The security of Hidden Field Equations (HFE)

Nicolas T. Courtois

Systèmes Information Signal (SIS), Université de Toulon et du Var
BP 132, F-83957 La Garde Cedex, France
courtois@minrank.org
http://hfe.minrank.org

Abstract. We consider the basic version of the asymmetric cryptosystem HFE from Eurocrypt 96.
We propose a notion of non-trivial equations as a tentative to account for a large class of attacks on one-way functions. We found equations that give experimental evidence that *basic* HFE can be broken in expected polynomial time for any constant degree d. It has been independently proven by Shamir and Kipnis [Crypto'99].
We designed and implemented a series of new advanced attacks that are much more efficient that the Shamir-Kipnis attack. They are practical for HFE degree $d \leq 24$ and realistic up to $d = 128$. The 80-bit, 500$ Patarin's 1st challenge on HFE can be broken in about 2^{62}.
Our attack is subexponential and requires $n^{\frac{3}{2}\log d}$ computations. The original Shamir-Kipnis attack was in at least $n^{\log^2 d}$. We show how to improve the Shamir-Kipnis attack, by using a better method of solving the involved algebraical problem MinRank. It becomes then in $n^{3\log d + \mathcal{O}(1)}$.
All attacks fail for modified versions of HFE: HFE$^-$ (Asiacrypt'98), HFEv (Eurocrypt'99), Quartz (RSA'2000) and even for Flash (RSA'2000).

Key Words: asymmetric cryptography, finite fields, one-way functions, Hidden Field Equation, HFE problem, basic HFE, MinRank problem, short signatures.

1 Introduction

The HFE trapdoor function Eurocrypt 96 [14], defined in 4, is one of the most serious alternative trapdoor functions. It generalizes the previous Matsumoto-Imai cryptosystem from Eurocrypt 88 [10] broken by Patarin in [13, 14].

HFE operates over finite fields. In this paper we restrict to the *basic* version of HFE, and to fields of characteristic 2. Thus we study a trapdoor function $F : GF(2^n) \rightarrow GF(2^n)$. We focus on the *cracking problem* of computing the inverse of the *basic* HFE encryption function, without trying to recover it's secret key.

In the section 2 we attempt to base a notion of a one-way function on algebraic criteria. We propose a "boosting model" which is nothing else that a kind of semantics of all deterministic cryptographic attacks. This approach, subsequently

narrowed down, proves particularly relevant to HFE attacks. The security is expressed in terms of properties of implicit equations that relate the inputs x_i and the outputs y_i of a function. An equation substituted with a given output value may, or may not, produce a new non-trivial equation on x_i. New equations boost the set of known linearly independent equations on the x_i, and at some point they should allow to compute the actual values of the x_i.

There is no doubt that our problem is closely related to polynomial elimination (Gröbner bases, XL algorithm [21]). Thus in section 3 we study the NP-complete problem of solving multivariate quadratic equations called sometimes MQ. A simple idea of linearizing and applying Gauss elimination can indeed be seen as eliminating equations (simple case of Gröbner bases algorithm), however we reinterpret it in section 3 in terms of implicit equations.

We distinguish between this 'elimination paradigm' and our approach called 'implicit equations paradigm'. Those methods ar different and complementary. We don't combine equations formally, trying to eliminate among all equations that we could construct within some size limitation. Instead of that, the problem is to find special subsets of such equations, that for algebraical reasons might be related. We are not limited (at all) by the size the equations, but only the size of the subset we selected (!).

The whole idea that it is interesting to do so, is the object of this paper. We may go back to the cryptanalysis of the Matsumoto-Imai cryptosystem described briefly in 4.1, to understand that algebraical reasons may suggest (or prove) the existence of some type of equations. The idea had several generalizations, such as the affine multiple attack by Jacques Patarin [13, 9] and other described here and in [4]. It was already known since [14] that some such equations will exist for *basic* HFE. In the present paper we show precisely what kind of equations exist and how to use them in realistic attacks.

Though it is very clear that the equations we have found in the present paper, exist for algebraical reasons, we were not able to explain them. They have been found on much more experimental basis, and it remains an open problem to understand them better. We did several months of extended computer simulations (section 5.6), to find memory-efficient types of equations that gave what is now the best known attack on *basic* HFE.

In the whole process of solving equations by finding other equations, we had to distinguish different types of equations. We denote them by expressions in x, y, X, Y, see section 5.1). We also distinguish several kinds of equations in terms of both their behaviour and a way they have been computed. Thus we had to invent some special vocabulary and notations, especially that some notions are informal.

A **glossary** of words that have special meaning in this paper, usually "double-quoted", along with common notations, is compiled at the end of the paper.

The section 5 shows precisely several classes of equations we have found and their immediate applications in an attack. Thus we get a strong experimental evidence that *basic* HFE can be broken in expected polynomial time if the degree

d is constant. The same result has just been independently found by Shamir and Kipnis at Crypto'99 [23].

