How is Polly? Revisiting the Differential Attack on Polly Cracker after 20 years

Christoph Strobl

Master’s thesis
2022:E39

Lund University

Faculty of Science
Centre for Mathematical Sciences
Mathematics

CENTRUM SCIENTIARUM MATHEMATICARUM

Abstract

When building cryptographic systems, one is constantly on the hunt for hard to solve problems, but are all problems suitable?

In this thesis we will take a look at Gröbner bases over finite fields and how they fail to provide a secure cryptosystem. We discuss several ways of attacking a Gröbner basis based cryptosystem named Polly Cracker, analyse the Differential Attack by Hofheinz and Steinwandt and show experimental results which suggest its extended range of application.

Keywords: polly cracker, differential attack, asymmetric cryptography, multivariate cryptography, public key cryptography

Danksagung

I would like to thank my supervisor Anna Torstensson for her guidance during my thesis.

Auch bei meiner dritten Arbeit soll der Dank welchen ich meinem UrgroSSvater und meinem GroSSvater gegenüber verspüre nicht unerwähnt bleiben. Beide haben mir mein Erststudium in Deutschland und mein jetziges Zweitstudium in Schweden ermöglicht. Desweiteren möchte ich dem Schwedischem Staat für die finanzielle Unterstützung während meines Studiums danken.

Contents

Notation

$\mathbf{K}$	A field.
$\langle x, y\rangle$	Ideal generated by x and y.
$\deg(x^\alpha)$	degree of monomial x^α.
$\mathrm{LCM}(p,q)$	Least common multiple of p and q.
$\mathrm{LM}(p)$	Leading monomial of the polynomial p.
$\mathrm{LT}(p)$	Leading term of the polynomial p.
grevlex	Graded Reversed Lexicographic Order.
grlex	Graded Lexicographic Order.
lex	Lexicographic Order.
$\mathrm{S}(f,g)$	S-polynomial of f and g.

In this work we will explore the world of multivariate polynomials and their possible applications in public key cryptography. We start with an introduction to some important aspects of multivariate polynomials: monomial orderings, Gröbner bases and Buchberger's algorithm. From there on we will continue by introducing cryptographic schemes mostly referred to as *Polly Cracker* and their cryptanalysis. In the cryptanalysis part we will discuss currently known methods of breaking the introduced schemes and explore their strengths and weaknesses.

1 Introduction

A general reference for this section is [1].

Definition 1.1. A monomial in $x_1, \dots, x_n$ is a product of the form

$$x_1^{\alpha_1} \cdot x_2^{\alpha_2} \dots x_n^{\alpha_n}$$

where all the exponents $\alpha_1, \dots, \alpha_n$ are non-negative integers. The total degree of the monomial, denoted by $\deg(x^\alpha)$, is the sum $\alpha_1 + \dots + \alpha_n$.

Definition 1.2. A polynomial p in $x_1, \dots, x_n$ with coefficients in the field $\mathbf{F}$ is a linear combination of monomials of the form:

$$p = \sum_\alpha a_\alpha x^\alpha \qquad \text{with } a_\alpha \in \mathbf{F}$$

where the sum is over a finite number of n-tuples $\alpha = (\alpha_1, \dots, \alpha_n)$. The set of all polynomials in the variables $x_1, \dots, x_n$ with coefficients in $\mathbf{F}$ is written $\mathbf{F}[x_1, \dots, x_n]$.

Example 1. An example of a polynomial in three variables in $\mathbf{Q}[x, y, z]$ is:

$$p(x, y, z) = \frac{7}{3}x^2y^2z + \frac{2}{3}x^3z^2 + x - 2y^2$$

Definition 1.3. Let $f = \sum_\alpha a_\alpha x^\alpha$ be a multivariate polynomial in $\mathbf{F}[x_1, \dots, x_n]$.

1. a_α is called the coefficient of the monomial x^α
2. $a_\alpha x^\alpha$ is called a term of f, whenever $a_\alpha \neq 0$
3. The total degree of f is denoted by $\deg(f)$ and is the maximum $\deg(x^\alpha)$ such that the coefficient a_α is nonzero.

Example 2. The polynomial $p(x, y, z) = x^2y^2z + x^3z^2 + x - y^2$ has four terms and a total degree of five.

Remark. A polynomial consisting only of monomials of degree greater than 0, will be called *n-monomial*, where n denotes the number of terms. So $x^2y^5 + y^3z^2$ is a 2-monomial.

Definition 1.4. A *Laurent-Polynomial* is an expression of the form:

$$p = \sum_{\alpha} a_\alpha x^\alpha \qquad \text{with } a_\alpha \in \mathbf{F} \tag{1}$$

where the coefficients a_α are elements in the field $\mathbf{F}$ and the n-tuples $\alpha = (\alpha_1, \ldots, \alpha_n)$ are integers. Using the multiplication and addition rules of regular polynomials, with the exception that also negative powers of x are permitted, Laurent polynomials fulfill the ring axioms. Such a ring is called *Laurent-Ring* over $\mathbf{F}$.

The proof that Laurent-Polynomials over $\mathbf{F}$ form a ring, which we denote by $\mathbf{F}[x, x^{-1}]$, is similar to the proof that the set of polynomials forms a ring except that in this case also negative exponents are permitted. Laurent-rings are mentioned for example in [2].

Definition 1.5. A subset I of $\mathbf{F}[x_1, \ldots, x_n]$ is called an ideal if the following conditions are satisfied:

1. $0 \in I$
2. If $f, g \in I$, then $f + g \in I$
3. If $f \in I$ and $h \in \mathbf{F}[x_1, \ldots, x_n]$ then $hf \in I$.

