Cryptanalysis of the HFE cryptosystem

George Kadianakis
supervisor Marilena N. Poulou

2012-05-20

Contents

"Cryptography is the discipline concerned with communication security (eg, confidentiality of messages, integrity of messages, sender authentication, non-repudiation of messages, and many other related issues), regardless of the used medium such as pencil and paper or computers.", according to Wiktionary.

1 Introduction

1.1 History

Cryptography, in the form of ensuring message confidentiality, was used since the ancient times, mainly to ensure secrecy in political or military communications. In modern days, cryptography, nowadays a lot more than message confidentiality, is a rapidly evolving field with deep connections to many areas of mathematics, computer science and other scientific fields.

Historically speaking, in Ancient Greece, the Spartan military used a scytale to perform *transposition encryption*, a primitive form of encryption were the ciphertext constitutes a permutation of the plaintext.
Centuries later, and possibly the most well known example of classical cryptography, the Caesar cipher was used by Julius Caesar to communicate with his generals. It was an example of *substitution encryption* in which each letter of the plaintext is replaced by a letter some fixed number of positions down the alphabet.
In later years during the first World War, many mechanical encryption machines were built, including the famous Enigma machine.

With the invention of electronics, the development of cryptography skyrocketed. During the 1970s, cryptography based on deep mathematical concepts was introduced. Examples are the RSA algorithm, the Diffie-Hellman key agreement protocol and the Data Encryption Standard (DES). Since then, the field of cryptography has been blossoming, enjoying deep academical attention and much practical use through computers.

Most modern cryptographic schemes are based on certain mathematical problems that are considered intractable. This allows a person who knows a certain secret to easily reverse such a problem, and read the encrypted message, while forcing adversaries who do not know the secret to manually solve the intractable problem.
Two such examples of mathematical problems are the following:

integer factorization The problem of breaking down a composite number into smaller non-trivial divisors, which when multiplied together equal the original integer. For very large numbers, this process is terribly slow and no efficient algorithms are known. Many cryptographic schemes, like RSA, make use of this problem.

discrete logarithm The problem of finding a solution x that satisfies the relation $g^x = h$ where g and h are elements of a finite cyclic group. Cryptographic schemes like ElGamal, Diffie-Hellman and DSA are based on the discrete logarithm problem.

It must be noted that there are no real proofs that the above two problems are actually hard; our only assurance is that no efficient algorithms solving them have been publicly announced.

Unfortunately, real life assymetric cryptography has been essentially restricted to cryptosystems using the above two problems.. This is a sad fact since an efficient factoring or discrete log solving algorithm would effectively annihilate most assymetric cryptography schemes.

Fortunately, during the past 20 years the cryptographic community - especially the academia - has developed many public key encryption and signature schemes based on other intractable problems. Examples are, schemes based on cryptographic hashes, lattices, linear codes, multivariate quadratic equations and linear codes. Unfortunately, even though some of these schemes provide excellent security, their use in real life applications has just started to surface.

The need for such schemes becomes even more urgent because of the development of quantum cryptography which will destroy any cryptographic scheme relying on the intractability of factorization or the discrete log problem. Fortunately, even though the quantum cryptography field is rapidly evolving, there is still a long way to go for the first functional quantum computer that will be able to break the above cryptosystems. [2]

1.2 Introduction to MQ

In the late 1980s, a number of new cryptosystems were introduced, exploiting the fact that solving systems of modular multivariate quadratic equations (**MQ**) over finite fields is a hard problem.

An early example of an MQ cryptosystem was a signature scheme developed by Ong Schnorr and Shamir, presented in their paper *"A fast signature scheme based on quadratic equations."* [9]. Another early example of MQ cryptosystems was the *"Matsumoto-Imai-Scheme"* by T. Matsumoto and H. Imai presented in the Eurocrypt conference [13]. A few years later, Jacques Patarin introduced two MQ algorithms, named *"Oil and Vinegar"* [11] and *""Hidden Monomial Cryptosystems"* [14]. In the beginning of the 21st century, Masao Kasahara and Ryuichi Sakai introduced yet another MQ cryptosystem on their paper *"A construction of 100 bit Public-Key Cryptosystem and Digital Signature Scheme"* [10].

Many of the above schemes have been broken, but some of them are still standing. [17][6][16][3][4][7][12].

This thesis studies the Hidden Field Equations (HFE) cryptosystem, introduced in the mid-90s by J. Patarin in his paper *"Hidden Field Equations (HFE) and Isomorphisms of Polynomials (IP): two new Families of Assymetric Algorithms"* [15]. According to J. Patarin, HFE is one of his strongest cryptographic algorithms based on MQ.

The HFE scheme has received many cryptanalytic blows. Examples of attack papers are: "Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization" by Kipnis and Shamir [12], "Algebraic cryptanalysis of HFE using Grbner bases" by Jean-Charles Faugre [4] and "Cryptanalysis of HFE with Internal Perturbation" by Vivien Dubois, Louis Granboulan, and Jacques Stern [7].

1.3 The aim

The aim of this thesis is to introduce the reader to the HFE cryptosystem, and afterwards present in an approachable way the attack detailed by Shamir and Kipnis in their paper "Cryptanalysis of the HFE Public Key Cryptosystem by Relinearization".

In section 2, we present the MQ problem itself and then we lay down a generic MQ cryptosystem and its operations.

In section 3, we focus on the HFE cryptosystem: We present its functions and its unique properties. We explain how and why it works and we set up the notation we will be using in Section 4.

In section 4, we start the cryptanalysis of HFE. We first give an overview of the attack and then we prove some essential lemmas that will be of use later on. Afterwards, we introduce two techniques of solving overdefined systems of equations, the *linearization* and *relinearization*. Finally, we explain how to apply these techniques to recover the private keys of the HFE cryptosystem.

In Appendix A, we try to provide all the mathematical definitions that one should need while reading this thesis. We do not prove any theorems, since the proofs can easily be found in many books of algebra, but we try to provide all the necessary definitions along with examples.

In Appendix B, we present examples that did not appear in the main text. Specifically, we show:

- An example of the public key generation of HFE.
- An example of lemma 4.2.2.

- An example of a technique introduced in Shamir and Kipnis' paper to convert a relation between the ranks of two matrices to a system of equations.
- An example of the relinearization technique.

2 MQ problem and MQ cryptosystems

This section presents the MQ problem and a generic MQ cryptosystem.

2.1 The MQ problem

All MQ cryptosystems are based on the problem of solving multivariate quadratic equations over a finite set of numbers. Specifically, it can be proved that the MQ problem over the finite field F_2 is hard, or in complexity theory's terms: NP-complete. [8].

MQ cryptosystems exploit the fact that the MQ problem is hard, by introducing a cryptographic trapdoor based on it. Essentially only someone who knows the trapdoor, can easily reverse the effects of MQ. An attacker who doesn't know the trapdoor has to solve the MQ-problem with brute force and ignorance, which is impractically slow.

The trapdoor of MQ cryptosystems is a set of polynomials P' and two invertible affine transformations S and T so that: $P(x) = S(P'(T(x)))$, where P is a multivariate quadratic equations system. Someone who knows the trapdoor - the (P', S, T) tuple - can easily reverse MQ.