We show that *basic* HFE is not secure for degree $d \leq 24$, while the original paper [14] suggested the HFE degree $d = 17$ as secure enough. Therefore, as we show in 5.10, in order to break the 500$ HFE challenge with $d = 96$ we need 2^{62} computations and 33 Tb of memory.

We introduced successive improvements to this attack. First, it is in fact possible to recover, recompose and use only parts of the equations ("reconciliation attack") - section 6.1. Secondly, the "distillation attack" of section 6.1-6.3 manages also to remove other, "interference" equations that unfortunately appear when the parts are too small. The final output is a method that uses very long equations without ever computing them, which dramatically reduces the memory requirements for Challenge 1 to 390 Gb.

In the section 7.1 we estimate the asymptotic complexity of our attacks. It is polynomial for a fixed HFE degree d and subexponential in general. If we go back to the Shamir-Kipnis attack on (basic) HFE from Crypto'99 [23], though it is very different, it gives similar results with much worse complexity. In the section 8 we introduce an improved version of it, that gives the asymptotic complexity similar that our attacks.

It is not true that HFE is broken. All attacks may have substantial complexity and completely fail for any modified version of HFE, see section 10.

2 Algebraic Paradigm for One-wayness

Let's consider any attack on any deterministic one-way function which we suppose described as a set of explicit arithmetic formulae $y_i = F_i(x_1, \ldots, x_n)$. We point out that following the first Gödel theorem, such equations can be written for any deterministic algorithm. The answer x we are looking for is also seen as a set of equations, though much simpler $x_i = \ldots$, which a hypothetical attack would evaluate to. Therefore any deterministic attack, is a series of transformations that starts from somewhat complex equations and eventually produces simpler ones. We call these "boosting transformations" as they boost the number of all equations with a know value, and produce simpler and therefore more "meaningful" equations. But what are *simple* or *complex* equations ? We must adopt a necessarily restrictive approach with a notion of complexity.

One possible notion of complexity is the non-linear degree. Every boolean function is a multivariate polynomial over $GF(2)$ (algebraic normal form). It seems to be an appropriate measure of complexity, especially to study HFE, based itself on bounded degree (quadratic) equations.

We would like to define a secure cryptographic primitive. However we don't say that they are no attacks, neither that all the attacks fail, which means little. We try to formalize **how** they fail.

The *random oracle* paradigm would be to ignore that the function formulae exist. It is used for a symmetric primitives but is meaningless for asymmetric

primitives. Indeed, they are usually described by some strikingly simple equations e.g. $x \mapsto x^e$. Thus, after all, this belief about every attack being kind of "completely puzzled by the irreducible randomness of answers to all possible questions", maybe it is not necessary at all to achieve security ?

We can even admit that some attacks exist, as long as they are hard to find and we cannot know the result before we executed the whole attack (experimental attacks without theoretical basis). For such general attacks, we suppose them to fail in most cases, even if they always do output some new equations. In fact it's very likely that we get only equations that are *trivial* combinations of those we have known and/or of higher degree that those given. Such a primitive would be considered secure.

Definition 2.0.1 (A one-way function - very informal). is a function that admits only trivial equations.

It is an attempt to give an algebraic definition of a one-way function. Still we need to precise what are "trivial" and "non-trivial" equations.

Definition 2.0.2 (Trivial equations - informal). are explicit bounded degree polynomials over the equations Y_i and variables x_i that does not exceed a given maximum $size_{max}$ (or of polynomial size) and such that their degree as a function of x_i **does not collapse**.

Definition 2.0.3 (Non-trivial equations -informal). are also bounded combinations of the Y_i and x_i, limited in size all the same, but their degree **does collapse**.

These equations, though could be generated **explicit**ly are obtained in an attack in an **implicit** way. We solve equations on their coefficients that come from the expressions of the Y_i or from a series of (cleartext,ciphertext) pairs (x, y).

3 Solving Quadratic Boolean Equations, MQ over GF(2)

In this paper we always consider n_b quadratic equations $y_i = Y_i(x_1, \ldots, x_{n_a})$ with n_a variables $x_i \in GF(q)$. If otherwise stated $n_a = n_b = n$ and $q = 2$.

The general problem of solving quadratic equations is called MQ and proved NP-complete, in [18, 7], which guarantees (only) worst-case security. However in the current state of knowledge, the MQ problem is hard even in average case, see [21] and about as hard as the exhaustive search in practice for $n < 100$ [21].

The Gaussian reduction that eliminates variables, can also be applied to MQ if $n_b > n_a(n_a - 1)/2$. Thus the so called *linearization* puts $z_i = x_i x_k$ and eliminates the new variables. We say rather that it implies the existence of at least $n_b - n_a(n_a - 1)/2$ equations of the form:

$$\sum \alpha_i y_i = \sum \beta_i x_i + \gamma$$

We call it equations of "type X+Y" later on, and the important point is that the fact that $n_b > n_a(n_a - 1)/2$ implies their existence, but the reverse is obviously false. They may exist even if for small n_b and it's always interesting to check if they do.