Definition 1.6. Let $f_1, \ldots, f_s$ be polynomials in $\mathbf{F}[x_1, \ldots, x_n]$, then we write:

$$\langle f_1, \ldots, f_s \rangle = \left\{ \sum_{i=1}^{s} h_i f_i \mid h_1, \ldots, h_s \in \mathbf{F}[x_1, \ldots, x_n] \right\}$$

We will know see that $\langle f_1, \ldots, f_s \rangle$ describes an ideal.

Lemma 1.7. *Let $f_1, \ldots, f_s$ be polynomials in $\mathbf{F}[x_1, \ldots, x_n]$, then $\langle f_1, \ldots, f_s \rangle$ is an ideal of $\mathbf{F}[x_1, \ldots, x_n]$. One says that $\langle f_1, \ldots, f_s \rangle$ is the ideal generated by $f_1, \ldots, f_s$.*

Proof. Clearly $0 \in \langle f_1, \ldots, f_s \rangle$ since $0 = \sum_{i=1}^{s} 0 \cdot f_i$. Now let $f = \sum_{i=1}^{s} p_i f_i$, $g = \sum_{i=1}^{s} q_i f_i$ and $h \in \mathbf{F}[x_1, \ldots, x_n]$. Then the equations:

$$\begin{aligned} f + g &= \sum_{i=1}^{s} (p_i + q_i) f_i \\ hf &= \sum_{i=1}^{s} (hp_i) f_i \end{aligned}$$

hold and this proofs that $\langle f_1, \ldots, f_s \rangle$ is an ideal. □

1.1 Orderings of Monomials

Definition 1.8. A monomial ordering on $\mathbf{F}[x_1, \ldots, x_n]$ is any relation $\succ$ on $\mathbf{Z}^n_{\geqslant 0}$ such that the following conditions are satisfied:

1. $>$ is a total (or linear) ordering on $\mathbf{Z}^n_{\geqslant 0}$. Meaning that for every pair of monomials x^α and x^β, exactly one of the following statements holds:

$$x^\alpha < x^\beta, \qquad x^\alpha = x^\beta, \qquad x^\alpha > x^\beta$$

2. If $\alpha > \beta$ and $\gamma \in \mathbf{Z}^n_{\geqslant 0}$, then $\alpha + \gamma > \beta + \gamma$

3. $>$ is a well-ordering on $\mathbf{Z}^n_{\geqslant 0}$. This means that every nonempty subset of $\mathbf{Z}^n_{\geqslant 0}$ has a smallest element under the relation $>$

Lemma 1.9. *An order relation $>$ on $\mathbf{Z}^n_{\geqslant 0}$ is a well-ordering if and only if every strictly decreasing sequence in $\mathbf{Z}^n_{\geqslant 0}$*

$$\alpha(1) > \alpha(2) > \alpha(3) > \dots$$

eventually terminates.

Proof. This will be proven in its contra positive form: An order relation $>$ is not a well-ordering if and only if there is an infinite strictly decreasing sequence in $\mathbf{Z}^n_{\geqslant 0}$. If $>$ is not a well-ordering, then some nonempty subset $S \subset \mathbf{Z}^n_{\geqslant 0}$ has no least element. Now pick $\alpha(1) \in S$. Since $\alpha(1)$ is not the least element, we can find $\alpha(1) > \alpha(2)$ in S. Then also $\alpha(2)$ is also not the least element, so there exists $\alpha(2) > \alpha(3)$ in S. This can be continued which leads to an infinite strictly decreasing sequence

$$\alpha(1) > \alpha(2) > \alpha(3) > \dots.$$

Conversely, given such an infinite sequence, $\{\alpha(1), \alpha(2), \alpha(3), \dots\}$ is a nonempty subset of $\mathbf{Z}^n_{\geqslant 0}$ with no least element and thus the relation $>$ is not a well-ordering. □

Definition 1.10 (Lexicographic Order)**.** Let us define $\alpha = (\alpha_1, \dots, \alpha_n)$ and $\beta = (\beta_1, \dots, \beta_n)$ be elements of $\mathbf{Z}^n_{\geqslant 0}$. We say that $\alpha >_{lex} \beta$ if, in the vector difference $\alpha - \beta \in \mathbf{Z}^n_{\geqslant 0}$, the left-most nonzero entry is positive. If $\alpha > \beta$, one writes $x^\alpha >_{lex} x^\beta$.

For example:
$(3, 2, 0) >_{lex} (1, 3, 4)$ since $\alpha - \beta = (2, -1, -4)$ or equivalently:

$$x_1^3 x_2^2 >_{lex} x_1 x_2^3 x_3^4$$

Proposition 1.11. *The lex ordering on $\mathbf{Z}^n_{\geqslant 0}$ is a monomial ordering.*

Proof. 1. That the ordering $>_{lex}$ is a total ordering follows directly from the definition and the fact that the usual numerical order on $\mathbf{Z}_{\geqslant 0}$ is a total ordering.

2. If $\alpha >_{lex} \beta$, then the left-most nonzero entry of $\alpha - \beta$, say $\alpha_k - \beta_k$ is positive. But $x^{\alpha} \cdot x^{\gamma} = x^{\alpha+\gamma}$ and $x^{\beta} x^{\gamma} = x^{\beta+\gamma}$. Then in $(\alpha + \gamma) - (\beta + \gamma) = \alpha - \beta$ the left-most nonzero entry is again $\alpha_k - \beta_n > 0$.

