $[k]$ to denote the set $\{1, \ldots, k\}$. Given a finite set S, the notation $X \xleftarrow{\$} S$ means a uniformly random assignment of an element of S to the variable X. Given a string $x \in \{0,1\}^k$ for some k and a subset σ of $[k]$, we let $x[\sigma]$ denote the subsequence of the bits of x whose index belongs to σ. Moreover, the i-th bit of $x[\sigma]$ will be denoted by x_{σ_i}.

2.1 Hypergraphs

Hypergraphs generalize the standard notion of graphs (which are defined by a set of nodes and a set of edges, an edge being a pair of nodes) to a more general object defined by a set of nodes and a set of *hyperedges*, each hyperedge being an arbitrary subset of the nodes. We define an (n, m, d)-hypergraph G to be a hypergraph with n vertices and m hyperedges, each hyperedge having cardinality d. The hyperedges are assumed to be ordered from 1 to m, and each hyperedge $\{i_1, i_2, \ldots, i_d\}$ is ordered and satisfies $i_j \neq i_k$ for all $j \leq d$, $k \leq d$, $j \neq k$. We will consider hypergraphs satisfying some expansion property, defined below.

Definition 1 (Expander Graph). *An (n, m, d)-hypergraph G, denoted $(\sigma^1, \ldots, \sigma^m)$, is (α, β)-*expanding *if for any $S \subset [m]$ such that $|S| \leq \alpha \cdot m$, it holds that $|\cup_{i \in S} \sigma^i| \geq \beta \cdot |S| \cdot d$.*

2.2 Predicates

The constructions of local pseudorandom generators that we will consider in this work rely on predicates satisfying some specific properties. Formally, a predicate P of arity d is a function $P : \{0,1\}^d \mapsto \{0,1\}$. We define below the two properties that were shown to be necessary for instantiating local PRGs:

- *Resiliency.* A predicate P is k-resilient if it has no non-trivial correlation with any linear combination of up to k of its inputs. An example of predicate with maximal resiliency is the parity predicate (*i.e.* the predicate which xors all its inputs).
- *Algebraic Immunity.* A predicate P has algebraic immunity e, referred to as $\mathsf{AI}(P) = e$, if the minimal degree of a non-null function g such that $Pg = 0$ (or $(P+1)g = 0$) on all its entries is e. A local PRG built from an AI-e predicate cannot be pseudorandom with a stretch n^e due to algebraic attacks.

Note that the algebraic immunity (also referred as rational degree in [AL16]) implies a lower bound on the degree and on the bit-fixing degree. Moreover, a high algebraic immunity implies at least the same degree. Hence, for now on, those two criteria are considered as the relevant criteria for evaluating the security of Goldreich's PRG.

We define a particular family of predicates which have been considered as a potential instantiation:

Definition 2 ($\mathsf{XOR}_\ell\mathsf{M}_k$ predicates). *We call $\mathsf{XOR}_\ell\mathsf{M}_k$ predicate a predicate P of arity $\ell + k$ such that M is a predicate of arity k and:*

$$P(x_1, \ldots, x_\ell, z_1, \ldots, z_k) = \sum_{i=1}^{\ell} x_i + M(z_1, \ldots, z_k) \ .$$

We also define a subfamily of $\mathsf{XOR}_\ell\mathsf{M}_k$ predicates, which have been considered in [AL16]:

Definition 3 ($\mathsf{XOR}_\ell\mathsf{MAJ}_k$ predicates). *We call $\mathsf{XOR}_\ell\mathsf{MAJ}_k$ predicate a predicate P of arity $\ell + k$ such that P is a $\mathsf{XOR}_\ell\mathsf{M}_k$ predicate such that M is the majority function in k variables:*

$$M(z_1, \ldots, z_k) = 1 \Leftrightarrow \mathsf{w}_H(z_1, \ldots, z_k) \geq \left\lceil \frac{k}{2} \right\rceil \ ,$$

where w_H denotes the Hamming weight.

2.3 Pseudorandom Generators

Definition. A pseudorandom generator is a deterministic process that expands a short random seed into a longer sequence, so that no efficient adversary can distinguish this sequence from a uniformly random string of the same length. Formally,

Definition 4 (Pseudorandom Generator). *A $m(n)$-stretch pseudorandom generator, for a polynomial m, is an efficient uniform deterministic algorithm PRG which, on input a seed $x \in \{0,1\}^n$, outputs a string $y \in \{0,1\}^{m(n)}$. It satisfies the following security notion: for any probabilistic polynomial-time adversary Adv,*

$$\begin{aligned} &\Pr[y \xleftarrow{\$} \{0,1\}^{m(n)} : \mathsf{Adv}(\mathsf{pp}, y) = 1] \\ \approx &\Pr[x \xleftarrow{\$} \{0,1\}^n, y \leftarrow \mathsf{PRG}(x) : \mathsf{Adv}(\mathsf{pp}, y) = 1] \end{aligned}$$

Here $\approx$ denotes that the absolute value of the difference of the two probabilities is negligible in the security parameters, and pp stands for the public parameters of the PRG. For any $n \in \mathbb{N}$, we denote PRG_n the function PRG restricted to n-bit inputs. A pseudorandom generator PRG is d-local *(for a constant d) if for any $n \in \mathbb{N}$, every output bit of PRG_n depends on at most d input bits.*

Goldreich's Pseudorandom Generator. Goldreich's candidate local PRGs form a family $\mathsf{F}_{G,P}$ of local PRGs: $\mathsf{PRG}_{G,P} : \{0,1\}^n \mapsto \{0,1\}^m$, parametrized by an (n, m, d)-hypergraph $G = (\sigma^1, \ldots, \sigma^m)$ (where $m = m(n)$ is polynomial in n), and a predicate $P : \{0,1\}^d \mapsto \{0,1\}$, defined as follows: on input $x \in \{0,1\}^n$, $\mathsf{PRG}_{G,P}$ returns the m-bit string $(P(x_{\sigma_1^1}, \ldots, x_{\sigma_d^1}), \ldots, P(x_{\sigma_1^m}, \ldots, x_{\sigma_d^m}))$.

Conjecture 1 (Informal). If G is a sufficiently expanding (n, m, d) hypergraph and P is a predicate with sufficiently high resiliency and high algebraic immunity, then the function $\mathsf{PRG}_{G,P}$ is a secure pseudorandom generator.

Note that picking an hypergraph G uniformly at random suffices to ensure that it will be expanding with probability $1-o(1)$. However, picking a random graph will always give a non-negligible probability of having an insecure PRG. To see that, observe that when the locality d is constant, a random hypergraph G will have two hyperedges containing the same vertices with probability $1/\mathsf{poly}(n)$; for any such graph G, the output of $\mathsf{PRG}_{G,P}$ on a random input can be trivially distinguished from random. Therefore, the security of random local functions is usually formulated non-uniformly, by stating that for a $1 - o(1)$ fraction of all hypergraphs G (and appropriate choice of P), no polytime adversary should be able to distinguish the output of $\mathsf{PRG}_{G,P}$ from random with non-negligible probability.

Fixed hypergraph versus random hypergraphs. Goldreich's candidates local pseudorandom generators require to use a sufficiently expanding hypergraph. Unfortunately, building concrete graphs satisfying the appropriate expansion properties is a non-trivial task. Indeed, all known concrete constructions of expanding bipartite hypergraphs fail to achieve parameters which would allow to construct a PRG with constant locality. Therefore, to our knowledge, in all works using local PRG (see e.g. [IKOS08, App13, Lin17, ADI+17b, BCG+17]), it is always assumed (implicitly or explicitly) that the hypergraph G of the PRG is picked uniformly at random (which makes it sufficiently expanding with probability $1 - o(1)$, even in the constant-locality setting) in a one-time setup phase. Therefore, this is the setting we assume for our cryptanalysis.

Notations. In the first part of this work, we focus on the predicate P_5, assuming that the subsets $\sigma^1, ..., \sigma^m$ are random subsets. The predicate P_5 can be regarded as a Boolean function of five variables:

$$P_5(x_1, x_2, x_3, x_4, x_5) = x_1 + x_2 + x_3 + x_4 x_5 \ .$$

The predicate P_5 has algebraic degree 2 and an algebraic immunity of 2, and is 2-resilient. Let n be the size of the input, *i.e.* the number of initial random bits. We define the stretch s and denote the size m of the output as $m = n^{\mathsf{s}}$. Let $x_1, \ldots, x_n \in \mathbb{F}_2$ be the input random bits and $y_1, \ldots, y_m \in \mathbb{F}_2$ be the output bits. The m public equations E_i for $1 \leq i \leq m$ are drawn as follows:

- a subsequence of $[n]$ of size 5 is chosen uniformly at random. Let us call it

$$\sigma^i = [\sigma^i_1, \sigma^i_2, \sigma^i_3, \sigma^i_4, \sigma^i_5] \ .$$

- E_i is the quadratic equation of the form

$$x_{\sigma^i_1} + x_{\sigma^i_2} + x_{\sigma^i_3} + x_{\sigma^i_4} x_{\sigma^i_5} = y_i \ .$$

The public system Σ that we consider is then defined with the m equations, that is $(E_i)_{1 \leq i \leq m}$.

Ordered and unordered. There are two different cases to consider:

1. (Ordered case) σ^i is ordered, *i.e.* $\sigma^i_1 < \sigma^i_2 < \sigma^i_3 < \sigma^i_4 < \sigma^i_5$.
2. (Unordered case) The order σ^i's elements is arbitrary.

However, in the core of the paper, we will consider the **unordered case**, as we will provide evidence that the vulnerabilities are even more important for the ordered case in Appendix B.

Matrix inversion complexity. Our attacks require a sparse matrix inversion algorithm. We consider the Wiedemann's algorithm [Wie86], the complexity of which is $O(n^2)$ in our context, since there are less than $d \cdot n$ non-zero elements of our matrices. Other algorithms could be used, but the complexity of our attacks would have to be modified accordingly.

3 Guess and Determine Cryptanalysis of Goldreich's PRG with P_5

3.1 The Attack - Asymptotic Description

We first describe a distinguishing attack, where our adversary outputs 1 when the challenged bitstream is considered as the PRG's output and 0 when it is considered as a random string.

At a high level, the attack works by collecting a large number of linear equations, by guessing well-chosen bits of the seed, seen as a vector x of n variables. When enough equations have been collected, two cases can occur.

- Either sufficiently many equations are linearly independent (as much as the number of variables); in this case, the attacker can invert a large subsystem of equations, obtain a candidate seed, and check it against the PRG output (therefore finding out whether the guesses were correct in the first place).

– Either most of the equations are linearly dependent. In this case, we show that this implies that the PRG output must pass a large number of linear tests, which a random string would be unlikely to all pass. We use this observation to mount a distinguishing attack.

We now proceed with the formal description of the attack. Our algorithm has the description of the PRG hardcoded (namely, an $(n, m, 5)$-hypergraph $G = (\sigma^1, \cdots, \sigma^m)$, where $m = n^{\mathsf{s}}$, and each $\sigma^i = (\sigma^i_1, \cdots, \sigma^i_5)$ is a size-5 subset of $[n]$). It takes as input an m-bit string $y = y_1 \cdots y_m$, and must distinguish whether y is a random string, or whether it is in the image of PRG_{G,P_5}. The algorithm starts by considering the following list of quadratic equations for $i = 1$ to m:

$$P_5(x_{\sigma^i_1}, \cdots, x_{\sigma^i_5}) = y_i.$$

We denote Q this list. The algorithm will proceed by constructing $O(m)$ linear equations from Q.

Selection Phase. The algorithm dynamically determines a "selected" subset of the quadratic equations and a subset Σ of $[m]$, which it will use in the guessing phase. The sets are constructed using the following greedy approach: set $j \leftarrow 1$ and mark all equations of Q as "unselected". In the jth step, find the variable that appears in the largest number of quadratic terms over all equations in Q which are marked "unselected". Mark all the equations in which this variable appears in a quadratic term as "selected", and add their indexes in Q to Σ, also add the linear equation corresponding to the affectation of this variable and count it as a "selected" equation. If the number s of equations marked as "selected" satisfies $s \geq n + j$, set $\ell \leftarrow j$, $y' \leftarrow y[\Sigma]$, and proceed to the guessing phase. Otherwise, set $j \leftarrow j + 1$ and continue.

Guessing Phase. In the previous phase, the algorithm has identified a subset of ℓ variables which appear overall in the quadratic term of $s \geq n+\ell$ selected quadratic (and linear) equations. In this step, the algorithm will enumerate over all 2^ℓ possible assignments for these variables, in some arbitrary fixed order. In each step, for $i = 1$ to 2^ℓ, the algorithm obtains a system s linear equations by assigning a value in $\{0, 1\}$ to each of the ℓ variables across all selected quadratic equations. Let A_i denote the matrix of this system. We distinguish two cases:

– **Case 1.** $\mathsf{rank}(A_i) = n$**.** In this case, there is an $n \times n$ invertible submatrix of A_i. The algorithm extract this submatrix, let us denote it E_i. We also denote by y'_i the subsequence of y' indexed by the position of the rows of E_i in A_i. Let $F_i \leftarrow E_i^{-1}$. The algorithm computes a candidate seed $x'_i \leftarrow F_i y'_i$, and checks whether $\mathsf{PRG}_{G,P_5}(x'_i) = y$. If it holds, it outputs 1 and halts. Else, it sets $i \leftarrow i + 1$.
– **Case 2.** $\mathsf{rank}(A_i) < n$**.** In this case, there exists at least $\ell + 1$ linearly dependent rows of A_i. Let B_i denote the row echelon form of A_i, obtained through Gaussian elimination, and let G_i denote the (invertible) matrix of this transformation; that is, $B_i = G_i A_i$. Let $(v_1^\intercal, \cdots, v_{\ell+1}^\intercal)$ denote the last $\ell + 1$ rows of G_i. The algorithm checks whether $v_k^\intercal y' = 0$ for $k = 1$ to $\ell + 1$. If all checks pass, it outputs 1 and halts. Else, it sets $i \leftarrow i + 1$.

If the algorithm reaches $i = 2^\ell + 1$, it outputs 0 and halts.

3.2 Complexity Analysis

We now analyze the complexity of the algorithm. We first estimate the average value of ℓ obtained in the selection phase. We consider the list Q of all quadratic equations. For all i such that $1 \leq i \leq n$ let denote N_i^1 the number of occurrences of x_i in degree-two monomials.

Proposition 1 (Number of guesses). *For any instance with n variables, m equations and c collisions, an upper bound on the sufficient number of guesses required to build $n - c$ linear equations is:*

$$\ell \leq \left\lfloor \frac{n^2}{2m} + 1 \right\rfloor . \tag{1}$$

Proof. Let us choose the variable with more occurrences, denoted w.l.o.g. x_1, as there are m equations, $\sum_{i=1}^{n} N_i^1 = 2m$, and therefore $N_1^1 \geq 2\frac{m}{n}$. Fixing the value of x_1 we get N_1^1 linear equations (plus the linear equation fixing the value of x_1). Since the value of x_1 is fixed, the remaining quadratic system of equations consists of $m - N_1$ equations in $n-1$ unknowns $(x_2, \ldots, x_n)$. We recursively use this strategy:

For all j $(2 \leq j \leq n)$ we denote N_i^j the number of occurrences of x_i in a degree-two monomial in the system of equations obtained after fixing the $j-1$ first most appearing variables (as previously described) w.l.o.g. $x_1, \ldots, x_{j-1}$. Then, choosing the variable with higher N_i^j, w.l.o.g. x_j, the remaining quadratic system of equations consists of $m - N_1^1 - N_2^2 - \ldots - N_j^j$ equations in $n-j$ unknowns. So for all $1 \leq j \leq n$:

$$N_j^j \geq 2\frac{m - N_1^1 - N_2^2 - \cdots - N_{j-1}^{j-1}}{n-j+1} \geq 2\frac{m}{n} ,$$

and we get $N_1^1 + N_2^2 + \cdots + N_j^j + j$ linear equations at this step with the value of $x_1, x_2, \ldots, x_j$ being fixed.

