Ariadne Thread and Salt: New Multivariate Cryptographic Schemes with Public Keys in Degree 3

Gilles Macario-Rat[1] and Jacques Patarin[2]

[1] Orange, Orange Gardens, 46 avenue de la République, F-92320 Châtillon, France
gilles.macariorat@orange.com
[2] Versailles Laboratory of Mathematics, UVSQ, CNRS, University of Paris-Saclay
jpatarin@club-internet.fr

Abstract. In this paper, we present two new perturbations for the design of multivariate schemes that we call "Ariadne Thread" and "Salt". From these ideas, we present some efficient multivariate encryption and signature schemes with explicit parameters that resist all known attacks. In particular they resist the two main (and often very powerful) attacks in this area: the Gröbner attacks (to compute a cleartext from a ciphertext) and the MinRank attacks (to recover the secret key). Ariadne Threat and Salt can also be seen as new "perturbations" that we can use to enforce many multivariate schemes. The "Salt" perturbation works only for public key equations of degree (at least) 3. Similarly at present the "Ariadne Thread" perturbation seems to be particularly powerful with public keys of degree 3. Despite this, the size of the public key may still be reasonable since we can use larger fields (and also maybe non dense equations). Ariadne Thread perturbation seems to be particularly interesting for encryption. This is unusual since in multivariate cryptography encryption is generally more difficult than signatures.

Keywords: public-key cryptography, post-quantum multivariate cryptography, UOV, HFE, AES.

1 Introduction

Many schemes in Multivariate cryptography have been broken. Among the most spectacular attacks we can mention that the C* scheme of Matsumoto and Imai [14] has been broken in [16], the SFlash scheme submitted to the NESSIE competition has been broken in [6, 10, 11] , the LUOV scheme [5] submitted to the Post-Quantum NIST competition has been broken in [8], and the GeMMS schemes [7] has been broken in [18]. At present the two main general attacks in multivariate cryptography are the use of Gröbner bases in "direct attacks" (in order to find a solution of the public equations involved without finding the secret key, cf [12]), and the MinRank attacks in order to find the secret key [13, 2]. In many schemes the Gröbner attack is dangerous because the degree of regularity

of the public equations is smaller than for random quadratic equations. Recently the MinRank attacks have become much more powerfull than before due to the introduction of the Minor equations [2].

Despite these dangerous and powerful attacks, multivariate cryptography remains an interesting area of research. This is mainly due to three facts. First, the schemes, if they can resist non-quantum attacks, are also expected to resist quantum computers, i.e. multivariate cryptography is one of the family of "post-quantum" cryptography (with lattices, codes, hash-based cryptography, isogenies, combinatorial schemes). Second, the MQ problem (solving a set of Multivariate Quadratic equations on finite field) is NP-hard on any finite field, and seems to be very difficult to solve when the equations are random and the number of variables is about the same as the number of equations. Third, some properties can be obtained at present only with multivariate cryptography such as ultra-short public-key signatures, or encryption with ultra-short blocs [15].

In this paper, we present two new tools that can be useful to design multivariate encryption scheme, or to enforce the security of any multivariate encryption scheme. We call these ideas "Ariadne Thread" and "Salt". At present it is easier to design public key multivariate signature schemes than public key multivariate encryption schemes, therefore it is nice to have new encryption possibilities.

2 Ariadne Thread and Salt: main ideas

2.1 Notations

As in all classical multivariate schemes, we use a finite field $\mathbb{F}_q$ with q elements and we deal with the ring of polynomials in n variables $(x_1, \ldots, x_n)$ (or simply x) over $\mathbb{F}_q$, noted $\mathbb{F}_q[x]$. Therefore here $\mathbb{F}_q[x]^m$ will refer to the ring module of n-ary m-dimensional polynomials, that we call (n, m)-polynomials for short. We note $\deg(f)$ the degree of a polynomial f. Classically by extension, the degree of a (n, m)-polynomial is the maximum degree of its $(n, 1)$-components. In the particular case $n = m$, by the mean of an adequate isomorphism between $\mathbb{F}_q^n$ and $\mathbb{F}_{q^n}$, we could note x^{q^i} the i-th iterate Frobenius mapping, considered as a (n, n)-polynomial of degree 1.