3. Suppose that $>_{lex}$ is not a well-ordering. Then by Lemma 1.9, there would be an infinite strictly descending sequence

   $$\alpha(1) >_{lex} \alpha(2) >_{lex} \alpha(3) >_{lex} \dots$$

   of elements of $\mathbf{Z}^n_{\geqslant 0}$. This leads to a contradiction.

   Consider the first entries of the vectors $\alpha(i) \in \mathbf{Z}^n_{\geqslant 0}$. By the definition of the lex order, these first entries form a non increasing sequence of non-negative integers. Since $\mathbf{Z}^n_{\geqslant 0}$ is well-ordered, the first entries of the $\alpha(i)$ must *stabilize* eventually. That is that there exist a k such that all the first components of $\alpha(i)$ with $i \geqslant k$ are equal.

   Beginning at $\alpha(k)$, the second and subsequence entries come into play in determining the lex order. The second entries of $\alpha(k), \alpha(k+1), \dots$ form a non increasing sequence. Like before, the second entries *stabilize* eventually too. Continuing this procedure for some l, we see that $\alpha(l)\alpha(l+1), \dots$ are all equal. This contradicts the fact $\alpha(l) > \alpha(l+1)$

□

In a similar fashion, it can also be proven that the next two orderings are monomial orderings.

Definition 1.12 (Graded Lexicographic Order)**.** Let $\alpha, \beta \in \mathbf{Z}^n_{\geqslant 0}$. We say $\alpha >_{grlex} \beta$ if

$$|\alpha| = \sum_{i=1}^{n} \alpha_i > |\beta| = \sum_{i=1}^{n} \beta_i, \qquad or \quad |\alpha| = |\beta| \ and \ \alpha >_{lex} \beta$$

Example 3.

1. $(1,3,4) >_{grlex} (4,2,0)$ since $|(1,3,4)| = 8 > |(4,2,0)| = 6$. Which is equivalent with writing: $x_1 x_2^3 x_3^4 >_{grlex} x_1^4 x_2^2$

2. $(1,2,5) >_{grlex} (1,1,6)$ since $|(1,2,5)| = |(1,1,6)| = 8$ and $(1,2,5) >_{lex} (1,1,6)$. Which is equivalent with writing: $x_1 x_2^2 x_3^5 >_{grlex} x_1 x_2 x_3^6$

Definition 1.13 (Graded Reversed Lexicographic Order)**.** Let $\alpha, \beta \in \mathbf{Z}^n_{\geqslant 0}$. We say $\alpha >_{grevlex} \beta$ if

$$|\alpha| = \sum_{i=1}^{n} \alpha_i > |\beta| = \sum_{i=1}^{n} \beta_i, \qquad or \quad |\alpha| = |\beta|$$

and, in $\alpha - \beta \in \mathbf{Z}^n$, the right-most nonzero entry is negative.

Example 4.

1. $(4,7,2) >_{grevlex} (4,2,4)$ since $|(4,7,2)| = 13 > |(4,2,4)| = 10$

Example 5. Let the polynomial $p(x, y, z) = x^3y^2z + x^2y^4 + x^5$ be ordered in different orderings, this is then:

$$p(x, y, z) = x^5 + x^3y^2z + x^2y^4$$

for lexicographic ordering, with $x > y > z$.

$$p(x, y, z) = x^3y^2z + x^2y^4 + x^5$$

for graded lexicographic ordering. And

$$p(x, y, z) = x^2y^4 + x^3y^2z + x^5$$

in graded reversed lexicographic ordering.

Additionally we will use the following terms throughout this work:

Definition 1.14. Let $f = \sum_\alpha a_\alpha x^\alpha$ be a nonzero polynomial in $\mathbf{F}[x_1, \ldots, x_n]$ and let $>$ denote a monomial ordering.

1. The multi degree of f is defined:

   $$\text{multideg}(f) = \max(\alpha \in \mathbf{Z}^n_{\geqslant 0} \mid a_\alpha \neq 0)$$

   where the maximum is taken with respect to the monomial ordering $>$.

2. The leading coefficient of f is defined:

   $$\text{LC}(f) = a_{\text{multideg}(f)} \in \mathbf{F}$$

3. The leading monomial of f is defined:

   $$\text{LM}(f) = x^{\text{multideg}(f)}$$

   where the coefficient is 1.

4. The leading term of f is therefore:

   $$\text{LT}(f) = \text{LC}(f) \cdot \text{LM}(f)$$

1.2 Gröbner Bases

When dividing univariate polynomials we use the division algorithm to divide polynomials in $\mathbf{F}[x]$ by one another. We possibly a product and a remainder upon successful execution. This makes it possible to then rewrite a polynomial $p(x)$ in the following way:

$$p(x) = q(x)b(x) + r(x) \quad \text{where } \deg(r) < \deg(b).$$

But this is not as easy with multivariate polynomials. Let's say we want to divide some polynomial $p \in \mathbf{F}[x_1, \dots, x_n]$ by $f_1, \dots, f_s \in \mathbf{F}[x_1, \dots, x_n]$, which means we would like to get an expression in the form:

$$p = a_1 f_1 + \cdots + a_s f_s + r.$$

There are several terms describing this process, for example *reduction* or *multivariate division*. Let us now take a look at the division algorithm:

Theorem 1.15 (Division Algorithm)**.** *Let $>$ be some fixed monomial ordering and let $F = (f_1, \dots, f_s)$ be an ordered s-tuple of polynomials in $\mathbf{F}[x_1, \dots, x_n]$. Then every $f \in \mathbf{F}[x_1, \dots, x_n]$ can be written in the following way:*

$$f = a_1 f_1 + \cdots + a_s f_s + r$$

where $a_i, r \in \mathbf{F}[x_1, \dots, x_n]$, and either $r = 0$ or r is a k-linear combination of monomials of which none is divisible by any of LT$(f_1), \dots,$ LT(f_s).