Take ℓ as the first value of j such that $N_1^1 + N_2^2 + \cdots + N_j^j + j \geq n + j$ (which is correctly defined as we only consider cases where $2m \geq n$). Then,

$$N_1^1 + N_2^2 + \cdots + N_{\ell-1}^{\ell-1} + \ell - 1 < n + \ell - 1 .$$

As for all $1 \leq j \leq \ell$, we have $N_j^j \geq 2\frac{m}{2}$ we get

$$2(\ell-1)\frac{m}{n} < n .$$

So, the number of variables to guess ℓ is at most:

$$\left\lfloor \frac{n^2}{2m} + 1 \right\rfloor .$$

Note that since we consider the regime of superlinear stretch ($m = n^{\mathsf{s}}$ with $\mathsf{s} > 1$), the above implies that $\ell = o(n)$ (in fact, $\ell = O(n^{2-\mathsf{s}})$). □

We show further in Section 3.6 that experimental results are much better. It is worth noticing that the value obtained at Proposition 1 is the extreme case for the attacker and does not reflect the average case. However, this frequency of appearance is linked to a well-known problem of combinatorics in the context of balls-into-bins. At the second order, the maximum load (*i.e.* the number of occurrences of the variable that appears the most) follows:

$$\Theta\left(\sqrt{\frac{m \ln n}{n}} + \frac{m}{n}\right) ,$$

where m corresponds to the number of balls and n to the number of bins (*e.g.* [JK77, KSC78]), which means we gain nothing asymptotically in average. This is related to us, but is not exactly the same, as in one monomial, one variable cannot be taken twice. However, we can lower the maximum that one variable appears with the classical setting of balls and bins by only considering the first variable, but also upper bound our exact probability distribution using twice the maximum load. Eventually, we can say that in average, the number of guesses is asymptotically the same as the worst case for the attacker.

Cost of the Selection Phase. The lemma below follows immediately:

Lemma 1. *The selection phase has complexity $O(\ell \cdot m)$ which is $O(n^2)$ with Equation 1 estimation.*

Cost of the Guessing Phase. For each $i \in \{1, \cdots, 2^\ell\}$, the algorithm executes either the procedure of case 1 or the procedure of case 2; finding out which case to execute requires computing the rank of an $s \times n$ matrix, with $s \approx n + \ell$. The cost of case 1 is dominated by the inversion of an $n \times n$ matrix (since this cost is at least n^2, it dominates the cost of evaluating PRG_{G,P_5}, which is $O(m)$); the cost of case 2 is dominated by the Gaussian elimination step. Observe that by construction, the matrix A_i (hence the submatrix E_i as well) is very sparse: each of its rows contains at most four nonzero entries. Therefore, we can apply Wiedemann algorithm [Wie86] and compute the rank of A_i, the inverse of E_i, or the row echelon form of A_i, in time $O(n \cdot (n + \ell)) = O(n^2)$ (since they can all be computed by making a constant number of black-box calls to an algorithm solving a sparse system of linear equations).

Combining the above calculations, the cost of the entire algorithm is dominated by

$$O(n^2 \cdot 2^\ell) = 2^{O(n^{2-s})}.$$

Lemma 2. *The asymptotic complexity of the attack is*

$$O\left(n^2 2^{\frac{n^{2-s}}{2}}\right).$$

3.3 Success Probability

We now analyze the success probability of the algorithm. Let us first assume that y is in the image of the PRG; that is, there exists x such that $y = \mathsf{PRG}_{G,P_5}(x)$. In this case, during the guessing phase, since the algorithm enumerates over all possible values for the ℓ selected variables, there must be an index i such that the selected variables have been assigned the correct value. Let i^* denote this index.

- If $\mathsf{rank}(A_{i^*}) = n$ (case 1), the algorithm exactly recovers the right seed x by inverting the $n \times n$ subsystem, hence the check that $\mathsf{PRG}_{G,P_5}(x'_i) = y$ necessarily passes, hence the algorithm outputs 1 and halts with probability 1.
- If $\mathsf{rank}(A_{i^*}) < n$ (case 2), observe that by construction, the last $\ell + 1$ rows of B_{i^*} are identically zero (since the number of zero rows at the end of the row echelon form of the matrix A_{i^*} is equal to the co-rank of A_{i^*}, which is at least $\ell + 1$). By assumption y is in the image of the PRG and i^* is the right guess, hence we have

  $$G_{i^*} y' = G_{i^*}(A_{i^*} x) = B_{i^*} x,$$

  which implies that $G_{i^*} y'$ ends with at least $\ell + 1$ zeroes (since the last $\ell + 1$ rows of B_{i^*} are identically zero). Therefore, all checks of the algorithm necessarily pass, and it outputs 1 and halts with probability 1.

Hence, if y is in the image of the PRG, the algorithm always outputs 1. Let us now assume that y is a uniformly random m-bit string. Let us fix an arbitrary i between 1 and 2^ℓ. We analyze the probability that the algorithm outputs 1 on this i, where the probability is over the uniformly random choice of y. As previously, two cases can happen.

- If $\mathsf{rank}(A_i) = n$ (case 1), the algorithm extracts a candidate seed x'_i. Note that this extraction is entirely independent of the choice of y. There are 2^n possible values of x'_i, hence 2^n possible values of $\mathsf{PRG}_{G,P_5}(x'_i)$. The probability (over a random choice of the m-bit string y) that y hits one of those values is equal to $2^n/2^m = 1/2^{m-n}$. Hence, the probability that the algorithm outputs 1 at step i, conditioned on case 1 happening, is upper bounded by $1/2^{m-n}$.
- If $\mathsf{rank}(A_i) < n$ (case 2), the algorithm obtains $\ell + 1$ vectors $(v_1, \cdots, v_{\ell+1})$. Note that since the v_i are rows of G_i, and G_i is invertible, the v_i are all linearly independent. Now, the probability that a uniformly random bit-vector y passes $\ell + 1$ linearly independent linear tests is at most $1/2^{\ell+1}$; therefore, the probability that the algorithm outputs 1 at step i, conditioned on case 2 happening, is upper bounded by $1/2^{\ell+1}$.

Since $\ell + 1 = O(n^2/m) = o(m-n)$, for a sufficiently large n we have

$$\frac{1}{2^{\ell+1}} > \frac{1}{2^{m-n}},$$

from which we get that for each i, the probability (over a random choice of y) that the algorithm outputs 1 is at most $1/2^{\ell+1}$. Taking a union bound over all possible choices of i, we get that the probability that *there exists an index* i for which the algorithm outputs 1 is at most $2^{\ell} \cdot 1/2^{\ell+1} = 1/2$. Hence, with probability at least $1/2$, the algorithm outputs 0.

Overall, the algorithm correctly distinguishes between $y = \mathsf{PRG}_{G,P_5}(x)$ and random y with probability at least $1/2(1 + 1/2) = 3/4$. Note that the success probability of the adversary can be made as close to 1 as one wishes, by collecting $n + \ell + \lambda - 1$ linear equations instead of $n + \ell$, for a security parameter λ; it is easy to check that this does not change the asymptotic complexity of the algorithm, and by the same analysis, the algorithm correctly outputs 0 when y is random with overwhelming probability at least $1 - 1/2^{\lambda}$.

3.4 Seed Recovery

The attack which we described above is a distinguishing attack: it breaks the pseudorandomness of the PRG in subexponential time $2^{O(n^{2-\mathsf{s}})}$. Observe that when $y = \mathsf{PRG}_{G,P_5}(x)$ for some x, if case 1 happens at the step i^* corresponding to the right guess, the attack gives something stronger: it actually breaks the one-wayness of the PRG, by recovering the seed. Furthermore, our experimental evaluations (which we will discuss in Section 3.6) show that this is actually *always* the case: the algorithm systematically ends up in case 1, and case 2 never happens, leading to a seed recovery attack. In this section, we provide some theoretical support for this observation:

- we put forth a combinatorial assumption and prove that, under this assumption, there is a seed recovery algorithm which is a slight variation of our algorithm (and has the same complexity);
- we provide heuristic support for our combinatorial conjecture by relating it to existing results in mathematics.

Combinatorial Conjecture. We consider the following conjecture: set $\beta \leftarrow \lfloor n^2/2m + 1 \rfloor$, and define, for $i = 1$ to 2^{β}, $\mathcal{D}_{n,i}$ to be the distribution over $\mathbb{F}_2^{n \times n}$ obtained by sampling the hypergraph of Goldreich's PRG at random (with $d = 5$), selecting ℓ variables that appear in $n+\ell$ quadratic equations using the *selection phase* algorithm (see Section 3.1), and outputting the $n \times n$ matrix M_n of the linear system obtained by setting all ℓ selected variables to the values indicated by the ℓ first bits of i (note that our analysis guarantees that $\ell \leq \beta$). We truncate to n equations for simplicity.

Hypothesis 1 *There exists a constant γ such that for every sufficiently large $n \in \mathbb{N}$, for every $i \leq 2^{\beta}$, the matrix M_i contains with overwhelming probability an invertible subsystem of $\gamma \cdot n$ equations, where the probability is taken over the coins of $M_i \xleftarrow{\$} \mathcal{D}_{n,i}$.*

Note that the conjecture is tailored to our particular attack, and could be easily generalized to more general PRG distributions and variable selection methods – indeed, we do consider generalizations and variants of this conjecture in the following Sections. We first show that if Hypothesis 1 is verified, then there is a seed recovery attack on Goldreich's PRG instantiated with P_5. The attack is a simple variation of our previous algorithm, where in the guessing phase we do not consider case 2. Instead, the algorithm extracts a $\gamma n \times \gamma n$ invertible submatrix E_i of A_i (whose existence is guaranteed by Hypothesis 1), and uses it to recover a subsequence of γn bits of the seed x. Now, by applying the result of Bogdanov and Qiao [BQ09] on recovering a preimage from an approximate preimage of Goldreich's PRG, there exists a black-box polynomial-time reduction from an algorithm that recovers (with no errors) $O(n^{(7-\mathsf{s})/8}) \ll \gamma \cdot n$ bits of the seed to an algorithm that fully recovers the seed.

Supporting the Conjecture. Unfortunately, the distributions $\mathcal{D}_{n,i}$ are quite complex, and it seems relatively difficult (and outside the scope of this paper) to prove our conjecture. However, we can

provide some heuristic support for the conjecture: variants of our conjecture with respect to simpler (and natural) distributions (which are close to the one we consider) follow from existing results in mathematics and computer science. Note that the $\mathcal{D}_{n,i}$ are distributions of random very sparse matrices, with at most 4 nonzero entries per row. We can consider two simpler natural distribution over very sparse matrices:

- the distribution D obtained by setting each entry of the matrix to be 1 with probability $4/n$, and 0 with probability $(n-4)/n$ (the Bernouilli distribution);
- the distribution D' obtained by sampling 4 random positions between 1 and n in each row, setting the entries at these positions to be 1, and setting all other entries of the row to be 0.

For the distribution D, simply looking at the entries that contain exactly a single 1 will give with high probability a $\gamma n \times \gamma n$ invertible submatrix (indeed, a permutation matrix),with $\gamma \approx 5 \cdot e^{-5}$. This gives a very loose lower bound on γ, but in fact, much stronger bounds are known for this distribution, at least in the case of random sparse *symmetric* matrices [BL10].

For the distribution D', the conjecture is very close to problems which have been studied in computer science under the name of Random XOR-SAT. In particular, the recent work of [PS16] gave a precise threshold value of c such that a random $c \cdot n \times n$ matrix contains an $n \times n$ invertible matrix with probability 1; this result implies in particular a (loose) lower bound of $\gamma = 1/c$ for our conjecture.

3.5 The Attack - Concrete Instantiation

We formulated our attack in an asymptotic sense, to obtain provable asymptotic efficiency guarantees. However, it is possible to obtain a much better concrete efficiency than the one achieved by our algorithm. A first observation is that even before the selection phase, we can collect several linear equations "for free" by looking at all quadratic equations where the quadratic terms are equal, and XORing them to cancel out the quadratic terms.

Finding All Collisions. We first define the notion of collisions between two quadratic equations.

Definition 5. *A collision is a couple* $(i,j) \in [m]^2$ *such that* $i \neq j$ *and* $\{\sigma_4^i, \sigma_5^i\} = \{\sigma_4^j, \sigma_5^j\}$.

Observe that any collision leads to a linear equation "for free": XORing the quadratic equations indexed by σ^i and σ^j, the terms $x_{\sigma_4^i} \cdot x_{\sigma_5^i}$ and $x_{\sigma_4^j} \cdot x_{\sigma_5^j}$ cancel out, leading to a linear equation. The algorithm first finds all collisions, and derives the corresponding linear equations. Let c be the number of linear equations obtained with this step. While the asymptotic number of such collisions is small, hence it does not change the asymptotic complexity, it turns out that this simple step already strongly reduces the concrete cost of the attack. Let c denote the number of linear equations obtained this way.

Note that finding all collisions can be reached with a tweaked sorting algorithm. The idea is to sort the equations $(E_i)_{1 \leq i \leq m}$ according to an order[1] on the quadratic term $x_{\sigma_4^i} x_{\sigma_5^i}$. And, each time an equality between two quadratic terms is found, one equation is removed and a new linear equation $E_i + E_j$ is derived. The complexity is dominated by the sorting complexity $O(m \cdot \log(m))$.

Avoiding the Bogdanov and Qiao Algorithm. Furthermore, as we already mentioned, we observe experimentally that case 2 never happens. In all our experiments, the algorithm always ends up in case 1, with a value of $\gamma > 0.90$. Note also that applying the result of Bogdanov and Qiao to obtain the seed from the approximate preimage is an overkill: this result actually only requires knowing an approximate preimage (but not necessarily which of the bits of the preimage are correct), while our attack gives us also the *exact position* of the correct bits of the preimage. Therefore, we can simply inject directly these $\gamma n > 0.90n$ values in our list of quadratic equations, which will turn a large fraction of them into linear equations, and hope to obtain the missing values directly from these linear equations. Our experiments show that this is indeed the case: after recovering a large fraction of the preimage, injecting the values in the quadratic equations always allows to recover the full seed. Our experiments show that this is the case with a large confidence gap: injecting only a small fraction $\gamma > 0.20$ of the preimage in the quadratic equations is sufficient to always recover the full seed.

[1] The order does not matter since only equalities are necessary, one can take the lexicographic order for example.

Collecting Less Equations. Lastly, since case 2 never happens, we do not need to collect $n+\ell$ linear equations: we can stop as soon as we collect $n-c$ linear equations in the guessing phase (leading to a total of n linear equations when adding the equations obtained through collisions – note that we were already truncating the matrices A_i and ignoring the last ℓ equations when formulating Hypothesis 1).

Assessing the Number of Collisions. For completeness, we analyze the asymptotic number of equations obtained through collisions. As previously noticed, collisions can be used to build linear equations. For example, let us assume we have the following two equations in Σ:

$$x_{\sigma_1^i} + x_{\sigma_2^i} + x_{\sigma_3^i} + x_{\sigma_4^i} x_{\sigma_5^i} = y_i \tag{2}$$

$$x_{\sigma_1^j} + x_{\sigma_2^j} + x_{\sigma_3^j} + x_{\sigma_4^i} x_{\sigma_5^i} = y_j \tag{3}$$

then adding equation (2) and equation (3) gives us the following linear equation:

$$x_{\sigma_1^i} + x_{\sigma_2^i} + x_{\sigma_3^i} + x_{\sigma_1^j} + x_{\sigma_2^j} + x_{\sigma_3^j} = y_i + y_j$$

However, we stress that if we had a third colliding equation:

$$x_{\sigma_1^k} + x_{\sigma_2^k} + x_{\sigma_3^k} + x_{\sigma_4^i} x_{\sigma_5^i} = y_k \tag{4}$$

then we could only produce a single other linear equation (w.l.o.g. (2) + (4)), since the other combination ((3) + (4)) would be linearly equivalent to the two previous linear equations.

Hence, this problem can be seen as a balls-into-bins problem: m balls are randomly thrown into $\binom{n}{2}$ bins and we want to know how many balls in average hit a bin that already contains at least one ball. Indeed, this number will approximate the value c of the algorithm.

Proposition 2 (Average number of collisions). *Let n be the number of variables, and m be the number of equations, let C be the random variable counting the number of collisions on the degree-two monomials in the whole system. Then, the average number of collisions is:*

$$\mathbb{E}(C) = m - \binom{n}{2} + \binom{n}{2}\left(\frac{\binom{n}{2}-1}{\binom{n}{2}}\right)^m \in O(n^{2(s-1)}) .$$

Proof. We first consider individually the $\binom{n}{2}$ degree-two possible monomials. For each equation, the two variables of the degree-two monomial are taken uniformly from the n variables (with replacement), therefore the probability that the monomial indexed by i, j is taken follows a Bernouilli law with parameter $p = \frac{1}{\binom{n}{2}}$.

The random variable counting how many times the monomial indexed by i, j is selected follows a binomial law of parameters m and p. As a collision happens when the monomial has already be taken, we consider the random variable $C_{i,j}$ counting 0 if the monomial has been taken 0 or 1 times, $k-1$ otherwise. The expectation of $C_{i,j}$ is therefore

$$\mathbb{E}(C_{i,j}) = \sum_{k=2}^{m} P_{[B(m,p)=k]} \cdot (k-1),$$

where $P_{[B(m,p)=k]}$ stands for the probability for a random variable following a binomial distribution of parameters m and p to take the value k. The total number of collisions is obtained by summing the expectations of all the $C_{i,j}$. The detailed calculations are in Appendix A. □

Tab. 1 gives the evaluation of this formula for some set of parameters. Our experimental results (see Section 3.6) corroborate these expectations and show that the number of collisions is always very close to this expected average.

