Preliminary Cryptanalysis of the Biscuit Signature Scheme

Charles Bouillaguet, Julia Sauvage

Preliminary Cryptanalysis of the Biscuit Signature Scheme

Charles Bouillaguet and Julia Sauvage

Sorbonne Université, CNRS, LIP6, Paris, France

Abstract. Biscuit is a recent multivariate signature scheme based on the MPC-in-the-Head paradigm. It has been submitted to the NIST competition for additional signature schemes. Signatures are derived from a zero-knowledge proof of knowledge of the solution of a structured polynomial system. This extra structure enables efficient proofs and compact signatures. This short note demonstrates that it also makes these polynomial systems easier to solve than random ones. As a consequence, the original parameters of Biscuit failed to meet the required security levels and had to be upgraded.

Keywords: algebraic attack · multivariate crypto · biscuit signature scheme

1 Introduction

Biscuit is a multivariate digital signature scheme recently proposed by Bettale, Kahrobaei, Perret and Verbel [BKPV23a, BKPV23b]. It has been submitted in July 2023 to the competition organised by NIST in order to standardize additional "post-quantum" signature schemes not based on lattices.

In Biscuit, the public key is a system of quadratic multivariate equations while the secret key is a solution of this system. Biscuit builds upon a multi-party protocol in which N parties each holding a share of a vector $\mathbf{x}$ determine if $\mathbf{x}$ is a solution of a structured polynomial system. This protocol is converted into a Zero-Knowledge Proof-of-Knowledge (ZKPoK) using the "Multi-Party Computation in the Head" paradigm. Finally, the Fiat-Shamir heuristic turns this interactive proof-of-knowledge protocol into a signature scheme. Producing a new signature is akin to proving knowledge of the solution of the polynomial system in the public key, using fresh randomness extracted from the message to sign.

The security of Biscuit thus relies on the computational hardness of solving the underlying system of multivariate quadratic equations. Indeed, finding a solution of the polynomial system that forms the public key yields an equivalent secret key and enables an attacker to produce valid signatures using the normal signing procedure. In general, solving multivariate quadratic systems over finite fields (the MQ problem) is well-known to be NP-hard [GJ79]. The practical computational hardness of the problem is well-established, and documented by a public collection of challenges that enables everyone to monitor the progress, or lack therefof, of available polynomial solving software [YDH+15].

Several digital signature schemes rely on the hardness of the general MQ problem such as MQDSS [CHR+16, SCH+19] or MQOM [BFR23]. Another common choice is to rely on the hardness of a special class of systems; this is notably the case in HFE [Pat96], UOV [KPG99], SFLASH [CGP03], ... In this case, the actual underlying computational problem, which is a special case of MQ, could potentially be easier. This has notably been demonstrated in the case of HFE, where an 80-bit public key has been practically broken

E-mail: charles.bouillaguet@lip6.fr (Charles Bouillaguet), julia.sauvage@lip6.fr (Julia Sauvage)

Date of this document: 2024-01-17.

by a Gröbner basis computation [FJ03]. In contrast, solving a random, unstructured system of 80 Boolean quadratic equations is still a formidable challenging and has not been done in practice yet.

Biscuit belongs to the second category of multivariate cryptosystems. To reduce the size of signatures, its designers use polynomials of a special shape. Each (quadratic) public polynomial can be written $f + g \times h$, where f, g and h are affine forms in n variables. The point is that evaluating this on some input vector $\mathbf{x}$ requires a single multiplication by a non-constant in the finite field. This is a very strong structure: while a generic quadratic polynomial is described by $(n+1)(n+2)/2$ coefficients, a "Biscuit-style" polynomial is fully described by only $3n+1$ coefficients.

The designers observed that this structure enable better attack algorithms than in the case of the general MQ problem. In the submission document [BKPV23a], they propose a simple combinatorial algorithm that solves Biscuit-style polynomial systems in n variables over a finite field with q elements using $\tilde{\mathcal{O}}\left(q^{3n/4}\right)$ operations. This is exponentially better than exhaustive search — it would require $\tilde{\mathcal{O}}\left(q^n\right)$ operations. No such improved combinatorial algorithm is known in the general case, and this is a strong hint that the additional structure makes the problem easier.

Contribution

We first describe a very simple combinatorial algorithm to solve "Biscuit-style" polynomial systems using only exhaustive and linear algebra. It requires $\tilde{\mathcal{O}}\left(q^{n/2}\right)$ operations. We then combine it with the use of the F_5 algorithm [Fau02] to obtain an improved "hybrid method" that combines exhaustive search Gröbner basis computation. This improved algorithm breaks the three parameter sets initially proposed by the designers of Biscuit, showing that they do not reach the claimed 128, 192 and 256 bits of security. Our new algorithms also improve the efficiency of the forgery attack proposed by Kales and Zaverucha in [KZ20], and we study the consequences on the security of Biscuit.

The designers of Biscuit updated the specification of their scheme following preliminary versions of our findings, and gave improved parameters in [BKPV23b], that we find to be secure.

Lastly, we analyze the asymptotic complexity of our algorithm for solving Biscuit-style polynomial systems and discuss additional algebraic properties of these special polynomial systems.

2 Preliminaries

Let $\mathbb{F}_q$ denote a finite field and $\mathbb{F}_q[x_1, \ldots, x_n]$ denote the ring of multivariate polynomials in n variables. We usually consider a sequence of m polynomials $f_1, \ldots, f_m$ in n variables with coefficients in $\mathbb{F}_q$. We write vectors in boldface letters, so that $\mathbf{x} = (x_1, \ldots, x_n$ and $\mathbf{f} = (f_1, \ldots, f_m)$. We write $\langle \mathbf{a}, \mathbf{x} \rangle$ the dot product of the two vectors $\mathbf{a}$ and $\mathbf{x}$.

2.1 Succinct Description of Biscuit

We provide a very high-level description of the Biscuit signature scheme and introduce the relevant notations. A first version of Biscuit has been submitted to the NIST competition for additional post-quantum signatures [BKPV23a]. We refer to it as "Biscuit v1". An improved version appears in [BKPV23b] with different parameters and we refer to it as "Biscuit v2". Biscuit has two sets of parameters for each security level: the "short" one produces the shortest possible signatures and the "fast" one yields faster signing and verification. Tables 1 and 2 show the parameters.

The security of Biscuit fundamentally relies on the hardness of solving structured polynomial systems. Given a sequence of constants $c_i \in \mathbb{F}_q$ and sequences of vectors $\mathbf{u}_i, \mathbf{v}_i, \mathbf{w}_i \in {\mathbb{F}_q}^n$, define the sequence of quadratic polynomials $\mathbf{f} = f_1, \ldots, f_m$ in n variables with coefficients over $\mathbb{F}_q$ of the following shape:

$$f_i(\mathbf{x}) = c_i + \langle \mathbf{u}_i, \mathbf{x} \rangle + \langle \mathbf{v}_i, \mathbf{x} \rangle \times \langle \mathbf{w}_i, \mathbf{x} \rangle \tag{1}$$

In the sequel, we refer to polynomials of the shape described by (1) as "Biscuit-style polynomials". Evaluating $f_i(\mathbf{x})$ requires a single multiplication in the finite field.

Given $f_1, \ldots, f_m$, the authors of Biscuit refer to the problem of finding a solution to an arbitrary subset of $\{f_1, \ldots, f_m\}$ of size $m - u$ as the "$\texttt{PowAff2}_u(\mathbf{f})$ problem". The security of Biscuit relies on the assumption that this problem is computationally hard.