2.2 The tweak

We start from some trapdoor scheme, $y = f(x)$, where f is a degree-d (n, m)-polynomial that we can invert somehow. Typical examples of f are HFE and UOV among many others, but for the sake of our presentation, we can stick to a generic f. Our purpose is to add a perturbation to f, in order to get a new scheme, with stronger security, and in particular dismiss all possible "low rank" issues. We will discuss this point hereafter. Classical schemes have degree 2 such as HFE or UOV; we will consider here larger degrees, mainly $d = 3$, that provide more interest to our new perturbation.

We now introduce our new "tweak": $p(x)P(x)$, where p and P are respectively random $(n, 1)$- and (n, m)-polynomials. We also require that $\deg(p)+\deg(P) = d$.

The new trapdoor can be expressed as $\tilde{f}(x) = f(x) + p(x)P(x)$. The advantages of this new trapdoor are the followings.

- Its degree is still at most d.
- Whatever the degree of p, $p(x)$ can only take q values (those of $\mathbb{F}_q$). In turn, it follows immediately that $p(x)P(x)$
  1. has probability about $1/q$ to be zero, relatively to x,
  2. can be approximate by a set : $\mathcal{S} = \{aP : a \in \mathbb{F}_q\}$ of q polynomials.
- It gives the opportunity to design "high rank" schemes, since the polynomials p and P may be chosen adequately, and therefore thwarts the MinRank attacks.

2.3 Ariadne Thread for encryption

The Ariadne perturbation is the conjunction of our new tweak and a special mode of operation. In this mode, to invert the trapdoor $y = f(x) + p(x)P(x)$, we assume that $p(x) = 0$. Then, we are led to simply solve $y = f(x)$. So, we compute a first set $\mathcal{S}_0$ of potential solutions of $y = f(x)$ with the appropriate method, and keep the only solutions $\mathcal{S}_1 = \{x \in \mathcal{S}_0 | p(x) = 0\}$. Notice here, that in a single inversion, there is only a probability of about $1/q$ that one gets a correct solution. The rest of the time, one gets unwanted solutions, or even no solution at all. However, with multiple inversions in an adequate protocol, this mode of operation is suitable for encryption as we will show later.

Remark 1. On the contrary, we explain here why Ariadne Thread is not suitable for the signature mode. Indeed, the signature mode would reveal publicly couples of the kind (m_i, σ_i) satisfying $m_i = \mathcal{P}(\sigma_i)$. Then all the signatures σ_i must also satisfy some equation $p'(\sigma) = 0$, where p' is a $(n, 1)$-polynomial related to the secret key. If enough signatures are known, a linear system leading to the coefficients of p' can be solved, so the key is broken.

2.4 Salt for signature

The Salt perturbation is the conjunction of the same tweak, but a different mode of inversion. In order to invert the trapdoor $y = f(x) + p(x)P(x)$, we make an exhaustive search on all the q possible values of $p(x)$. For one possible value a, we simply invert $y = f(x) + aP(x)$, and keep only the solutions satisfying $p(x) = a$. Of course, we have made here the additional assumption that $y = f(x) + aP(x)$ is indeed efficiently invertible. This assumption is highly related to the choice of the initial scheme f but not difficult to fulfil. For instance, for HFE, it suffices to assume that the polynomial P has "HFE" shape, or for UOV, it is "UOV-compatible", etc. This mode of operation is more suitable for signature, and costs also a factor q compared to the original scheme.