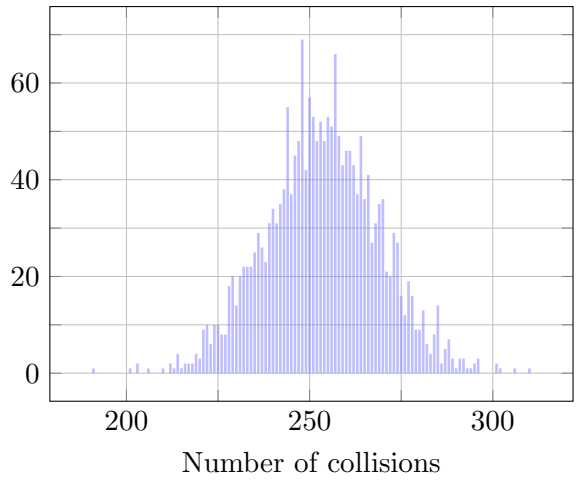

Fig. 1. Number of collisions for $n = 1024$ and $\mathsf{s} = 1.4$ with 2000 tests

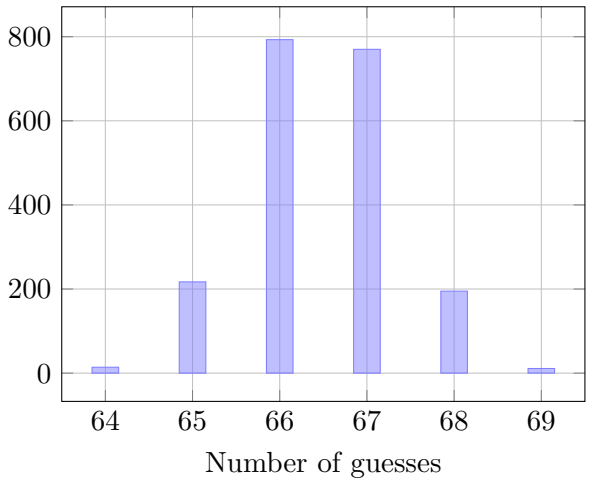

Fig. 2. Number of guesses for $n = 2048$ and $\mathsf{s} = 1.3$ with 2000 tests

3.6 Experiments

Distribution of the number of collisions. The theoretical results of Table 1 are verified in practice, as shown in Fig. 1 for the particular case of $n = 1024$ and $\mathsf{s} = 1.4$. As expected with the analytical formula, the number of collisions is very close to 254 in average. Moreover, our experimental results are very dense around the average, suggesting that the distribution has a low variance.

Implementation of the attack. Since the study of this paper is the concrete security of Goldreich's PRG, it is important to practically check if the attack presented in Section 3.4 can be efficient when implemented. For this purpose, we provide a proof of concept in Python[1].

We first analyzed experimentally Hypothesis 1 and observed that we always obtain an invertible subsystem of at least $0.90 \cdot n$ equations, for all tested parameters ($2^8 \leq n \leq 2^{14}$ and $1 < \mathsf{s} < 1.5$). We also experimented that knowing only 20% of the seed allows to inject it in the quadratic system and to recover the remaining 80%, showing a large gap of confidence in our hypothesis.

One can note that the practical attack should be on average more efficient than assessed theoretically. Indeed, the asymptotic complexity of Proposition 2 is estimated in the worst case and pessimistic approximations were made on $n - c$ and on the value of ℓ. Hence, we experimented this attack for different stretches and different values of n and we effectively noticed that the complexity in average is much smaller than the expected complexity. Table 2 represents the theoretical number of guesses necessary to recover the seed and Table 3 represents the average number of guesses actually needed in the experiment. Moreover, we also noticed that the number of guesses needed to invert the system has a very low variance, as shown in Fig. 2.

With this experiment, we were able to estimate the practical security of Goldreich's PRG against the guess and determine approach with 80 bits of security. Indeed, for one instance of the PRG, the complexity of the seed recovery can be easily derived from the number ℓ of guesses as $2^{\ell} n^{\omega}$. So to assess the 80 bits security, one can evaluate the average number of guesses necessary for one choice of (n, s) and check if the complexity is lower than 2^{80}. For that, for 30 values of $n \in [2^7, 2^{14}]$, we delimited the smallest stretch for which the average number of guesses allows a 80 bits attack. Each average has

[1] Our proof of concept can be found at https://github.com/LuMopY/SecurityGoldreichPRG

Table 1. Average number of collisions

n	256	512	1024	2048	4096
$\mathsf{s} = 1.45$	142	269	506	946	1771
$\mathsf{s} = 1.4$	83	145	254	442	773
$\mathsf{s} = 1.3$	28	42	64	97	147

Table 2. Theoretical number of guesses (worst case)

n	256	512	1024	2048	4096
$\mathsf{s} = 1.45$	4	7	11	18	27
$\mathsf{s} = 1.4$	9	15	23	37	58
$\mathsf{s} = 1.3$	20	34	56	94	156

Table 3. Experimental number of guesses (average)

n	256	512	1024	2048	4096
$\mathsf{s} = 1.45$	4	6	9	14	21
$\mathsf{s} = 1.4$	6	11	17	27	44
$\mathsf{s} = 1.3$	13	23	39	65	110

been done on 1000 measurements because the variance was very small. Fig. 3 represents the limit on vulnerable (n, s) parameters. Above the line, the parameters are on average insecure against the guess and determine attack.

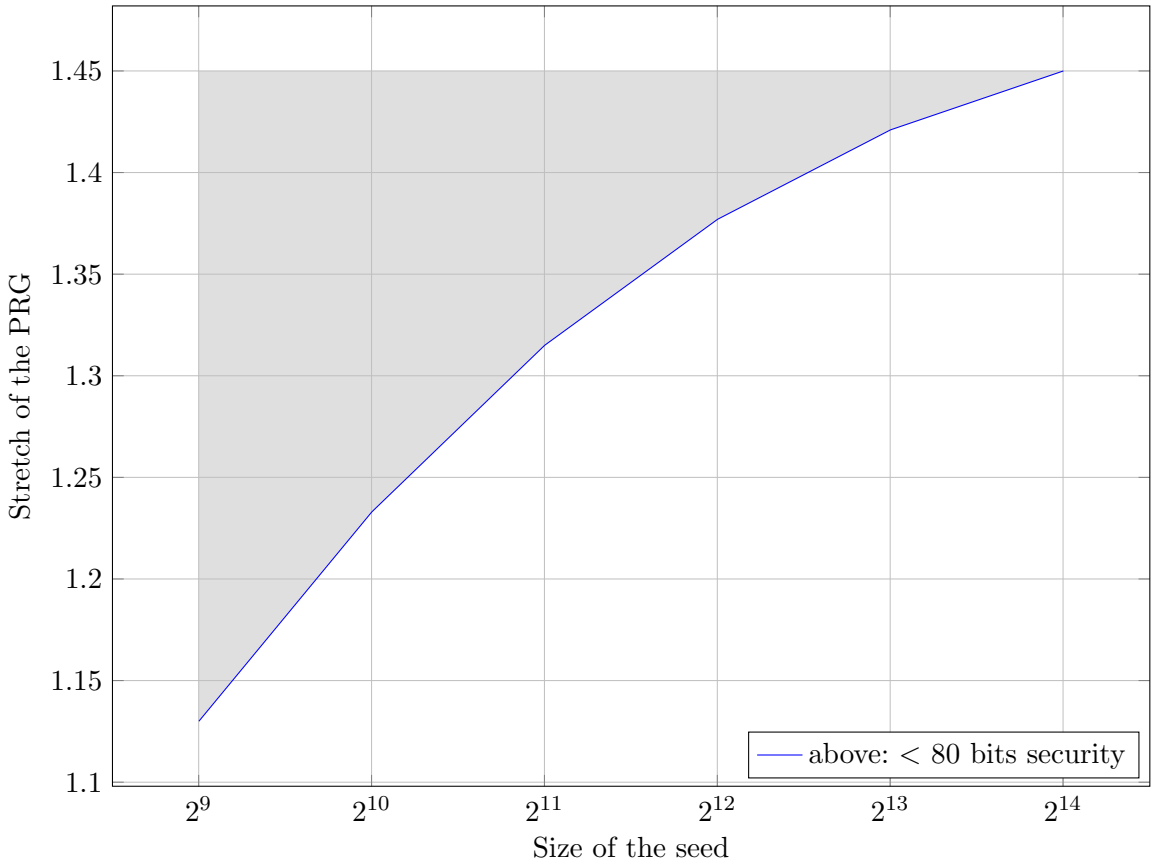

Fig. 3. Limit stretch for vulnerable instances. The grey zone above the curve denotes the insecure choices of parameters.

Candidate Non-Vulnerable Parameters. We were able to estimate the practical range of parameters that appear to resist to this attack. To assess them, we estimated the number of guesses necessary and deduced the bit security. With many measurements (1024 for each set of parameters), we could find the limit stretch for parameters that are, not vulnerable to our attack. The couples (n, s) that possess the maximal s with an expected security of 80 or 128 bits[1] are conjectured to be the limit for non-vulnerable parameters. These couples[2] are represented by the two lines in Fig. 4.

We also introduce certain parameters in Table 4 as challenges for improving the cryptanalysis of Goldreich's PRG. These parameters correspond to choices of the seed size and the stretch which cannot be broken in less than 2^{80} (resp. 2^{128}) operations with the attacks of this paper. Further study is required to assess confidence in the security level given by these parameters.

[1] We actually took a margin of 10% to take into account the possible improvements of our implementation.
[2] This curve should not be extrapolated because outside of its range, Gröbner attacks seem more powerful, see Fig. 10

Table 4. Challenge parameters for seed recovery attacks. The first line contains the parameter n and below are represented the associated stretches s.

Elementary operations	512	1024	2048	4096
$< 2^{80}$	1.120	1.215	1.296	1.361
$< 2^{128}$	1.048	1.135	1.222	1.295

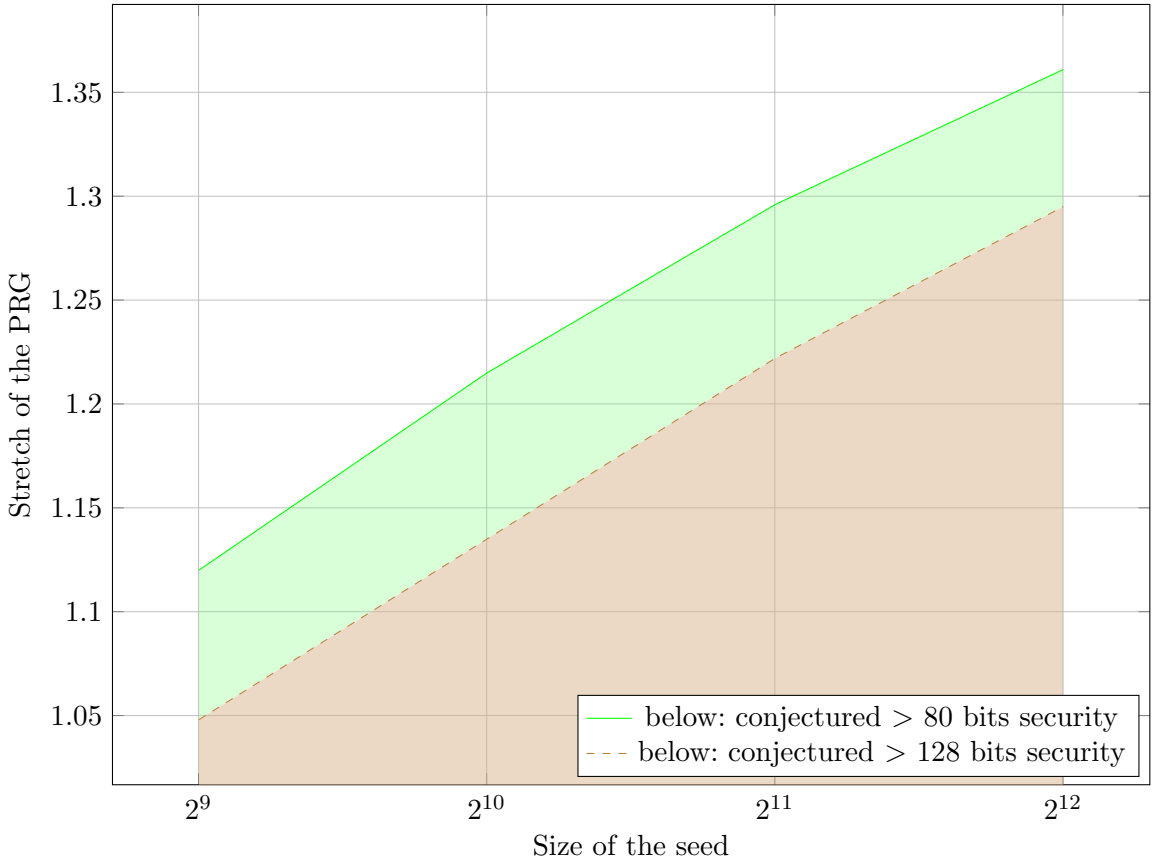

Fig. 4. Limit stretch for conjectured non-vulnerable instances.

4 Algebraic Cryptanalysis of Goldreich's PRG with P_5

To complement the attacks of Section 3.1, we also provide an analysis of the efficiency of algebraic attacks with Gröbner basis on Goldreich's PRG. While it is known that Goldreich's PRG (and its variants) provably resists such attacks for appropriate choices of (asymptotic) parameters ([AL16], Theorem 5.5), little is known about its exact security against such attacks for concrete choices of parameters.

In this section, we study the existence of polynomial attacks for $\mathsf{s} < 1.5$. In fact, with the current literature, either $\mathsf{s} \geq 1.5$ and there is a polynomial inversion, or $\mathsf{s} < 1.5$ and the only known attack is subexponential. The idea of this section is to offer some granularity on the parameters (n, s) instead of this abrupt limit for polynomial inversion. For this, we opt for a different algebraic approach without guess and determine. Instead of guessing values to transform the public system into a linear system in the seed, one might want to generate enough equations in order to linearize. This standard method has been introduced by Macaulay in [Mac64] and Lazard in [laz81]. The idea behind linearization is the assignment of an unknown variable for each of the monomials appearing in the system. For example, to each monomial $x_i x_j$, a variable $X_{i,j}$ will be assigned. Thereby, a linear system of equations with more unknowns of type $X_{i,j}$ remains to be solved. This linearization method has been improved in Gröbner basis computations due to Buchberger [Buc76] and later by Faugère with F4 [Fau99] and F5 [Fau02] algorithms.

Performance of a Gröbner basis strategy is hard to assess for the specific case of Goldreich's PRG with the existing theory (see [BFSyY] for complexity bounds on Boolean random quadratic systems). Indeed, Goldreich's PRG is far from a Boolean random quadratic system, it has a strong structure and is very sparse. These features should make Goldreich's PRG an easier target. In a first step, in order to give an intuition on how Gröbner basis algorithms would behave on Goldreich's PRG with predicate P_5, we provide an easy-to-understand degree-two linearization attack. This polynomial attack leads to a practical seed recovery for certain parameters (n, s) and we can derive a heuristic bound for vulnerable (n, s) for 80 bits of security. The existence of such an attack allows to estimate Gröbner basis algorithm complexity. Indeed, Gröbner basis algorithms use an optimized method to generate polynomials. So, their performance is at least as good as our linearization attack. Thus, from our linearization attack performance and complexity, we derive a heuristic bound on vulnerable (n, s) parameters against a Gröbner basis technique. This heuristic bound shows that a Gröbner basis approach may attack more parameters than the guess-and-determine technique (of Section 3) for high values of n.

4.1 A Polynomial Attack with Degree-Two Linearization

For a degree-two linearization, the number of variables will highly increase in comparison to the Section 3 case. Indeed, the total variables will include linear terms of shape x_i and quadratic terms of shape $x_i x_j$ where $i \neq j$. Thus, the total number of variables is

$$\mathcal{N}_{var}(n) = n + \binom{n}{2}.$$

To get a chance to invert a system with so many linearized variables, one needs to generate as many quadratic equations as possible. Fortunately, Goldreich's PRG with P_5 predicate has such a structure that allows any attacker to create a certain number of new equations from the original system. Before showing how to generate these equations, in a first step, let us introduce the principle of the attack assuming that a certain number of equations is drawn.