It is straightforward that deciding if a system of Biscuit-style polynomials has a solution is NP-hard. Take an arbitrary quadratic polynomial $g(\mathbf{x}) = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}x_ix_j + \sum_{i=1}^{n} b_ix_i + c$. Introduce n^2 new variables y_{ij}. Then, there exist $\mathbf{x}$ such that $g(\mathbf{x}) = 0$ if and only if there exist $\mathbf{x}, \mathbf{y}$ such that:

$$\begin{cases} 0 &= y_{ij} - x_ix_j & (1 \leq i, j, n) \\ 0 &= \displaystyle\sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}y_{ij} + \sum_{i=1}^{n} b_ix_i + c \end{cases}$$

This is a system of Biscuit-style polynomials. Therefore, deciding if an arbitrary system of quadratic polynomial admits a solution reduces to a (polynomially larger) instance of the same problem on Biscuit-style polynomials. The former problem is NP-hard, and therefore the latter has to be.

Suppose that a secret vector $\mathbf{x} \in {\mathbb{F}_q}^n$ has been additively shared between a set of N parties. Biscuit is designed around an efficient MPC protocol Π that enables the parties to check whether $\mathbf{f}(\mathbf{x}) = 0$ (in this case, the parties "accept" $\mathbf{x}$). It is shown in [BKPV23b, Proposition 1] that if $\mathbf{f}(\mathbf{x}) \neq 0$, the parties accept $\mathbf{x}$ with probability q^{-u} if $\mathbf{x}$ is the solution of a subset of size exactly $m - u$ of the polynomials. This probability is taken over the randomness used in the protocol: the parties need m common random elements of $\mathbb{F}_q$.

This multi-party protocol is transformed into an interactive 5-pass zero-knowledge proof of knowledge protocol for $\mathbf{x}$ using the "MPC-in-the-Head" paradigm:

1. [**commitment**] The prover $\mathcal{P}$ performs an additive sharing of the secret $\mathbf{x}$ among N fictitious parties and commits the share of each party.
2. [**challenge 1**] The verifier $\mathcal{V}$ sends the randomness to use in the MPC protocol (m elements of $\mathbb{F}_q$).
3. [**response 1**] $\mathcal{P}$ completes the execution of the MPC protocol for the N parties and commits their views.
4. [**challenge 2**] $\mathcal{V}$ targets a given party i (integer in $0, \ldots, N-1$).
5. [**response 2**] $\mathcal{P}$ opens the shares and the views of all other parties.

Using $\mathbf{x}$ as the secret key, this ZKPoK protocol can be seen as a 5-pass identification scheme by iterating it τ times in order to increase its soundness. Finally, it is turned into a signature scheme using the Fiat-Shamir heuristic. The public key is a collection of Biscuit-style polynomials $f_1, \ldots, f_m$, the secret is a vector $\mathbf{x} \in {\mathbb{F}_q}^n$ such that $\mathbf{f}(\mathbf{x}) = 0$. The challenges sent by the verifier are instead generated deterministically from the message to sign and the round number.

A slighlty simplified description of the signing procedure follows. To produce a signature for a given message:

1. Choose a random `salt`. Derive from it τ pseudo-random additive sharing of the secret key x among N parties. Hash all the shares of all the N parties for all the τ rounds toghether along with the message m; call h_1 the result.

Table 1: Parameters of Biscuit v1 as given in [BKPV23a].

Security level	Version	N	τ	q	n	m	Claimed bit security
I	short	256	18	16	64	67	143
	fast	16	34				
II	short	256	30		87	90	208
	fast	16	54				
III	short	256	40		118	121	274
	fast	16	73				

Table 2: Parameters of Biscuit v2 as given in [BKPV23b].

Security level	Version	N	τ	q	n	m	Claimed bit security
I	short	256	18	256	50	52	143
	fast	32	28				
II	short	256	25		89	92	207
	fast	32	40				210
III	short	256	33		127	130	272
	fast	32	53				275

2. Generate the first challenge of all τ rounds: derive τm pseudo-random element of $\mathbb{F}_q$ from h_1.

3. Run the MPC protocol τ times. Hash together the views of all the N parties in all the τ rounds along with h_1, the message and the public key; call the result h_2.

4. Generate the second challenge of all τ rounds: derive τ pseudo-random integers $i_0, \ldots, i_{\tau-1}$ in $\{0, \ldots, N-1\}$ from h_2.

5. The signature is made of h_1, h_2 and the opened views of all parties except the i_k-th in round k.

2.2 Gröbner Bases and Semi-Regular Polynomial Systems

Some of the best methods to solve systems of polynomial use Gröbner bases. A Gröbner basis of some polynomial ideal I is a set of generators of I that enjoy additional desirable properties. It is beyond the scope of this paper to discuss Gröbner basis; we refer the interested reader to a standard textbook such as [CLO91].

We will just point out that in some cases, solutions of the corresponding polynomial system can be easily read off a Gröbner basis. This is in particular the case when the polynomial system $f_1 = \cdots = f_m = 0$ has a single solution in the algebraic closure of the field. In this specific case, any Gröbner basis of the ideal spanned by the f_i's is $x_1 - a_1, \ldots, x_n - a_n$, where $\mathbf{a}$ is the unique solution of the system. Thus obtaining any Gröbner basis reveals the solution of the system.

The historical method to compute such bases is known as Buchberger's algorithm [Buc65]. Nowadays, the most efficient method to compute Gröbner bases is the F_5 algorithm [Fau02]. If the *degree of regularity* of the ideal is D, then this algorithm terminates in less than

$$\mathcal{O}\left(m \cdot \binom{n+D}{D}^{w}\right) \quad \text{operations over } \mathbb{F}_q \tag{2}$$

where w denote the linear algebra constant. In general, finding the degree of regularity D of a given polynomial ideal is difficult. However, when the input polynomials satisfy certain

genericity assumptions, then the degree of regularity can be deduced from n and m. These assumptions imply that the Hilbert series of the ideal is a function of m, n and the degrees of the input polynomials, but that it does not depend from the actual values of their coefficients. When all the input polynomials are quadratic, then the "generic" degree of regularity depends only on n and m, and we denote it by $\mathrm{d}_{reg}(n, m)$.

These genericity assumptions essentially mean that the polynomials are not bound by "unexpected" algebraic relations. They are usually well-verified in practice on unstructured systems, and are therefore standard in all the cryptographic litterature. The interested reader is referred to [Bar04, BFS15] for more details.

If $m \leq n$ and the polynomials form a *regular sequence*, then the degree of regularity is given by the Macaulay bound: $D = 1 + \sum_i (\deg f_i - 1)$. This implies in particular that if $m = n$, then the degree of regularity of a sequence of regular quadratic polynomials is $\mathrm{d}_{reg}(n, n) = n + 1$.

When $m > n$, *i.e.* when the system is overdetermined, the polynomials cannot form a regular sequence. The concept has been extended to *semi-regular* sequence in [Bar04, BFS15]. The degree of regularity of a semi-regular sequence of polynomials can also be easily determined from m and n. Write the series expansion of

$$\sum_j a_j z^j = \prod_{i=1}^{m} \frac{1 - z^{\deg f_i}}{1 - z}$$

The smallest index j such that $a_j < 0$ is the degree of regularity of the semi-regular sequence $f_1, \ldots, f_m$. This provides an algorithm to compute $\mathrm{d}_{reg}(n, m)$ effectively.

It is important to note that if n is fixed, then it is a decreasing function of m: the more overdetermined the system, the easier it is to solve. [Bar04] gives an asymptotic equivalent of the degree of regularity for quadratic polynomials when $n \to +\infty$:

$$\mathrm{d}_{reg}(n, \theta n) = \left(\theta - \frac{1}{2} - \sqrt{\theta(\theta - 1)}\right) n + \mathcal{O}\left(n^{1/3}\right) \tag{3}$$

If the polynomial system can form a *semi-regular* sequence, the complexity of the algorithm F_5 is:

$$C_{F_5}(n, m) = \mathcal{O}\left(m \cdot \binom{n + \mathrm{d}_{reg}(n, m)}{\mathrm{d}_{reg}(n, m)}^{w}\right) \tag{4}$$

2.3 The Hybrid Method

The "hybrid method" [BFP09, BFP12] is usually the best technique for solving polynomial systems over finite fields. Its principle is simple:

1. Choose $0 \leq k \leq n$.
2. "Guess" the value of k variables.
3. Compute a Gröbner basis of the remaining system of m equations $n - k$ variables using the F_5 algorithm.
4. If no solution has been found, return to step 2.