3 Security analysis of Ariadne and Salt in degree 2

We first give a word about these new perturbations when $d = 2$, and show that they have not much interest in this case. We analyse them particularly in the light of the HFE and UOV schemes, but the analyse can certainly be generalized to other schemes. The most restrictive condition we have to fulfil is $\deg(p) + \deg(P) = 2$, then we can imagine:

- $\deg(p) = 0$, $\deg(P) = 2$. We skip this case, since p must be constant and therefore does not add entropy.
- $\deg(p) = 1$, $\deg(P) = 1$. In this case p and P are linear. For HFE and regarding the MinRank Attack, this perturbation simply increases the rank of the secret polynomial by an amount of 1, which is not competitive, since the perturbation already costs a factor q. For UOV, this perturbation amounts to transform 1 'oil' variable into 1 'vinegar' variable, which is useless compared to the original scheme.
- $\deg(p) = 2$, $\deg(P) = 0$. Here, p is a random quadratic form, P is a random element of $\mathbb{F}_q^n$. This case has already been described in literature as the "+" perturbation, i.e. adding a small amount of random quadratic polynomials to the original scheme. The effect of this perturbation can be cancelled by considering the adequate projection, and the natural isomorphism between $\mathbb{F}_q^n$ and $\mathbb{F}_{q^n}$:

$$\begin{aligned} y &= f(x) + p(x)P \\ y^q &= f(x)^q + p(x)P^q \\ yP^q - y^qP &= f(x)P^q - f(x)^qP \end{aligned}$$

  In particular, regarding the Gröbner basis computation attack, this shows, that the degree of regularity does not increase compared to the one of $y = f(x)$

4 Security analysis of Ariadne and Salt in degree 3: Gröbner and MinRank

4.1 Grobner

Our motivation for this new perturbation and particularly our interest in degree 3, lies in the behavior of Gröbner basis algorithms (see [1]). First, theory and also experiment can show that for "big enough" q, the degree D_{reg} of regularity of a random (n, n)-polynomial f of degree 3 is $2n+1$. This can be infer from the index of the first non-negative coefficient of the Hilbert series: $H(z) = (1-z^3)^n/(1-z)^n$ [1]. The complexity of solving a random system by Gröbner basis, of n equations in n variables of degree 3 over $\mathbb{F}_q$ can be estimated as $\binom{n+D_{reg}}{n}^\omega$, where ω is the greatest lower bound for the exponent of matrix multiplication algorithm. Taking into account the field equations $x_i^q - x_i = 0$, we get a more accurate formula for our case: $D_{reg} = \min(2n+1, q)$. So for q big enough, we clearly hope to have smaller n in degree 3 than in degree 2 for same level of security.

4.2 Minrank

In degree 3, we consider that the most useful definition of rank that should apply to a (n, n)-polynomial f in our case, is the minimum r of "elementary" products occurring in a sum that amounts to f, or so to say : $f(x) = \sum_{i=1}^{r} a_{3i-2}(x)a_{3i-1}(x)a_{3i}(x)$, where $a_i(x)$ are degree-1 (n, n)-polynomials. Since $d = 3$, the condition $\deg(p) + \deg(P) = d$ can non trivially be fulfilled with ($\deg(p) = 2$ and $\deg(P) = 1$) or ($\deg(p) = 1$ and $\deg(P) = 2$). The (classical) rank of a random degree-2 $(n, 1)$- or (n, n)- polynomial is n, with overwhelming probability. Hence, the rank of the tweak and therefore of the secret trapdoor, can be provably set to a value greater than n.

5 HFE-Ariadne in degree 3: examples of parameters and security results

We now propose a variant of HFE in degree 3 using the Ariadne Thread perturbation, for encryption mode.