An attack and its complexity. Suppose that a Goldreich's PRG is drawn with parameters (n, s) and with c collisions. Suppose also that one can create a set of quadratic equations that contains $\mathcal{N}_{indep\ eqns}$ linearly independent ones. Only equations of degree exactly 2 are counted in $\mathcal{N}_{indep\ eqns}$. We sketch a seed recovery attack assuming that

$$0 \leq \mathcal{N}_{var}(n) - \mathcal{N}_{indep\ eqns} \leq c$$

and assess its complexity.

step 1 From the system of $\mathcal{N}_{indep\ eqns}$, we create a linear system in matrix form.

step 2 We rewrite this system by separating the quadratic part and creating submatrices. Let q_i be the quadratic part of this new system and b_i be its linear part and y_i be its constant term.

$$\begin{aligned} q_1 + b_1 &= y_1 \\ &\vdots \\ q_{\mathcal{N}_{indep\ eqns}} + b_{\mathcal{N}_{indep\ eqns}} &= y_{\mathcal{N}_{indep\ eqns}} \end{aligned}$$

The linearization consists in solving $(q_i - b_i = y_i)_{i \in [\mathcal{N}_{indep\ eqns}]}$ by replacing each monomial with a variable and trying to invert a linear system of size $\mathcal{N}_{indep\ eqns} \cdot \mathcal{N}_{var}(n)$. We then rewrite the system in terms of matrices. Let $Q \in \mathbb{F}_2^{\mathcal{N}_{indep\ eqns} \cdot \binom{n}{2}}$ represent the coefficients of the quadratic polynomials q_i and $B \in \mathbb{F}_2^{\mathcal{N}_{indep\ eqns} \cdot n}$ represent the coefficients of the linear part b_i. Figure 5 represents such matrices. The grey vector represents the list of quadratic variables of type $(x_i x_j)$, the light-grey vector represents the linear and constant variables.

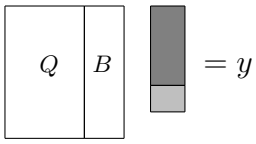

Fig. 5. Linearized System

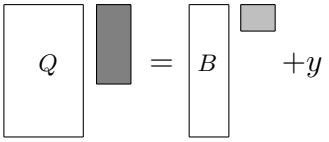

Fig. 6. Rewritten linearized system

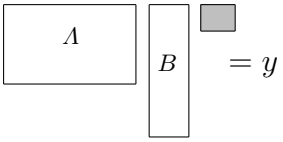

Fig. 7. Linear system

step 3 We compute the rank of matrix Q.

- If Q is full rank[1], then we invert the system by applying Gaussian elimination on Q which is enough to recover the secret seed $x_1, \ldots, x_n$.

- Else Q is not full rank but the condition $\mathcal{N}_{var}(n) - \mathcal{N}_{indep\ eqns} \leq c$ still allows to recover the secret because the rank defect is small. Indeed, the condition can be reformulated as $\mathcal{N}_{indep\ eqns} - \binom{n}{2} \geq n - c$, so at least $n - c$ linear equations can be created:
  step 3.a We rewrite the system differently as in Figure 6.
  step 3.b We derive a matrix Λ for left kernel of Q such that $\Lambda Q = 0$. Λ has at least $n - c$ independent rows.
  step 3.c We multiply the system in Figure 6 by Λ and obtain a linear system as in Figure 7.
  step 3.d With high probability $\mathsf{rank}(\Lambda B) \geq n - c$, with the c linear equations obtained by collisions, we invert the linear system in x_i, giving the secret seed.

Since the costliest step in attack is the inversion of a matrix of size $\binom{n}{2}$, the complexity is $O(n^{2\cdot\omega})$. It then leads to the following proposition.

Proposition 3. *Let Goldreich's PRG be instantiated with n, s, and P_5. Let c be the number of collisions and $\mathcal{N}_{indep\ eqns}$ be the number of linearly independent quadratic polynomials generated with the previous generation. If $0 \leq \mathcal{N}_{var}(n) - \mathcal{N}_{indep\ eqns} \leq c$ the previous algorithm recovers the seed with high probability with time complexity $O(n^{2\cdot\omega})$.*

Creating and counting quadratic equations. In order to satisfy Proposition 3's hypothesis, one must draw $\mathcal{N}_{indep\ eqns}$ linearly independent quadratic equations such that

$$\mathcal{N}_{indep\ eqns} \geq \mathcal{N}_{var}(n) - c\ .$$

In order to achieve it, in the following we introduce a (non exhaustive) list of ways to create new quadratic polynomials. In each case, equations are grouped in a type. We denote by $\mathcal{N}_{T_i}$ the number of equations following from Type i. Unfortunately, predicting the linear dependencies with these new equations is a difficult task for a system with such a structure. For each type, we will remove all redundant equations (also with other types) and assess the number. The linear independence will only be conjectured from experiments.

Let us suppose that an instance of Goldreich's PRG with (n, s) is drawn and gives $m = n^{\mathsf{s}}$ equations $E_1, \ldots, E_m$ evaluated in the secret seed $x_1, \ldots, x_n$ such that for $i \in [m]$,

$$x_{\sigma_1^i} + x_{\sigma_2^i} + x_{\sigma_3^i} + x_{\sigma_4^i} x_{\sigma_5^i} = y_i \qquad (E_i)$$

where $y_1, \ldots, y_m \in \mathbb{F}_2$ is the output.

Type 0. The original system. The first quadratic equations are the system itself composed of n^{s} quadratic equations. If the system has linear dependencies between equations, then a distinguisher is found and the PRG is broken. We then consider that all equations are linearly independent. All the

[1] Some quadratic monomial may never appear in the system, which induces columns of zeros in the matrix. The system is considered full rank if it is when these columns are removed.

new quadratic equations will come from this system. To avoid redundancy in the next constructions, we remove one equation from each collision, thus $\mathcal{N}_{T_0} = n^{\mathsf{s}} - c$.

Type 1. Generated individually. New quadratic polynomials can be derived directly from each equation E_i with $i \in [m]$. Let us fix $i \in [m]$. In the field $\mathbb{F}_2$, the equation $x^2 = x$ gives

$$\begin{aligned} x_{\sigma_1^i} + x_{\sigma_2^i} + x_{\sigma_3^i} + x_{\sigma_4^i} x_{\sigma_5^i} = y_i \rightarrow x_{\sigma_1^i} x_{\sigma_5^i} + x_{\sigma_2^i} x_{\sigma_5^i} + x_{\sigma_3^i} x_{\sigma_5^i} + x_{\sigma_4^i} x_{\sigma_5^i} = y_i x_{\sigma_5^i} \\ \rightarrow x_{\sigma_1^i} x_{\sigma_4^i} + x_{\sigma_2^i} x_{\sigma_4^i} + x_{\sigma_3^i} x_{\sigma_4^i} + x_{\sigma_4^i} x_{\sigma_5^i} = y_i x_{\sigma_4^i} \end{aligned}$$

Thus, the set of quadratic equations generated from E_i is

$$\{zE_i \mid \forall z \in \{x_{\sigma_4^i}, x_{\sigma_5^i}\}\} .$$

Then, considering all i's in $[m]$, $2 \cdot \mathcal{N}_{T_0} = 2n^{\mathsf{s}} - 2c$ new equations can be created. A linear dependence in these equations would also lead to a distinguisher, then we consider that all these equations are linearly independent, thus $\mathcal{N}_{T_1} = 2n^{\mathsf{s}} - 2c$.

Remark 1. If we combine equations of Type 0 with equations of Type 1, a small number of linear equations can follow. Indeed, take the following example

$$x_{\sigma_1^i} + x_{\sigma_2^i} + x_{\sigma_3^i} + x_{\sigma_4^i} x_{\sigma_5^i} = y_i \rightarrow x_{\sigma_1^i} x_{\sigma_5^i} + x_{\sigma_2^i} x_{\sigma_5^i} + x_{\sigma_3^i} x_{\sigma_5^i} + x_{\sigma_4^i} x_{\sigma_5^i} = y_i x_{\sigma_5^i} .$$

If the quadratic monomials $x_{\sigma_1^i} x_{\sigma_5^i}$, $x_{\sigma_2^i} x_{\sigma_5^i}$ and $x_{\sigma_3^i} x_{\sigma_5^i}$ appear in Type 0 equations, then each quadratic term can be replaced by the linear part. Thus, a new linear equation of weight up to 13 is created. The expected number of such linear equations is

$$\mathcal{N}_{extra\ lin}(n, \mathsf{s}) = 2 \cdot \mathcal{N}_{T_0} \cdot \left(\frac{\mathcal{N}_{T_0}}{n\binom{n}{2}} \right)^3 \approx 2^4 \cdot n^{4\mathsf{s}-6} .$$

This number is low, so these equations are added to the linear equations coming from collisions. In other words, from now on, $c \leftarrow c + \mathcal{N}_{extra\ lin}(n, \mathsf{s}) \approx c$.

Type 2. From collisions. According to Definition 5, a collision is a couple (i, j) such that the sum of E_i and E_j generates a linear equation of shape $x_{\sigma_1^i} + x_{\sigma_2^i} + x_{\sigma_3^i} + x_{\sigma_1^j} + x_{\sigma_2^j} + x_{\sigma_3^j} = y_i + y_j$. Thus, the set of quadratic equations generated from a linear equation L is

$$\{zL \mid \forall z \in \{x_1, \ldots, x_n\}\} .$$

Then, $n \cdot c$ quadratic equations can be created. A linear dependence in these equations would lead to a distinguisher with success probability higher than $1/2$, then we consider that all these equations are linearly independent, thus $\mathcal{N}_{T_2} = n \cdot c$.

Type 3. From semi-collisions. Let us first introduce the definition of a semi-collision.

Definition 6 (semi-collision). *A semi-collision is a couple $(i, j) \in [m]^2$ such that*

- *$i \neq j$*
- *(i, j) is not a collision*
- *there exists a $k \in [n]$ such that*

$$x_k | x_{\sigma_4^i} x_{\sigma_5^i} \text{ and } x_k | x_{\sigma_4^j} x_{\sigma_5^j}$$

Example 1. The following equations,

$$\begin{aligned} x_1 + x_2 + x_3 + x_7 x_{10} = y_1 \quad (E_1) \\ x_4 + x_5 + x_6 + x_7 x_8 = y_2 \quad (E_2) \end{aligned} \tag{5}$$

induces $(1, 2)$ as a semi-collision because $x_7 | x_7 x_{10}$ and $x_7 | x_7 x_8$.

Lemma 3. *When a semi-collision (i, j) occurs, an extra quadratic equation of shape $x_{\sigma_{5\ or\ 4}^{j}} E_i + x_{\sigma_{4\ or\ 5}^{i}} E_j$ can be generated.*

This can be easily seen on an example. In Example 1, one can generate a new quadratic equation:

$$x_8x_1 + x_8x_2 + x_8x_3 + x_{10}x_4 + x_{10}x_5 + x_{10}x_6 = x_8 \cdot y_1 + x_{10} \cdot y_2 \qquad (x_8 \cdot E_1 + x_{10} \cdot E_2)$$

Lemma 4. *The total number of semi-collisions can be approximated by*

$$\mathcal{N}_{semi\ collisions} = n\binom{2n^{-1}(n^{\mathsf{s}} - c)}{2} .$$

Proof. Let p be the probability that a fixed variable x_i appears in the quadratic term of a fixed Type 0 quadratic equation. Thus, $\mathsf{p} = \frac{2}{n}$. For a variable x_i, there are on average $(m - c)\mathsf{p} = 2n^{-1}(n^{\mathsf{s}} - c)$ elements[1] that have x_i in their quadratic term. Inside this set of $2n^{-1}(n^{\mathsf{s}} - c)$ elements, there are $\binom{2n^{-1}(n^{\mathsf{s}}-c)}{2}$ couples. To get all the semi-collisions and collisions, we multiply the previous equation by n. □

Removing redundant equations inside Type 3 If naively generated following Proposition 4's proof, many equations are redundant. To compute a correct assessment of the significant Type 3 equations, we will remove several redundant equations. Let us study a phenomenon that is at the origin of many redundancies. Look at the following example :

$$x_1 + x_2 + x_3 + x_{10}x_{11} = y_1 \qquad (E_1)$$
$$x_4 + x_5 + x_6 + x_{11}x_{12} = y_2 \qquad (E_2)$$
$$x_7 + x_8 + x_9 + x_{10}x_{12} = y_3 \qquad (E_3)$$

Among the three semi-collisions concerning x_{10}, x_{11} and x_{12}, one is exactly the sum of both other. Then, when a "cycle" of size 3 appears in the quadratic terms, one semi-collision should be ignored. This makes $\mathcal{N}_{T3}$ significantly smaller than $\mathcal{N}_{semi\ collisions}$. Let $\mathcal{N}_{cycles}$ be the expected number of these "cycles" of size 3 in a random instance Goldreich's PRG . $\mathcal{N}_{cycles}$ can be approximated by the following:

$$\mathcal{N}_{cycles} \approx \frac{1}{\binom{\binom{n}{2}}{3}} \cdot \binom{n}{3} \cdot \binom{m}{3} \in O(n^{3\mathsf{s}-3}) .$$

Then, the remaining number of linearly independent equations is upper bounded by $\mathcal{N}_{semi\ collisions} - \mathcal{N}_{cycles}$. One equation per cycle is removed and all other equations are kept and counted in $\mathcal{N}_{T3}$.

$$\mathcal{N}_{T3} = \mathcal{N}_{semi\ collisions} - \mathcal{N}_{cycles}$$

Proposition 4. *The total number of linearly independent quadratic equations that can be generated with the previous types of equations is upper-bounded by*

$$\mathcal{N}_{indep\ eqns} \leq \mathcal{N}_{T0} + \mathcal{N}_{T1} + \mathcal{N}_{T2} + \mathcal{N}_{T3} := \mathcal{N}_{eqn}(n, \mathsf{s}) \in O(n^{2\mathsf{s}-1}) .$$

Asymptotically, $\mathsf{s} < 1.5 \implies \mathcal{N}_{eqn}(n, \mathsf{s}) < \mathcal{N}_{var}(n)$ which makes the linearization impossible. This result comes with no surprise since it is part of the asymptotic security assumptions. However, for many instances (when $n < 2^{14}$), $\mathcal{N}_{eqn}(n, \mathsf{s}) \approx \mathcal{N}_{var}(n)$. In the next section, we provide conditions on n and s such that a polynomial seed recovery is possible with non-negligible probability.

Conjectured bound on vulnerable parameters. Proposition 3 condition ($\mathcal{N}_{var}(n) - \mathcal{N}_{indep\ eqns} \leq c$) does not easily give a bound in terms of parameters. Indeed, $\mathcal{N}_{indep\ eqns}$ is hard to assess because the linear independence of equations form types 0, 1, 2 and 3 is non-trivial to prove.
However, extensive experiments on small parameters support $\mathcal{N}_{eqn}(n, \mathsf{s}) \approx \mathcal{N}_{indep\ eqns}$. That is what allows us to make the following conjectured limit parameters for this polynomial attack:

$$\mathcal{N}_{eqn}(n, \mathsf{s}) > \mathcal{N}_{var}(n) - c \qquad \text{(Heuristical limit)}$$

[1] This is a worst-case approximation

Experiment. We implemented this attack with a proof of concept using Magma CAS.[1] For each value $n \in \{100, 110, 120, \ldots, 240\}$, we found out that if (n, s) are such that $\mathcal{N}_{eqn}(n, \mathsf{s}) \gg \mathcal{N}_{var}(n)$, the attack succeeds with high probability which corroborates the theory. For a given n, we measured the limit stretch s for which the success probability goes under 50%. Indeed, in Fig. 8, the dots represent the experiments, the line corresponds to the equality $\mathcal{N}_{eqn}(n, \mathsf{s}) = \mathcal{N}_{var}(n) - c$ (Heuristical limit) which was computed discretely in another Magma code. The estimation of Heuristical limit was a worst-case assessment, so it is not surprising that some experimental limits are actually slightly below the line.