The point is that the sub-systems that are actually solved in step 3 are more overdetermined than the original input system, and therefore computing a Gröbner basis is faster. With the optimal choice of k, the complexity of the resulting algorithm is:

$$C_{hyb}(n, m) = \min_{0 \leq k \leq n} \left(q^k \left(C_{F_5}(n - k, m)\right)\right) \tag{5}$$

This is necessarily less than $C_{F_5}(n, m)$. The authors also compute the asymptotic complexity of this hybrid combination of exhaustive search and F_5 when $n \to +\infty$ with q and $\alpha = m/n$ fixed. Define $\beta = k/n$; we have $C_{hyb}(n, \alpha n) \sim 2^{Kn}$ with:

$$\begin{aligned} K = &\log_2(q) + \omega \log_2(1-\beta) \\ &- \frac{\omega}{2}\left(1 + \sqrt{\frac{\alpha}{\alpha+\beta-1}}\right) \log_2 D_1(\alpha, \beta) \\ &- \frac{\omega}{2}\left(1 - \sqrt{\frac{\alpha}{\alpha+\beta-1}}\right) \log_2 D_2(\alpha, \beta), \\ D_1(\alpha, \beta) = &\alpha + \frac{1-\beta}{2} - \sqrt{\alpha(\alpha+\beta-1)}, \\ D_2(\alpha, \beta) = &\alpha - \frac{1-\beta}{2} - \sqrt{\alpha(\alpha+\beta-1)}. \end{aligned}$$

Given values of q and α, the best choice of β is one that minimizes the value of K. It can easily be found numerically.

For exemple, with $\alpha = 1$, i.e. $n = m$ and $q = 16$, we have an asymptotic complexity of $2^{2.01n}$ with $\beta = 0.182$. When $q = 256$, we have an asymptotic complexity of $2^{2.39n}$ with $\beta = 0.049$.

2.4 Estimating the Concrete Hardness of Solving Polynomial Systems

While the global asymptotic complexity of solving systems of semi-regular quadratic polynomials is relatively well-understood, estimating the concrete number of operations that is required is more an art than a science.

Several software tools provide estimates of the number of operations required to execute polynomial system solving algorithms. The `MQEstimator` [BMSV22a] that is notably available in the `CryptographicEstimators` software library [EVZB23] has been used by the designers of Biscuit to estimate the complexity of attacks that involve solving polynomial systems.

The estimates provided by these tools are crude at best. One of the main obstacles is that the asymptotic expression of the number of operations is only known up to some constant factors. These factors depend on the actual implementation of the algorithm, on the choice of data structures, etc. and also on the hardware. The `MQEstimator` for instance ignores these constants and assumes that they are very small (all equal to one). This yields meaningful *lower-bounds* on the hardness of running the corresponding algorithms, and this is usually the conservative approached used by the designers of multivariate cryptographic schemes. Note that sometimes, in addition, potentially significant lower-order terms are ignored asymptotically, as in (3).

The `MQEstimator` estimates the number of arithmetic operations (call it N) in $\mathbb{F}_q$. It then estimates the total number of bit operations as $N \log_2^\theta q$, where $\theta = 2$ by default. The $\log_2^\theta q$ term approximates the cost of multiplication in the finite field. Considering that these finite fields are usually small, the choice of $\theta = 2$ is reasonable and amounts to schoolbook multiplication. However, this induces an other inaccuracy: not all arithmetic operations are multiplications, and the constant is arbitrarily set to 1.

Another potential issue is the linear algebra constant. The best known value is $w = 2.3728596$ [AW21] but it is well-known that algorithms that multiply matrices in $\mathcal{O}(n^w)$ operations are so impractical that they have never been implemented. The MQEstimator uses $w = 2$, which is again a safe lower-bound. However, when the polynomial system is overdetermined, and therefore likely has a single solution, simpler algorithms can be used to solve it, such as variants of XL [CKPS00]. They enjoy the same time bound

as F_5 in this case. When these algorithms are implemented using the Block-Wiedemann algorithm [Cop94] to solve linear systems, as in [CCNY12], it is reasonable to assume that the linear algebra constant is $w = 2$.

In this paper, we will take the same approach as the designers of Biscuit and take the crude estimates provided by these tools as an order of magnitude of the number of operations required to solve a polynomial system.

2.5 The Kales-Zaverucha Forgery Attacks Against Signature Schemes Constructed From Five-Pass Identification Schemes

Kales and Zaverucha proposed in [KZ20] a forgery attack on signature schemes derived from 5-pass Fiat-Shamir identification schemes, provided that these meet certain mild conditions. This attack can be adapted to Biscuit, and it was taken into account in the choice of parameters. We summarize it in this section. The main idea of the attack is that a cheating prover can convince a honest verifier into accepting $\mathbf{x}$ in the ZKPoK protocol by either guessing the first *or* the second challenge correctly.

Suppose that the adversary knows a vector $\mathbf{x}$ that solves a subset of size $m - u$ of the polynomial equations in the public-key. Obtaining such an $\mathbf{x}$ gets easier when u increases. This $\mathbf{x}$ is accepted by the N parties in the MPC protocol with probability q^{-u}. To run the attack, the adversary chooses an integer partition $\tau = \tau_1 + \tau_2$ and proceeds as follows:

1. Choose a random `salt`. Run the signature algorithm until the first response.
2. Set $R \leftarrow \{0 \leq i < \tau \mid \text{the } N \text{ parties accept } x \text{ in the } i\text{-th iterations of the MPC protocol}\}$.
3. If $|R| < \tau_1$, return to step 1.
4. Guess the second challenge (*i.e.* the identity of the single party whose view remains concealed) for all rounds not in R.
5. Produce a forgery by running the protocol normally in all rounds in R and exploiting knowledge of the second challenge in the other rounds.
6. If this is not successful, return to step 4.

The cardinality of R follows a binomial distribution of parameters (τ, q^{-u}). The probability that $|R| \geq \tau_1$ is therefore $\sum_{i=\tau_1}^{\tau} \binom{\tau}{i} q^{-ui} \left(1 - q^{-u}\right)^{\tau - i}$. The probability of guessing correctly the τ_2 second challenges is simply $1/N^{\tau_2}$. It follows that the expected time complexity of this attack is of order

$$KZ_{\tau_1,\tau_2} = \frac{1}{\sum_{i=\tau_1}^{\tau} \binom{\tau}{i} q^{-ui} \left(1 - q^{-u}\right)^{\tau - i}} + N^{\tau_2} \tag{6}$$

To each value of u corresponds an optimal choice of (τ_1, τ_2). The concrete cost of the attack can be estimated by trying all possible $1 \leq u \leq m$ and all $0 \leq \tau_1 \leq \tau$.

3 Simple Combinatorial Algorithms for Biscuit-Style Polynomial Systems

In this section, we present simple algorithms to solve systems of Biscuit-style polynomials, *i.e.*, polynomials of the shape described by equation (1).