4 The HFE Problem

We give a simple mathematical description of the so called "HFE problem". More details on various aspects of HFE can be found in [14, 5, 4, 15, 18].

The HFE problem defined below is defined as finding one reverse image for a basic version of the HFE cryptosystem exactly as initially proposed at Eurocrypt 1996 [14]. First we recall two basic facts from [14]:

Fact 4.0.4. Let P be a polynomial over $GF(q^n)$of the special form:

$$P(a) = \sum_i \alpha_i \cdot a^{q^{s_i}+q^{t_i}}. \tag{1}$$

Then P can be written as n multivariate quadratic equations equations over the $a_i \in GF(q)$.

Fact 4.0.5 (HFE trapdoor). If P is a polynomial of degree at most d that $P^{-1}(\{b\})$ can be computed in time $d^2(\ln d)^{\mathcal{O}(1)}n^2\ GF(q)$ operations, see [14, 8].

Definition 4.0.6 (HFE Problem). Let S and T be two random secret bijective and affine multivariate variable changes. Let

$$F = T \circ P \circ S. \tag{2}$$

We believe that it's difficult to compute F^{-1} as far as it's decomposition $F^{-1} = S^{-1} \circ P^{-1} \circ T^{-1}$ remains secret.

4.1 Examples of HFE Problem

The simplest non-linear case of *basic* HFE is $P = a^{q^\alpha+q^\beta}$. It is called the Matsumoto-Imai cryptosystem (or C^*) [10] from Eurocrypt'88. A toy example of public equations can be found in [13].

It has been broken 7 years after the proposal [13]. The cryptanalysis ([13, 9, 14]) shows that there exist at least $2/3n$ of what we describe later as equations of "type XY", and what are simply implicit bi-affine equations involving input and output variables x_i and y_i:

$$\sum \alpha_{ij}x_iy_j + \sum \beta_i x_i + \sum \gamma_j y_j + \delta = 0$$

The Attack is as follows: first we recover these equations by Gaussian elimination on their coefficients. Then we recover x substituting y in these equations.

4.2 HFE Challenge 1

It has been proposed by Jacques Patarin in the extended version of [14].

The HFE polynomial is of degree $d = 80$ over $GF(2^n)$ with $n = 80$ bits. The price of 500$ is promised for breaking the signature scheme that amounts to computing F^{-1} three times. An example of F can be downloaded from [5].

5 Implicit Equations Attack

5.1 Types of equations

We have a convention to describe an equation type:

1. The equation type is a union of terms in formal variables x, y, X, Y, for example: $XY \cup x^2$.
2. A term $x^k y^l$ denotes all the terms of degree **exactly k** in all the $x_i, i = 1..n_a$ and of degree **exactly l** in $y_i, i = 1..n_b$.
   Important: If the variables are in $GF(q)$, the degrees must be in $[0..q-1]$.
3. The capital X, Y describe equation sets that include all the lower degree terms. For example: $XY \cup x^2 \equiv 1 \cup x \cup y \cup xy \cup x^2$.
4. If necessary we distinguish by $\{XY \cup x^2\}$ the set of terms used in the corresponding equation type, while $[XY \cup x^2]$ denotes the set of equations of this type.

5.2 Invariant Equations

Definition 5.2.1 (Invariant equations). Set of equations with their set of terms invariant modulo any bijective affine S and T variable changes.

For example $[X^2Y]$ is invariant but not $[x^2y]$. The definition states that the sets of terms involved are invariant, that implies that the number of equations that exist for a given type is invariant (but each of the equations is invariant).

If the equations are invariant, the number of equations of a given type will be the same for any output value. Thus we can assume that we are solving $F^{-1}(y)$ with $y = 0$ without loss of generality. We make this assumption for all subsequent attacks. The problem of the invariant equations of higher degree is that they are still at least quadratic after substituting y.

5.3 "Biased" Equations

Definition 5.3.1 (Biased). equations are the equations that after substitution of $y = 0$ reduce to a affine equation of the x_i (type X).

Proposition 5.3.2. If there is "enough" invariant equations, there exist "enough" biased equations.

Enough means the equal to the number of terms remaining after substitution of $y = 0$. The proposition is trivial, we eliminate in a set of implicit equations all the terms of $\{X^{\infty} - X\}$ before the substitution of $y = 0$. The important point that biased equations may exist even if it is not guaranteed by the above proposition. Our experiences in 5.6 has indeed shown they do.

Another important property of the biased equations is that they allow a single round attack. The result of substitution of $y = 0$ are linear in the x_i. The drawback is that they are made for a single y value. The whole attack must be re-iterated to compute several $F^{-1}(y)$ for different y.

Important: The "biased" equations does not need to be computed completely in an attack. Only the coefficients of the terms in x_i as well as constant parts are needed (!)

5.4 The *size* of the Equations

We call *size* the number of terms of type $x_i x_j y_k$ etc.. that are used in the type of equations considered. The implicit equations attack requires huge quantities of memory, because the length *size* of the equations is polynomial in n of degree at least $3 - 4$, and the attack memory requirements are quadratic still in *size*.