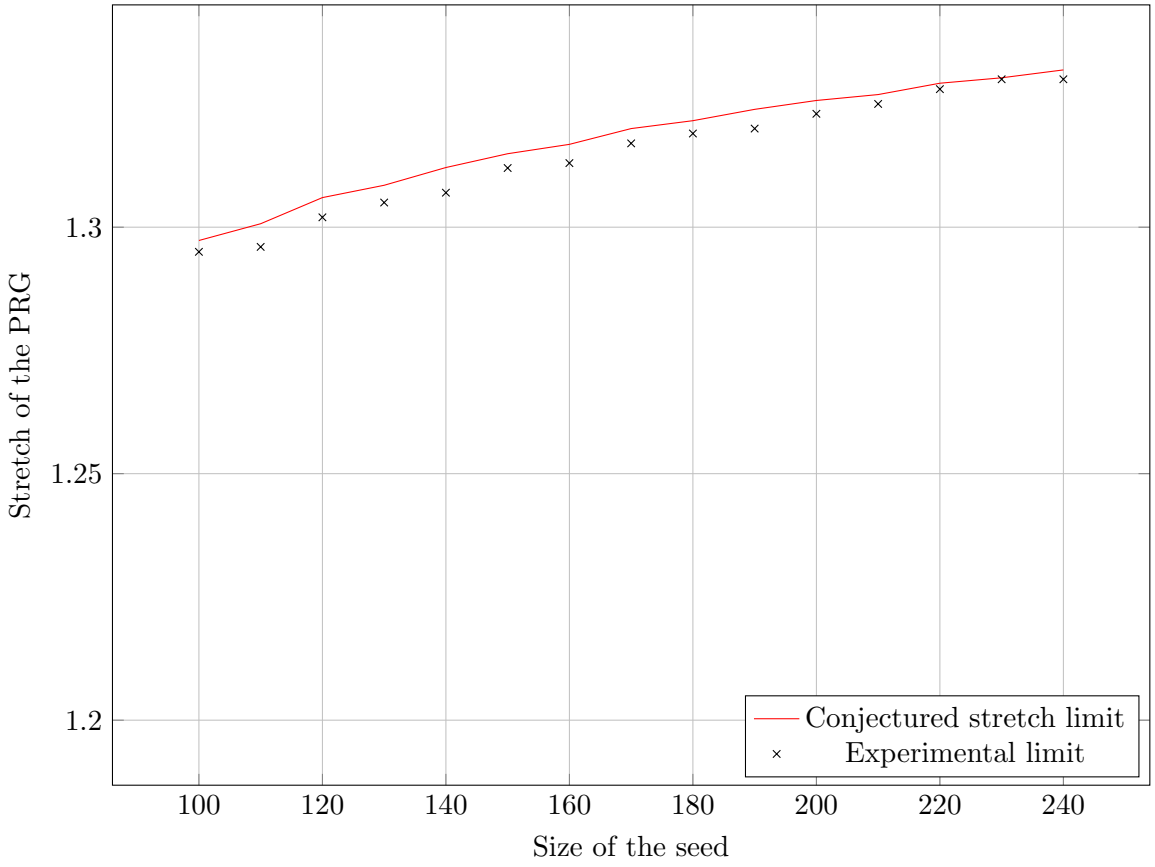

Fig. 8. Experiment

Heuristic 1 ***Extrapolation for higher*** n.
For any set of parameters (n, s) *such that Equation Heuristical limit is verified, we conjecture that there is a polynomial seed recovery attack for Goldreich's PRG with* P_5 *with cost* $O(n^{2\omega})$.

We can notice that if $n < 2^{14}$ then the complexity is lower than 2^{80}.

In Fig. 9, we represent the extrapolated heuristic bound on (n, s). Above the line, the sets of parameters are conjectured to be vulnerable to this polynomial attack.

4.2 Gröbner Approach

The most efficient algebraic attack is using efficient Gröbner basis algorithms such as Faugères F4 [Fau99] and F5 [Fau02]. It consists in a succession of linearization attempts where the degree of the linearization is incremented at each step. For each linearization attempt, all polynomial combinations are exhausted in a smart way in order to generate as many new equations as possible. However hard to

[1] The Magma code can be found at https://github.com/LuMopY/SecurityGoldreichPRG

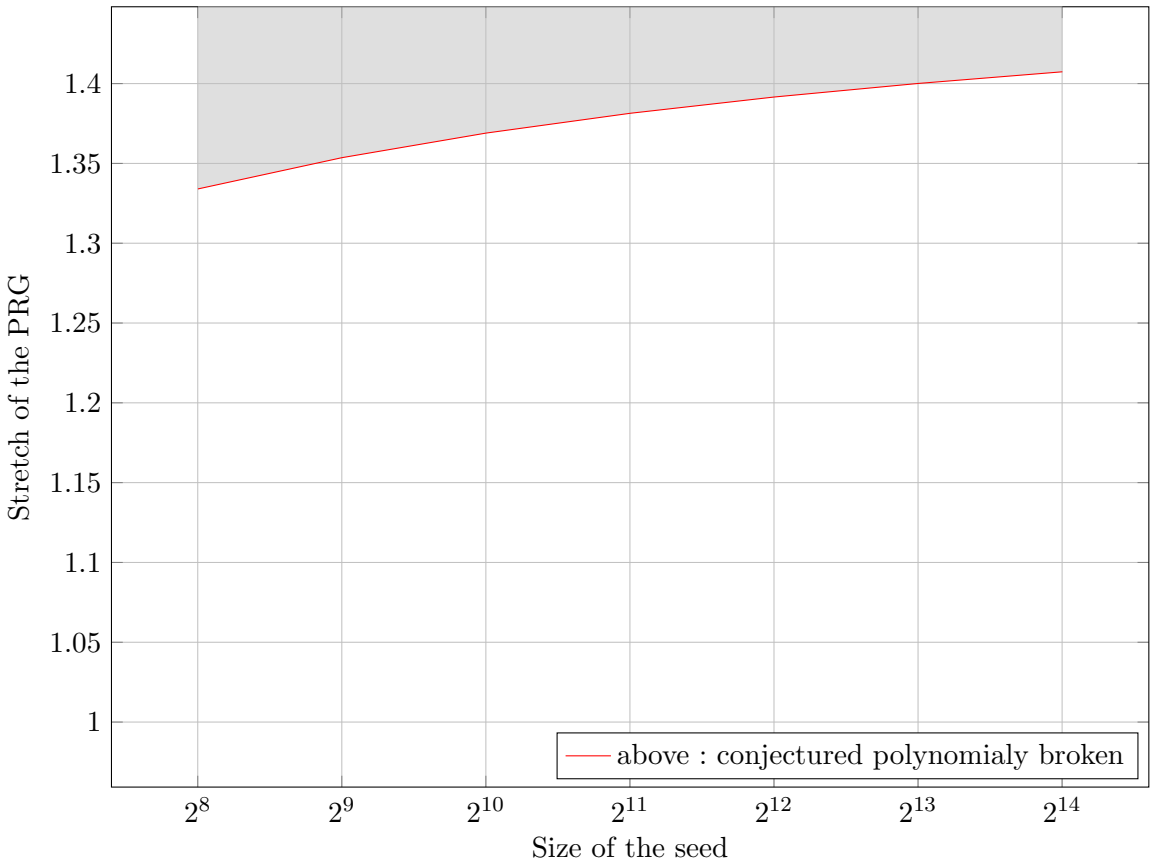

Fig. 9. Extrapolation graph

assess (see Bardet, Faugère, Salvy and Yag's work [BFSyY]), Gröbner basis computation's complexity is dominated by Gaussian elimination on the smallest invertible Macaulay matrix. This Macaulay matrix contains coefficients associated with the monomials of a fixed degree. We denote by *degree of regularity* or D_{reg}, the degree of the monomials associated with the invertible Macaulay matrix. In [BFSyY], under certain hypotheses, the degree of regularity for a random Boolean quadratic system is bounded by

$$-n^{\mathsf{s}} + \frac{n}{2} + \frac{n}{2}\sqrt{2n^{2\mathsf{s}-2} - 10n^{\mathsf{s}-1} - 1 + 2(n^{\mathsf{s}-1}+2)\sqrt{n^{\mathsf{s}-1}(n^{\mathsf{s}-1}+2)}}\,.$$

This bound is too generic and does not represent what happens for practical (n, s). Goldreich's PRG structure allows to drastically reduce the degree of regularity. We conjecture an upper bound on the degree of regularity for certain parameters based on Section 4.1 attack results and that is observed to be true in our experiments.

Proposition 5. *If the attack of Section 4.1 recovers the secret for one instance of Goldreich's PRG , the degree of regularity D_{reg} is 3 and drops to 2 for the resolution on this instance.*

Indeed, the performance of Faugères F4 or F5 algorithm on Goldreich's PRG is strictly superior to the attack presented in Section 4.1. Indeed, the three types of equations found in Step 1 form a subset of the equations derived from Gröbner basis algorithm up to degree 3. Then, if the subsystem is invertible with a degree-two linearization, Gröbner basis algorithm will also be able to invert it with a degree-two linearization. There is a subtlety because when computing the Gröbner basis, the maximal degree of polynomials involved is actually 3: for finding semi-collisions, the quadratic polynomials need to be multiplied by a monomial. But then, once enough semi-collisions are found, the Gröbner basis algorithm falls back into solving a degree-two system. This phenomenon is called a *degree fall*.

Experimental results. To experiment the performance, we used the Gröbner basis algorithms of Magma CAS. The Magma code is then very simple as it consists in computing GroebnerBasis(System,3) which calls a Boolean variant of Faugère F4 algorithm. For each computation, we checked that the degree fall happened and the inversion was done with a degree 2. For each value $n \in \{100, 110, 120, \ldots, 240\}$ and the conjectured limit stretch for 50% success, we ran 100 seed recoveries and Gröbner basis algorithm was able to recover around than 90% of the seeds. We finally conclude that according to the conducted experiments, Heuristic 1 is observed to be true for small values of n.

Remark 2. Gröbner basis performance was able to attack more parameters with lower stretches (often below $\mathsf{s} = 1.25$) with degree of regularity 2. So, some parameters below the heuristic bound may also be vulnerable.

Increasing the degree of regularity. Since we consider 80 bits of security, we want the cost of a degree D_{reg} linearization to be doable with at most 2^{80} operations. A degree D_{reg} linearization corresponds to a Gaussian elimination on a system with $\binom{n}{n-D_{reg}}$ variables. Then, D_{reg} should verify:

$$\binom{n}{n-D_{reg}}^{\omega} < 2^{80} .$$

This implies that D_{reg} cannot be higher than 2 for $n > 512$. For $n \leq 512$, a degree-three linearization might solve more (n, s) instances. We leave this study as future work.

4.3 Conclusion

We described in Section 3 a guess-and-determine attack against Goldreich's PRG. In this section, we complement this result with an analysis of the security of Goldreich's PRG against a degree-two linearization attack (*à la* Gröbner). We represent on Figure 10 the range of parameters for which Goldreich's PRG is conjectured to have 80 bits of security against those two attacks. As illustrated in the graph, the guess-and-determine approach targets more parameters for low n while the linearization attack performs better for $n > 4000$. Although Goldreich's PRG is conjectured to be theoretically secure for a stretch approaching 1.5 by an arbitrary constant, our analysis shows that a very large seed must be used to achieve at least 80 bits of security with such a stretch. In particular, if a stretch of 1.4 is needed, no seed smaller than 5120 bits should be used. Similarly, for a stretch as small as 1.1, the seed must be at least 512 bits long.

5 Generic Attacks against Goldreich's PRG

Beyond the predicate P_5 we investigate the security of other predicates for higher stretches, and show that the considered criteria are not sufficient to determine the security. In Section 6, we prove that the number of independent annihilators of the predicate has to be taken into account. Hence, the algebraic immunity is not enough, as we provide a new bound on the stretch that refines the theorem of Applebaum and Lovett [AL16]. On the other side, we provide in this section an improvement of the guess-and-determine technique, combined with an algebraic attack. This generalization can be seen as a hybrid attack as defined in [Bet11].

5.1 A Subexponential-Time Algorithm

The theorem of Applebaum and Lovett for polynomial-time algorithms regarding algebraic attacks can be improved, as shown in Section 6. In this section, we focus on subexponential-time algorithms. The idea here is to generalize the seed recovery attack of Section 3 against the PRG instantiated with the predicate P_5, to all other considered predicates. Therefore we generalize the attack to all $\mathsf{XOR}_\ell\mathsf{M}_k$ predicates and then more particularly to the $\mathsf{XOR}_\ell\mathsf{MAJ}_k$ predicates.

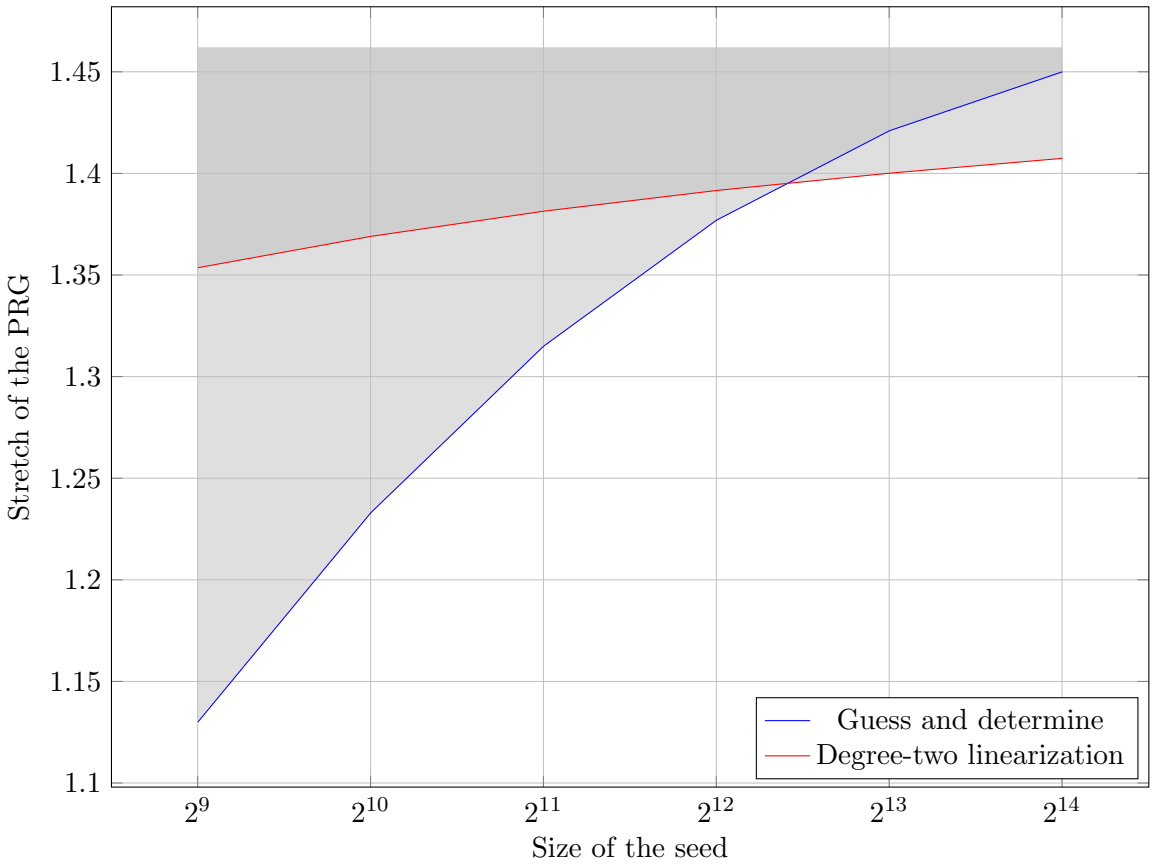

Fig. 10. Limit stretch for vulnerable parameters with 80 bits of security against both guess and determine (Section 3) and degree-two linearization attacks (See Appendix 4). The grey zone above the curves denotes the insecure choices of parameters.

Combinatorial Conjecture. As for Section 3, the success of the seed recovery attack will depend on the inversibility of a large subsystem of equations related to the variables. We give a generalization of the combinational conjecture of Section 3, and of Hypothesis 1. For each attack, we will refer to the particular parameters of this conjecture and hypothesis.

We consider the following conjecture, parametrized by the bound β, the locality d, and j the order of linearization. Define, for $i = 1$ to 2^β, $\mathcal{D}_{n',i}$ to be the distribution over $\mathbb{F}_2^{n' \times n'}$ obtained by sampling the hypergraph of Goldreich's PRG at random (with locality d), taking a subset of ℓ variables which suffices to recover $n' + \ell$ linear equations in the n' linearized variables ($n' = \sum_{i=1}^{j} \binom{n}{j}$) (using a straightforward generalization of the *selection phase* algorithm (see Section 3.1)), and outputting the $n' \times n'$ matrix $M_{n'}$ of the linear system obtained by assigning bit values to the selected linearized variables (according to an arbitrary injection of $\{0,1\}^\ell$ into $[1..2^\beta]$, which exists when $\beta \geq \ell$). We truncate to n' equations for simplicity).

Hypothesis 2 *There exists a constant γ such that for every sufficiently large $n' \in \mathbb{N}$, for every $i \leq 2^\beta$, the matrix M_i contains with overwhelming probability an invertible subsystem of $\gamma \cdot n'$ equations, where the probability is taken over the coins of $M_i \xleftarrow{\$} \mathcal{D}_{n',i}$.*

Remark 3. Note that for the attack of Section 3, the specific parameters of the conjecture are $\beta = \lfloor n^2/2m + 1 \rfloor$ for the bound, $d = 5$ for the locality, and $j = 1$ for the order of linearization, meaning that no linearization is applied.

The principle. Let n be the size of the seed of the PRG with stretch s, and let P be a predicate with locality d. The general idea is to guess r variables of the seed, and solve the corresponding system of equations for each possible value of those r bits. For each equation obtained, an equation of smaller or equal degree can be derived using the principle of the algebraic immunity. Then, the complexity of

the attack mainly depends on the values of r and the algebraic immunity of the functions we obtain. It corresponds to the general principle of algebraic attacks with guess and determine ([MJSC16]), for which we can affine the complexity in the particular case of $\mathsf{XOR}_\ell\mathsf{M}_k$ predicates. We begin by considering the complexity of an attack targeting the degree of the M predicate after guessing some bits, based on the following remark:

Remark 4. As soon as $k-1$ variables among the k variables of M are fixed, a linear equation can be found, as the output of M depends on only one variable and as XOR_ℓ is linear.