3.1 Main Idea

The usual hybrid method to solve such polynomial systems consists in "guessing" k variables, which brings a system of m polynomials in $n-k$ variables. Suppose that

$$c_i + \langle \mathbf{u}_i, \mathbf{x} \rangle + \langle \mathbf{v}_i, \mathbf{x} \rangle \times \langle \mathbf{w}_i, \mathbf{x} \rangle = 0.$$

We observe that if we "guess" the value of $\langle \mathbf{v}_i, \mathbf{x} \rangle$, which is an element of $\mathbb{F}_q$ (let us call it a_i), then we end up with *two* linear equations:

$$\begin{cases} c_i + \langle \mathbf{u}_i, \mathbf{x} \rangle + a_i \times \langle \mathbf{w}_i, \mathbf{x} \rangle &= 0 \\ \langle \mathbf{v}_i, \mathbf{x} \rangle &= a_i. \end{cases} \tag{7}$$

From this two linear equations we can eliminate two of the n variables. The resulting system has $m-1$ polynomials in $n-2$ variables. Losing a polynomial is regrettable, but it is more than compensated for by the elimination of *two* variables, as the resulting system is even more overdetermined.

A direct application of this idea would be to "guess" a_i for $1 \leq i \leq n/2$. This yields a system of n linear equations, plus $m - n/2$ extra quadratic equations. Solving the linear system takes time $\mathcal{O}\left(n^3\right)$; if it does not have any solution, then the initial guess was wrong. If it has a single solution, then it is sufficient to test this solution against the original polynomial system. Otherwise, the set of its solution is expected to have a small dimension d, so that the remaining quadratic part can be reduced to only d variables and can most likely be solved by linearization. The process has to be repeated until a solution of the original system is found. The expected complexity of this simple algorithm is $\mathcal{O}\left(n^3 q^{n/2}\right)$.

This improves upon a more complex algorithm of complexity $\tilde{\mathcal{O}}\left(q^{3n/4}\right)$ given by the designers of Biscuit in [BKPV23a].

3.2 Application to Underdetermined Systems

If there are more variables than polynomial equations ($n \leq m$), then the same idea can be used even more efficiently. In this case, the expected number of solutions is q^{n-m} and therefore with high probability a solution will remain even if we impose $n-m$ extra linear constraints.

Denote by $e = n - m$ the *excess*, namely the number of extra variables. To find a solution, set $h \leftarrow \min(e, m)$ and choose $a_1, \ldots, a_h$ at random. Impose that $\langle \mathbf{v}_i, \mathbf{x} \rangle = a_i$ for $1 \leq i \leq h$. This yields $2h$ linear equations using (7) and this enable us to eliminate $2h$ variables. This yields a smaller system of $m-h$ polynomials in $n-2h$ variables, that has a solution with high probability. Note that as soon as $n \geq 2m$, there is nothing more to do since no equation remains. In that case, a random solution of the system can be found in time $\mathcal{O}\left(n^3\right)$.

If $n < 2m$, any technique can be used to solve the resulting subsystem, including the one discussed in section 3.1. This yields an algorithm of expected complexity $\mathcal{O}\left(n^3 q^{m-n/2}\right)$ that produces a random solution of the system.

3.3 Application to the "$\texttt{PowAff2}_u$" Problem

Given a sequence of Biscuit-style polynomial equations $f_1 = \cdots = f_m = 0$, we consider the problem of finding a vector that satisfies at least $m-u$ equations. This is relevant in the context of the Kales-Zaverucha attack described in section 2.5.

If $m - u \leq n$, then we are in the domain of underdetermined systems and the technique of section 3.2 can be applied. It follows from the discussion in the previous section that the problem can be solved in polynomial time as soon as $u \geq m - n/2$. Therefore we assume in the sequel that $u < m - n/2$, and in particular that $n < 2m$.

In [BKPV23a], the designers of Biscuit exploit the following idea. Suppose that $\mathbf{x}$ is known to satisfy the first $m-\ell$ polynomial equations; it heuristically satisfies each of the ℓ remaining equation with probability $1/q$. Therefore $\mathbf{x}$ can be assumed to solve approximately ℓ/q extra equations out of chance, in addition to the $m-\ell$ that it is already known to satisfy.

We combine this observation with the algorithm of section 3.2 to obtain another simple algorithm for the $\texttt{PowAff2}_u$ problem:

1. Let $h \leftarrow \lfloor n/2 \rfloor$
2. Pick a random h-subset $\{i_1, \ldots, i_h\}$ of $\{1, \ldots, m\}$.
3. Find a random solution $\mathbf{x}$ of $f_{i_1} = \cdots = f_{i_h} = 0$ using the technique of section 3.2.
4. If $\mathbf{x}$ also satisfies $m-h-u$ additional equations among $f_{h+1} = \cdots = f_m = 0$, then return $\mathbf{x}$. Otherwise, return to step 2.

Solving the subsystem in step 3 takes time $\mathcal{O}\left(n^3\right)$. We heuristically assume that $\mathbf{x}$, being a random solution of the first h equations, is sufficiently random to satisfy each of the remaining ones with probability $1/q$. The problem thus appears to be very easy if $u \geq (m-n/2)(1-1/q)$.

Denote by Y the random variable that gives number of equations satisfied by $\mathbf{x}$ among the $m-h$ last ones. By hypothesis, Y follows a binomial distribution of parameters $(m-h, 1/q)$. The probability of success of each iteration is therefore

$$\Pr(Y \geq m-h-u) = \sum_{i=m-h-u}^{m-h} \binom{m-h}{i} q^{-i}(1-1/q)^{m-h-i}$$

and the expected number of iterations is $1/\Pr(Y \geq u)$.

The special case where $m = n$ and $u = \gamma n$ with $\alpha < \frac{1}{2}\left(1 - \frac{1}{q}\right)$ is both relevant, non-trivial and more amenable to analysis.

First, we note that directly using the technique of section to solve the underdetermined system of $(1-\alpha)n$ polynomials in n variables formed by $f_1, \ldots, f_{m-u}$ yields an asymptotic time complexity of $n^3 q^{\left(\frac{1}{2}-\gamma\right)n}$. In the technique presented above, the lower-bound on the tail of the binomial distribution given in [Ash90, Lemma 4.7.2] tells us that:

$$\Pr\left[Y \geq \left(\frac{1}{2} - \gamma\right) n\right] \geq \frac{1}{\sqrt{8\,(1-2\gamma)\,\gamma n}} \exp\left(-\frac{n}{2} D\left(1-2\gamma, \frac{1}{q}\right)\right)$$

where

$$D(a,p) = a \ln \frac{a}{p} + (1-a) \ln \frac{1-a}{1-p}$$

denotes the Kullback-Leibler divergence between an a-coin and a p-coin. This yields an exponential algorithm of asymptotic complexity $\sqrt{n} \exp\left(\frac{n}{2} D\left(1-2\gamma, \frac{1}{q}\right)\right)$. With $q = 16$ and $\alpha = 1/3$ for instance, this is $e^{0.1654n} = q^{0.05964n}$, and this improves significantly over the direct use of the technique of the previous section that yields $q^{n/6}$.

4 Hybrid Combination With the F_5 Algorithm

In this section, we extend the mostly combinatorial algorithms of section 3 beyond simple linear algebra by hybridizing them with the F_5 algorithm. Instead of guessing values until a full linear system has been obtained, we will use the F_5 algorithm after a judicious number of guesses. These algorithms take extra parameters $(k, \ell, \ldots)$ that describe how many values should be "guessed".

```
1: function BiscuitZero(f_1, ..., f_m, k)
2:     for all (a_1, ..., a_k) ∈ F_q^k do
3:         Eliminate 2k variables using (7)
4:         Assemble the resulting subsystem S of m − k polynomials in n − 2k variables
5:         Solve S using the F_5 algorithm
6:         for each solution x of S do
7:             if x satisfies the original equations then
8:                 return x
9:     return ⊥ (no solution found)
```

Figure 1: Hybrid algorithm for Biscuit-style polynomial systems with $m \geq n$.

```
1: function BiscuitSolve(f_1, ..., f_m, k)
2:     if n ≤ m then
3:         return BiscuitZero(f_1, ..., f_m, k)
4:     loop
5:         Pick (a_1, ..., a_{n−m}) at random in F_q^{n−m}
6:         Eliminate 2(n − m) variables using (7)
7:         Assemble the resulting subsystem S of 2m − n polynomials in 2m − n variables
8:         Solve S using BiscuitZero
9:         if a solution has been found then
10:            return it
```

Figure 2: Hybrid algorithm for arbitrary Biscuit-style polynomial systems.