2.2 The MQ cryptosystem components

This section introduces the basic components of a generic MQ cryptosystem.

The private key of an MQ cryptosystem is a tuple (P', S, T) containing:

- P': A system of quadratic multivariate polynomials over F_{q^n}.
- S, T: Two invertible affine transformations over F_{q^n}.

The public key of an MQ cryptosystem is (P):

- P: A system of quadratic multivariate polynomials over F_q.

During the key creation of an MQ cryptosystem, P' is generated appropriately so that $P(x) = S(P'(T(x)))$ holds.
The affine transformations S, T are used to hide the polynomial structure of P' in the above relation.

2.3 MQ cryptosystem properties

Figure 1: MQ

Figure 1 presents a generic MQ cryptosystem. It shows how the private key (P', S, T) works as a trapdoor, effectively allowing someone with knowledge of the private key to reverse the MQ-problem where otherwise it would have been an intractable problem.

Some important MQ properties are:
The degree of P':
Since we are dealing with multivariate **quadratic** cryptosystems, the polynomials in P can not be of degree higher than 2 over F_q.
Looking at the $P(x) = S(P'(T(x)))$ relation, this means, that S and T are equations of degree 1 and P' is also a system of quadratic polynomials over F_q.
The invertibility of S, T and $\mathbf{P'}$:
S^{-1}, P'^{-1} and T^{-1} are used during the decryption and signature generation operations and it's important that S, T and P' are swiftly invertible so that these operations are practical in real life applications:
The affine transformations **S** and **T** are invertible if the matrix of their linear transformation is also invertible.
The system of polynomials $\mathbf{P'}$ and its inversion depends on the specific nature of the polynomials, which differs for every MQ cryptosystem. The invertibility of P' in HFE will be studied in section 3.1.

2.4 MQ operations

Every public key cryptosystem has 4 standard operations: *Encryption*, *Decryption*, *Signature generation* and *Signature verification*. This section briefly describes the purpose of each operation and analyzes how it is performed in an MQ cryptosystem.

2.4.1 Encryption

Encryption is the cryptographic operation where a message x is given as input, and the operation's output y is an encrypted copy of that message.

In an MQ cryptosystem, to encrypt an n-bit message x, we assign each bit of the message x to a polynomial variable x_i and evaluate the n quadratic polynomials of the public key P. The encrypted output y, is the concatenation of the results of the n polynomial evaluations:

$$
\begin{aligned}
y_1 &= p_1(x_1, ..., x_n) \\
&\vdots \\
y_n &= p_n(x_1, ..., x_n)
\end{aligned}
$$

2.4.2 Decryption

Decryption is the cryptographic operation where given an encrypted message y as input, the operation's output x is the original message.

In an MQ cryptosystem, to decrypt an n-bit message y, we assign each bit of y to a variable y_i, and we try to find the polynomial variables x_i that satisfy the equations:

$$
\begin{aligned}
y_1 &= S^{-1}(p'_1(T^{-1}(x_1, ..., x_n))) \\
&\vdots \\
y_n &= S^{-1}(p'_n(T^{-1}(x_1, ..., x_n)))
\end{aligned}
$$

where p'_i with $i = 1, \ldots, n$, are the n quadratic multivariate polynomials of the inverse of the private key P^{-1}.

The decrypted message x, is the concatenation of the values of the x_i polynomial variables.

2.4.3 Signature generation

Signature generation is the cryptographic operation where given a message x, we generate a *signature* s.

A signature can later be used to verify that a message x with signature s was signed by the holder of a specific private key. For this reason, the signature generation procedure uses the private key of an MQ cryptosystem and it is similar to the decryption operation.

Specifically, to generate a signature s for an n-bit message x we assign each bit of x to a polynomial variable x_i, and we solve the equations:

$$
\begin{aligned}
s_1 &= S^{-1}(p'_1(T^{-1}(x_1, ..., x_n))) \\
&\vdots \\
s_n &= S^{-1}(p'_n(T^{-1}(x_1, ..., x_n)))
\end{aligned}
$$

The signature s is the string composed by the concatenation of the values of the polynomial variables s_i.

2.4.4 Signature verification

Signature verification is the cryptographic operation where given a message x and a signature s, we output *True* if the signature s was uniquely created from x by the bearer of a specific private key. It is used to validate that a specific entity that holds a private key had access to the information in the message x. The signature verification procedure uses the public key of an MQ cryptosystem and for this reason it's quite similar to the encryption operation.

To verify that a signature s was uniquely created from a message x by the bearer of a private key (P', S, T), we assign each bit of s to a polynomial variable s_i and evaluate the n quadratic polynomials of the public key P:

$$\begin{aligned} y_1 &= p_1(s_1, ..., s_n) \\ &\vdots \\ y_n &= p_n(s_1, ..., s_n) \end{aligned}$$

If y, derived from concatenating the results y_i of the polynomial evaluations, is equal to the message x, then the signature is valid. Otherwise, the signature is invalid.

3 Hidden Field Equations

This section specifies the HFE cryptosystem. It defines the cryptosystem's parameters and components and matches them with the generic MQ cryptosystem that was described in the previous section.

3.1 The HFE cryptosystem components

The parameters of the HFE cryptosystem are:

- A field F_q (typically F_2)
- An extension of degree n over F_q, called F_{q^n} (typically $n = 128$)
- A security parameter r (typically $r \leq 1024$)

The private key of the HFE cryptosystem is a tuple (f, S, T) containing:

- S, T: Two invertible affine transformations
- A function $f : F_{q^n} \to F_{q^n}$:

  $$f : x \to \sum_{i=0}^{n-1} \beta_i x^{q^{\theta_i}+q^{\phi_i}} + \sum_{i=0}^{n-1} \alpha_i x^{q^{\xi_i}} + \mu_0$$

  where β_i,α_i,μ_0 are elements of F_{q^n} and θ_i,ϕ_i,ξ_i are elements of F_q.

  The exponents of f are carefully chosen so that the degree of the polynomial is not greater than the security parameter r. This bound is required so that f can be inverted using Berlekamp's algorithm [1].

The public key of the HFE cryptosystem is (P):

- P: A system of n multivariate quadratic polynomials p_i so that:

  $$S(f(T(x))) = (p_1(x_1, ..., x_n), ..., p_n(x_1, ..., x_n))$$

  where $p_i \in F_q[x_1, ..., x_n]$

  The public key generation procedure will be studied in section 3.3.

3.2 HFE notation

This section sets up the notation that we will be using through the rest of this text.

- The basis of F_{q^n} in F_q is $\Omega = (\omega_i, ...\omega_n)$ so that $\sum_{i=0}^{n-1} x_i\omega_i \leftrightarrow (x_0, ..., x_n)$.
- The security parameter of HFE is r as defined in 3.1.

3.3 HFE public key generation

Generating an HFE public key essentially consists of finding n polynomials p_i in n variables, so that:

$$S(f(T(x))) = (p_1(x_1, ..., x_n), ..., p_n(x_1, ..., x_n))$$

According to 2.3, S and T are equations of degree 1 and f should be expressable as a system of quadratic polynomials over F_q.
This means that the composition $S \circ f \circ T$ is also a system of quadratic polynomials over F_q.