The attack. Our subexponential time algorithm works as follows:

step 1 Fix r variables of the seed $(x_{i_1}, \ldots, x_{i_r})$, with $r \in O\left(n^{\frac{k-s}{k-1}}\right)$.
step 2 For all 2^r possible values of $x_{i_1}, \ldots, x_{i_r}$, recover the corresponding linear system of equations.
step 3 Solve the system in $(n-r)^\omega$ operations; if there is a contradiction, go back to step 2, otherwise add the solution to the list.
step 4 Return the list of solutions.

This attack works as long as the system of linear equations obtained in Step 3 above contains an invertible subsystem of size sufficiently large to recover the seed. It is equivalent to assuming Hypothesis 2 relatively to the parameters $\beta \in O\left(n^{\frac{k-s}{k-1}}\right)$, $d = k + \ell$, and $j = 1$.

Complexity analysis. The complexity is dominated by Step 3, as we repeat this step 2^r times (we have to solve a system of linear equations of size $n - r$ for each possible values of the r bits), the complexity of this algorithm is subexponential: $O(n^\omega 2^r)$. Eventually, the final complexity is determined by the following proposition:

Proposition 6. *For an overwhelming proportion of Goldreich's* PRG *instantiated with a* $\mathsf{XOR}_\ell\mathsf{M}_k$ *predicate, under Hypothesis 2 with parameters* $\left(O\left(n^{\frac{k-s}{k-1}}\right), d = k + \ell, j = 1\right)$ *on step 2 system, the complexity order of the previous algorithm can be approximated by :*

$$2^{n^{\frac{k-s}{k-1}}} \cdot n^\omega \ .$$

Proof. First, we need to approximate the number of linear equations we can get: for a given output bit, the probability that at least $k-1$ over the k bits of M are fixed is given by:

$$\frac{\binom{r}{k-1}\binom{n-r}{1}}{\binom{n}{k}} + \frac{\binom{r}{k}\binom{n-r}{0}}{\binom{n}{k}} \ ,$$

as to get j over k bits of M fixed there are $\binom{r}{j}$ possibilities to select these fixed bits, $\binom{n-r}{k-j}$ to select the $k-j$ non-fixed bits, over the $\binom{n}{k}$ possibilities. Multiplying this probability by n^s gives us the average number of linear equations that we get using this technique. As soon as the number of those equations linearly independent is greater than or equal to n, we can solve the corresponding linear system (which is the case when we assume that a negligible fraction of those equations are linearly dependent, see Hypothesis 2). If there is no solution the guess is refuted, otherwise it is compared to the PRG's output (Note that it is sufficient to compare $O(n)$ output bits, as no more than one seed gives this exact output with overwhelming probability).

As k is smaller than the locality (remember that $d = k + \ell$), and by taking $r \simeq n^{\frac{k-s}{k-1}}$ we can make the following approximations: $\binom{r}{k-1} \simeq r^{k-1}$, $\binom{r}{k} \simeq r^k$, and $\binom{n}{k} \simeq n^k$. Hence, the number of linear equations that we get is approximately:

$$n^s \cdot \left(\frac{r^{k-1}(n-r) + r^k}{n^k}\right) = n^s \cdot \left(\frac{r^{k-1}}{n^{k-1}}\right) \ .$$

Hence, the number of variables that we need to guess is roughly $n^{\frac{k-s}{k-1}}$, giving the corresponding complexity for the algorithm.

□

Remark 5. It is important to notice that the parameter of this attack does not rely directly on the locality, but only on the number k of variables that appear in the non-linear part M, hence, it improves the complexity of [BQ09]. Indeed, the generic complexity of Bogdanov and Qiao is roughly $O(2^{n^{1-(\mathtt{s}-1)/2d}})$ where d denotes the locality, as our algorithm has a complexity that is in $O\left(n^\omega \cdot 2^{n^{1-(\mathtt{s}-1)/(k-1)}}\right)$, with $k-1 < d$, by definition of k.

Moreover, the predicate requires a high resiliency to avoid linear attacks, and one of the most natural constructions to build a resilient function is to add an independent linear part to a function. It corresponds to the $\mathsf{XOR}_\ell\mathsf{M}_k$ predicates, which have a resiliency of at least $\ell - 1$ given by the xor part. It is also possible to build resilient functions differently, which seems to be a better choice regarding this attack. For the case of P_5, we have $k = 2$, that gives us an attack in $O(n^\omega 2^{n^{2-\mathtt{s}}})$.

Possible improvement. This algorithm only relies on the number of variables of the non-linear part, but not on its algebraic immunity. Instead of fixing variables in order to obtain linear equations in the non-linear part of a $\mathsf{XOR}_\ell\mathsf{M}_k$ predicate, an attacker can fix variables in order to recover equations of degree greater than 1. Indeed, using the algebraic immunity of the M predicate, the attacker can recover such equations by fixing less than k bits in the M part. By doing so, it appears that the relevant criterion regarding this attack is no longer the algebraic immunity, neither the r-bit fixing degree defined in [AL16], but a generalization of the two. The efficiency of the attack will depend on the algebraic immunity of the predicates obtained after doing some guesses, and on the probability of getting predicates (in fewer variables) with this algebraic immunity (or smaller). A lower bound on the algebraic immunity that can be obtained with r guesses is given by the r-bit fixing algebraic immunity (introduced first in terms of recurrent algebraic immunity in [MJSC16] to bound the complexity of algebraic attacks combined with guess and determine) defined in the following sense:

Definition 7. *(r-bit fixing algebraic immunity) Let f be a Boolean function with d variables. For any $0 \leq r \leq d$, and $b = (b_1, \ldots, b_r) \in \{0,1\}^r$, $i = (i_1, \ldots, i_r) \in [d]^r$ such that $i_1 < i_2 < \cdots < i_r$, we note $f_{(b,i)}$ the restriction of f where the r variables indexed by $i_1, \ldots, i_r$ are fixed to the value $b_1, \ldots, b_r$. Then f has r-bit fixing algebraic immunity a if*

$$\min\left(\mathsf{AI}(f_{(b,i)}) : i = (i_1, \ldots, i_r) \in [d]^r,\, i_1 < i_2 < \cdots < i_r,\, b \in \{0,1\}^r\right) = a$$

where AI denotes the algebraic immunity.

For the case of $\mathsf{XOR}_\ell\mathsf{M}_k$ predicates we prove in the next part an upper bound on the r-bit fixing algebraic immunity. Thereafter, determining the number of predicates with this algebraic immunity that could be reached guessing r variables will lead to other subexponential time algorithms. We give the description and analysis of this algorithm applied on $\mathsf{XOR}_\ell\mathsf{M}_k$ predicates. However, this algorithm only generalizes the result given by the first algorithm as it considers systems of equations of degree greater than one. But, it does not assume any property on the M predicate, and leads to consider the maximum algebraic immunity that can be provided by this part when some variables are fixed. Considering the principle of the r-bit fixing algebraic immunity, we can try to find guesses which lower this algebraic immunity, leading to an attack with even better complexity.

5.2 Improvement of the subexponential-time algorithm

Proposition 7. *Let P be a $\mathsf{XOR}_\ell\mathsf{M}_k$ predicate, let $j \leq k$, then for all set of j variables of the M part, $z_{i_1}, \ldots, z_{i_j}$ and all $b \in \{0,1\}^j$, the Boolean function f with $d - j$ variables defined as:*

$$f = P|_{z_{i_1}=b_1,\ldots,z_{i_j}=b_j}\ ,$$

has an algebraic immunity smaller than or equal to $\left\lceil \frac{k-j}{2} \right\rceil + 1$.

Proof. $f = \mathsf{XOR}_\ell + g$, where g is a Boolean function with $k-j$ variables. Hence, its algebraic immunity is upper bounded by the value of $\left\lceil \frac{k-j}{2} \right\rceil$. Let h be a non-null Boolean function that cancels g. Then h

is of degree at most $\left\lceil \frac{k-j}{2} \right\rceil$. It appears that the Boolean function h' of $\ell + k - j$ variables defined by:

$$h' = (1 + \mathsf{XOR}_\ell)h \ ,$$

is an annihilator of f. Indeed, if $\mathsf{XOR}_\ell = 1$, then $h' = 0$. Else, if $\mathsf{XOR}_\ell = 0$, then $h' = h$, and $f = g$, so $h'f = hg = 0$, by definition of h. Moreover, as h is non-zero, and h' is constructed using independent variables, h' is necessarily also non-zero. In the same way, if h is canceling $g + 1$, then h' is canceling f and is non-zero, finishing the proof. □

Using Proposition 7, we can generalize the previous algorithm on all $\mathsf{XOR}_\ell\mathsf{M}_k$ predicates, generalizing the result of Proposition 6. For each value of the r-bit fixing algebraic immunity of the M predicate we consider a subexponential time algorithm. However, this generalization is a slightly improvement as it makes sense only if the stretch is sufficiently big.

Proposition 8. *Let j be an integer such that $0 < j \leq k$ and $\mathsf{s} > \lceil \frac{k-j}{2} \rceil + 1$, for an overwhelming proportion of Goldreich's PRG instantiated with a $\mathsf{XOR}_\ell\mathsf{M}_k$ predicate, under Hypothesis 2 with parameters $\left(O\left(n^{\frac{1+j-s+\lceil (k-j)/2 \rceil}{j}} \right), k + \ell, \lceil \frac{k-j}{2} \rceil + 1 \right)$, the seed can be recovered in time complexity of order:*

$$2^r n^{\omega(\lceil \frac{k-j}{2} \rceil + 1)} \ ,$$

where $r \simeq n^{\frac{1+j-s+\lceil (k-j)/2 \rceil}{j}}$.

Proof. The proof is similar to the proof of Proposition 6:

First, we approximate the number of equations of degree at most $\left\lceil \frac{k-j}{2} \right\rceil + 1$ we can get using Proposition 7. The probability of recovering a predicate of degree at most $\left\lceil \frac{k-j}{2} \right\rceil + 1$ when r variables are fixed is at least:

$$\sum_{i=j}^{k} \frac{\binom{r}{i}\binom{n-r}{k-i}}{\binom{n}{k}} \ ,$$

as it corresponds to the probability of fixing j variables in the M part.

By multiplying this with n^{s}, that is the total number of equations, we get the total number of equations that we get in average of degree at most $\left\lceil \frac{k-j}{2} \right\rceil + 1$. As soon as the number of those equations linearly independent is greater than or equal to:

$$\sum_{i=1}^{\lceil \frac{k-j}{2} \rceil + 1} \binom{n}{i} \ ,$$

we can solve the corresponding linearized system (which is the case when we assume that a low number of those equations are linearly dependent, *i.e.* assuming Hypothesis 3), giving a complexity order of $n^{\omega \cdot (\lceil \frac{k-j}{2} \rceil + 1)}$.

If there is no solution the guess is refuted, otherwise it is compared to the PRG's output.

As k is a constant, we can approximate the number of linearly independent equations as:

$$n^{\mathsf{s}} \sum_{i=j}^{k} \frac{r^i (n-r)^{k-i}}{n^k} \ .$$

Now taking $r \simeq n^{\frac{1+j-\mathsf{s}+\lceil (k-j)/2 \rceil}{j}}$, and as $\mathsf{s} > 1 + \lceil (k-j)/2 \rceil$, it allows us to approximate this expression by:

$$n^{\mathsf{s}} \sum_{i=j}^{k} \frac{r^i n^{k-i}}{n^k} \simeq n^{\mathsf{s}} \frac{r^j}{n^j} \sum_{i=0}^{k-j} \frac{r^i}{n^i} \geq n^{\mathsf{s}} \frac{r^j}{n^j} \ .$$

Replacing by the value of r, we obtain

$$n^{\mathsf{s}} \frac{r^j}{n^j} \simeq n^{\lceil \frac{k-j}{2} \rceil + 1} .$$

Hence, we get enough equations of degree at most $\left\lceil \frac{k-j}{2} \right\rceil + 1$, that can be solved using the linearization technique.

□

In the following, we show on the XOR-MAJ predicates how only taking into account specific values of guessed bits (but changing the positions that we guess) enables to target a low algebraic immunity with enough equations.

5.3 Application to XOR-MAJ Predicates.

In the previous algorithms, we fix r bits that never change, but we test all possible values for those bits. However, it might be of interest to change the bits that we guess, by taking into account a specific value for those bits, such that we decrease more drastically the degree of the equations that we get. Using the notations of Definition 7, it boils down to finding values of $b \in \{0,1\}^r$ such that $\mathsf{AI}(f_{(i,b)})$ is low for enough i.

Let us consider the $\mathsf{XOR}_\ell\mathsf{MAJ}_k$ predicate (Definition 3), then the initial subexponential algorithm breaks the construction with complexity $O(n^\omega 2^{n^{(k-\mathsf{s})/(k-1)}})$, and its generalization with complexity:

$$O\left(2^{n \frac{1+j-\mathsf{s}+\lceil (k-j)/2 \rceil}{j}} n^{\omega(\lceil \frac{k-j}{2} \rceil + 1)}\right),$$

for all integers j such that $1 \leq j \leq k$. Moreover, this algorithm is an improvement only for bigger stretches. In the following, we change the way we make our guesses, in order to capture how the r-bit fixing algebraic immunity is a relevant criterion.

In these algorithms, one can notice that fixing j bits among the k variables that appear in the majority function can derive different degrees of equations, depending on the value of the bits that are guessed: fixing $\lceil \frac{k}{2} \rceil$ bits all to 0 (or all to 1) will derive directly linear equations. Indeed, for the majority function, if strictly more than half of the bits are supposed to be all zero, then the corresponding output has to be 0 by definition of the majority, and respectively 1 if all these bits are ones. On the other side, fixing a quarter of bits to be ones and a quarter of bits to be zero will derive another majority function taken other half of the bits, which is clearly non-linear for k big enough.

Hence, instead of fixing r bits and guess all possible values of those bits, we choose r bits, guessing that all those bits are all one or all zero, and repeat this until the guess is right (the position of the r guessed variables changes, not the value). This particular guess-and-determine is exactly what Duval, Lallemand and Rotella investigated in [DLR16] on the FLIP family of stream ciphers (and which complexity can be bounded through the r-bit fixing algebraic immunity, [MJSC16] Section 3.4).

Description of the algorithm.

step 1 Fix randomly r variables of the seed $(x_{i_1}, \ldots, x_{i_r})$.
step 2 Assume that all of them are equal to zero, solve the corresponding linear system, add the solution to the list.
step 3 Assume that all the r variables are equal to one, solve the corresponding linear system, add the solution to the list.
step 4 If in the solution list there is one with no contradiction with the PRG output, output the solution as the seed. Otherwise, empty the list and go back to Step 1.

As for the other seed recovery attacks, the success depends on the invertibility of a large proportion of a linear system. More particularly, the attack is effective here if a sufficiently important subpart of the linear systems described in step 2 or step 3 are invertible. Note that this is not directly related to Hypothesis 2, which focuses on distribution of systems where at most β variables are selected, and where all the affectations of these variables are considered. Here, the affectation of the variables is fixed, all 0 or all 1, and we focus on the systems given by randomly selecting β variables.

Combinatorial Conjecture. We consider the following conjecture, parametrized by the bound β, the locality d, j the order of linearization, and V the set of affectations. Define, for $i = 1$ to $|V|\binom{n}{\beta}$, $\mathcal{D}'_{n',i}$ to be the distribution over $\mathbb{F}_2^{n' \times n'}$ obtained by sampling the hypergraph of Goldreich's PRG at random (with locality d), taking a subset of β over n variables, affecting to it a value of V, and outputting the $n' \times n'$ matrix $M_{n'}$ of the linearized system obtained (where $n' = \sum_{i=1}^{j} \binom{n}{j}$), according to an arbitrary fixed ordering of all assignments.