We express *size* as a function of the number of input and output variables, respectively n_a and n_b. In [4] one can find a complete reference table that allows to compute size values. For example, for fields of characteristic 2:

$$size_{XY \cup x^2 y \cup x y^2 \cup x^3 y \cup x^2 y^2} = \frac{7}{12} n_a n_b + \frac{1}{4}(n_a n_b^2 - n_a^2 n_b + n_a^2 n_b^2) + \frac{1}{6} n_a^3 n_b + n_a + n_b + 1.$$

5.5 Trivial Equations

Since the y_i are quadratic, therefore we have n_b equations of the type $[1 \cup x \cup x^2 \cup y]$. All the equations that are the consequence of these equations are called trivial. In practice, when n_a is bigger than some initial threshold, the number of trivial equations is always the number that we get when we pick all quadratic equations at random. Example:

In $[XY \cup x^2]$ there are n trivial equations, the same as in $[1 \cup x \cup x^2 \cup y]$.

Trivial equations, though they mix with "non-trivial' equations" used in cryptanalysis, are predictable and harmless. When the y_i values substituted to the linear mix of the non-trivial and trivial equations, we eliminate the interference as trivial equations always reduce to 0.

The exact number $trivial_{\text{type}}$ of trivial equations is not obvious to compute. Those that come from the interaction of different components of the 'type' expression, may overlap and thus type $\mapsto trivial_{\text{type}}$ is not an additive function. In [4] we compute $trivial_{\text{type}}$ for all the equation types we consider.

5.6 Results

In the following table on page 8, we show the number of equations of different types found for *basic* HFE. We did much more such computations in [4].

Table 1. Non-trivial equations found for *basic* HFE

n=21	Equation type					
d	XY	$XY \cup x^2y$	$XY \cup x^2y \cup xy^2$	X^2Y	$X^2Y \cup XY^2 \cup X^3$	$XY \cup x^2y \cup xy^2 \cup x^3y \cup x^2y^2$
3	$42 \to 19$	$693 \to 19$	$1995 \to 19$	$882 \to 210$	$2688 \to 484$	...
4	$21 \to 21$	$441 \to 21$	$1995 \to 21$	$630 \to 210$	$2688 \to 484$	...
5	$1 \to 1$	$232 \to 18$	$1177 \to 18$	$357 \to 144$	$1806 \to 484$	...
8	$1 \to 1$	$170 \to 20$	$1094 \to 20$	$336 \to 184$	$1764 \to 484$	...
9	$0 \to 0$	$126 \to 18$	$672 \to 18$	$231 \to 124$	$1134 \to 337$	...
16	$0 \to 0$	$43 \to 20$	$568 \to 20$	$168 \to 144$	$1092 \to 379$	...
17	$0 \to 0$	$0 \to 0$	$63 \to 16$	$84 \to 84$	$357 \to 169$	...
24	$0 \to 0$	$0 \to 0$	$22 \to 18$	$84 \to 84$	$315 \to 311$	...
32	$0 \to 0$	$0 \to 0$	$0 \to 0$	$64 \to 64$	$315 \to 315$	...
33	$0 \to 0$	$0 \to 0$	$0 \to 0$	$0 \to 0$	$147 \to 147$	...
64	$0 \to 0$	$0 \to 0$	$0 \to 0$	$0 \to 0$	$147 \to 147$	$4739 \to 20$
65	$0 \to 0$	$0 \to 0$	$0 \to 0$	$0 \to 0$	$42 \to 42$	$1911 \to 17$
96	$0 \to 0$	$0 \to 0$	$0 \to 0$	$0 \to 0$	$42 \to 42$	$1638 \to 21$
128	$0 \to 0$	$0 \to 0$	$0 \to 0$	$0 \to 0$	$42 \to 42$	$1547 \to 20$
129	$0 \to 0$	$0 \to 0$	$0 \to 0$	$0 \to 0$	$0 \to 0$	$0 \to 0$

Legend:

We write the equation number found as **A** $\to$ **B** with:

A is the number of non-trivial equations found, which means we have subtracted the number of trivial equations. This convention allows, at least as long as n is not too small, to have 0 at places where HFE behaves exactly as a random multivariate quadratic function (MQ).

B Is the number of the above equations that remain linearly independent after substitution of a randomly chosen y value. We apply an analogous convention for the origin, trivial equations are subtracted.

The memory needed to do these computations was up to 1.2 Gbyte and for this reason we had to skip some irrelevant cases.

Interpretation in terms of security:

If we get somewhere more that 0 equations, it is a weakness, but not necessarily a working attack.
The only HFE that can pretend to be secure, should give **0** non-trivial equations for all the types we can compute within realistic memory limits.

5.7 Interpretation of the Results

In the computations on page 7 more and more complex equations exist when d increases. In [4] we consider many more different equation types and other $q \neq 2$.. The subtypes of types $[X^l Y]$ prove the best because at constant $size$, their degree in x is smaller.