We must now show that f can indeed be expressed as n **quadratic** polynomials in n variables over F_q:

We approach this problem, by first showing that f can be expressed as n polynomials in n variables and then proving that the polynomials are quadratic.

We observe that we can express any $x \in F_{q^n}$, as an n-tuple of F_q elements, with respect to basis Ω: $x \to (x_1, ..., x_n)$.
Applying the above to function f:

$$f(x) = f(x_1, ..., x_n) = (p_1(x_1, ..., x_n), ..., p_n(x_1, ..., x_n))$$

where $p_i \in F_q[x_1, ..., x_n]$.

Now, we have to show that the polynomials p_i are quadratic:
Since the map $\phi : F_{q^n} \to F_{q^n} : x \to x^{q^i}$ is a linear function, the first sum of the univariate representation of f becomes: $\sum_{i=0}^{n-1} \beta_i x^{q^{\theta_i}+q^{\phi_i}} = \sum_{i=0}^{n-1} \beta_i (x^{q^{\theta_i}} x^{q^{\phi_i}})$

According to the Frobenius Endomorphism, $x = x^{q^i}$ is an automorphism in F_{q^n} (see, lemma A.1.1) thus the highest degree that the first sum of f can take is 2.

An example of this operation can be found in the Appendix.

Since we showed that f can be expressed as a system of n polynomials in n variables, we can also express $S(f(T(x)))$ the same way, thereby generating the HFE public key.

3.4 HFE operations

The operations of HFE are the same as the operations of the generic MQ scheme introduced in 2.4.
Note that: P' of 2.2 corresponds to f of HFE.

4 Kipnis and Shamir HFE cryptanalysis

This section studies the attack of Aviad Kipnis and Adi Shamir. It starts with an overview of the attack and proceeds with describing the whole methodology.

4.1 Overview of attack

The attack is based on the observation that any given system of n multivariate polynomials in n variables over a field F_q can be expressed by a single univariate polynomial over F_{q^n}.
Kipnis and Shamir use this property to transform the original system of multivariate polynomials to an *overdefined* system of equations.
Finally, a novel algebraic technique is introduced, called *relinearization*, which is able to solve overdefined [1] systems of equations and recover the keys of the HFE cryptosystem.

4.2 Transforming a system of multivariate polynomials into a univariate polynomial

This section proves that any system of n multivariate polynomials of degree d in n variables over a field F_q can be represented as a single sparse univariate polynomial of a special form over F_{q^n}.
We begin the proof by showing that any linear mapping in F_q can be represented as a univariate polynomial in F_{q^n}. We note that since the map $x \rightarrow x^{q^i}$ is a linear function of $F_{q^n} \rightarrow F_{q^n}$, any mapping of the form $x \rightarrow \sum_{i=0}^{n-1} \alpha_i x^{q^i}$ is also a linear mapping.

Lemma 4.2.1
Let A be a linear mapping from n-tuples to n-tuples of values in F_q. Then there are coefficients $a_0, ..., a_n$ in F_{q^n} such that for any two n-tuples $(x_0, ..., x_{n-1})$ and $(y_0, ..., y_{n-1})$ of elements in F_q, $(y_0, ..., y_{n-1}) = A(x_0, ..., x_{n-1})$ if and only if $y = \sum_{i=0}^{n-1} \alpha_i x^{q^i}$, where $x = \sum_{i=0}^{n-1} x_i \omega_i$ and $y = \sum_{i=0}^{n-1} y_i \omega_i$ are the elements of F_{q^n} which correspond to the two vectors over F_q.

Proof.

$$(y_0, ..., y_{n-1}) = A(x_0, ..., x_{n-1}) \Leftrightarrow$$

$$\begin{bmatrix} y_o \\ \vdots \\ y_{n-1} \end{bmatrix} = \begin{bmatrix} A \end{bmatrix} \begin{bmatrix} x_o \\ \vdots \\ x_{n-1} \end{bmatrix} \Longrightarrow$$

[1] A system $m \times n$ is called overdefined when $m > n$

$$\begin{bmatrix} y_o \\ \vdots \\ y_{n-1} \end{bmatrix} = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n-1} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n-1} \\ & \vdots & & \\ a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n-1} \end{bmatrix} \begin{bmatrix} x_o \\ \vdots \\ x_{n-1} \end{bmatrix} \Leftrightarrow$$

$$\begin{cases} y_0 = a_{1,1}x_0 + a_{1,2}x_1 + ... + a_{1,n-1}x_{n-1} \\ \vdots \\ y_{n-1} = a_{n-1,1}x_0 + a_{n-1,2}x_1 + ... + a_{n-1,n-1}x_{n-1} \end{cases} \Leftrightarrow$$

$$\begin{cases} y_0 = \sum_{i=0}^{n-1} \alpha_{1,i}x_i \\ y_1 = \sum_{i=0}^{n-1} \alpha_{2,i}x_i \\ \dots \\ y_{n-1} = \sum_{i=0}^{n-1} \alpha_{n-1,i}x_i \end{cases}$$

Expresing the above with respect to basis Ω:

$$\begin{cases} y_0\omega_0 = \sum_{i=0}^{n-1} \alpha_{1,i}x_i\omega_i \\ y_1\omega_1 = \sum_{i=0}^{n-1} \alpha_{2,i}x_i\omega_i \\ \dots \\ y_{n-1}\omega_{n-1} = \sum_{i=0}^{n-1} \alpha_{n-1,i}x_i\omega_i \end{cases}$$

Since $x = \sum_{i=0}^{n-1} x_i\omega_i$ and $y = \sum_{i=0}^{n-1} \alpha_i x^{q^i}$:

$y = \sum_{i=0}^{n-1} \alpha_i x$

and applying the Frobenius Automorphism we obtain:

$y = \sum_{i=0}^{n-1} \alpha_i x^{q^i}$

which proves the lemma.

It should be noted that we can also use a counting argument to prove this lemma, as the original paper does:

Since there are $q^{(n^2)}$ $n \times n$ matrices over F_q and $(q^n)^n$ sums of n monomials over F_{q^n}, the number of linear mappings and the number of polynomials of this form are the same. We can map every polynomial to a linear mapping and we only have to show that the mapping is injective to prove the lemma:
Let two distinct polynomials represent the same linear mapping, their difference being a non-zero polynomial of degree q^{n-1} with q^n roots in the field F_{q^n}. Which is a contradiction since ... ¡XXX APPENDIX¿ □

We have shown that each linear mapping is represented by a univariate polynomial; we now generalize the previous lemma from linear mappings to any system of multivariate polynomials:

Lemma 4.2.2
Let $P = \{P_0(x_0, ..., x_{n-1}), ..., P_{n-1}(x_0, ..., x_{n-1})\}$ be any set of multivariate polynomials in n variables over F_q. Then there are coefficients $a_0, ..., a_n$ in F_{q^n} such that for any two n-tuples $(x_0, ..., x_{n-1})$ and $(y_0, ..., y_{n-1})$ of elements in F_q, $y_j = P_j(x_0, ..., x_{n-1})$ for all $0 \leq j \leq n-1$ if and only if $y = \sum_{i=0}^{n-1} \alpha_i x^{q^i}$, where $x = \sum_{i=0}^{n-1} x_i \omega_i$ and $y = \sum_{i=0}^{n-1} y_i \omega_i$ are the elements of F_{q^n} which correspond to the two vectors over F_q.