5.1 Description of HFE-Ariadne

In this particular case, we have $n = m$, which enables us to use a natural isomorphism between $\mathbb{F}_q^n$ and $\mathbb{F}_{q^n}$. We set $f(x) = x^{q+2} + \alpha x^3$, a cubic binomial, α being chosen randomly in $\mathbb{F}_{q^n}$. The secret trapdoor is then $\tilde{f}(x) = x^{q+2} + \alpha x^3 + p(x)P(x)$, with $\deg(p) = 1$, and $\deg(P) = 2$. The public key is a composition $\mathcal{P} = S \circ \tilde{f} \circ T$ where S and T are two secret linear bijective maps of $\mathbb{F}_q^n$. The secret key is a description of α, p, P, S and T. Typical values of the parameters q and n are below 100, around 30 for instance. See below the Rationale section for parameter selection and further explanation.

5.2 Encryption protocol

We now describe the process for encryption and decryption. Alice and Bob first agree on a public symmetric cipher such as AES, that we note $E(K, X)$ and where K is the symmetric key and X is a binary representation of the clear text. The role of the symmetric cipher it to prevent an attacker from deriving algebraic equations issued from many related inputs. Then Alice and Bob agree on a sufficiently large amount of symmetric keys K_i, and also a value l. The way Alice and Bob make the agreement is not important, the cipher and the keys have just to be public, or at least known by Alice and Bob at the right time. Alice wants to send confidentially a message to Bob. To do so, she embeds the message into X and sends the list of the first l computed values $y_i = \mathcal{P}(E(K_i, X)), i = 1, \ldots, l$. The reason for doing so, is that, with high probability the Ariadne Thread condition be satisfied, namely one or more values satisfy $p(T(E(K_i, X))) = 0$. Individually, the condition has roughly a probability $1/q$ to be fulfilled, so the list should

have around q elements, that is $l \approx q$. Then for each received value y_i, Bob computes all the solutions in z of $f(z) = S^{-1}(y_i)$ and keeps the potential messages $E^{-1}(K_i, T^{-1}(z))$, for z satisfying $p(z) = 0$. To manage the detection of correct decryption, Alice and Bob can adopt for instance one of these strategies:

- X is the message and Bob decrypts until two potential messages are equal.
- A part of X is reserved for a message authentication code (MAC), and Bob decrypts until one of the potential messages has the correct MAC.

In the event of failure, Bob may ask Alice to renew the list by using the next l keys. Fortunately, the probability of many retries decreases exponentially. Moreover Bob should also randomly ask Alice for a retry even in case of success, with an adequate probability, so that an eavesdropper may not distinguish the cases with failure from the successful ones.

5.3 Rationale, sizes and performance

One major interest of the Ariadne-perturbation is that it thwarts the MinRank attack. So we should only be concerned by direct attacks. We observed for small values of n (up to 7 for which the complexity is still achievable), that the behavior of F4 algorithm which computes solutions of $\mathcal{P}(x) = y$ is the same for random systems with same dimensions (same number of equations and variables). Therefore, we hypothesized that this result could be generalized to higher values of n, and also that hybrid attacks [3] (combination of Gröbner basis and exhaustive search), should be the best option for an attacker. We used the Magma script in appendix that gives the complexity of such attack, based on the fact that the complexity can be upper bounded by $\binom{n+D_{reg}}{n}^{\omega}$, and that D_{reg}, the degree of regularity of the system can be evaluated by a Hilbert series.

For the HFE polynomial, we can choose the smallest degree possible (over $\mathbb{F}_{q^n}$), but we do not recommend only one monomial, so it should have at least two monomials, in order to avoid attacks such as the ones against C^* and SFlash, hence the choice $x^{q+2} + \alpha x^3$. If one needs a non homogeneous version, then it is possible to add the monomials in x^{q+1}, x^q, x^2, and x.

We note λ the security level. We should choose the parameters n and q such as the least complexity of hybrid attacks is above 2^λ. Then, there are many ways to choose (q, n), which give different balances between size and performance. For instance we propose the following sets of parameters for a level of security $\lambda = 128$ (assuming $\omega = 2.37$).