Proof. The complete proof for this can be seen for example in [1]. Pseudocode for the division is presented here. □

Algorithm 1 Division algorithm in $\mathbf{F}[x_1, \dots, x_n]$

```
Require: $f_1, \dots, f_s, f$
  procedure Division($f, [f_1, \dots, f_s]$)
      $a_1 \leftarrow 0, a_2 \leftarrow 0, \dots, a_s \leftarrow 0, r \leftarrow 0$
      $p \leftarrow f$
      while $p \neq 0$ do
          $i \leftarrow 1$
          divisionoccurred := false
          while $i \leqslant s$ AND divisionoccurred = false do
              if LT($f_i$) divides LT($p$)  then
                  $a_i := a_i + \mathrm{LT}(p)/\mathrm{LT}(f_i)$
                  $p := p - (\mathrm{LT}(p)/\mathrm{LT}(f_i))f_i$
                  divisionoccurred := true
              else
                  $i := i + 1$
              end if
          end while
          if divisionoccurred = false then
              $r := r + \mathrm{LT}(p)$
              $p := p - \mathrm{LT}(p)$
          end if
      end while
  end procedure
```

Although this seems similar to the process for univariate polynomials, we can observe three main differences:

1. The result of the division process depends on the ordering of the divisors $f_1, \ldots, f_n$.

2. The ordering is not automatically induced as it is in the univariate case.

3. A leading monomial m in f_i is unable to divide the leading term of the polynomial p but does divide another term of p, which is not possible in the univariate case.

For the first difference, let us look at an example from [1]:

Example 6. Let us divide $p = xy^2 + 1$ by $f_1 = xy - 1$ and $f_2 = y^2 - 1$ in the first case, using lex ordering and then in reverse lex ordering:

$$\begin{array}{rrl}
a_1\colon & x + y & \\
a_2\colon & 1 & \qquad r \\
\begin{array}{r} xy - 1 \\ y^2 - 1 \end{array} & \sqrt{x^2y + xy^2 + y^2} & \qquad \underline{\qquad\qquad} \\
& \underline{x^2y - x \qquad\qquad} & \\
& xy^2 + x + y^2 & \\
& \underline{xy^2 - y \qquad\quad} & \\
& \underline{x + y^2 + y} & \qquad \to x \\
& y^2 + y & \\
& \underline{y^2 - 1} & \\
& \underline{y + 1} & \\
& \underline{1} & \qquad \to x + y \\
& 0 & \qquad \to x + y + 1
\end{array}$$

$$\begin{array}{rrl}
a_1\colon & x + 1 & \\
a_2\colon & x & \qquad r \\
\begin{array}{r} y^2 - 1 \\ xy - 1 \end{array} & \sqrt{x^2y + xy^2 + y^2} & \qquad \underline{\qquad\qquad} \\
& \underline{x^2y - x \qquad\qquad} & \\
& xy^2 + x + y^2 & \\
& \underline{xy^2 - x \qquad\quad} & \\
& \underline{2x + y^2} & \qquad \to 2x \\
& y^2 & \\
& y^2 \underline{- 1} & \\
& \underline{1} & \\
& 0 & \qquad \to 2x + 1
\end{array}$$

To address the second difference, we introduced total orderings in the previous section, which has to be decided prior to performing any calculations. For the last one we are going to take a look at the concept of *Gröbner Bases.*

The concept of Gröbner bases was discovered Bruno Buchberger and Heisuke Hironaka in the 1960s independently of each other. Buchberger named the concept after his PhD supervisor Wolfgang Gröbner, while Hironaka used the term *standard bases.* Gröbner bases solve several problems in Mathematics.

Two of them are:

1. The Ideal Membership Problem: Given a polynomial $p \in \mathbf{F}[x_1, \ldots, x_n]$ and an ideal $I = \langle f_1, \ldots, f_s \rangle$, is $p \in I$?
2. The Problem of Solving Polynomial Equations: Find all common solutions in k^n of a system of polynomial equations:

$$f_1(x_1, \ldots, x_n) = \cdots = f_s(x_1, \ldots, x_n) = 0$$

Definition 1.16. Let $I \subset \mathbf{F}[x_1, \ldots, x_n]$ be an ideal other than $\{0\}$. We then:

1. denote by $\mathrm{LT}(I)$, the set of leading terms of elements of I:

$$\mathrm{LT}(I) = \{cx^\alpha \mid \text{there exists } f \in I \text{ with } \mathrm{LT}(f) = cx^\alpha\}.$$

2. denote by $\langle \mathrm{LT}(I) \rangle$ the ideal generated by the elements of $\mathrm{LT}(I)$.

Definition 1.17. Fix a monomial ordering. A finite subset $G = \{g_1, \ldots, g_t\}$ of an ideal I is said to be a Gröbner basis if:

$$\langle \mathrm{LT}(g_1), \ldots, \mathrm{LT}(g_t) \rangle = \langle \mathrm{LT}(I) \rangle.$$

A Gröbner basis can be calculated by using the Buchberger Algorithm: But

Algorithm 2 Buchberger algorithm

```
Require: F = finite subset of $\mathbf{F}[x_1, \ldots, x_n]$
  procedure BUCHBERGER(F)
    $G \leftarrow F$
    $B \leftarrow \{\{g_1, g_2\} \mid g_1, g_2 \in G$ with $g_1 \neq g_2\}$
    while $B \neq \varnothing$ do
      select $\{g_1, g_2\} \in B$
      $B \leftarrow B \backslash \{\{g_1, g_2\}\}$
      $h \leftarrow$ spoly$(g_1, g_2)$
      $h_0 \leftarrow$ some normal form of $h$ modulo $G$
      if $h_0 \neq 0$ then
        $B \leftarrow B \cup \{\{g, h_0\} \mid g \in G\}$
        $G \leftarrow G \cup \{h_0\}$
      end if
    end while
  end procedure
```

a Gröbner base is not necessarily unique. To achieve uniqueness, one needs to reduce the Gröbner basis to a reduced Gröbner basis.