4.1 Zero-Dimensional Biscuit-Style Polynomial Systems

Assume that $f_1, \ldots, f_m$ is a semi-regular sequence of biscuit-style polynomials, with $m \geq n$. Therefore they span an ideal of dimension zero. Figure 1 shows our fundamental procedure to solve such systems. It takes an additional parameter k, which is the number of values to "guess".

We assume that adding extra linear equations still yields a semi-regular sequence. An equivalent hypothesis is made in the analysis of the hybrid method in [BFP12] or in [BFSS13] where it is called "strong semi-regularity". Under this assumption, the complexity of using the F_5 algorithm in step 5 can be assumed to be $C_{F5}(n-2k, m-k)$. The complexity of BiscuitZero is therefore $q^k \cdot C_{F5}(n-2k, m-k)$.

The problem is to find the optimal value of k. Given a concrete input system of n variables and m polynomial, finding the best k is not difficult: just try all possible values, estimate the running time of the F_5 algorithm (for instance using the `MQEstimator`) and pick the best solution.

4.2 General Biscuit-Style Polynomial Systems

The algorithm BiscuitSolve(f shown in Figure 2 is capable of dealing with underdetermined polynomial systems that do not span ideals of dimension zero. The algorithm is again parametrized by the number k of values to "guess".

The number of iterations of the loop is heuristically expected to be small, for reasons exposed in section 3.2. If $n \geq m$, then its running time is $q^k \cdot C_{F5}(2m-n-2k, 2m-n-k)$, and again, there is an optimal value of k. If $n < m$, then its running time is the same as that of BiscuitZero.

```
1: function BiscuitPowAff2(f_1, ..., f_m, u, ℓ, k)
2:     loop
3:         x ← BiscuitSolve(f_1, ..., f_{m−ℓ}, k)
4:         if x satisfies at least ℓ − u extra equations among f_{m−ℓ+1}, ..., f_m then
5:             return x
```

Figure 3: Hybrid algorithm for the $\texttt{PowAff2}_u$ problem.

4.3 $\texttt{PowAff2}_u$ Problem

The strategy is the same as in section 3.3: choose a parameter $\ell \geq u$, sample a random solutions to a random subset of $m - \ell$ polynomials equations and check them against the remaining ℓ ones, hoping that some of them will stochastically be satisfied. Repeat the process until at least $m - u$ equations are satisfied in total.

Denote again by Y the random variable that gives number of equations satisfied by $\mathbf{x}$ among the ℓ ones that are not known to be deterministically satisfied. The same reasoning as in section 3.3 shows that the probability of success of each iteration is

$$\Pr(Y \geq \ell - u) = \sum_{i=\ell-u}^{\ell} \binom{\ell}{i} q^{-i}(1-1/q)^{\ell-i}$$

and the expected number of iterations is $1/\Pr(Y \geq \ell - u)$. The complexity is dominated by the calls to BiscuitSolve, and their cost depends on whether $m - \ell$ is greater or smaller than n. Let us assume that $m - \ell \leq n$, so that solving the subsystems uses the strategy of section 3.2. Then the total complexity is

$$\frac{1}{\Pr(Y \geq \ell - u)} q^k \cdot C_{F5}(2m - 2\ell - n - 2k, 2m - 2\ell - n - k) \tag{8}$$

Given n, m and u, there are optimal values of ℓ and k that minimize this complexity.

4.4 Asymptotic Complexity Analysis of BiscuitZero

We now provide an asymptotic equivalent when $n \to +\infty$ of the complexity of BiscuitZero that we will denote C'_{hyb}, as shown in Figure 1, and show that it is better than applying the classic hybrid method. Our analysis follows the lines of [BFP12]. We define α and β such that $m = \alpha n$ and that we guess $k = \beta n$ variables, with $0 \leq \beta \leq \frac{1}{2}$. With the result of section4.1, we get:

$$C'_{hyb}(\alpha, n) = \min_{0 \leq \beta \leq 1/2} \left(q^{\beta n} \left(C_{F_5}\left((1-2\beta)n, (\alpha-\beta)n \right) \right) \right) \tag{9}$$

First we need an asymptotic equivalent of the degree of regularity for the reduced Biscuit system. Applying (3), we have:

$$\mathrm{d}_{reg}((1-2\beta)n, (\alpha-\beta)n) = \left(\alpha - \beta - \frac{1-2\beta}{2} - \sqrt{(\alpha-\beta)(\alpha+\beta-1)} \right) n + o(n)$$

We define $\gamma = \alpha - \beta - \frac{1-2\beta}{2} - \sqrt{(\alpha-\beta)(\alpha+\beta-1)}$.

We have our asymptotic equivalent for the degree of regularity. We now need a asymptotic equivalent for C_{F_5}. With equation (4) and the Stirling's formula, i.e. $n! \sim \sqrt{2\pi n}\left(\frac{n}{e}\right)^n$, we get that:

$$C'_{F_5}(n, m) = \frac{1}{\sqrt{2\pi}} \left(\frac{(n + \mathrm{d}_{reg}(n,m))^{n+\mathrm{d}_{reg}(n,m)+\frac{1}{2}}}{\mathrm{d}_{reg}(n,m)^{\mathrm{d}_{reg}(n,m)+\frac{1}{2}} \cdot n^{n+\frac{1}{2}}} \right)$$

With the formula (9), we have:

$$C'_{hyb}(\alpha, n) = q^{\beta n}\left(\frac{1}{\sqrt{2\pi}}\left(\frac{(1-2\beta+\gamma)\, n^{(1-2\beta)n+\gamma n+\frac{1}{2}}}{((1-2\beta)n)^{(1-2\beta)n+\frac{1}{2}}\,(\gamma n)^{\gamma n+\frac{1}{2}}}\right)^{\omega}\right)$$

Taking logarithms give:

$$\begin{aligned}\log_2 C'_{hyb}(\alpha, n) =& \beta n \log_2(q)\\ &+ \omega n\left(1-2\beta+\gamma+\frac{1}{2n}\right)(\log_2(n)+\log_2(1-2\beta+\gamma))\\ &- \omega n\left(\gamma+\frac{1}{2n}\right)(\log_2(n)+\log_2(\gamma))\\ &- \omega n\left(1-2\beta+\frac{1}{2n}\right)(\log_2(n)+\log_2(1-2\beta))\end{aligned}$$

Most of the $\log_2(n)$ terms cancel each other. We get:

$$\begin{aligned}\log_2 C'_{hyb}(\alpha, n) =& \beta n \log_2(q) - \frac{1}{2}\omega \log_2(n)\\ &+ \omega n\left(1-2\beta+\gamma+\frac{1}{2n}\right)\log_2(1-2\beta+\gamma)\\ &- \omega n\left(\gamma+\frac{1}{2n}\right)\log_2(\gamma)\\ &- \omega n\left(1-2\beta+\frac{1}{2n}\right)\log_2(1-2\beta)\end{aligned}$$

When $n \to \infty$, the $\frac{1}{n}$ terms vanish, and we get:

$$\begin{aligned}\log_2 C'_{hyb}(\alpha, n) =& \beta n \log_2(q) + \omega n\,(1-2\beta+\gamma)\log_2(1-2\beta+\gamma)\\ &- \omega n\gamma \log_2(\gamma) - \omega n(1-2\beta)\log_2(1-2\beta)\end{aligned}$$

The asymptotic complexity of this new hybrid approach is 2^{Kn}, with:

$$K = (\beta \log_2(q) + \omega\,(1-2\beta+\gamma)\log_2(1-2\beta+\gamma)) - \omega\cdot\gamma\log_2(\gamma) - \omega(1-2\beta)\log_2(1-2\beta)$$

We have a formula for the asymptotic complexity of our new hybrid approach when q and α are given. We considered that $\omega = 2$. $K(\beta)$ appears to be convex on $]0, \frac{1}{2}[$. To find the β where $K(\beta)$ is minimal, we can use the golden section method. In practice, we used `scipy.optimize.minimize_scalar` [VGO+20] that use this method with some optimizations. We have some example of complexity in Table 3. This examples match with the parameters used in the Biscuit signature scheme.