Proof.
Without loss of generality, we assume that the first coordinate of the first vector of the basis of F_{q^n} is $\omega_1 = 1$.

The mapping:

$$(x_0, ..., x_{n-1}) \rightarrow (x_i, 0, ..., 0) \text{ , where } 0 \leq i \leq n-1 \tag{1}$$

is linear over F_q and thus it has a univariate polynomial representation over F_{q^n} according to the previous lemma.
In this lemma we will start with the mapping (1) and using allowed actions we will form a vector which contains a multivariate polynomial P_i in each of its coordinates, effectively representing the whole set P. At the same time we will be making the same actions to the univariate polynomial representation of mapping (1), so that in the end we end up with a univariate polynomial representation for the set of multivariate polynomials.

Step [1].- We represent the mapping $(x_0, ..., x_{n-1}) \rightarrow (\prod_i^{n-1} x_i^{c_i}, 0, ..., 0)$, by multiplying all the univariate representations of mapping (1) with their multiplicities c_i.

Step [2].- We represent the mapping $(\prod_i x_i^{c_i}, 0, ..., 0) \rightarrow (k \prod_i x_i^{c_i}, 0, ..., 0)$ by summing the univariate representations of Step [1] with appropriate coefficients. At this point we have created a vector with any multivariate polynomial in the first coordinate of the vector and zeroes elsewhere. In vector form this looks like: $(P_i, 0, ..., 0)$.

Step [3].- We can now create the mapping $(P_i, 0, ..., 0) \rightarrow (0, ..., P_i, 0, ..., 0)$ by multiplying all the coefficients of the univariate representation of Step [2] by the basis element ω_k where k is the coordinate we shift the polynomial to.

Step [4].- We can now create $(P_0, P_1, ..., P_{n-1})$ by repeating Steps 1 and 2 for each polynomial, shifting them to the approprioate coordinate and adding the resulting univariate representations.

A numeric example of this lemma can be found on Appendix C. □

With the following last lemma we reach our goal for this section.

Lemma 4.2.3
Let P be a system of n homogeneous multivariate polynomials of degree d in n variables over F_q. Then the only powers of x which can occur with non-zero coefficients in its univeriate polynomial representation $G(x)$ over F_{q^n} are sums of exactly d (not necessarily distinct) powers of q: $q^{i_1} + q^{i_2} + ... + q^{i_d}$. If d is a constant, then $G(x)$ is sparse and its coefficients can be found in polynomial time.

Proof.
A monomial of degree 1 is a linear functions and according to lemma 4.2.1 it can be represented as a univariate polynomial $\sum_{i=0}^{n-1} \alpha_i x^{q^i}$ - which is the sum of monomials of the form x^{q^i} with each monomial containing a single power of q. When we multiply d such monomials - to create a multivariate polynomial of degree d - we get only powers of x which are the sums of exactly d powers of q. Since $G(x)$ is $\sum_{i=0}^{n-1} \alpha_i x^{q^i}$ - which is the sum of such polynomials multiplied by constants - the same is true for $G(x)$.

To prove that $G(x)$ is sparse, we observe that the construction of $G(x)$ allows it to have at most $O(n^d)$ non-zero coefficients, which is a polynomial number x, if d is a constant. This makes $G(x)$ x-sparse and allows us to find its coefficients in polynomial time by interpolation. □

4.3 Cryptanalytic representation of the public key

In this section we create a new representation of the HFE public key based on the previous lemmas. This new form will be used in subsequent sections to cryptanalyze the cryptosystem.

We notice that each polynomial $P_i \in F_q[x_1, ..., x_n]$ of the HFE public key, can be written as the quadratic form $x^t P_i x$ where P_i is an $n \times n$ matrix of coefficients, x is the column vector of variables and x^t is its transpose.

However, will also be using an alternative representation of the public key, based on the results of the previous section:

$$G(x) = \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} g_{ij} x^{q^i + q^j} = x^t G x \text{ where } G = [g_{ij}] \text{ and } x = (x^{q^0}, x^{q^1}, ..., x^{q^{n-1}})$$

4.4 Cryptanalytic representation of the $\mathbf{P(x) = T(f(S(x)))}$ equation

Through this section, we will be using the univariate representation of the linear mappings S as $\sum_{k=0}^{n-1} s_k x^{q^k}$ and T as $\sum_{k=0}^{n-1} t_k x^{q^k}$ according to lemma 4.2.1. We will also be using the $x^t P x$ and $x^t G x$ form of $P(x)$ and $G(x)$ respectively, according to the previous section.

We start by rewriting the equation: $P(x) = T(f(S(x))) \rightarrow T^{-1}(G(x)) = P(S(x))$, where T^{-1} is also a linear mapping and thus it has a univariate representation according to lemma 4.2.1.

Lemma 4.4.1
The matrix of the quadratic form in x which represents the polynomial composition $T^{-1}(G(x))$ is $\sum_{k=0}^{n-1} t_k G^{*k}$ where G^{*k} is obtained from the $n \times n$ matrix representation of G by raising each one of its entries to the power q^k in K, and cyclically rotating forwards by k steps both the rows and the columns of the result. The matrix of the quadratic form in x which representes the polynomial composition $P(S(x))$ is WPW^t in which $W = [w_{ij}]$ is an $n \times n$ matrix defined by $w_{ij} = (s_{j-i})^{q^i}$, where $j - i$ is computed modulo n.

Proof.

We begin by finding the polynomial composition of $T^{-1}(G(x))$:
According to the Frobenius Automorphism, raising sums to the power q^i is a linear operation:

$$T^{-1}(G(x)) = \sum_{k=0}^{n-1} t_k (\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} g_{ij} x^{q^i+q^j})^{q^k} = \sum_{k=0}^{n-1} t_k \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} (g_{ij})^{q^k} x^{(q^i+q^j)q^k}$$

The exponents of q can be reduced modulo n since $x^{q^n} = x^{q^0} = x$ and the summation indices can be cyclically rotated if they are computed modulo n:

$$T^{-1}(G(x)) = \sum_{k=0}^{n-1} t_k \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} (g_{ij})^{q^k} x^{q^{i+k}+q^{j+k}} = \sum_{k=0}^{n-1} t_k \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} (g_{i-k,j-k})^{q^k} x^{q^i+q^j}$$

The matrix of the quadratic form representation of this polynomial is indeed $G' = \sum_{k=0}^{n-1} t_k G^{*k}$.