We observed that the degrees $d = q^k + 1..q^{k+1}$ behave almost the same way and that the number of non-trivial equations found behaves as $\mathcal{O}(n^{\alpha(\lceil \log_q d \rceil, type)})$ with a constant $\alpha(\lceil \log_q d \rceil, type)$. We postulate that:

Conjecture 5.7.1. A *basic* HFE (or the HFE problem) of degree d admits $\mathcal{O}(n)$ equations of type $[X \cup x^2 y \cup \ldots \cup x^{\frac{1}{2}\lceil \log_q d \rceil - 1} y]$.

In a later attack we will "cast' these equations over a smaller subspace, but we will see in the section 6 that we can only recover them starting from a threshold $n_a = n_{art}(n, type)$, a threshold memory (usually in Terabytes) and a threshold computing power. It means that today's computers are not powerful enough to find what happens for the equations more complex that the one we have already studied (!)

5.8 The Complexity of the Attacks

The memory used in the attack is quadratic in $size$ and is equal to $size^2/8$ bytes.

In terms of speed, the essential element of all the attacks is the Gaussian elimination. Though better algorithms exist in theory, [3], they are not practical. We have implemented a trivial algorithm in $\mathcal{O}(size^3)$. A structured version of it can go as fast as CPU clock while working on a huge matrix on the disk (!). Assuming that a 64-bit XOR in done in one clock cycle, we estimate that the structured elimination takes $2 \cdot size^3/64$ CPU clocks.

5.9 Realistic HFE Attacks when $d \leq 24$

We see in 5.6 that for $d <= 24$ equations of type $XY \cup x^2 y \cup xy^2$ give between $\mathcal{O}(n)$ and $\mathcal{O}(n^2)$ equations, enough to break *basic* HFE. For example we consider an attack for $n = 64$ bits HFE with the degree $d \leq 24$:

$$size_{XY \cup x^2 y \cup xy^2}(64, 64) = \mathcal{O}(n^3) \tag{3}$$

The precise computation yields $size = 262\,273$ and thus the memory required in the attack is $size^2/8 = \mathcal{O}(n^6) = 8$ Gb. The running time is $2 \cdot size^3/64 \approx 2^{48}$ CPU clocks, few days on a PC, and it is not our best attack yet.

Thus *basic* HFE is not secure for $d \leq 24$. The asymptotic complexity is at most $\mathcal{O}(n^9)$.

5.10 Direct Attack on Challenge 1

Now we try to use the equations of type $XY \cup x^2y \cup xy^2 \cup x^3y \cup x^2y^2$ to break this degree 96 *basic* HFE. We have

$$size_{XY \cup x^2y \cup xy^2 \cup x^3y \cup x^2y^2}(80, 80) = 17\ 070\ 561 \tag{4}$$

The memory required is not realistic: $size^2/8 = 33$ Terabytes. The running time is $2 \cdot size^3/64 \approx 2^{62}$ CPU clocks.

6 Advanced Attacks

6.1 Reconciliation Technique

Since the main problem of the attacks is the *size* of the equations, it is a very good idea to compute these equations only partly. We fix to zero all x_i except n_a of them. We call "cast" equations the equations we get from the initial equations.

Unfortunately if n_a is too small, there are some more equations that we call "artificial" equations. We show that the "cast" equations of trivial equations are trivial and the "cast" equations of artificial equations are artificial. In [4] we have managed to predict the number of artificial equations with a great accuracy.

For example, if $n = n_b = 80$ we computed:

$$n_{art}(XY \cup x^2y \cup xy^2 \cup x^3y \cup x^2y^2) = 38 \tag{5}$$

It means that the "cast" (and "non-trivial") equations are known modulo a linear combination of some "interference" equations (artificial equations), that make the resulting mix unusable for $n_a < 38$.

The **reconciliation attack** works before the threshold when artificial equation arise. The necessary condition is thus $n_a \geq n_{art}$.

Moreover the equations are recovered modulo a linear combination, and we need to, make sure that it is possible to generate "cast" equations, such that their intersections are big enough to recover uniquely their corresponding linear combinations. This leads to an additional condition.

Thus we will recover the equations from different "casts". In fact we do not exactly recover the whole equations but only a part of them that contains firstly enough terms to combine different casts, and secondly their constant coefficients and coefficients in x_i, as **only** those are necessary to compute x and break HFE.
.

6.2 The Distillation Technique

In the **distillation attack** we show that there is another, strictly lower threshold, and HFE can be broken in spite of the "interference" equations. The idea is very simple, the artificial equations alone doesn't have any sense with relation to initial (huge) equations and can be eliminated from different "casts'.

In [4] we show that if the following distillation condition is true:

$$artificial(n_a - 1, n_b) \geq artificial(n_a, n_b). \tag{6}$$

then a successful attack can be lead.