We can see that the structure of Biscuit systems have a great impact on its asymptotic security, especially when q is small.

4.5 Discussion

The fact that structured polynomial systems, such as those used in Biscuit, can be solved more efficiently than generic, random systems, does not come as a surprise to anyone familiar with the literature about algebraic cryptanalysis.

Table 3: Comparaison of the asymptotic of the "classic" hybrid method with BiscuitZero on Biscuit-style polynomial system with $m = n$

	Classical		BiscuitZero	
q	β	K	β	K
16	0.182	2.01	0.269	1.59
256	0.049	2.39	0.086	2.24

There are both (heuristic) theoretical arguments and empirical evidence that suggest that sequence of random Biscuit-style polynomials form regular or semi-regular sequence. Empirical evidence is given in [ACF+15] and [BKPV23a]. In particular, under this assumption, their degree of regularity is the same as that of a completely random (semi-regular) polynomial system. As such systems of Biscuit-style polynomials should not be much easier to solve than unstructured ones by directly a Gröbner basis computation. This was one of the main arguments of the designers of Biscuit to use them.

Despite this fact, the algorithms presented in this paper exploit the specific structure of Biscuit-style polynomial systems in a different way, and obtain an exponential speedup compared to the state-of-the-art for fully random systems.

Reasoning about the degree of regularity of a polynomial system, and claiming that it is high, shows that a direct Gröbner basis computation will be hard. But this does not rule out the possibility that different algorithms can be more efficient, as it is the case here. The designers of Biscuit made this observation themselves in [BKPV23a] when they presented their $\tilde{\mathcal{O}}\left(q^{3n/4}\right)$ algorithm for Biscuit-style polynomial systems.

What is even more remarkable is the fact that can solve Biscuit-style polynomial systems more efficiently than by directly computing a Gröbner basis with the F_5 algorithm, while we only apply the F_5 algorithm on semi-regular systems in a block-box way.

5 Application to the Security of Biscuit

We now use the algorithms of section 4 to study the security of Biscuit. We rely on estimates provided by the MQEstimator, just like the designers.

5.1 Key-Recovery Attack

Solving the Biscuit-style polynomial system exposed by a Biscuit public-key reveals the secret key. These systems are slightly overdetermined, so that using BiscuitZero is sufficient for direct key-recovery attack on Biscuit. To choose the number of variables to guess (the k parameter of BiscuitZero), we simply did an exhaustive search.

We see in Table 4 that, as a consequence of the algorithms presented in this article, Biscuit v1 (as in [BKPV23a]) loses 36 to 62 bits of security, depending of the version. This initial set of parameters does not provide the expected security level.

As a consequence of preliminary versions of our results, the designers of Biscuit proposed a new set of parameters in [BKPV23b]. These new parameters are more secure and essentially resist a direct key-recovery attack using BiscuitZero. Increasing the size of the finite field has the effect of diminishing the efficiency of hybrid methods: guessing values gets more costly, and the advantage this provides compared to a direct Gröbner basis computation is reduced.

In some cases, our estimates of the cost of our attacks are slightly below the security claims of Biscuit. Recall, however, that these are crude estimates, as discussed in section 2.4, and we believe that the difference falls within the error margin.

The optimal k we obtained are consistent with the theoretic β we obtained with our asymptotic complexity analysis. (When $q = 16$, k/n = $17/64 = 0.266$, $26/87 = 0.299$ and

Table 4: Key-recovery attacks on Biscuit. `v1` denotes the version of Biscuit in [BKPV23a] and `v2` is in [BKPV23b]. The "Hybrid method" columns is the algorithm of [BFP12]. Column T shows the log in base 2 of the required number of bit operation, computed using the estimator for the MQ problem available in the `CryptographicEstimators` software library [EVZB23]. Column k shows the number of "guessed" values. Bit complexities have been rounded to the closest integer.

Version		Parameters				Hybrid method		BiscuitZero	
	Level	q	n	m	Bit-security	T	k	T	k
v1	I	16	64	67	160	151	11	124	17
	II		87	90	210	201	13	163	26
	III		118	121	276	266	21	215	31
v2	I	256	50	52	143	140	0	133	3
	II		89	92	207	232	3	222	5
	III		127	130	272	326	4	312	9

$31/118 = 0.263$ for $\beta = 0.269$. When $q = 256, k/n = 17/64 = 0.266$, and X for $\beta = X$). We observe that BiscuitZero guesses more values than the classic hybrid method.

5.2 Universal Forgery Attack

We discuss the consequence of the Kales-Zaverucha forgery attack described in section 2.5 combined with the BiscuitPowAff2 algorithm of Figure 3. The attack has two phases, and depends on four parameters (u, k, ℓ and τ_1):

Offline Get $\mathbf{x} \leftarrow \text{BiscuitPowAff2}(f_1, \ldots, f_m, u, k, \ell)$.

Online Given m and $\mathbf{x}$, run the Kales-Zaverucha attack of section 2.5 with $\tau = \tau_1 + \tau_2$.

We estimate the cost of the attack as the maximum of the running times of these two phases. The cost of the offline phase is given by (8) and that of the online phase is given by (6). Only u enables a trade-off between the two phases; choosing the best k and ℓ speeds up the offline phase, while choosing the best τ_1 speeds up the online phase. Given concrete parameters m, n, q, we find the values of all parameters of the attack by trying all possible values of u; for each value of u, we find the best possible (k, ℓ) on the one hand, and the best possible τ_1 on the other hand.

Table 5 shows the results. This forgery attack cost almost as much as recovering the secret key. Here are a few comments. When $u = \ell = 0$ then the forgery attack degenerates into the key-recovery attack. When $\ell \approx n/2$ and $k \approx 0$ (as in the first two rows), then the algorithm of section 4.1 degenerates into its simpler version of section 3.3, that does not use the F_5 algorithm.

6 Other Properties of Biscuit-Style Polynomial Systems

We mention in this section other interesting properties of Biscuit-Style polynomial systems. Both hint at the fact that the structure they contain is so strong that it can potentially be exploited. The two properties we demonstrate below rely on well-chosen linear changes of variables.

6.1 Free Gröbner Basis in the Underdetermined Case

If $n \geq 2m$ (underdetermined system), we observed in section 3.2 that it is possible to sample random solutions in polynomial time. But in fact, there is more: with high probability,

Table 5: Forgery attacks on Biscuit. V1 denotes the version of Biscuit in [BKPV23a] and V2 is in [BKPV23b]. The "sec." column shows the level of bit-security claimed in [BKPV23a] and [BKPV23b]. Column T shows the log in base 2 of the required number of bit operation, as provided by the `MQEstimator` software [BMSV22b]. Bit complexities have been rounded to the closest integer.