Definition 1.18. A reduced Gröbner basis for a polynomial ideal I is a Gröbner basis G for I such that:

1. $\mathrm{LC}(p) = 1$ for all $p \in G$
2. For all $p \in G$, no monomial of p lies in $\langle \mathrm{LT}(G \setminus \{p\}) \rangle$

This is done with doing a polynomial reduction on the Gröbner basis.

Algorithm 3 Polynomial reduction

```
Require: F = finite subset of $\mathbf{F}[x_1, \ldots, x_n]$
  procedure REDUCTION(P)
      $G \leftarrow F$
      while there is $p \in G$ which is reducible modulo $G \backslash \{p\}$ do
          select $p \in G$ which is reducible modulo $G \backslash \{p\}$
          $G \leftarrow G \backslash \{p\}$
          $h \leftarrow$ some normal form of $h$ modulo $G$
          if $h \neq 0$ then
              $G \leftarrow G \cup \{h\}$
          end if
      end while
      $G \leftarrow \{LC(q)^{-1} \cdot q \mid q \in G\}$
  end procedure
```

This step is repeated until all elements of G satisfy the constraints.

Algorithm 4 Reduced Gröbner Basis

```
Require: G = a Gröbner basis in $\mathbf{F}[x_1, \ldots, x_n]$
  procedure GRÖBNER-REDUCED(G)
      $H \leftarrow \varnothing$;   $F \leftarrow G$
      while $F \neq \varnothing$ do
          select $f_0 \in F$
          $F \leftarrow F \backslash \{f_0\}$
          if $\mathrm{LT}(f) \nmid \mathrm{LT}(f_0)$ for all $f \in F$ AND $\mathrm{LT}(h) \nmid \mathrm{LT}(f_0)$ for all $h \in H$ then
              $H \leftarrow H \cup \{f_0\}$
          end if
      end while
      $H \leftarrow$ REDUCTION($H$)
  end procedure
```

The algorithms presented are from [3] on the pages 203 and 214.

2 Cryptography

Now that we are done with the hard part, we can continue with the other, equally hard, part, namely Cryptography. But before we get there, let us start with a bit of background on the use of Gröbner bases in Cryptography. One of the first papers which warns about the use of Gröbner bases in Cryptography is [4]. The authors explain that there seems to be a misconception about the connection between the ideal membership problem and the computation of Gröbner bases. While Gröbner bases provide a solution to ideal membership problem, they don't give an exclusive solution to the ideal membership problem. There can be easier methods which give a solution that can be used to break a proposed cryptographic system, of which two are mentioned in the paper. The first one makes use of a modified Buchberger algorithm which only allows computations of Gröbner basis elements below a predefined degree. The second involves the use of linear algebra avoiding the user of Gröbner bases totally. Their analysis is focused on schemes with dense polynomials and ends with the conjecture that sparse schemes are even easier to defeat.

One of the proposed sparse schemes is *Polly Cracker*, will be described next.

2.1 Polly Cracker

To describe the cryptographic schemes, we will use the standard notation in which Alice and Bob want to communicate and Eve wants to intercept their communication.

There are several different encryption schemes published under the umbrella of the term *Polly Cracker*. The generalization as presented by Koblitz in [5] can be described in the following way:

Let $G = \{g_1, \ldots, g_n\}$ be a Gröbner basis of an ideal I in the polynomial ring $\mathbf{F}[x_1, \ldots, x_n]$. Let $S \subset \mathbf{F}[x_1, \ldots, x_n]$ be a subset which cannot be reduced further modulo G. This subset S is publicly known, whereas G is kept a secret, as well as the term ordering. An arbitrary message m consists of a linear combination of elements of S.

1. Alice chooses a set $B = \{q_i\}$ of polynomials which are contained in the ideal I. Let J denote the ideal generated by the elements in B.

2. To encrypt his plaintext message, Bob forms α a linear combinations of elements in S with coefficients in $\mathbf{F}$. He then adds α to an element of the ideal J, such that $c = \alpha + \sum h_j q_j, q_j \in B, h_j \in \mathbf{F}$ is the ciphertext he transmitting to Alice.

3. Alice can now decipher the transmission from Bob by reducing $c \mod G$ to receive the plaintext message.

In this work we will lay focus on the more specified scheme described in [5] and [6], which works as follows:

Let $\mathbf{F}_{p^s}[x] := \mathbf{F}_{p^s}[x_1, \ldots, x_n]$ denote be a polynomial ring in n variables over a finite field $\mathbf{F}_{p^s}$.

1. Alice chooses polynomials $q_1, \ldots, q_r \in \mathbf{F}_{p^s}[x]$, which all have some zero $\sigma \in \mathbf{F}_{p^s}$ in common, such that $q_1(\sigma) = \ldots q_r(\sigma) = 0$. Alice then publishes the polynomials, but keeps the common zero σ, a secret.