Hypothesis 3 *There exists a constant γ such that for every sufficiently large $n' \in \mathbb{N}$, for every $i \leq |V|\binom{n}{\beta}$, the matrix M_i contains with overwhelming probability an invertible subsystem of $\gamma \cdot n'$ equations, where the probability is taken over the coins of $M_i \xleftarrow{\$} \mathcal{D}'_{n',i}$.*

Complexity analysis. The complexity is dominated by the number of repetitions of Step 2 and Step 3, we determine it through the following proposition:

Proposition 9. *For an overwhelming proportion of Goldreich's* PRG *instantiated with a* $\mathsf{XOR}_\ell\mathsf{MAJ}_k$ *predicate, assuming Hypothesis 3 with parameters $\beta = O(n^{1+\frac{1-s}{\lceil \frac{k}{2} \rceil + 1}}), d = \ell + k, j = 1, V = \{0^n, 1^n\}$ for Step 2 and 3 systems, the seed can be recovered in time complexity of order:*

$$n^\omega 2^{n^{1-\frac{s-1}{\lceil \frac{k}{2} \rceil + 1}}} .$$

Proof. Using the precedent proof, we need to have $j = \lceil \frac{k}{2} \rceil + 1$. Hence, the probability that we recover a linear equation is equal to

$$\sum_{i=j}^{k} \frac{\binom{r}{i}\binom{n-r}{k-i}}{\binom{n}{k}} .$$

Multiplying the expression above with n^{s}, we get the number of linear equations. When the number of those equations linearly independent is greater than or equal to n, we can invert the system. As k is a constant, we can approximate this as

$$n^{\mathsf{s}} \sum_{i=j}^{k} \frac{r^i (n-r)^{k-i}}{n^k} .$$

Now taking $r \simeq n^{\frac{1+j-s}{j}}$ allows us to approximate this expression to

$$n^{\mathsf{s}} \sum_{i=j}^{k} \frac{r^i n^{k-i}}{n^k} \simeq n^{\mathsf{s}} \frac{r^j}{n^j} \sum_{i=0}^{k-j} \frac{r^i}{n^i} \geq n^{\mathsf{s}} \frac{r^j}{n^j} .$$

Replacing by the value of r we have, we get that

$$n^{\mathsf{s}} \frac{r^j}{n^j} \simeq n .$$

Eventually, the number of positions that we have to guess is roughly $n^{\frac{1+j-s}{j}}$, with $j = \lceil \frac{k}{2} \rceil + 1$.

Note that the probability that a guess is correct depends on the Hamming weight of the seed. We circumvent this issue by considering the maximum over the probabilities of the two events the r fixed variables are all 0 and the r fixed variables are all 1. As the seed has at least $\lceil \frac{n}{2} \rceil$ 0 or 1, the maximum is at least:

$$\frac{\binom{\lceil n/2 \rceil}{r}}{\binom{n}{r}},$$

which is equal to 2^{-r} when we approximate $\binom{\lceil n/2 \rceil}{r}$ by $(n/2)^r$ and $\binom{n}{r}$ by n^r.

Hence, the final time complexity order is

$$n^\omega 2^{n^{1-\frac{s-1}{\lceil \frac{k}{2} \rceil + 1}}} .$$

□

This algorithm captures something else than the previous ones, as it shows that one has to consider all possible choices of guesses in order to evaluate exactly the security of such constructions. In other words, it shows that the r-bit fixing algebraic immunity is exactly the relevant criterion to resist our attack, as it defines the smallest algebraic immunity that can be considered for an attack. However, one must also take the probability that a corresponding guess happens on the equations into account. Hence there exists a trade-off between the choice of the good guesses, and the probability that the corresponding equation of small degree can be derived.

6 Independent Annihilators and Polynomial Attack

6.1 On the number of independent annihilators

In [AL16, Theorem 1.4], Applebaum and Lovett showed that if the stretch s is strictly smaller than the *rational degree* e (also called Algebraic Immunity), then there exists a polynomial-time refutation attack.

In symmetric cryptography, the algebraic immunity is a relevant criterion, as it defines the complexity of inverting a certain system of nonlinear equations, but the data complexity is often not an issue for the attacker. However, the context of Goldreich's PRG is very different. Indeed, the degree of the equations is constant, but the constraint is the number of equations we aim at solving, and that is why the stretch has to be smaller than the algebraic immunity.

Moreover, ever since the first attacks using algebraic immunity were found (see [Cou03, CM03]), it has been pointed out that the number of annihilators of the corresponding degree can improve the data complexity of (fast) algebraic attacks on stream ciphers. Since then, this criterion has been studied: for example in [Car06], there are some results on the number of independent equations that can be obtained by only one nonlinear equation. As we are very limited in the number of data here, this criterion is in fact a more relevant one in our setting than the algebraic immunity only.

Example. In order to be more understandable, let us explain an example: we assume that we have the following equation:

$$x_1 + x_2x_3x_4x_5 = 0.$$

Noticing that $1 + x_2$ is an annihilator of $x_2x_3x_4x_5$: $(1 + x_2)(x_2x_3x_4x_5) = x_2x_3x_4x_5 + x_2x_3x_4x_5 = 0$, one can remark that the following equation can be derived multiplying by $1 + x_2$:

$$x_1(1 + x_2) = 0.$$

As x_2, x_3, x_4 and x_5 are playing a symmetric role in the first equation, we get four equations of degree 2 of the type $x_1(1 + x_i) = 0$. All those four equations are derived from the first one, but they are all linearly independent, implying that one equation of degree 4 is *hiding* four equations of degree 2, linearly independent.

Remark 6. Note that in the attack of Section 4 we are already using the fact that a function can have multiple linearly independent annihilators of the same degree. Indeed, the type-1 equations imply degree-two annihilators of P_5: for such equations of the type $P_5x_j = Q_i$ where Q_i is a quadratic polynomial, we get $P_5x_j = P_5Q_i$, implying that $x_j + Q_i$ is a non-null annihilator of P_5.

Generic attack. We focus on a construction that follows Goldreich's idea: We take a predicate P of locality d, of algebraic immunity e, n denotes the size of the seed, and s the stretch. Moreover, we denote N_a^0 the number of independent annihilators of the predicate P of degree at most a, and N_a^1 the number of independent annihilators of the predicate $P + 1$ of degree at most a. By definition, we know that $\forall a < e, N_a^1 = N_a^0 = 0$.

Theorem 1. *Consider a* PRG *following Goldreich's construction with predicate P, let e be the algebraic immunity of P and for $a \geq e$, let N_a^0 and N_a^1 be respectively the number of linearly independent annihilators of P (respectively $P + 1$) of degree at most a, and $N_a = \min(N_a^1, N_a^0)$. If*

$$\mathsf{s} \geq a - \frac{\log(N_a)}{\log(n)} ,$$

then, assuming Hypothesis 2 with parameters $\beta = O\left(c_0 N_a^0 + c_1 N_a^1\right), d, j = a$*, there exists a polynomial-time algorithm that finds a preimage for a given output* $y \in \{0,1\}^{n^{\mathsf{s}}}$ *(or certifies that* y *has no preimage).*

Proof. Let $a \geq e$. For each output bit, the attacker can derive either N_a^0 or N_a^1 linearly independent equations of degree at most a, depending on the value of this bit, let us denote c_0 the number of zeros and c_1 the number of ones. Then the attacker can recover $c_0 N_a^0 + c_1 N_a^1$ equations.

Assuming Hypothesis 2 with parameters $\beta = O\left(c_0 N_a^0 + c_1 N_a^1\right), d, j = a$, we can solve the linearized system, with a total time complexity of roughly $O(n^{a \cdot \omega})$.

To link this number to the stretch, recall that $c_0 + c_1 = n^{\mathsf{s}}$, then $c_0 N_a^0 + c_1 N_a^1 \geq n^{\mathsf{s}} N_a$. As soon as this number of equations is greater than $O(n^a)$, we can apply the linearization technique.

The condition $n^{\mathsf{s}} N_a \geq n^a$, implies directly the following condition

$$\mathsf{s}\log(n) + \log\left(N_a\right) \geq a\log(n)\ ,$$

finishing the proof.

□

However, when considering $\mathsf{XOR}_\ell\mathsf{M}_k$ predicates, we can improve this theorem with the following proposition.

Proposition 10. *Let* P *be a* $\mathsf{XOR}_\ell\mathsf{M}_k$ *predicate with* $\ell > 0$ *then:*

$$\forall a,\ N_a^0 = N_a^1 = N_a\ .$$

Proof. We have $P(x_1+1,\ldots,x_d) = 1 + P(x_1,\ldots,x_d)$. Let f be such that $f(x_1,\ldots,x_d)P(x_1,\ldots,x_d) = 0$, then:

$$f(x_1+1,x_2,\ldots,x_d)(P+1) = f(x_1+1,x_1,\ldots,x_d)P(x_1+1,x_1,\ldots,x_d) = 0\ ,$$

Which implies that if $f(x_1,\ldots,x_d)$ is an annihilator of P, then:

$$f' = f(x_1+1,x_2,\ldots,x_d)\ ,$$

is an annihilator of $P+1$.

Moreover, $f \neq f'$. Indeed, f is a non-null annihilator of P and f' of $P+1$, so if $f = f'$, we get that $f(P+P+1) = 0$ implying $f = 0$. Last but not least, the transformation $x_1 \mapsto x_1 + 1$ is bijective and linear, so it preserves the degree, finishing the proof. □

Putting Theorem 1 and Proposition 10 together, we can improve the theorem of Applebaum and Lovett for $\mathsf{XOR}_\ell\mathsf{M}_k$ predicates by taking $e = a$ and considering only the annihilators of P, which gives us the following corollary.

Corollary 1. *Let* P *be a* $\mathsf{XOR}_\ell\mathsf{M}_k$ *predicate with* $\ell > 0$*, let* e *be the algebraic immunity of* P *and* N_e *the dimension of the vectorial space of the annihilators of* P*. Then if*

$$\mathsf{s} \geq e - \frac{\log(N_e)}{\log(n)}\ ,$$

then there exists a polynomial-time algorithm that finds a preimage for the output or refute the output as an image of the PRG*.*

Note that this attack uses a property of the predicate which is not exactly its algebraic immunity, as for any function of n variables and algebraic immunity e we have $0 \leq N_e \leq \binom{n}{e}$. We study this quantity N_e in the particular case of $\mathsf{XOR}_\ell\mathsf{MAJ}_k$ predicates:

Proposition 11. *Let* P *be a* $\mathsf{XOR}_\ell\mathsf{MAJ}_k$ *predicate, then:*

$$N_{\lceil \frac{k}{2} \rceil + 1} \geq \binom{k}{\lceil \frac{k}{2} \rceil}.$$

Proof. Let consider first the function $1 + \mathsf{MAJ}_k$, this function is equal to 1 on all inputs of Hamming weight lesser than $\lceil \frac{k}{2} \rceil$ and equal to 0 on all other inputs. For any monomial function of degree $\lceil \frac{k}{2} \rceil$, the product with $1 + \mathsf{MAJ}_k$ is then 0, as these functions are equal to 1 only on input with Hamming weight greater than or equal to $\lceil \frac{k}{2} \rceil$. As these monomials are linearly independent, it gives for $1 + \mathsf{MAJ}_k$:

$$N_{\lceil \frac{k}{2} \rceil} \geq \binom{k}{\lceil \frac{k}{2} \rceil}.$$

Then, $1 + \mathsf{XOR}_\ell$ admits the linear function XOR_ℓ as annihilator, so the direct sum $\mathsf{XOR}_\ell\mathsf{MAJ}_k$ admits as annihilators all the products of XOR_ℓ by a degree $\lceil \frac{k}{2} \rceil$ monomial in the k variables of MAJ_k, which are linearly independent functions of degree $\lceil \frac{k}{2} \rceil + 1$, finishing the proof.

□

Putting all together, we can then bring the following corollary for $\mathsf{XOR}_\ell\mathsf{MAJ}_k$ predicates.

Corollary 2. *Let a* PRG *following Goldreich's construction with predicate* $\mathsf{XOR}_\ell\mathsf{MAJ}_k$ *with* $\ell > 0$*, then if*

$$s \geq \left\lceil \frac{k}{2} \right\rceil + 1 - \frac{\log\left(\binom{k}{\lceil \frac{k}{2} \rceil}\right)}{\log n},$$

then there exists a polynomial-time algorithm that finds a preimage for a given output $y \in \{0,1\}^{n^s}$ *(or certifies that* y *has no preimage).*

6.2 Open Questions

The attacks and their variants described in this section and the precedent one ask many open questions. For the polynomial time algorithm using the number of linearly independent annihilators, we do not take into account some dependencies into different equations as explained in Section 4. Hence, the condition on the stretch that we gave could be improved by considering dependencies on the subsets.

For the subexponential-time attack of Section 5 that uses the r bit fixing algebraic immunity, we do not know if the bound given in Proposition 7 is tight, that is if there exist predicates, such that fixing any bits will still derive Boolean functions with fewer variables that reach the maximal algebraic immunity. In other words, is it possible to have a perfect predicate regarding the r bit fixing algebraic immunity? Recalling that it is the relevant criterion in this context.

Moreover, this bound does not depend on the value of the bits that are guessed, whereas this might have an influence, as shown on the XOR-MAJ predicate. For example, the Boolean function $x_0 + x_1x_2x_3x_4$ is of algebraic immunity 2, but fixing x_1 to be 1 will derive a Boolean function that is still of algebraic immunity 2, but fixing $x_1 = 0$ will bring directly a linear equation. Hence, all choices of guess are not equivalent, implying that different choices of guesses could improve the complexity of our subexponential-time algorithm, depending strongly on the predicate.

Last but not least, how the first idea of using different annihilators can improve the subexponential-time algorithms using guess and determine?

Acknowledgments

We thank Jean-Pierre Tillich and Benny Applebaum for useful discussions and observations. We also are indebted to Henri Gilbert and Guénaël Renault for fruitful discussions about Gröbner basis approaches, to the users Kevin P. Costello and loup blanc on the math.stackexchange network who took the time to answer our questions related to the probability of having large invertible subsystems in random sparse matrices, and to the reviewers of ASIACRYPT for their useful comments. This research has been partially funded by ANRT under the programs CIFRE N 2015/1158 and 2016/1583. We acknowledge the support of the French Programme d'Investissement d'Avenir under national project RISQ P141580. The first author was supported by ERC grant 724307 (project PREP-CRYPTO). The third author was supported by the Chist-era project SECODE. The fifth author was partially supported by the French Agence Nationale de la Recherche through the BRUTUS project under Contract ANR-14-CE28-0015.

References

ABR12. B. Applebaum, A. Bogdanov, and A. Rosen. A dichotomy for local small-bias generators. In *TCC 2012*, *LNCS* 7194, pages 600–617. Springer, Heidelberg, March 2012.

ABR16. B. Applebaum, A. Bogdanov, and A. Rosen. A dichotomy for local small-bias generators. *Journal of Cryptology*, 29(3):577–596, July 2016.

ADI+17a. B. Applebaum, I. Damgård, Y. Ishai, M. Nielsen, and L. Zichron. Secure arithmetic computation with constant computational overhead. Cryptology ePrint Archive, Report 2017/617, 2017. http://eprint.iacr.org/2017/617.

ADI+17b. B. Applebaum, I. Damgård, Y. Ishai, M. Nielsen, and L. Zichron. Secure arithmetic computation with constant computational overhead. In *Crypto'17*, pages 223–254, 2017.

AHI05. M. Alekhnovich, E. A. Hirsch, and D. Itsykson. Exponential lower bounds for the running time of dpll algorithms on satisfiable formulas. *Journal of Automated Reasoning*, 35(1-3):51–72, 2005.

AIK04. B. Applebaum, Y. Ishai, and E. Kushilevitz. Cryptography in NC^0. In *45th FOCS*, pages 166–175. IEEE Computer Society Press, October 2004.

AIK08. B. Applebaum, Y. Ishai, and E. Kushilevitz. On pseudorandom generators with linear stretch in nc 0. *Computational Complexity*, 17(1):38–69, 2008.

AL16. B. Applebaum and S. Lovett. Algebraic attacks against random local functions and their countermeasures. In *48th ACM STOC*, pages 1087–1100. ACM Press, June 2016.

App12. B. Applebaum. Pseudorandom generators with long stretch and low locality from random local one-way functions. In *44th ACM STOC*, pages 805–816. ACM Press, May 2012.

App13. B. Applebaum. Pseudorandom generators with long stretch and low locality from random local one-way functions. *SIAM J. Comput.*, 42(5):2008–2037, 2013.

App15. B. Applebaum. The cryptographic hardness of random local functions – survey. Cryptology ePrint Archive, Report 2015/165, 2015. http://eprint.iacr.org/2015/165.

ARS+15. M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. Ciphers for MPC and FHE. In *EUROCRYPT 2015, Part I*, *LNCS* 9056, pages 430–454. Springer, Heidelberg, April 2015.

BCG+17. E. Boyle, G. Couteau, N. Gilboa, Y. Ishai, and M. Orrù. Homomorphic secret sharing: Optimizations and applications. In *ACM CCS 17*, pages 2105–2122. ACM Press, 2017.

Bet11. L. Bettale. *Cryptanalyse algebrique : outils et applications*. PhD Thesis, 2011.

BFSyY. M. Bardet, J.-C. Faugere, B. Salvy, and B. y. Yang. Asymptotic behaviour of the degree of regularity of semi-regular polynomial systems. In *MEGA05, 2005. Eighth International Symposium on Effective Methods in Algebraic Geometry*.

BGI+01. B. Barak, O. Goldreich, R. Impagliazzo, S. Rudich, A. Sahai, S. P. Vadhan, and K. Yang. On the (im)possibility of obfuscating programs. In *CRYPTO 2001*, *LNCS* 2139, pages 1–18. Springer, Heidelberg, August 2001.