- $q = 5$, $n = 56$, the size of public keys is 648 Kbytes, average decryption takes 0.09s, and average encryption 2.5s.
- $q = 7$, $n = 47$, size is 325 Kbytes, average decryption takes 0.28s, and average encryption 1.7s.
- $q = 11$, $n = 40$, size is 230 Kbytes, average decryption takes 1s, and average encryption 1,3s.
- $q = 13$, $n = 39$, size is 208 Kbytes, average decryption takes 2s, and average encryption 1.4s.

Of course, encryption and decryption are highly parallelizable, and could benefit from multi-core CPUs.

6 UOV-Salt in degree 3: example of parameters and security results

We now propose a variant of UOV using the Salt perturbation and degree 3, for signature mode.

6.1 Description of UOV-Salt

In what follows, n_o and n_v are respectively the number of oil and vinegar variables. We note $n = n_o + n_v$, and the different variables $x_o = (x_1, \ldots, x_{n_o})$, $x_v = (x_{1+n_o}, \ldots, x_n)$, $x = (x_1, \ldots, x_n)$.

We set $f(x) = \sum_{i=1}^{n_o} x_i Q_i(x_v) + C(x_v)$, the "classical" UOV trapdoor extended in degree 3, where Q_i are random degree-2 (n_v, n_o)-polynomials, C is a random degree-3 (n_v, n_o)-polynomial. This is the same idea as in the original scheme: the trapdoor is linear in x_o, hence it can easily be inverted if x_v is set. The UOV perturbed scheme becomes: $\tilde{f}(x) = f(x) + p(x)P(x)$, where p is a degree-2 $(n, 1)$-polynomial, P is a degree-1 (n, n_o)-polynomial compatible with the UOV structure. More precisely $P(x) = \sum_{i=1}^{n_o} \alpha_i x_i + L(X_v)$ where the α_i are random elements of $\mathbb{F}_q^{n_o}$ and L is a random degree-1 (n_v, n_o)-polynomial. One can easily check that when x_v is set and p is guessed, the perturbed trapdoor can be inverted as claimed. The public key is a composition $\mathcal{P} = \tilde{f} \circ T$ where T is some secret linear bijective map over $\mathbb{F}_q^n$. The secret key is a description of Q_i, C, p, P and T. See below the Rationale section for parameter selection and further explanation.

6.2 Signature protocol

To sign a value y, the signer aims to find a value z such as $\mathcal{P}(z) = y$, or equivalently find x such as $\tilde{f}(x) = y$. In order to invert the trapdoor, the signer chooses at random x_v, makes all the q possible "guesses" on $p(x)$, invert each linear system on x_o, (if possible), and stops as soon as one guess on $p(x)$ is correct, otherwise makes a new choice for x_v. Then, once a value x is found, the signer outputs $z = T^{-1}(x)$. To verify a signature z of y, the verifier, simply checks that $\mathcal{P}(z) = y$.

6.3 Rationale, sizes and performance

The direct attack using Gröbner basis computation can be performed by fixing n_v variables out of n. The resulting system has n_o equations in n_o variables and we estimate with simulation that it behaves with F4 like a random system as soon as $n_v \geq n_o$. So, the solving complexity of hybrid attacks can be used when $n_o = n_v$. We estimate that the Salt perturbation thwarts the Kipnis-Shamir

attack, and as a rule of thumb, that $q^{n_v} > 2^\lambda$ and $q^{n_o} > 2^{2\lambda}$ should be sufficient to thwart other obvious attacks. We expect to have a more precise evaluation of these points in the future.

There are many ways to choose the parameters. So, for a security level of $\lambda = 128$, (assuming $\omega = 2.37$) we propose the following parameters: $q = 821$, $n_o = 27$, $n_v = 27$. The public key size is 914 Kbytes, a signature size is $(27+27)*10 = 540$ bits (each element of $\mathbb{F}_{821}$ is coded on 10 bits). On our Magma simulation, on average, a signature takes 330 ms and a verification 130 ms.

Here also the signature is parallelizable and could take benefit from multi-core CPUs.