2. In order for Bob to send a message $\alpha \in \mathbf{F}_{p^s}$ to Alice, Bob selects some elements from the ideal generated by $q_1, \ldots, q_n$ in $\mathbf{F}_{p^s}[x]$. These selected polynomials $h_1, \ldots, h_r$ are contained in $\mathbf{F}_{p^s}[x]$. He then computes $\tilde{c} = \sum_{n=1}^{r} h_i \cdot q_i$ and adds α such that his ciphertext $c = \alpha + \tilde{c}$.

3. Decryption the ciphertext c is a simple evaluation of the transmitted polynomial at the secret zero σ. By construction:

$$c(\sigma) = (\alpha + \tilde{c}(\sigma)) = \alpha + \sum_{n=1}^{r} \underbrace{h_i(\sigma)q_i(\sigma)}_{=0} = \alpha \tag{2}$$

This scheme corresponds to setting $G = \{x_1 - \sigma_1, x_2 - \sigma_2, \ldots, x_n - \sigma_n\}$ and $S = 1$ in the more general scheme, where $\sigma \in \mathbf{F}^n$ is the common zeros. But how could Eve intercept Alice and Bob's communication?

2.2 Linear Algebra Attacks

There are several possible ways to attack the Polly Cracker system using Linear Algebra.

A first version of the *Linear Algebra Attack* takes advantage of the fact that it is also possible for us the recover the message if we find the h_i's, which Bob chooses.

Let us assume that there exists a *characteristic* monomial $m_i := x_1^{\nu_1} \cdot \cdots \cdot x_n^{\nu_n}$ such that m_i only appears in one public polynomial q_i. This can enable the attacker, if h_i which contains such an m_i is not chosen in a suitable way, that this h_i, can be extracted from the ciphertext c easily. A easy example:

Example 7. Assume Alice publishes the following elements in $\mathbf{F}_{53}[x, y, z]$ as her public polynomials and let her secret zero be $\sigma = (4, 10, 32)$.

$$q_1 := x^{13}y^{11}z^{13} + 48$$
$$q_2 := -x^{22}y^{25}z^{22} + 4$$

And Bob sends her the following ciphertext:

$$c := -x^{22}y^{27}z^{23} - 2x^{22}y^{25}z^{22} + x^{15}y^{11}z^{15} + 4x^{13}y^{11}z^{13} + 48x^2z^2 + 4y^2z + 45$$

When looking for example at the term $4x^{15}y^{11}z^{15}$, we see that this term is not divisible by a monomial from q_2, hence it must be a multiple of $x^{13}y^{11}z^{13}$. This leads us to the conclusion that $x^{13}y^{11}z^{13}$ got multiplied by x^2z^2. A similar analysis can be done for the rest of the terms, which in the end reveals that the polynomials h_1, h_2 Bob choose are $x^2z^2 + 4$ and $y^2z + 2$. From this we then can deduce that Alice received 5 as a message from Bob.

In [7] it is noted that this can be avoided by using the same monomials with different coefficients across the public polynomials.

A more advanced version of the Linear Algebra attack is the *Intellegent Linear Algebra Attack*. Assume that m_c is a monomial contained in the cipher text c. Then it is reasonable to assume that there exists a monomial m_h, which Bob chooses, such that $m_c = m_h \cdot m_q$, where m_q is a monomial in the public key of Alice. We can determine a superset $\mathcal{M}$ which contains more than just the monomials h_i. The knowledge of this set reduces the deciphering process to the problem of solving a system of linear equations over the respective polynomial ring.

For $1 \leqslant i \leqslant r$ and $m \in \mathcal{M}$ let A_{im} be the unknown variables. The coefficients in the equation:

$$c = A_0 + \sum_{n=1}^{r} \left(\sum_{m \in \mathcal{M}} A_{im} \cdot m \right) \cdot q_i \tag{3}$$

form a linear system of equations with the unknowns A_{im} and A_0, with A_0 denoting the unknown plaintext. The system is solvable by construction and all solutions give the correct plaintext $A_0 = \alpha$.

2.3 Hidden monomials

In order to strengthen our cryptosystem against the Intelligent Linear Algebra Attack, Lenstra suggested to Koblitz in [5, Chapter 5, §6]the following modification:

> ..., Bob must artfully create at least one monomial d' in his h_i's such that d' times any term in q_i is cancelled in the ciphertext c. This protection is obviously defeated if there exist too few of those monomials or they are easy to guess, ...

This prevents the attacker from forming a valid superset $\mathcal{M}$, hence making it unable to form the correct system of equations.

Example 8. An example immune against the intelligent linear algebra attack over $\mathbf{F}_{79}[x,y]$ with $\sigma = (63,0)$ and the hidden message is 26:

$$\begin{aligned}
q_1 &:= 12x^4y^4 + 22x^7 \\
q_2 &:= x^9y^2 + x^7y^4 + x^3y^7 + x^9 + xy^6 + x^3y + 10 \\
h_1 &:= 65x^{48}y^{39} + 21x^{23}y^{29} + 42 \\
h_2 &:= 10x^{43}y^{41} + 71x^{48}y^{35} + 52x^{30}y^{17} + 53 \\
c &:= 10x^{50}y^{45} + 71x^{57}y^{37} + 10x^{46}y^{48} + 10x^{52}y^{41} + 71x^{51}y^{42} \\
&\quad + 71x^{57}y^{35} + 10x^{44}y^{47} + 71x^{49}y^{41} + 10x^{46}y^{42} + 71x^{51}y^{36} \\
&\quad + 21x^{43}y^{41} - x^{48}y^{35} + 15x^{27}y^{33} + 67x^{30}y^{29} + 52x^{39}y^{19} \\
&\quad + 52x^{37}y^{21} + 52x^{33}y^{24} + 52x^{39}y^{17} + 52x^{31}y^{23} + 52x^{33}y^{18} \\
&\quad + 46x^{30}y^{17} + 53x^9y^2 + 53x^7y^4 + 53x^3y^7 + 53x^9 + 30x^4y^4 \\
&\quad + 55x^7 + 53xy^6 + 53x^3y + 3
\end{aligned}$$

The monomial $x^{48}y^{39}$, leading monomial in h_1, is not contained in the set $\left\{\frac{m_c}{m_q} \mid m_c \in M(c), m_q \in M(q_1) \cup M(q_2)\right\}$.