6.3 Distillation Attack on Challenge 1

For $n_b = 80$ and type $XY \cup x^2y \cup xy^2 \cup x^3y \cup x^2y^2$, the solution for the distillation condition above is computed in [4] to be $n_a \geq 30$.

The working *size* of the attack is:

$$size_{XY \cup x^2y \cup xy^2 \cup x^3y \cup x^2y^2}(30, 80) = 1\ 831\ 511. \tag{7}$$

We need only $size^2/8 = 390$ Gb of memory instead of 33 Tb in the direct attack of section 5.10. Following [4], the running time is computed as $(80 - 30 + 1) \cdot 2 \cdot size^3/64 \approx 2^{62}$ CPU clocks.

6.4 Sparse methods

In the attacks above, we have to solve systems of several million equations with several million variables. Such equations could be sparse, if we try to recover them in a slightly different way. We build a matrix with columns corresponding to each component of the equation, for example y_1y_4 or $x_2y_{55}y_9$. Each line of the equation will correspond to a term, for example $x_3x_5x_7x_{16}$. We only need to consider about as many terms as *size*, (there is much much more) though sparse methods [Lanczos, Wiedemann] could take advantage if we generated more.

Such a system of equations is sparse, for example the column $x_2y_{55}y_9$ contains non-zero coefficients only for terms containing x_2, therefore for about $1/n$ of all terms.

In [12] we hear that with $size = 1.3M$ (million), a system over $GF(2)$ could be solved in few hours on one processor of CrayC90 using modified Lanczos algorithm. Their system had only $39M$ non-zero coefficients, i.e. about $1/40000$ of them. Assuming that sparse methods would combine with reconciliation and distillation, for our systems of $size = 1.8M$ we have about $1/80$ non-zero coefficients, much more.

Thus it is unclear if any of the aforementioned sparse methods could improve on the attack.

7 Asymptotic Security of *basic* HFE

First, if d is fixed, we have found in 5.6 an experimental evidence that *basic* HFE can be broken in expected polynomial time. The same result has just been independently shown by Shamir and Kipnis at Crypto'99, see [23].

Our attack in a basic version based on conclusions form 5.7 (no reconciliation, no distillation) gives about:

$$size \approx n^{\frac{1}{2}\log_q d}. \tag{8}$$

In [4] we show that the distillation attack gives roughly:

$$size \approx n\left(\sqrt{n}\right)^{\frac{1}{2}\log_q d} \approx n^{\frac{1}{4}\log_q d}. \tag{9}$$

We retain a conservative approximation:

$$size \leq n^{\frac{1}{2}\log_q d}. \tag{10}$$

7.1 Results

Therefore the security of *basic* HFE is not better than:

$$security \leq n^{\frac{3}{2}\log_q d}. \tag{11}$$

If the distillation attack works as well as estimated in [4], it would give even:

$$security \leq n^{\frac{3}{4}\log_q d}. \tag{12}$$

First, we compare it to the secret key operations of HFE. It requires to factorise the degree d polynomial P over a finite field. The asymptotically fastest known algorithm to solve a polynomial equation P over a finite field of von zur Gathen and Shoup [8] requires about $d^2(\log_q d)^{\mathcal{O}(1)}n^2$ operations. At any rate we need $d = n^{\mathcal{O}(1)}$ to enable secret key computations [14]. Thus:

$$security \leq n^{\mathcal{O}(\log_q n)} \approx e^{(\log_q^2 n)}. \tag{13}$$

In [4] it has been shown that the complexity of Shamir-Kipnis attack is rather in $n^{\mathcal{O}(\log_q^2 d)}$ which gives $e^{\mathcal{O}(\log_q^3 n)}$. We are going to improve it to get a similar result.

8 Shamir-Kipnis Attack Revisited

The starting point here is the Shamir-Kipnis attack for *basic* HFE, [23] that we do not describe due to lack of space. It shows there exist $t_0, \ldots, t_{n-1} \in GF(q^n)$ such that the rank of

$$G' = \sum_{i=0}^{n-1} t_k G^{*k} \tag{14}$$

collapses to at most $r = 1 + \lceil \log_q d \rceil$, with G^{*k} being n public matrices $n \times n$ over $GF(q^n)$.

The underlying problem we are solving is called MinRank [6]. Shamir and Kipnis solved it by what is called 'relinearization', see [21] for improvements on it. We do not use it, and instead we solve MinRank directly. Our method is identical as previously used by Coppersmith, Stern and Vaudenay in [1, 2].

We write equations in the $t_0, \ldots, t_{n-1}$ saying that every $(r+1)$x$(r+1)$ submatrix has determinant 0. Each submatrix gives a degree $(r+1)$ equation on the $t_0, \ldots, t_{n-1}$ over $GF(q^n)$. There are as much as $\binom{n}{r+1}^2$ such equations and we hope that at least about $\binom{n}{r+1}$ of them are linearly independent. We get about $\binom{n}{r+1}$ equations which have $\binom{n}{r+1}$ terms, and are simply linearized and solved by Gaussian reduction.