We now proceed with finding the polynomial composition of $P(S(x))$:

$$P(S(x)) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} p_{ij} (\sum_{k=0}^{n-1} s_k x^{q^k})^{q^i+q^j} = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} p_{ij} ((\sum_{u=0}^{n-1} s_u x^{q^u})^{q^i})((\sum_{u=0}^{n-1} s_u x^{q^u})^{q^j})$$

We use the same arguments of linearity and cyclic index shifting as above to get:

$$\sum_{i=0}^{n-1}\sum_{j=0}^{n-1} p_{ij} (\sum_{u=0}^{n-1} s_u^{q^i} x^{q^{u+i}})(\sum_{u=0}^{n-1} s_u^{q^j} x^{q^{u+j}}) = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} p_{ij} (\sum_{u=0}^{n-1} s_{u-i}^{q^i} x^{q^u})(\sum_{u=0}^{n-1} s_{u-j}^{q^j} x^{q^u})$$

We rearrange the order of the sums and the multiplied terms to derive:

$$P(S(x)) = \sum_{u=0}^{n-1}\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\sum_{u=0}^{n-1} x^{q^u} s_{u-i}^{q^i} p_{ij} s_{u-j}^{q^j} x^{q^u} = xWPW^t x^t$$

where W is the matrix specified in the lemma. □

4.5 Recovering HFE keys

4.5.1 The fundamental equation

The rest of the attack is based on the matrix equation: $G' = WPW^t$ from the previous section, which the original paper calls *the fundamental equation*.
In this section we will do some observations on the fundamental equation.

- For $G' = \sum_{k=0}^{n-1} t_k G^{*k}$ we observe that:

  G can be computed by representing the public key as a univariate polynomial like 4.3 described, and then representing the polynomial using its quadratic form.

  G^{*k} can be computed by raising the elements of G to the appropriate powers and cyclically rotating its rows and columns.

  This makes G' an $n \times n$ matrix whose entries are linear combination of known values (public key) with unknown coefficients $(t_0, ..., t_{n-1})$ from F_{q^n}.

- The matrix P is an $n \times n$ matrix where only the top left $r \times r$ elements are non-zero, where r is the security parameter of HFE.

- The matrix W is unknown since its elements are the elements of S.

Our goal is to use the above observations to solve the fundamental equation in polynomial time so that we can fully determine the private key tuple (f, S, T).

4.5.2 Recovering T

The first step of the attack is to determine the affine map T. We will do this by recovering $t_0, ..., t_{n-1}$ from the fundamental equation $G' = WPW^t$, since each entry of G' is a linear combination of the t_k variables.
We will use the following observation to achieve this:

- As mentioned in section 4.5.1, matrix P contains at most r non-zero rows, thus both its rank and subsequently the rank of WPW^t cannot exceed r.

- We also notice that for random choices of t_k the expected rank of G' is close to n. This is because choices of t_k that produce linearly independent rows or columns are extremely rare.

We can use the two observations above to distinguish the correct t_k values from the wrong ones. We can express this in a system of equations, as follows:
A G' matrix with the correct t_k values has rank at most r and thus its left nullspace has dimension $n - r$, according to lemma A.2.2.
This means that its basis is composed by $n - r$ linearly independent vectors $x_1, ..., x_{n-r}$. We, then, force the first $n - r$ coordinates of each vector to an arbitrary value and consider the remaining r coordinates as new variables.

Now, by considering each vector equation $xG' = 0$ as n scalar equations over F_{q^n}, with the above considerations we get a total of $n(n-r)$ equations in $r(n-r)+n$ variables.
The above procedure produced a system of multivariate quadratic equations (MQ) which was our original problem. What makes this particular system solvable, is that instead of a system of n variables in n equations, we now have an *overdefined* system of n^2 equations in about rn variables where $r << n$.
Fortunately, techniques exist that solve overdefined systems of multivariate quadratic equations. We will now introduce the *linearization* technique:

Linearization

The linearization technique replaces each product of variables $x_i x_j$ by a new independent variable y_{ij}. The number of these new variables is $n(n+1)/2$ and each quadratic equation in the x_i variables is rewritten as a linear equation in the y_{ij} variables.

If after the above transformation, there are still more equations than variables, we expect the system to be uniquely solvable using Gaussian elimination. After finding all the y_{ij} values, we can derive two possible values for each x_i by extracting the square root of y_{ij}, and use the values of y_{ij} to combine correctly the roots of y_{ii} and y_{ij}. XXX
If after the transformation, the variables are more than the number of equations, the system produces an exponential number of parasitic y_{ij} solutions which do not correspond to any real x solution.

Unfortunately, in our case, the linearization method is not sufficient, and the original paper has to introduce a new technique for solving overdefined MQ systems called *relinearization*.

Relinearization

The relinearization technique increases the number of equations, while decreasing the number of variables, in an already *linearized* MQ system. It achieves that, by adding additional constraints that create relations between the y_{ij}:

For any a, b, c, d, the variables: $x_a x_b x_c x_d$ can be parenthesized in three different ways:

$$(x_a x_b)(x_c x_d) = (x_a x_c)(x_b x_d) = (x_a x_d)(x_b x_c) \Rightarrow y_{ab} y_{cd} = y_{ac} y_{bd} = y_{ad} y_{bc}$$

By using the above relations of y_{ij} variables on the fundamental equation, we increase the number of equations, and produce a solvable system of equations, considering a normal security parameter d.

The relinearization technique is thoroughly studied in the Appendix, where an example is also provided.

4.5.3 Recovering S

Since T is now known, the last part of the attack is to recover S and f.

We notice that the left hand side of the fundamental equation $G' = WPW^t$ is now a known matrix.
As mentioned in section 4.5.1, the matrix P contains at most r non-zero rows, and thus the rank of the right hand side of the fundamental equation cannot exceed r.
Reasoning similarly to the previous section, we shall now translate this rank situation to a system of equations.

Without loss of generality we assume that the rank of P is r and the rank of W is n. Let $u_1, .., u_{n-r}$ be a basis for the left nullspace of G'. Since W^t is invertible, the left nullspace of WPW^t is equal to the left nullspace of WP. The left nullspace of P consists of the vectors which are zero in their first r entries and thus each u_i is mapped by W to a vector of this form. Since G' is known, we can compute its left nullspace, which each of its $n - r$ basis vectors gives rise to r equations in the unknown entries of W.
We now have an *underdefined* system of $r(n-r)$ linear equations in n^2 variables. We can reduce the number of variables from n^2 to n by replacing each w_{ij} by $s_{j-i}^{q^i}$ but then we get nonlinear equations. The crucial observation is that these nonlinear equations over K become linear if reinterpreted over F. Altogether there are $r(n-r)n$ equations in the n^2 new variables over F and for any $r > 1$ the system is greatly overdefined since $r(n-r)n >> n^2$.

A Mathematical Background

The aim of this section is to introduce the reader to the necessary mathematical background. Basic knowledge of algebra and shit is recommended to fully comprehend all the terms and ideas introduced. Tack!