2.4 Differential Attack

In [7] Hofheinz and Steinwandt present an attack which defeats the approach with hidden monomials. Their attempt to exploit the structure of the ciphertext starts by defining a function Δ, which maps from the polynomial ring $\mathbf{F}_{p^s}[x]$ into the power set $2^{\mathbf{F}_{p^s}[x,x^{-1}]}$ of Laurent Polynomials.

$$\begin{aligned}
\Delta : \qquad \mathbf{F}_{p^s}[x] &\longrightarrow 2^{\mathbf{F}_{p^s}[x,x^{-1}]} \\
\sum_{\nu \in \mathbf{N}_0^n} \gamma_\nu \cdot x^\nu &\longmapsto \left\{\frac{\gamma_\mu}{\gamma_\eta} \cdot x^{\mu-\eta} | \mu \succ \eta, \gamma_\mu \cdot \gamma_\eta \neq 0\right\}
\end{aligned}$$

Remark. In the following, a, b are polynomials in $\mathbf{F}_{p^s}[x]$ and have no monomial in common.

We can deduce the following properties from Δ:

1. $\Delta(a) = \varnothing$ iff a consists of one term or $a = 0$
2. $|\Delta(a)| \leqslant \frac{|M(a)|^2 - |M(a)|}{2}$
3. $\Delta(a) = \Delta(\gamma_\nu x^\nu \cdot a)$
4. $\Delta(a+b) \supseteq \Delta(a) \cup \Delta(b)$

Proof. 1. $\Rightarrow$ Assume that a contains two or more non-zero terms $\gamma_\mu \cdot x^\mu, \gamma_\eta \cdot x^\eta$. Then for $\mu \succ \eta$, the term $\frac{\gamma_\mu}{\gamma_\eta} \cdot x^{\mu-\eta}$ is an element in $\Delta(a)$. $\Leftarrow$ holds trivially.

2. A total ordering is induced by $\succ$, and therefore the terms in a

$$|\Delta(a)| \leqslant \sum_{i=1}^{|\mathrm{M}(a)|-1} i$$

The sum can be evaluated to $(|\mathrm{M}(a)|^2 - |\mathrm{M}(a)|)/2$, hence the result holds.

3. Follows from the definition of Δ, as $\succ$ is a monomial ordering.

4. Follows from the assumption that $\mathrm{M}(a) \cap \mathrm{M}(b) = \varnothing$.

□

Let the polynomial $c = \alpha + \sum_{i=1}^{r} h_i \cdot q_i$ be a ciphertext which Bob sends to Alice.

We can assume that with some luck there exists an $i \in \{1, \ldots, r\}$ such that:

$$\Delta(q_i) \cap \Delta(c) \neq \varnothing.$$

This will be the case if we assume there is some $i \in \{1, \ldots, r\}$ for which there exist terms $\gamma_{\mu_i} x^{\mu_i}$ and $\gamma_{\nu_i} x^{\nu_i}$ in q_i such that the following two conditions hold:

1. $\delta_i := \gamma_{\mu_i} x^{\mu_i} / \gamma_{\eta_i} x^{\eta_i} \in \Delta(q_i) \backslash \left(\bigcup_{i \neq j} \Delta(q_j) \right)$

2. There exists a term $\gamma_{\eta_i} x^{\eta_i}$ in h_i such that the monomials $x^{\eta_i} x^{\mu_i}$ and $x^{\eta_i} x^{\nu_i}$ do not occur in the set of monomials $\mathrm{M}(c - \gamma_{\eta_i} x^{\eta_i} q_j)$.

Then

$$\delta_i := \frac{\gamma_{\eta_i} x^{\eta_i} \cdot \gamma_{\mu_i} x^{\mu_i}}{\gamma_{\eta_i} x^{\eta_i} \cdot \gamma_{\nu_i} x^{\nu_i}} \in \Delta(c) \tag{4}$$

This process does not depend on the occurrence of the monomials x^{η_i} in the ciphertext. If we find some terms $t_1, t_2 \in c$ for which $x^{\mu_i} \mid t_1$ and the quotient t_1/t_2 is equal to an element in $\Delta(q_i)$, it is sufficient to assume that the two previously mentioned conditions hold.

$$t_h := \frac{t_1}{\gamma_{\mu_i} x^{\mu_i}} = \frac{t_2}{\gamma_{\nu_i} x^{\nu_i}} \tag{5}$$

Now the attacker, can replace c with $c' = c - t_h \cdot q_i$ and in this way obtain another valid encryption of the message. Although there is no guarantee that t_h is a term of h_i, a decrease in number of terms of the ciphertext c can be taken as evidence for the correctness of the guess.

In order to decide which potential hidden monomial to use to obtain the best simplified ciphertext, [7] suggests the monomial which results in the smallest simplified ciphertext $c' = c - t_h \cdot q_i$. From this point on, one can try an intelligent linear algebra attack or repeat the above process to further reduce the number of terms in c until a potential message is obtained.