The *size* of the equations to solve is

$$size \approx \binom{n}{r+1} \approx n^{r+\mathcal{O}(1)} \approx n^{\log_q d+\mathcal{O}(1)}, \tag{15}$$

which gives similar results as our attacks:

$$security \leq n^{\mathcal{O}(\log_q d)}. \tag{16}$$

9 Is *basic* HFE likely to be polynomial ?

The MinRank is an NP-complete problem for e.g. $r = n-1$ [24, 6]. It seems therefore unlikely that our attack for MinRank in $n^{\mathcal{O}(r)}$ could ever be improved to remain polynomially bounded when r grows.

The same remark applies to our equational attacks. When d grows, the HFE problem (i.e. *basic* HFE) tends to the NP-complete MQ problem of solving random quadratic equations, see [14, 15, 4].

10 Conclusion

The best known HFE attack is our distillation attack for *basic* HFE. It's not proven to work for $d >> 129$ but relies on an extensive experimental evidence. we have also the Shamir-Kipnis attack, and rather our improved version of it, that though worse in practice comes with a proof [23].

They both give the complexities in $n^{\mathcal{O}(\log_q d)}$ to break the *basic* HFE version. It is polynomial when d is fixed and subexponential in general. Both presented attacks on HFE are much better that any previously known.

Even with the significant progress we have made, the attacks still have the complexity and memory requirements that can quickly go out-of-range. Though it is certain that attacks will be improved in the future, HFE can be considered secure for $d > 128$ and $n > 80$.

Perspectives

The *basic* version of HFE is broken for the initially proposed degree $d \geq 17$ [14] and even for $d \geq 24$. Our attacks has been tested to work for $d \leq 128$, and thus the HFE Challenge 1 is broken in 2^{62}.

HFE modifications that resist to all known attacks.

Several HFE problem-based cryptosystems avoid all the attacks described in the present paper. We verified that our attacks rapidly collapse for these schemes:

HFE$^-$: It is a basic HFE with several public equations removed, see [16].

HFEv: Described in a paper presented at Eurocrypt'99, [17]. It consists of adding new variables to HFE, as in the Oil and Vinegar algorithm partially broken at Crypto'98 [22].

HFEv-: Combines both above ideas. There are many other variants of HFE proposed by Jacques Patarin in the extended version of [14] and in [15, 18].

Quartz: Presented at RSA'2000 [19] and submitted to the european Nessie call for primitives. An unique 128-bit long signature scheme, based on HFEv-, designed for long-term security. Our best attack applied to the *basic* HFE subsystem of Quartz, with $d = 129$ and $n = 103$, gives about 2^{114}. However it does not apply at all to the whole Quartz.

Flash, Sflash Also at RSA'2000 [20] and submitted to Nessie. A signature scheme based on C^{*-}, designed for speed. The security is an open problem.

References

1. Don Coppersmith, Jacques Stern, Serge Vaudenay: *Attacks on the birational permutation signature schemes*; Crypto 93, Springer-Verlag, pp. 435-443.
2. Don Coppersmith, Jacques Stern, Serge Vaudenay, *The Security of the Birational Permutation Signature Schemes*, in Journal of Cryptology, 10(3), pp. 207-221, 1997.
3. Don Coppersmith, Samuel Winograd: "Matrix multiplication via arithmetic progressions"; J. Symbolic Computation (1990), 9, pp. 251-280.
4. Nicolas Courtois: *La sécurité des primitives cryptographiques basées sur les problèmes algébriques multivariables MQ, IP, MinRank, et HFE*, PhD thesis, Paris 6 University, to appear in 2001, partly in English.
5. Nicolas Courtois: The HFE cryptosystem home page. Describes all aspects of HFE and allows to download an example of HFE challenge. http://hfe.minrank.org
6. Nicolas Courtois: *The Minrank problem.* MinRank, a new Zero-knowledge scheme based on the NP-complete problem. Presented at the rump session of Crypto 2000, available at http://www.minrank.org
7. Michael Garey, David Johnson: *Computers and Intractability, a guide to the theory of NP-completeness*, Freeman, p. 251.
8. J. von zur Gathen, Victor Shoup, "Computing Fröbenius maps and factoring polynomials", Proceedings of the 24th Annual ACM Symposium in Theory of Computation, ACM Press, 1992.
9. Neal Koblitz: "Algebraic aspects of cryptography"; Springer-Verlag, ACM3, 1998, Chapter 4: "Hidden Monomial Cryptosystems", pp. 80-102.
10. Tsutomu Matsumoto, Hideki Imai: "Public Quadratic Polynomial-tuples for efficient signature-verification and message-encryption", Eurocrypt'88, Springer-Verlag 1998, pp. 419-453.