6.4 Shorter signature thanks to UOV

In the previous section, we assumed that the input size of the signature function was twice the size of the security level, which enables to "plug" the output of a hash function of the same size, instead of the message itself, whenever the size of the message exceeds this size. This is the classical mode of signature using a hash function. In this section, we show how to reduce the size of the signature by exploiting the special property of UOV, which is that the inversion of UOV is based on the linearity of the oil variables.

Let's now suppose that y is the hash to sign, its size is at least 2λ bits as expected. Say it can be expressed as n' elements of $\mathbb{F}_q$. We define an UOV-Salt scheme $\mathcal{P}$ as previously, with a parameter n_o satisfying $q^{n_o} \approx 2^\lambda$ (instead of $2^{2\lambda}$ previously). So $\mathcal{P}(x)$ is a (n, n_o)-polynomial. We now introduce a public function $L(x, y) : \mathbb{F}_q^n \times \mathbb{F}_q^{n'} \mapsto \mathbb{F}_q^{n_o}$. We require this function to have the following properties : to be affine in x, to be easy to compute, and to be collision-free with good probability. First we may write : $L(x, y) = g_0(y) + \sum_{i=1}^{n} x_i g_i(y)$ which is by design affine in x. Then for instance the public functions g_i may be random degree-1 (n', n_o)-polynomials. Or, in order to avoid to describe them, they can be simulated by a pseudo random generator with adequate output size, with y used as a seed.

Now, to sign a value y, the signer must find a value z satisfying $\mathcal{P}(z) = L(z, y)$ and proceeds as follows. As y is given, the term $L(z, y)$ is evaluated in y and the equation becomes $\mathcal{P}(z) = \alpha_0 + \sum_{i=1}^{n} \alpha_i z_i$, where the α_i belong to $\mathbb{F}_q^{n_o}$. Solutions in z can be found with the same method as before: make the change of variables $z = T^{-1}(x)$, draw vinegar variables at random, guess $p(x)$, solve the system in degree 1 obtained in the oil variables, check if $p(x)$ is correct, then go back in z variable. To verify a signature z of y, the verifier, simply checks that $\mathcal{P}(z) = L(z, y)$.

For a security level of $\lambda = 128$, we propose the following parameters: $q = 8$, $n_o = 45$, $n_v = 45$. The public key size is 2070 Kbytes, the signature is $(45 + 45) * 3 = 270$ bits.

6.5 Variants and further directions

An immediate generalization of our idea is to replace the tweak $p(x)P(x)$ by a sum with similar terms: $\sum_{i=1}^{s} p_i(x)P_i(x)$, where s is a small number. This variant seems to have significant effects on the signature scheme. First, the impact on the performance is clearly a slow down of factor q^s on the search of a signature. However, it has also an impact on the security, since it "hides" more deeply the trapdoor inside the public key. It seems that it could compensate for a decrease of the number of vinegar variables. For instance, for the same level $\lambda = 128$, we may propose the following parameters: $q = 8$, $s = 3$, $n_o = 45$, $n_v = 42$. The public key size is 1917 Kbytes, the signature is $(45 + 42) * 3 = 261$ bits. Further investigations are ongoing.

7 Conclusion

"Ariadne Thread" and "Salt" are two new tool in Multivariate cryptography. From these ideas we have obtained multivariate scheme for encryption and signature with very nice properties: they are fast, have a reasonable size of public key, and they resist all the known attacks in multivariate cryptography. We think that it is particularly important to notice that MinRank attacks are thwarted. However since these ideas of "Ariadne Thread" and "Salt" are very new, and since many failures have occurred in multivariate cryptography it is natural to be suspicious about these ideas and to want to wait for more analysis before using these schemes in real life applications. The fact in this paper we use public equations of degree 3 (instead of 2 usually in multivariate cryptography) might also open the door to new and powerful attacks...