A.1 Fields

Definition 1 (Field)
Let F be a set of $q \in N$ elements with the two operations addition $+ : F \times F \to F$ and multiplication $\dot{:} F \times F \to F$. Note that this definition implies closure for addition and multiplication. We call $(F, +, \dot{})$ a field if the following axioms are fulfilled:

1. $(F, +)$ is an additive abelian group, which means that the following properties are satisfied:

   associativity: For every $a, b, c \in F : ((a + b) + c) = (a + (b + c))$

   additive neutral: There is an element $e \in F$ so that for every $a \in F$: $a + e = a$.

   additive inverse: For every element $a \in F$ there is an element $b \in F$ so that $a + b = 0$.

   commutativity: For every $a, b \in F : a + b = b + a$

2. $(F \backslash \{0\}, \dot{})$ is a multiplicative Abelian group, which means that the following properties are satisfied:

   associativity: For every $a, b, c \in F : ((a\dot{b})\dot{c}) = (a\dot{(}b\dot{c}))$

   multiplicative neutral: There is an element $e \in F$ so that for every $a \in F : a\dot{e} = a$.

   multiplicative inverse: For every element $a \in F \backslash \{0\}$ there is an element $b \in F \backslash \{0\}$ so that $a\dot{b} = 1$.

   commutativity: For every $a, b \in F : a\dot{b} = b\dot{a}$

3. distributivity: For every $a, b, c \in F : a\dot{(}b + c) = a\dot{b} + a\dot{c}$

Example A.1.1
The set of real numbers $\mathbb{R}$ under the usual operations of addition and multiplication forms a field.

Example A.1.2
The set of complex numbers $\mathbb{C}$ forms a field.

Definition 2 (Finite Field)
A field F that contains a finite number of elements n is called a *finite field* and symbolized as F_n. The number of elements is caleld *order of the finite field.*

Example A.1.3
The set $\{0, 1, .., n-1\}$ or Z_n, with the operations of addition and multiplication modulo n, defines a finite field of order n.

Definition 3 (Prime Field)
A field F_p is called a *prime field* if p is prime.

Example A.1.4
The field Z_5 of elements $\{0, 1, 2, 3, 4\}$ is a prime field.

Definition 4 (Field Characteristic)
If for a ring R a positive integer n exists such that $na = 0$ for all $a \in R$, then the least such positive integer is called *characteristic of the ring R*. If no such positive integer exists, then R is of *characteristic* 0.

Example A.1.5
Any prime field Z_p has characteristic p. For example, in Z_5:

$$0 : 5 * 0 = 0 + 0 + 0 + 0 + 0 = 0$$
$$1 : 5 * 1 = 1 + 1 + 1 + 1 + 1 = 5 \bmod 5 = 0$$
$$2 : 5 * 2 = 2 + 2 + 2 + 2 + 2 = 10 \bmod 5 = 0$$
$$3 : 5 * 3 = 3 + 3 + 3 + 3 + 3 = 15 \bmod 5 = 0$$
$$4 : 5 * 4 = 4 + 4 + 4 + 4 + 4 = 20 \bmod 5 = 0$$

Definition 5 (Ring of Polynomials)
Let F be a field. The set of all polynomials with coefficients in F forms a commutative ring denoted $F[X]$ and is called the *ring of polynomials over F*.

Example A.1.6
$x^2 + 2x + 2 \in Z_5[x]$ and $x^2 + 3 \in Z_5[x]$

Note that operations are defined normally under the properties of the field:
$(x^2 + 2x + 2) + (x^2 + 3) = 2x^2 + 2x$

Definition 6 (Monomial)
For a single variable x, we call x^n for $n \in N$ a *monomial*

There are many different definitions of monomial through the literature, but this is the one we use through this document

Example A.1.7
The polynomial x^4 is a monomial.

Definition 7 (Extension Field)
Let E be a field and let F be a subset of the underlying set of E, while respecting all the field operations and inverses of E. Then we call E an *extension field of F*.

E can be regarded as a *vector space* over F, considering elements of F as scalars and elements of E as vectors. The dimension of this vector space is called the *degree of the extension field*.

Example A.1.8
The field of complex numbers $\mathbb{C}$ is an extension field of the field of real numbers $\mathbb{R}$.

Example A.1.9
The real coordinate space $\mathbb{R}^n$ of degree n, is a vector space of degree n over $\mathbb{R}$. Hence each element of $\mathbb{R}^n$ can be represented as an n-tuple of elements of $\mathbb{R}$.

Lemma A.1.1 (Frobenius Automorphism)
Let F be a field of characteristic p. Then the map $\phi_p : F \to F$ defined by $\phi_p(x) = x^p$ for $a \in F$ is an automorphism, called the *Frobenius Automorphism* of F.

A.2 Linear/Affine transformations

Definition 8 (Linear Transformation)
Let V and W be vector spaces over a field F. A *linear transformation from V into W* is a function $T : V \to W$ such that:

$T(ca + b) = c(Ta) + Tb$

for all a and b in V and all scalars c in F.

Example A.2.1
The function $T : \mathbb{R}^2 \to \mathbb{R}^2 : x \to 2x$ is a linear transformation since:

$$T(ca + b) = 2(ca + b) = 2ca + 2b = c2a + 2b = cT(a) + T(b)$$

Definition 9 (Transformation Matrix of Linear Transformation)
If $T : R^n \to R^m$ is a linear transformation and x is a column vector with n elements, then:

$$T(x) = Ax$$

for some $m \times n$ matrix A. We call A the *transformation matrix of T*.

Example A.2.2
The linear transformation T from the previous example has transformation matrix: $\left[\begin{smallmatrix} 2 & 0 \\ 0 & 2 \end{smallmatrix}\right]$.
As a check:

$$Tx = \left[\begin{smallmatrix} 2 & 0 \\ 0 & 2 \end{smallmatrix}\right] \left[\begin{smallmatrix} x_1 \\ x_2 \end{smallmatrix}\right] = \left[\begin{smallmatrix} 2x_1 \\ 2x_2 \end{smallmatrix}\right] = 2 \left[\begin{smallmatrix} x_1 \\ x_2 \end{smallmatrix}\right] = 2x = T(x)$$

Definition 10 (Affine Transformation)
Let V and W be vector spaces over a field F. An *affine transformation from V to W* is a function $f : V \to W$ consisting of a linear transformation followed by a translation.

Let A be a matrix and $b \in V$ a vector, so that:

$f(x) = Ax + b$

We call the right hand side of the above construction, the *matrix representation of the affine transformation f*

Example A.2.3
The function $A : \mathbb{R}^2 \to \mathbb{R}^2 : x \to 2x + 1$ is an affine transformation.

We suppose that the concepts of vector spaces, dimension, rank of matrix, nullity, etc. is known to the reader.

Definition 11 (Nullspace and nullity)
For a matrix A, we call its *nullspace* the set of all vectors x that satisfy $Ax = 0$. The dimension of the nullspace is called *nullity*.

Definition 12 (Left Nullspace)
The left nullspace of a matrix A consists of all vectors x such that $x^t A = 0^t$.
The left nullspace of A is the same as the nullspace of A^t:
$x^t A = 0^t \to (x^t A)^t = (0^t)^t \to Ax = 0$
since $(AB)^t = B^t A^t$.

Lemma A.2.1 (Rank-Nullity Theorem)
If A is an $m \times n$ matrix we have:
$\text{rank}(A) + \text{nullity}(A) = n$

Lemma A.2.2 (Corollary of Definition ABOVE)
By definition we have rank(A) = rank(A^t) and since we are dealing with a square matrix: nullity(A) = nullity(A^t).