11. Tsutomu Matsumoto, Hideki Imai:"A class of asymmetric cryptosystems based on polynomials over finite rings"; 1983 IEEE International Symposium on Information Theory, Abstract of Papers, pp.131-132, September 1983.
12. Peter L. Montgomery: A Block Lanczos Algorithm for Finding Dependencies over GF(2); Eurocrypt'95, LNCS, Springer-Verlag.
13. Jacques Patarin: "Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88"; Crypto'95, Springer-Verlag, pp. 248-261.
14. Jacques Patarin: "Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): two new families of Asymmetric Algorithms"; Eurocrypt'96, Springer Verlag, pp. 33-48. The extended version can be found at http://www.minrank.org/∼courtois/hfe.ps
15. Jacques Patarin: *La Cryptographie Multivariable*; Mémoire d'habilitation à diriger des recherches de l'Université Paris 7, 1999.
16. Jacques Patarin, Nicolas Courtois , Louis Goubin: "C*-+ and HM - Variations around two schemes of T. Matsumoto and H. Imai"; Asiacrypt 1998, Springer-Verlag, pp. 35-49.
17. Jacques Patarin, Aviad Kipnis , Louis Goubin: "Unbalanced Oil and Vinegar Signature Schemes"; Eurocrypt 1999, Springer-Verlag.
18. Jacques Patarin, Louis Goubin, Nicolas Courtois, + papers of Eli Biham, Aviad Kipnis, T. T. Moh, et al.: *Asymmetric Cryptography with Multivariate Polynomials over a Small Finite Field*; known as 'orange script', compilation of different papers with added materials. Available from J.Patarin@frlv.bull.fr.
19. Jacques Patarin, Louis Goubin, Nicolas Courtois: Quartz, **1**28-*bit long digital signatures*; Cryptographers' Track Rsa Conference 2001, San Francisco 8-12 April 2001, LNCS2020, Springer-Verlag.
20. Jacques Patarin, Louis Goubin, Nicolas Courtois: *Flash, a fast multivariate signature algorithm*; Cryptographers' Track Rsa Conference 2001, San Francisco 8-12 April 2001, LNCS2020, Springer-Verlag.
21. Nicolas Courtois, Adi Shamir, Jacques Patarin, Alexander Klimov, *Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations*, In Advances in Cryptology, Eurocrypt'2000, LNCS 1807, Springer-Verlag, pp. 392-407.
22. Adi Shamir, Aviad Kipnis: "Cryptanalysis of the Oil and Vinegar Signature Scheme"; Crypto'98, Springer-Verlag.
23. Adi Shamir, Aviad Kipnis: "Cryptanalysis of the HFE Public Key Cryptosystem"; Crypto'99. Can be found at http://www.minrank.org/∼courtois/hfesubreg.ps
24. J.O. Shallit, G.S. Frandsen, J.F. Buss, *The computational complexity of some problems of linear algebra*, BRICS series report, Aarhus, Denmark, RS-96-33. Available at http://www.brics.dk/RS/96/33

11 Common Terms and Notations

about equations: We consider multivariate equations over $GF(q)$, usually with $q = 2$. n_a/n_b are the numbers of input/output variables x_i/y_i. We note $size_{\text{type}}(n_a, n_b)$ the length of equations of a given "type". The "type" is specified by a convention using expressions in variables x, y, X, Y detailed in the section 5.1.

artificial* equations are due to the small dimension n_a of the x sub-space and the small degree of y_i expressions. They become visible if they are more that trivial+non-trivial equations. Their number $artificial_{\text{type}}(n_a, n_b)$ can be correctly computed and does not depend on the HFE degree.

biased equations - for one particular value $y = 0$ they become affine in x_i.

boosting - general notion of an operation that starting with some equations on the unknowns, finds some other equations on them that are not trivial (e.g. linear) combinations of the initial equations.

cast* equations are non-trivial equations with some x_i fixed to 0, usually for $i = n_a + 1, \ldots, n$.

distillation - eliminating artificial "interference" equations between different casts of the same equation, see 6.1-6.3.

HFE stands for the Hidden Field Equations cryptosystem [14]. P denotes the hidden univariate HFE polynomial over $GF(q^n)$. S and T are affine multivariate bijective variable changes over $GF(q)$ and $F = T \circ P \circ S$.

interference* equations - any complementary space of cast equations in artificial equations.

invariant equations - equations that are still of the same type after an affine variable change because their set of terms is invariant.

non-trivial* equations - any complementary space of trivial equations found implicitly by Gaussian reduction. The implicit equations we are able to recover must be of small degree in both the y_i and x_i. An implicit equation in the x_i and y_i may be viewed as a point such that, an expression in the y_i and x_i, re-written as as a polynomial in x_i, has unusually small degree.
For cryptanalysis we look for equations that have small degree in the x_i after substitution of one value y (or all possible y). The equations mixed with trivial equations are still useful for cryptanalysis. Their existence is a definite weakness of any one-way function candidate.

reconciliation - recomposing different "casts" of the same equations, see 6.1.

trivial equations - explicit small degree combinations of given equations and the variables that are due to the quadratic character of y_i. Their number is $trivial_{\text{type}}(n_a, n_b)$.

$*$ - informal categories, doesn't make sense for equations regardless how they have been computed.