References

1. Bardet M., Faugère JC., Salvy B. (2015). On the complexity of the F5 Gröbner basis algorithmJournal of Symbolic Computation, 70, 49 - 70.
2. Bardet M. et al. (2020) Improvements of Algebraic Attacks for Solving the Rank Decoding and MinRank Problems. In: Moriai S., Wang H. (eds) Advances in Cryptology – ASIACRYPT 2020. ASIACRYPT 2020. Lecture Notes in Computer Science, vol 12491. Springer, Cham. https://doi.org/10.1007/978-3-030-64837-4_17
3. Bettale L., Faugère JC., Perret L. (2009). Hybrid approach for solving multivariate systems over finite fieldsJ. Math. Cryptol., 3(3), 177–197.
4. Beullens W. (2020). Improved Cryptanalysis of UOV and Rainbow. IACR Cryptol. 2020: 1343. https://eprint.iacr.org/2020/1343
5. Beullens W., Preneel B. (2017) Field Lifting for Smaller UOV Public Keys. In: Patra A., Smart N. (eds) Progress in Cryptology – INDOCRYPT 2017. INDOCRYPT 2017. Lecture Notes in Computer Science, vol 10698. Springer, Cham. https://doi.org/10.1007/978-3-319-71667-1_12
6. Bouillaguet C., Fouque PA., Macario-Rat G. (2011) Practical Key-Recovery for All Possible Parameters of SFLASH. In: Lee D.H., Wang X. (eds) Advances in Cryptology – ASIACRYPT 2011. ASIACRYPT 2011. Lecture Notes in Computer Science, vol 7073. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-25385-0_36

7. Casanova A., Faugère JC., Macario-Rat G., Patarin J., Perret L., Ryckeghem J. (2017). GeMSS: A Great Multivariate Short Signature. Research Report. UPMC - Paris 6 Sorbonne Universités ; INRIA Paris Research Centre, MAMBA Team, F-75012, Paris, France ; LIP6 - Laboratoire d'Informatique de Paris 6, https://hal.inria.fr/hal-01662158,
8. Ding J., Deaton J., Schmidt K., Vishakha, Zhang Z. (2020) Cryptanalysis of the Lifted Unbalanced Oil Vinegar Signature Scheme. In: Micciancio D., Ristenpart T. (eds) Advances in Cryptology – CRYPTO 2020. CRYPTO 2020. Lecture Notes in Computer Science, vol 12172. Springer, Cham. https://doi.org/10.1007/978-3-030-56877-1_10
9. Ding J., Perlner R., Petzoldt A., Smith-Tone D. (2018) Improved Cryptanalysis of HFEv- via Projection. In: Lange T., Steinwandt R. (eds) Post-Quantum Cryptography. PQCrypto 2018. Lecture Notes in Computer Science, vol 10786. Springer, Cham. https://doi.org/10.1007/978-3-319-79063-3_18
10. Ding J., Schmidt D. (2005) Rainbow, a New Multivariable Polynomial Signature Scheme. In: Ioannidis J., Keromytis A., Yung M. (eds) Applied Cryptography and Network Security. ACNS 2005. Lecture Notes in Computer Science, vol 3531. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11496137_12
11. Dubois V., Fouque PA., Shamir A., Stern J. (2007) Practical Cryptanalysis of SFLASH. In: Menezes A. (eds) Advances in Cryptology - CRYPTO 2007. CRYPTO 2007. Lecture Notes in Computer Science, vol 4622. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-74143-5_1
12. Faugère JC., Joux A. (2003) Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems using Gröbner bases. In: Boneh D. (eds) Advances in Cryptology - CRYPTO 2003. CRYPTO 2003. Lecture Notes in Computer Science, vol 2729. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-45146-4_3
13. Kipnis A., Shamir A. (1999) Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization. In: Wiener M. (eds) Advances in Cryptology — CRYPTO' 99. CRYPTO 1999. Lecture Notes in Computer Science, vol 1666. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-48405-1_2
14. Matsumoto T., Imai H. (1988) Public Quadratic Polynomial-Tuples for Efficient Signature-Verification and Message-Encryption. In: Barstow D. et al. (eds) Advances in Cryptology — EUROCRYPT '88. EUROCRYPT 1988. Lecture Notes in Computer Science, vol 330. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-45961-8_39
15. Patarin J., Macario-Rat G., Bros M., and Koussa E. (2020). Ultra-Short Multivariate Public Key SignaturesIACR Cryptol. ePrint Arch. 2020: 914. https://eprint.iacr.org/2020/914
16. Patarin J. (1996) Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms. In: Maurer U. (eds) Advances in Cryptology — EUROCRYPT '96. EUROCRYPT 1996. Lecture Notes in Computer Science, vol 1070. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-68339-9_4
17. Patarin J. (1995) Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88. In: Coppersmith D. (eds) Advances in Cryptology — CRYPT0' 95. CRYPTO 1995. Lecture Notes in Computer Science, vol 963. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-44750-4_20
18. Tao C., Petzoldt A., Ding, J. (2020). "Improved Key Recovery of the HFEv- Signature Scheme". IACR Cryptol. ePrint Arch. 2020: 1424. https://eprint.iacr.org/2020/1424