By using the above theorem for a matrix A with $\text{rank}(A) = r$:
$\text{rank}(A^t) + \text{nullity}(A^t) = n \Rightarrow$
$\text{nullity}(A^t) = n - r$

Definition 13 (System of Linear Equations)
A collection of linear equations involving a set of variables is called a *system of linear equations*.
A system of linear equations with more variables than equations is called *underdefined*.
A system of linear equations with more equations than variables is called *overdefined*

Definition 14 (Quadratic Forms)
A homogeneous polynomial of second degree in a number of variables, is called a *quadratic form*.
In general, for A a symmetric $n \times n$ matrix and x an $n \times 1$ column vector of variables, any quadratic form can be expressed in matrix form in the form of $x^t Ax$. We call this the *quadratic form associated with A*.

Example A.2.4

$$2x^2 + 6xy - 5y^2 = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} 2 & 3 \\ 3 & -5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

A.3 Relinearization

In this section we study the relinearization technique of solving overdefined systems of equations.
Considering a system of e homogeneous quadratic equations in m variables $x_1, ..., x_m$.
Using the linearization method – replacing any product of two variables $x_i x_j$ by a new variable y_{ij} – on the system we get a system of e equations in approximately $m^2/2 y_{ij}$ variables.
The solution space (the column space) of such a system is a linear subspace of dimension $(1/2 - \epsilon)m^2$ – according to corollary A.2.2 – and each solution can be expressed as a linear function of $(1/2 - \epsilon)m^2$ new variables z_k.
If the number of equations satisfies $e \geq m^2/2$ we expect the system to have a unique solution, but if $e << m^2/2$ we expect the linear system to have an exponential number of parasitic y_{ij} solutions which do not correspond to any real x_i solutions.
The relinearization technique helps us solve the latter case. Specifically, we add additional equations by relating the various y_{ij} variables to each other in the way implied by their definition as $y_{ij} = x_i x_j$.
For any a, b, c, d the x_i variables: $x_a x_b x_c x_d$ can be parenthesized in three different ways:
$(x_a x_b)(x_c x_d) = (x_a x_c)(x_b x_d) = (x_a x_d)(x_b x_c) \Rightarrow y_{ab} y_{cd} = y_{ac} y_{bd} = y_{ad} y_{bc}$
There are about $m^4/4!$ different ways to choose sorted 4-tuples of distinct indices and each choice gives rise to 2 equations. We thus get about $m^4/12$ quadratic equations in $m^2/2$ variables y_{ij}.
We can also reduce the number of variables to $m^2/2 - e$ by replacing the y_{ij} variables with their parametric represenations as a linear combination of the new z_k variables
An interesting fact about this technique is that it's recursive. For example, we can linearize again the final state of above system by replacing each product $z_i z_j$ with a new variable u_{ij} and restart the procedure.

B Examples

B.1 Univariate to multivariate representation

This example was found in Nicolas Courtois' webpage [5] and demonstrates how to express the function f of the HFE cryptosystem as a system of multivariate polynomials (section 3.3).
Let F be a field and E an extension field of degree 3 over F.

We will now see how we can express a univariate polynomial f over E as a system of multivariate polynomials over F:
Let $f : E \to E$

$a \to a + a^3 + a^5$

Since E is an extension field over F of degree 3, we can represent any element of E as a 3-tuple over F.
Also, since E is isomorphic to the polynomial ring $F[x]/(x^3 + x^2 + 1)$, we can represent any element a of E as a polynomial over $F[x]/(x^3 + x^2 + 1)$, using the basis $B = \{x^2, x, 1\}$.
This means that we have:

$a \leftrightarrow (a_0, a_1, a_2) \leftrightarrow a_2x^2 + a_1x + a_0$

So, we have: $f(a) = f(a_0, a_1, a_2) = f(a_2x^2 + a_1x + a_0)$
where

$$f(a_2x^2 + a_1x + a_0) = (a_2x^2 + a_1x + a_0) + (a_2x^2 + a_1x + a_0)^3 + (a_2x^2 + a_1x + a_0)^5 \pmod{x^3 + x^2 + 1} =$$
$$(a_2 + a_2a_1 + a_2a_0 + a_1)x^2 + (a_2a_1 + a_1a_0 + a_2)x + (a_0 + a_2 + a_1a_0 + a_2a_0)$$

Expressing this result with respect to basis B we get a vector with a multivariate polynomial in each coordinate:

$((a_2 + a_2a_1 + a_2a_0 + a_1), (a_2a_1 + a_1a_0 + a_2), (a_0 + a_2 + a_1a_0 + a_2a_0))$

B.2 Example of lemma 2

In this example we will convert step by step the multivariate polynomials: $f(x, y, z) = 2x^2y + yz$ and $h(x, y, z) = y^2z + xy$ to a single univariate polynomial according to lemma 4.2.2.

Initially, according to the lemma we have the mappings:

$(x,y,z) \rightarrow (x,0,0)$ $\quad \sum_i \alpha_i x^{q^i}$

$(x,y,z) \rightarrow (y,0,0)$ $\quad \sum_i \beta_i x^{q^i}$

$(x,y,z) \rightarrow (z,0,0)$ $\quad \sum_i \gamma_i x^{q^i}$

Step [1].- *We represent the mapping* $(x_0,...,x_{n-1}) \rightarrow (\prod_i x_i^{c_i},0,...,0)$*, by multiplying all the univariate representations of mapping* $(x_0,...,x_{n-1}) \rightarrow (x_i,0,...,0)$ *with their multiplicities* c_i*.*

We now represent the mappings:

$(x,y,z) \rightarrow (x^2y,0,0)$ $\quad \sum_i \alpha_i x^{q^i} \sum_i \alpha_i x^{q^i} \sum_i \beta_i x^{q^i}$

$(x,y,z) \rightarrow (yz,0,0)$ $\quad \sum_i \beta_i x^{q^i} \sum_i \gamma_i x^{q^i}$

$(x,y,z) \rightarrow (y^2z,0,0)$ $\quad \sum_i \beta_i x^{q^i} \sum_i \beta_i x^{q^i} \sum_i \gamma_i x^{q^i}$

$(x,y,z) \rightarrow (xy,0,0)$ $\quad \sum_i \alpha_i x^{q^i} \sum_i \beta_i x^{q^i}$

Step [2].- *We represent the mapping* $(\prod_i x_i^{c_i},0,...,0) \rightarrow (k\prod_i x_i^{c_i},0,...,0)$ *by summing the univariate representations of Step [1] with appropriate coefficients. At this point we have created a vector with any multivariate polynomial in the first coordinate of the vector and zeroes elsewhere. In vector form this looks like:* $(P_i,0,...,0)$*.*

We now represent the mappings:

$(x,y,z) \rightarrow (2x^2y+yz,0,0)$ $\quad 2(\sum_i \alpha_i x^{q^i} \sum_i \alpha_i x^{q^i} \sum_i \beta_i x^{q^i}) + \sum_i \beta_i x^{q^i} \sum_i \gamma_i x^{q^i}$

$(x,y,z) \rightarrow (y^2z+xy,0,0)$ $\quad \sum_i \beta_i x^{q^i} \sum_i \beta_i x^{q^i} \sum_i \gamma_i x^{q^i} + \sum_i \alpha_i x^{q^i} \sum_i \beta_i x^{q^i}$

Step [3].- *We can now create the mapping* $(P_i,0,...,0) \rightarrow (0,...,P_i,0,...,0)$ *by multiplying all the coefficients of the univariate representation of Step [2] by the basis element* ω_k *where* k *is the coordinate we shift the polynomial too.*

So we have: $(y^2z+xy,0,0) \rightarrow (0,y^2z+xy,0)$ with no effect on its univariate representation.