Version			Parameters						KZ attack				
			N	τ	q	n	m	sec.	T	u	ℓ	k	τ_1
v1	I	short	256	18	16	64	67	143	116	4	33	1	6
		fast	16	34					120	4	33	1	8
	II	short	256	30		87	90	208	162	3	12	21	13
		fast	16	54					163	1	1	26	33
	III	short	256	40		118	121	274	215	3	7	32	18
		fast	16	73					215	0	0	31	73
v2	I	short	256	18	256	50	52	143	131	4	4	5	4
		fast	32	28					133	0	0	3	28
	II	short	256	25		89	92	207	199	10	11	7	2
		fast	32	40				210	205	9	9	8	2
	III	short	256	33		127	130	272	265	16	18	9	2
		fast	32	53				275	271	14	15	8	2

a well-chosen linear change of variables directly yields a Gröbner basis. Indeed, suppose that the $2m$ vectors $\mathbf{v}_1, \ldots, \mathbf{v}_m, \mathbf{w}_1, \ldots, \mathbf{w}_m$ are linearly independent (this happens with high probability if they are randomly chosen). Then consider the linear change of variable that sends $\langle \mathbf{v}_i, \mathbf{x} \rangle$ to a new variable y_i as well as $\langle \mathbf{w}_j, \mathbf{x} \rangle$ to z_j. Write $\mathbf{y} = (y_1, \ldots, y_m)$ and $\mathbf{z} = (z_1, \ldots, z_{n-m})$. This change of variable turns the k-th polynomial of the system into

$$f'_k := y_k z_k + \langle \mathbf{u}'_k, \mathbf{y} \rangle + \langle \mathbf{v}'_k, \mathbf{z} \rangle + c_k. \tag{10}$$

where $\mathbf{u}'_k$ and $\mathbf{v}'_k$ are vectors with coordinates in $\mathbb{F}_q$.

Lemma 1. *Let G denote the collection of polynomials $f'_1, \ldots, f'_m$ described by (10). G is a Gröbner basis for the lexicographic order (and any graded order).*

Proof. In the lexicographic order (or any graded order), the leading monomials of f'_k is automatically $y_k z_k$. It follows that the leading monomials of all polynomials in G only contain distinct variables, and therefore the S-polynomial of any two polynomials f'_i and f'_j is simply

$$S(f'_i, f'_j) = y_i z_i f'_j - y_j z_j f'_i = y_i z_i \left(f'_j - y_j z_j\right) - y_j z_j \left(f'_i - y_i z_i\right)$$

Let us compute the multivariate division of $g := y_i z_i \left(f'_j - y_j z_j\right)$ by the order sequence of polynomials G. g contains monomials of the shape $y_i y_k z_i$ and $y_i z_i z_k$. These can only be reduced by f'_i — they are divisible by the leading monomial of f'_i and not divisible by the leading monomial of any other polynomial in G. It follows that the result of running the division algorithm is:

$$g = f'_i \left(f'_j - y_j z_j\right) + \left(y_i z_i - f'_i\right) \left(f'_j - y_j z_j\right)$$

(the remainder is the second summand). The division algorithm is linear [CLO91, Exercise 2.3.12], and this implies that the multivariate division of the S-polynomial by G yields:

$$S(f'_i, f'_j) = f'_i \left(f'_j - y_j z_j\right) - f'_j \left(f'_i - y_i z_i\right)$$

And the remainder is zero. By Buchberger's criterion [CLO91, Thm. 2.6.6], G is then a Gröbner basis. □

6.2 Free Elimination of One Third the Variables

For simplicity, assume that n is a multiple of 3, set $\ell := n/3$ and $m \geq n/3$ (not too underdetermined). Suppose that the n vectors $\mathbf{u}_1, \dots \mathbf{u}_\ell, \mathbf{v}_1, \dots, \mathbf{v}_\ell, \mathbf{w}_1, \dots, \mathbf{w}_\ell$ are linearly independent. Then consider the linear change of variable that sends $\langle \mathbf{u}_i, \mathbf{x} \rangle$ to a new variables w_i, $\langle \mathbf{v}_i, \mathbf{x} \rangle$ to y_i as well as $\langle \mathbf{w}_i, \mathbf{x} \rangle$ to z_u. Write $\mathbf{w} = (w_1, \dots, w_\ell)$, $\mathbf{y} = (y_1, \dots, y_\ell)$ and $\mathbf{z} = (z_1, \dots, z_\ell)$. This change of variable turns the first ℓ polynomials of the system into

$$q_k := y_k z_k + w_k + c_k. \tag{11}$$

In a solution of the polynomial system, we therefore have $w_k = -y_k z_k - c_k$ for $1 \leq k \leq \ell$. Therefore, in the remaining $m - \ell$ polynomials, we replace all occurences of w_k by $-y_k z_k - c_k$. This eliminates $n/3$ variables and yields a new polynomial system $m - n/3$ polynomials of degree up to 4 in $2n/3$ variables (the y_i's and the z_i's). Call $\mathcal{S}$ the resulting system. Note that no known technique is capable of obtaining this result starting from an arbitrary system of quadratic polynomials in polynomial time.

Unfortunately, we could not find any way to solve $\mathcal{S}$ faster than the original problem.

Naively trying to attack $\mathcal{S}$ directly does not yield any improvement. It can be solved in $q^{2n/3}$ operations by exhaustive search, but this is worse than the simple technique discussed in section 3.2. It is also not easier to solve than the original one using algebraic techniques, as the increase in degree offsets the reduction in the number of variables.

$\mathcal{S}$ is nevertheless quite structured. Consider the following kind of polynomials:

$$r_k := \sum_{i=1}^{n} e_i y_i z_i + f_i y_i + g_i z_i \tag{12}$$

We say that r_k is a polynomial in "shortbread form". This is quite reminiscent of the canonical form of quadratic polynomials over fields of characteristic two. Note that this is different from the "Biscuit form" of (1) — the stucture is even stronger. Polynomials in $\mathcal{S}$ can be written as $f_k = u_k + v_k \times w_k$ where u_k, v_k and w_k are in shortbread form (the $y_i z_i$ terms come from the replacement of w_i by $y_i z_i$).

Our attemps at exploiting this structure also proved unsuccessful. Guessing all the y_i's turns $\mathcal{S}$ back into a collection of Biscuit-style polynomials in $n/3$ variables — but we could have obtained the exact same result by applying the technique of section 3.

Consider the following variant. Guess $n/3$ scalars α_k such that $h_k - \alpha_k = 0$. This yields $2n/3$ polynomial equations in shortbread form in $2n/3$ variables $(y_1, \dots, y_{n/3}, z_1, \dots, z_{n/3})$. Because of the shortbread structure, only $n/3$ distinct quadratic monomials occur in these $2n/3$ polynomials. Performing linear combinations of the polynomials to eliminate the quadratic terms yield $n/3$ linear equations — for instance this allows to express the z_i as linear functions of the y_i. But this again yield a collection of $n/3$ Biscuit-style polynomials in $n/3$ variables.

7 Conclusion and Future Work

As a conclusion, we proposed an improved hybrid approach for solving Biscuit-style polynomial system. As a direct application, we have shown that the first set of parameters proposed by the Biscuit design team signature scheme does not offer the security level expected by NIST. The designers of Biscuit updated the specification of their scheme following preliminary versions of our findings. Biscuit is still a competitive signature scheme, but parameters have to be enlarged and the new version is slightly less efficient.

Biscuit-style polynomial systems are similar to the those one obtain by applying the Arora-Ge algebraic attack [AG11] to LWE with binary error. One possible future work could be to design a specific hybrid method to run algebraic attacks on LWE with binary error, with the aim of improving the results presented in [ACF+15].

Acknowledgements

We are grateful to Ludovic Perret for his support during the preparation of this work.

We acknowledge financial support from the French *Agence Nationale de la Recherche* under project "GORILLA" (ANR-20-CE39-0002) and "PostCryptum" (ANR20-ASTR-0011).