BL10. C. Bordenave and M. Lelarge. The rank of diluted random graphs. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete algorithms*, pages 1389–1402. Society for Industrial and Applied Mathematics, 2010.

BQ09. A. Bogdanov and Y. Qiao. On the security of goldreich's one-way function. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 392–405. Springer, 2009.

Buc76. B. Buchberger. A theoretical basis for the reduction of polynomials to canonical forms. *SIGSAM Bull.*, 10(3):19–29, August 1976.

Car06. C. Carlet. On the higher order nonlinearities of algebraic immune functions. In *CRYPTO 2006*, *LNCS* 4117, pages 584–601. Springer, Heidelberg, August 2006.

CCF+16. A. Canteaut, S. Carpov, C. Fontaine, T. Lepoint, M. Naya-Plasencia, P. Paillier, and R. Sirdey. Stream ciphers: A practical solution for efficient homomorphic-ciphertext compression. In *FSE 2016*, *LNCS* 9783, pages 313–333. Springer, Heidelberg, March 2016.

CEMT14. J. Cook, O. Etesami, R. Miller, and L. Trevisan. On the one-way function candidate proposed by goldreich. *ACM Transactions on Computation Theory (TOCT)*, 6(3):14, 2014.

CM01. M. Cryan and P. B. Miltersen. On pseudorandom generators in nc 0. In *International Symposium on Mathematical Foundations of Computer Science*, pages 272–284. Springer, 2001.

CM03. N. Courtois and W. Meier. Algebraic attacks on stream ciphers with linear feedback. In *EUROCRYPT 2003*, *LNCS* 2656, pages 345–359. Springer, Heidelberg, May 2003.

Cou03. N. T. Courtois. Fast algebraic attacks on stream ciphers with linear feedback. In *CRYPTO*, pages 176–194, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

DGM05. D. K. Dalai, K. C. Gupta, and S. Maitra. Cryptographically significant Boolean functions: Construction and analysis in terms of algebraic immunity. In *FSE 2005*, *LNCS* 3557, pages 98–111. Springer, Heidelberg, February 2005.

DLR16. S. Duval, V. Lallemand, and Y. Rotella. Cryptanalysis of the FLIP family of stream ciphers. In *CRYPTO 2016, Part I*, *LNCS* 9814, pages 457–475. Springer, Heidelberg, August 2016.

DMS05. D. K. Dalai, S. Maitra, and S. Sarkar. Basic theory in construction of Boolean functions with maximum possible annihilator immunity. Cryptology ePrint Archive, Report 2005/229, 2005. http://eprint.iacr.org/2005/229.

Fau99. J.-C. Faugere. A new efficient algorithm for computing grobner bases (f4). *Journal of Pure and Applied Algebra*, 139(1):61 – 88, 1999.

Fau02. J. C. Faugere. A new efficient algorithm for computing grobner bases without reduction to zero (f5). In *Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, ISSAC '02, pages 75–83, New York, NY, USA, 2002. ACM.

GGM84. O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions (extended abstract). In *25th FOCS*, pages 464–479. IEEE Computer Society Press, October 1984.

Gol00. O. Goldreich. Candidate one-way functions based on expander graphs. Cryptology ePrint Archive, Report 2000/063, 2000. http://eprint.iacr.org/2000/063.

GRR+16. L. Grassi, C. Rechberger, D. Rotaru, P. Scholl, and N. P. Smart. MPC-friendly symmetric key primitives. In *ACM CCS 16*, pages 430–443. ACM Press, October 2016.

IKOS08. Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai. Cryptography with constant computational overhead. In *40th ACM STOC*, pages 433–442. ACM Press, May 2008.

IPS08. Y. Ishai, M. Prabhakaran, and A. Sahai. Secure arithmetic computation with no honest majority. Cryptology ePrint Archive, Report 2008/465, 2008.

JK77. N. Johnson and S. Kotz. *Urn models and their application: an approach to modern discrete probability theory*. Wiley Series in Probability and Statistics: Applied Probability and Statist ICS Sesction Series. Wiley, 1977.

KSC78. V. Kolchin, B. Sevastianov, and V. Chistiakov. *Random allocations*. Scripta series in mathematics. V. H. Winston, 1978.

laz81. D. lazard. Resolution des systemes d'equations algebriques. *Theoretical Computer Science*, 15(1):77 – 110, 1981.

Lin17. H. Lin. Indistinguishability obfuscation from SXDH on 5-linear maps and locality-5 PRGs. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 599–629. Springer, Heidelberg, August 2017.

LT17. H. Lin and S. Tessaro. Indistinguishability obfuscation from trilinear maps and block-wise local PRGs. In *CRYPTO 2017, Part I*, *LNCS* 10401, pages 630–660. Springer, Heidelberg, August 2017.

LV17. A. Lombardi and V. Vaikuntanathan. Limits on the locality of pseudorandom generators and applications to indistinguishability obfuscation. In *TCC 2017, Part I*, LNCS, pages 119–137. Springer, Heidelberg, March 2017.

Mac64. F. Macaulay. *The Algebraic Theory of Modular Systems*. Cambridge tracts in mathematics and mathematical physics. Stechert-Hafner Service Agency, 1964.

MJSC16. P. Méaux, A. Journault, F.-X. Standaert, and C. Carlet. Towards stream ciphers for efficient FHE with low-noise ciphertexts. In *EUROCRYPT 2016, Part I*, *LNCS* 9665, pages 311–343. Springer, Heidelberg, May 2016.

MST03. E. Mossel, A. Shpilka, and L. Trevisan. On e-biased generators in NC0. In *44th FOCS*, pages 136–145. IEEE Computer Society Press, October 2003.

OW14. R. ODonnell and D. Witmer. Goldreich's prg: evidence for near-optimal polynomial stretch. In *Computational Complexity (CCC), 2014 IEEE 29th Conference on*, pages 1–12. IEEE, 2014.

PS16. B. Pittel and G. B. Sorkin. The satisfiability threshold for k-xorsat. *Combinatorics, Probability and Computing*, 25(2):236–268, 2016.

Sie84. T. Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications (corresp.). *IEEE Transactions on Information theory*, 30(5):776–780, 1984.

SW14. A. Sahai and B. Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *46th ACM STOC*, pages 475–484. ACM Press, May / June 2014.

Wie86. D. Wiedemann. Solving sparse linear equations over finite fields. *IEEE transactions on information theory*, 32(1):54–62, 1986.

Appendices

A Detailed calculations for the proof of Proposition 2

$$\begin{aligned}\mathbb{E}(C) &= \sum_{i=1}^{n}\sum_{j=i+1}^{n}\mathbb{E}(C_{i,j}) = \sum_{i=1}^{n}\sum_{j=i+1}^{n}\sum_{k=2}^{m} P_{[B(m,p)=k]}\cdot(k-1)\\ &= \binom{n}{2}\sum_{k=2}^{m} P_{[B(m,p)=k]}\cdot(k-1)\\ &= \binom{n}{2}\sum_{k=2}^{m}\binom{m}{k}p^k(1-p)^{m-k}\cdot(k-1).\end{aligned}$$

Working on the sum:

$$\begin{aligned}\mathbb{E}(C) &= \left[\binom{n}{2}\sum_{k=0}^{m}\binom{m}{k}p^k(1-p)^{m-k}\cdot(k-1)\right] - \left[\binom{n}{2}\sum_{k=0}^{1}\binom{m}{k}p^k(1-p)^{m-k}\cdot(k-1)\right]\\ &= \left[\binom{n}{2}\sum_{k=0}^{m}\binom{m}{k}p^k(1-p)^{m-k}\cdot(k-1)\right] + \binom{n}{2}p^0(1-p)^m\\ &= \left[\binom{n}{2}\sum_{k=0}^{m}k\binom{m}{k}p^k(1-p)^{m-k}\right] - \binom{n}{2} + \binom{n}{2}(1-p)^m.\end{aligned}$$

Using the value of p:

$$\begin{aligned}\mathbb{E}(C) &= \left[\sum_{k=0}^{m}k\binom{m}{k}\left(\frac{1}{\binom{n}{2}}\right)^{k-1}\left(\frac{\binom{n}{2}-1}{\binom{n}{2}}\right)^{m-k}\right] - \binom{n}{2} + \binom{n}{2}(1-p)^m\\ &= \left[\sum_{k'=-1}^{m-1}(k'+1)\binom{m}{k'+1}\left(\frac{1}{\binom{n}{2}}\right)^{k'}\left(\frac{\binom{n}{2}-1}{\binom{n}{2}}\right)^{m-1-k'}\right] - \binom{n}{2} + \binom{n}{2}(1-p)^m\\ &= \left[\sum_{k'=0}^{m-1}m\binom{m-1}{k'}\left(\frac{1}{\binom{n}{2}}\right)^{k'}\left(\frac{\binom{n}{2}-1}{\binom{n}{2}}\right)^{m-1-k'}\right] - \binom{n}{2} + \binom{n}{2}\left(\frac{\binom{n}{2}-1}{\binom{n}{2}}\right)^{m}\\ &= m - \binom{n}{2} + \binom{n}{2}\left(\frac{\binom{n}{2}-1}{\binom{n}{2}}\right)^{m}.\end{aligned}$$

Eventually this number can be estimated with a limited development:

$$\begin{aligned}
\mathbb{E}(C) &= n^{\mathsf{s}} - \binom{n}{2} + \binom{n}{2}\left(1 - \frac{1}{\binom{n}{2}}\right)^{n^{\mathsf{s}}} \\
&= n^{\mathsf{s}} - \binom{n}{2} + \binom{n}{2} e^{ln\left(1-\frac{1}{\binom{n}{2}}\right)n^{\mathsf{s}}} \\
&= n^{\mathsf{s}} - \binom{n}{2} + \binom{n}{2} e^{\left(-\frac{1}{\binom{n}{2}} - \frac{1}{2\binom{n}{2}^2} + o\left[\frac{1}{\binom{n}{2}^3}\right]\right)n^{\mathsf{s}}} \\
&= n^{\mathsf{s}} - \binom{n}{2} + \binom{n}{2} e^{\left(-\frac{n^{1+\mathsf{s}}}{\binom{n}{2}} - \frac{n^{1+\mathsf{s}}}{2\binom{n}{2}^2} + o\left[\frac{n^{\mathsf{s}}}{\binom{n}{2}^3}\right]\right)} \\
&= n^{\mathsf{s}} - \binom{n}{2} + \binom{n}{2}\left(1 - \frac{n^{\mathsf{s}}}{\binom{n}{2}} - \frac{n^{\mathsf{s}}}{2\binom{n}{2}^2} + o\left[\frac{n^{\mathsf{s}}}{\binom{n}{2}^3}\right] + \frac{n^{2\mathsf{s}}}{2\binom{n}{2}^2} + o\left[\frac{n^{2\mathsf{s}}}{\binom{n}{2}^2}\right]\right) \\
&= n^{\mathsf{s}} - \binom{n}{2} + \binom{n}{2} - n^{\mathsf{s}} - \frac{\binom{n}{2}n^{\mathsf{s}}}{2\binom{n}{2}^2} + \frac{\binom{n}{2}n^{2\mathsf{s}}}{2\binom{n}{2}^2} + o\left[\frac{\binom{n}{2}n^{2\mathsf{s}}}{\binom{n}{2}^2}\right] \\
&= -\frac{n^{\mathsf{s}}}{2\binom{n}{2}} + \frac{n^{2\mathsf{s}}}{2\binom{n}{2}} + o\left[\frac{n^{2\mathsf{s}}}{2\binom{n}{2}}\right] \\
&= O(n^{2(\mathsf{s}-1)})
\end{aligned}$$

B The Case of Ordered Goldreich's PRG with P_5

In this section, we give evidence that the additional structure given by an ordered Goldreich's PRG brings a lower security level than the unordered case.

B.1 Guess and Determine

Although the ordered case seems highly non-trivial to analyze from a theoretical point of view, we give evidence that it brings a lower security level than the unordered case. Then, we also give some experimental measures to support our studies. We remind that each subset is of the form:

$$\sigma^i = [\sigma_1^i, \sigma_2^i, \sigma_3^i, \sigma_4^i, \sigma_5^i], \text{ where } \sigma_1^i < \sigma_2^i < \sigma_3^i < \sigma_4^i < \sigma_5^i \ ,$$

and that the equations are of the form:

$$x_{\sigma_1^i} + x_{\sigma_2^i} + x_{\sigma_3^i} + x_{\sigma_4^i} x_{\sigma_5^i} = y_i \ .$$

In this particular case, the average number of collisions is much higher than in the unordered case, since the lasts bits of the seed are drawn with a higher probability. More formally, the average number of collisions is given by the following proposition:

Proposition 12 (Average number of collisions in the ordered case). *Let n be the number of variables, and m be the number of equations, let C be the random variable counting the number of collisions on the degree-two monomials in the whole system. Then, the average number of collisions is:*

$$\mathbb{E}(C) = \sum_{i=1}^{n-1} (n-i)\left(-1 + mp_i + (1-p_i)^m\right) \ ,$$

where $p_i = \frac{\binom{i-1}{3}}{\binom{n}{5}}$.

Proof. We first consider individually the $\binom{n}{2}$ degree-two possible monomials. For each equation, the two variables of the degree-two monomial are taken after the three degree-one monomials, therefore the probability that the monomial indexed by i, j is taken follows a Bernouilli law with parameter $p_i = \frac{\binom{i-1}{3}}{\binom{n}{5}}$.

The random variable counting how many times the monomial indexed by i, j is selected follows a binomial law of parameters m and p_i. As a collision happens when the monomial has already be taken, we consider the random variable $C_{i,j}$ counting 0 if the monomial has been taken 0 or 1 times, $k-1$ otherwise. The expectation of $C_{i,j}$ is therefore:

$$\mathbb{E}(C_{i,j}) = \sum_{k=2}^{m} P_{[B(m,p_i)=k]} \cdot (k-1),$$

where $P_{[B(m,p_i)=k]}$ stands for the probability for a random variable following a binomial distribution of parameters m and p_i to take the value k.

The total number of collisions is obtained by summing the expectations of all the $C_{i,j}$:

$$\begin{aligned}
\mathbb{E}(C) &= \sum_{i=1}^{n-1}\sum_{j=i+1}^{n} \mathbb{E}(C_{i,j}) = \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\sum_{k=2}^{m} P_{[B(m,p_i)=k]} \cdot (k-1) \\
&= \sum_{i=1}^{n-1}(n-i)\sum_{k=2}^{m}\binom{m}{k}p_i^k(1-p_i)^{m-k}\cdot(k-1) \\
&= \sum_{i=1}^{n-1}(n-i)\left[\left(\sum_{k=0}^{m}\binom{m}{k}p_i^k(1-p_i)^{m-k}\cdot(k-1)\right) - (1-p_i)^m\right] \\
&= \sum_{i=1}^{n-1}(n-i)\left[\left(\sum_{k=0}^{m}k\binom{m}{k}p_i^k(1-p_i)^{m-k}\right) - 1 - (1-p_i)^m\right] \\
&= \sum_{i=1}^{n-1}(n-i)\left[\left(\sum_{k=0}^{m}m\binom{m-1}{k-1}p_i^k(1-p_i)^{m-k}\right) - 1 - (1-p_i)^m\right] \\
&= \sum_{i=1}^{n-1}(n-i)\left[\left(m\sum_{k'=0}^{m-1}\binom{m-1}{k'}p_i^{k'+1}(1-p_i)^{m-1-k'}\right) - 1 - (1-p_i)^m\right] \\
&= \sum_{i=1}^{n-1}(n-i)\left[mp_i - 1 - (1-p_i)^m\right]
\end{aligned}$$

The penultimate line is obtained by fixing $k' = k-1$.

In this very particular case, the average number of collisions and the number of guesses are hard to determine. Intuitively, we expect the last bits of the seed to be drawn more often in the monomials of degree two. As a consequence, the number of collisions is likely to be much higher. Also, the number of guesses should be greatly reduced , since we guess the bits of the seed that appears the most.

Our experimental results, shown in Table 5 and Table 6, support this intuition. Even better, for $\mathsf{s} = 1.45$ we could not find a seed size n that forces the attacker to make at least one guess.

B.2 Algebraic attack on the ordered case

Algebraic attacks are also more efficient for the ordered case since there is an additional structure. One can figure that, in that case, the number of equations derived from the three types method of Section 4.1 will find more collisions and semi-collisions. So, the limit stretch can be lowered in comparison to the non ordered case.

Table 5. Average number of collisions for the ordered case

n	256	512	1024	2048	4096
$\mathsf{s} = 1.45$	458	890	1703	3251	6162
$\mathsf{s} = 1.4$	271	488	873	1539	2709
$\mathsf{s} = 1.3$	95	145	221	341	520

Table 6. Average number of guesses for the ordered case

n	256	512	1024	2048	4096
$\mathsf{s} = 1.45$	0	0	0	0	0
$\mathsf{s} = 1.4$	0	1	2	5	9
$\mathsf{s} = 1.3$	6	10	17	30	50