Step [4].- *We can now create* $(P_0,P_1,...,P_{n-1})$ *by repeating Steps 1 and 2 for each polynomial, shifting them to the approprioate coordinate and adding the resulting univariate representations.*

By summing the above we have the univariate representation:

$$2(\sum_i \alpha_i x^{q^i} \sum_i \alpha_i x^{q^i} \sum_i \beta_i x^{q^i}) + \sum_i \beta_i x^{q^i} \sum_i \gamma_i x^{q^i} \sum_i \beta_i x^{q^i} \sum_i \beta_i x^{q^i} \sum_i \gamma_i x^{q^i} + \sum_i \alpha_i x^{q^i} \sum_i \beta_i x^{q^i}$$

which represents the mapping

$(x, y) \rightarrow (2x^2y + yz, y^2z + xy)$

B.3 Linearization/Relinearization

This example is taken off the original paper by Kipnis and Shamir and demonstrates the linearization and relinearization techniques.
We will attempt to solve a system of 5 quadratic equations in three variables x_1, x_2, x_3 modulo 7.

$$\begin{aligned}
3x_1x_1 + 5x_1x_2 + 5X_1x_3 + 2x_2x_2 + 6x_2x_3 + 4x_3x_3 &= 5 \bmod 7 \\
6x_1x_1 + 1x_1x_2 + 4x_1x_3 + 4x_2x_2 + 5x_2x_3 + 1x_3x_3 &= 6 \bmod 7 \\
5x_1x_1 + 2x_1x_2 + 6x_1x_3 + 2x_2x_2 + 3x_2x_3 + 2x_3x_3 &= 5 \bmod 7 \\
2x_1x_1 + 0x_1x_2 + 1x_1x_3 + 6x_2x_2 + 5x_2x_3 + 5x_3x_3 &= 0 \bmod 7 \\
4x_1x_1 + 6x_1x_2 + 2x_1x_3 + 5x_2x_2 + 1x_2x_3 + 4x_3x_3 &= 0 \bmod 7
\end{aligned}$$

After replacing each x_1x_j by y_{ij} we obtain a system of 5 equations in 6 variables. By solving this system using elimination we obtrain a parametric solution in an unknown variable z:
$y_{11} = 2 + 5z$, $y_{12} = z$, $y_{13} = 3 + 2z$, $y_{22} = 6 + 4z$, $y_{23} = 6 + z$, $y_{33} = 5 + 3z$
Solving this system would produce many parasitic solutions.

We can now use the relinearization technique to impose additional constraints to the system:

$$y_{11}y_{23} = y_{12}y_{13}, y_{12}y_{23} = y_{13}y_{22}, y_{12}y_{33} = y_{13}y_{23} \Rightarrow \quad (2)$$

$$(2 + 5z)(6 + z) = z(3 + 2z), z(6 + z) = (3 + 2z)(6 + 4z), z(5 + 3z) = (3 + 2z)(6 + z) \Rightarrow \quad (3)$$

$$3z^2 + z + 5 = 0, 0z^2 + 4z + 4 = 0, z^2 + 4z + 3 = 0 \quad (4)$$

The relinearization technique introduces two new variables $z_1 = z$ and $z_2 = z^2$ and treats them as independent variables. This means that we have three linear equations in two variables. Solving that we obtain $z_1 = 6$ and $z_2 = 1$.
We can now work backwards to find the x_i solutions that correspond to them: Using the values of z_1 and z_2 in (2) we get $y_{11} = 4, y_{22} = 2, y_{33} = 2$ and by extracting their square roots modulo 7 we find that $x_1 = \pm 2, x_2 = \pm 3, x_3 = \pm 3$. Finally, we use the values $y_{12} = 6, y_{23} = 5$ to combine these roots in just two possible ways to obtain: $x_1 = 2, x_2 = 3, x_3 = 4$ and $x_1 = 5, x_2 = 4, x_3 = 3$ which solve the original quadratic system.

References

[1] E. R. Berlekamp. Factoring Polynomials over Finite Fields. In *Bell System Technical Journal*, volume 46, pages 1853–1859, 1967.

[2] Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen. *Post Quantum Cryptography*. Springer Publishing Company, Incorporated, 1st edition, 2008.

[3] An Braeken, Christopher Wolf, and Bart Preneel. A study of the security of unbalanced oil and vinegar signature schemes, 2004.

[4] Jean charles Faugre and Antoine Joux. Algebraic cryptanalysis of hidden field equation (HFE) cryptosystems using grbner bases. In *In Advances in Cryptology CRYPTO 2003*, pages 44–60. Springer, 2003.

[5] Nicolas T. Courtois. HFE page. `http://www.minrank.org/hfe/`.

[6] Y. Desmedt, A. Odlyzko, P. Piret, P. Delsarte, and P. Delsarte. Fast cryptanalysis of the matsumoto-imai public key scheme. In *Advances in Cryptology EuroCrypt84, volume 209 of Lecture Notes in Computer Science*, pages 142–149. Springer-Verlag, 1985.

[7] Vivien Dubois, Louis Granboulan, and Jacques Stern. J.stern. cryptanalysis of HFE with internal perturbation. In *In PKC 07*, pages 249–265. Springer-Verlag, 2006.

[8] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[9] A. Shamir H. Ong, C. Schnorr. A fast signature scheme based on quadratic equations.

[10] Masao Kasahara and Ryuichi Sakai. A construction of 100 bit public-key cryptosystem and digital signature scheme. In *IMann and Thompson, 19871 W.C. Mann and S.A*, pages 495–531, 2003.

[11] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced oil and vinegar signature schemes. In *IN ADVANCES IN CRYPTOLOGY EUROCRYPT 1999*, pages 206–222. Springer, 1999.

[12] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem by relinearization. In *Proceedings of the 19th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '99, pages 19–30, London, UK, 1999. Springer-Verlag.

[13] Tsutomu Matsumoto and Hideki Imai. Public quadratic polynomial-tuples for efficient signature-verification and message-encryption. pages 419–453. Springer-Verlag, 1998.

[14] Jacques Patarin. Asymmetric cryptography with a hidden monomial. In *CRYPTO'96*, pages 45–60, 1996.

[15] Jacques Patarin. Hidden fields equations (HFE) and isomorphisms of polynomials (IP): Two new families of asymmetric algorithms. In Ueli Maurer, editor, *Advances in Cryptology EUROCRYPT 96*, Lecture Notes in Computer Science, pages 33–48. Springer Berlin / Heidelberg, 1996.

[16] Jacques Patarin. Cryptanalysis of the matsumoto and imai public key scheme eurocrypt 98. *Des. Codes Cryptography*, 20:175–209, June 2000.

[17] John M. Pollard and Claus P. Schnorr. An efficient solution of the congruence $x^2+ky^2 = m(modn)$. *IEEE Trans. Inf. Theor.*, 33:702–709, September 1987.