References

[ACF+15] Martin R. Albrecht, Carlos Cid, Jean-Charles Faugère, Robert Fitzpatrick, and Ludovic Perret. Algebraic algorithms for LWE problems. *ACM Commun. Comput. Algebra*, 49(2):62, 2015. doi:10.1145/2815111.2815158.

[AG11] Sanjeev Arora and Rong Ge. New algorithms for learning in presence of errors. In Luca Aceto, Monika Henzinger, and Jirí Sgall, editors, *Automata, Languages and Programming - 38th International Colloquium, ICALP 2011, Zurich, Switzerland, July 4-8, 2011, Proceedings, Part I*, volume 6755 of *Lecture Notes in Computer Science*, pages 403–415. Springer, 2011. doi:10.1007/978-3-642-22006-7_34.

[Ash90] R.B. Ash. *Information Theory*. Dover books on advanced mathematics. Dover Publications, 1990.

[AW21] Josh Alman and Virginia Vassilevska Williams. A refined laser method and faster matrix multiplication. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 522–539. SIAM, 2021. doi:10.1137/1.9781611976465.32.

[Bar04] Magali Bardet. *Étude des systèmes algébriques surdéterminés. Applications aux codes correcteurs et à la cryptographie*. PhD thesis, Pierre and Marie Curie University, Paris, France, 2004. URL: https://tel.archives-ouvertes.fr/tel-00449609.

[BFP09] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Hybrid approach for solving multivariate systems over finite fields. *J. Math. Cryptol.*, 3(3):177–197, 2009. doi:10.1515/JMC.2009.009.

[BFP12] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Solving polynomial systems over finite fields: improved analysis of the hybrid approach. In Joris van der Hoeven and Mark van Hoeij, editors, *International Symposium on Symbolic and Algebraic Computation, ISSAC'12, Grenoble, France - July 22 - 25, 2012*, pages 67–74. ACM, 2012. doi:10.1145/2442829.2442843.

[BFR23] Ryad Benadjila, Thibauld Feneuil, and Matthieu Rivain. MQ on my mind: Post-quantum signatures from the non-structured multivariate quadratic problem. *IACR Cryptol. ePrint Arch.*, page 1719, 2023. URL: https://eprint.iacr.org/2023/1719.

[BFS15] Magali Bardet, Jean-Charles Faugère, and Bruno Salvy. On the complexity of the F5 gröbner basis algorithm. *J. Symb. Comput.*, 70:49–70, 2015. doi:10.1016/j.jsc.2014.09.025.

[BFSS13] Magali Bardet, Jean-Charles Faugère, Bruno Salvy, and Pierre-Jean Spaenlehauer. On the complexity of solving quadratic boolean systems. *J. Complex.*, 29(1):53–75, 2013. doi:10.1016/J.JCO.2012.07.001.

[BKPV23a] Luk Bettale, Delaram Kahrobaei, Ludovic Perret, and Javier Verbel. Biscuit. Technical report, National Institute of Standards and Technology, 2023. available at https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures.

[BKPV23b] Luk Bettale, Delaram Kahrobaei, Ludovic Perret, and Javier Verbel. Biscuit: New mpcith signature scheme from structured multivariate polynomials. Cryptology ePrint Archive, Paper 2023/1760, 2023. https://eprint.iacr.org/2023/1760. URL: https://eprint.iacr.org/2023/1760.

[BMSV22a] Emanuele Bellini, Rusydi H. Makarim, Carlo Sanna, and Javier A. Verbel. An estimator for the hardness of the MQ problem. In Lejla Batina and Joan Daemen, editors, *Progress in Cryptology - AFRICACRYPT 2022: 13th International Conference on Cryptology in Africa, AFRICACRYPT 2022, Fes, Morocco, July 18-20, 2022, Proceedings*, volume 13503 of *Lecture Notes in Computer Science*, pages 323–347. Springer Nature Switzerland, 2022. doi:10.1007/978-3-031-17433-9_14.

[BMSV22b] Emanuele Bellini, Rusydi H. Makarim, Carlo Sanna, and Javier A. Verbel. An estimator for the hardness of the MQ problem. pages 323–347, 2022. doi:10.1007/978-3-031-17433-9_14.

[Buc65] B. Buchberger. *Ein Algorithmus zum Auffinden der Basiselemente des Restklassenringes nach einem nulldimensionalen Polynomideal (An Algorithm for Finding the Basis Elements in the Residue Class Ring Modulo a Zero Dimensional Polynomial Ideal)*. PhD thesis, Mathematical Institute, University of Innsbruck, Austria, 1965. English translation in J. of Symbolic Computation, Special Issue on Logic, Mathematics, and Computer Science: Interactions. Vol. 41, Number 3-4, Pages 475–511, 2006.

[CCNY12] Chen-Mou Cheng, Tung Chou, Ruben Niederhagen, and Bo-Yin Yang. Solving quadratic equations with XL on parallel architectures. pages 356–373, 2012. doi:10.1007/978-3-642-33027-8_21.

[CGP03] Nicolas T. Courtois, Louis Goubin, and Jacques Patarin. SFLASHv3, a fast asymmetric signature scheme. Cryptology ePrint Archive, Report 2003/211, 2003. https://eprint.iacr.org/2003/211.

[CHR+16] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. From 5-pass MQ-based identification to MQ-based signatures. pages 135–165, 2016. doi:10.1007/978-3-662-53890-6_5.

[CKPS00] Nicolas Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. pages 392–407, 2000. doi:10.1007/3-540-45539-6_27.

[CLO91] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra, (Undergraduate Texts in Mathematics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1991.

[Cop94] Don Coppersmith. Solving homogeneous linear equations over gf(2) via block wiedemann algorithm. *Mathematics of Computation*, 62(205):333–350, 1994. URL: http://www.jstor.org/stable/2153413.

[EVZB23] Andre Esser, Javier A. Verbel, Floyd Zweydinger, and Emanuele Bellini. `CryptographicEstimators`: a software library for cryptographic hardness estimation. *IACR Cryptol. ePrint Arch.*, page 589, 2023. URL: https://eprint.iacr.org/2023/589.

[Fau02] Jean-Charles Faugère. A New Efficient Algorithm for Computing Gröbner Bases Without Reduction to Zero (F5). In T. Mora, editor, *ISSAC '02: Proceedings of the 2002 International Symposium on Symbolic and Algebraic Computation*, pages 75–83, New York, NY, USA, July 2002. ACM Press. isbn: 1-58113-484-3. URL: https://doi.org/10.1145/780506.780516.

[FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using gröbner bases. pages 44–60, 2003. doi:10.1007/978-3-540-45146-4_3.

[GJ79] M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar signature schemes. pages 206–222, 1999. doi:10.1007/3-540-48910-X_15.

[KZ20] Daniel Kales and Greg Zaverucha. An attack on some signature schemes constructed from five-pass identification schemes. pages 3–22, 2020. doi:10.1007/978-3-030-65411-5_1.

[Pat96] Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms. pages 33–48, 1996. doi:10.1007/3-540-68339-9_4.

[SCH+19] Simona Samardjiska, Ming-Shing Chen, Andreas Hulsing, Joost Rijneveld, and Peter Schwabe. MQDSS. Technical report, National Institute of Standards and Technology, 2019. available at https://csrc.nist.gov/projects/post-quantum-cryptography/post-quantum-cryptography-standardization/round-2-submissions.

[VGO+20] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi:10.1038/s41592-019-0686-2.

[YDH+15] Takanori Yasuda, Xavier Dahan, Yun-Ju Huang, Tsuyoshi Takagi, and Kouichi Sakurai. MQ challenge: Hardness evaluation of solving multivariate quadratic problems. Cryptology ePrint Archive, Report 2015/275, 2015. https://eprint.iacr.org/2015/275.