Appendices

A Magma code for the complexity

```
HilbertSeries:= function(q,n,DEGS);
//Input
//  q size of the finite field
//  n number of variables
//  DEGS list of degree of equations
//Ouput
//  The index of the first non negative coefficient
//  of the Hilbert Series of the corresponding system
if #DEGS lt N then return "Under determined system"; end if;
R<z>:= PowerSeriesRing(Rationals());
if q eq 2 then return "Not implemented"; end if;
HS:= &*[ 1-z^d : d in DEGS] / (1-z)^N;
DREG := 0;
while Coefficient(HS,DREG) gt 0 do
    DREG +:= 1;
end while;
return DREG;
end function;

complexity:=function(q,n,Degs: omega := 2.37);
//Input
//  q size of the finite field
//  n number of variables
//  DEGS list of degree of equations
//  Optional: algebraic constant
//Output
//  Log2 of complexity
//  Estimated degree of regularity
//  Use of Field equations
dreg:=HilbertSeries(n,Degs);
dreg2:=HilbertSeries(n,Degs cat [q: i in [1..n]]);
res:=dreg2 lt dreg;
dreg:=Minimum(dreg,dreg2);
return Log(2,Binomial(n+dreg,n)^omega),dreg, res;
end function;

hybrid:=function(q,n,Degs);
//Input
//  q size of the finite field
//  n number of variables
//  DEGS list of degree of equations
//Output
```

```
//  Log2 of complexity
//  Estimated degree of regularity
//  Number of variables to fix : best trade-off
//  Use of Field equations
compmin:=1000;
dregmin:=0;
kmin:=0;
fieldmin:=false;
for k:=0 to n do;
    comp,dreg,field:=complexity(q,n-k,Degs);
    comp+:=k*Log(2,q);
    if comp lt compmin then
        compmin:=comp;
        kmin:=k;
        dregmin:=dreg;
        fieldmin:=field;
    end if;
end for;
return compmin, dregmin, kmin, fieldmin;
end function;
```

Table 1. Complexity ($\log_2$) of hybrid attacks for random systems of n equations in n variables of degree 3. k: number of variables to fix for the best trade-off, D_{reg}: degree of regularity for the best trade-off. Comp=$q^k \binom{n-k+D_{reg}}{D_{reg}}^{\omega}$

q	n	Comp	D_{reg}	k
5	56	130.03	1	56
7	47	129.72	5	36
8	45	130.90	6	32
9	42	128.60	9	24
11	40	128.63	9	22
13	39	131.30	8	23
16	37	130.41	11	17
17	37	131.90	11	17
19	36	130.38	11	16
23	34	128.42	10	16
29	33	128.74	16	9
37	32	129.24	18	7
49	31	128.19	16	8
73	30	128.06	18	6
121	29	128.31	19	5
277	28	128.04	22	3
821	27	128.02	24	2