Algorithm 5 Differential Attack

```
Require: $Q$ the public polynomials, $c$ the cipertext
  procedure DifferentialAttack(Q,c)
    $\Delta(c) \leftarrow \{\frac{\gamma_\mu}{\gamma_\nu} \cdot x^{\mu-\nu} \mid \mu \succ \nu,\ \gamma_\mu \cdot \gamma_\nu \neq 0\}$ for each monomial in c
    for $q_i \in Q$ do
      $\Delta(q_i) \leftarrow \{\frac{\gamma_\mu}{\gamma_\nu} \cdot x^{\mu-\nu} \mid \mu \succ \nu,\ \gamma_\mu \cdot \gamma_\nu \neq 0\}$ for each monomial in $q_i$
    end for
    for $\Delta(q_i)$ do
      $M_i \leftarrow \Delta(c) \cap \Delta(q_i)$
    end for
    for $t_1 \in M_i$ do
      Calculate $t_{h_i} \leftarrow \frac{t_1}{\gamma_{\mu_i} x_i^\mu}$
    end for
    for $t_{h_i}$ do
      $c' \leftarrow c - t_{h_i} \cdot q_i$ s.t. $c'$ is smallest
    end for
  end procedure
```

Time complexity

Let us now examine the time complexity for one step of the Differential Attack. As a reference for time complexity we used [8].

Let c be a ciphertext in the polynomial ring $\mathbf{F}_p[x_1, \ldots, x_v]$ with n terms and an upper limit for the degree of each variable of γ. We assume c and the set of $q_i's$ to be ordered. If this was not the case, then ordering the public polynomials only needs to be done once during the whole process, while the simplified message c' might need to be ordered after each step. For further information on the time complexity comparing monomials we refer to [9].

The process of calculating the Δ function, for c and the $q_i's$ is dominated by the calculation of $\Delta(c)$ as we can assume c being much longer than the $q_i's$. So we perform $\frac{n(n-1)}{2}$ times division of coefficients $\mod p$ and v subtractions, each costing $\mathcal{O}\left((\log p)^2\right)$ and $\mathcal{O}\left(v \log \gamma\right)$ bit operations respectively. This leads to a total of $\mathcal{O}\left(n^2\left((\log p)^2 + v \log \gamma\right)\right)$.

The comparison step $\Delta(c) \cap \Delta(q_i)$, depends mostly on the size of $\Delta(c)$, which consists of $\frac{n(n-1)}{2}$ elements. We approximate the comparison step by first sorting the lists and then iterating through them. While sorting is in the range of $\mathcal{O}\left(n^2 \log\left(\frac{n(n-1)}{2}\right)\right)$, iterating should not be more than $\mathcal{O}\left(n^2\right)$, which costs $\mathcal{O}\left(n^2 \log\left(\frac{n(n-1)}{2}\right)\right)$ bit operations in total.

The calculation of t_i has the same cost as calculating the Δ terms, but can be neglected due to it being a much smaller set of terms than $\Delta(c)$.

Comparing the length of the simplified ciphertexts, which are in the range slightly smaller than n can be bounded by $\mathcal{O}(n)$.

Summing up all gives $\mathcal{O}\left(n^2\left((\log p)^2 + v \log \gamma + \log\left(\frac{n(n-1)}{2}\right)\right)\right)$.

Taking into account the suggestions in [5] for number of variables and polynomial degree, we can derive $n \gg v \gg \gamma$, which leaves us with:

$$\mathcal{O}\left(n^2\left((\log p)^2 + \log\left(\frac{n(n-1)}{2}\right)\right)\right).$$

In [10] it is claimed that the Differential Attack does not work when one of the public polynomials consists of just a monomial or two monomials (2-monomial). In their earlier paper [11], their motivation for this is explained:

> [...] F contains monomials or binomials. In this case, a reduction may replace one monomial with another, or even with nothing; encyphering [sic], one can perform long chains of reductions without exponential growth. An eavesdropper can only assume that that every operation has one element in common with the others, but this does not help: every reconstruction has multiple possibilities, and the possibility of long simplification chains entail an exponential reconstruction. [...]

The case in which one public polynomial consists of only a monomial is not strictly applicable in our case, as the decryption process on Alice's side depends on a common zero among all public polynomials. In order to make this work in the setting which [7] and we use, we choose one part of the zero to be zero. The downside for this is that we leak information about the secret key, but this attack is about recovering the message and not about recovering the secret key Alice uses.

2.5 Experiments

In order to run check the conjecture of [10], we needed an implementation of Polly Cracker and the Differential Attack. This was a main part of this work. The procedure for this is the following:

We generate polynomials in several variables which all have a zero in common and sum up powers of them or powers of sums of them in order to create public polynomials q_i. A subset of them is chosen to contain one or more hidden monomials. In the next step, we assign randomly constructed monomials m_i^*'s as hidden monomials. These monomials m_i^*'s get multiplied with their corresponding q_i^*'s giving the terms which we want to erase with the rest of the q_i. Now the $m_i^* q_i^*$'s are divided by q_i's. The negative of the resulting monomials $m_i^x = \frac{m_i^* q_i *}{q_i}$ are part of the h_i's. More random monomials are constructed and added to enlarge the h_i's. Finally we also add a constant to each h_i, in order to help to disguise the message.

To be sure that the generated instance would withstand an Intelligent Linear

Algebra Attack, we checked if the h_i^*'s are contained in the set

$$\left\{ \frac{m_c}{m_q} \mid m_c \in M(c), m_q \in \bigcup_i M(q_i) \right\}.$$

. If so, we save the field size, number of variables, public polynomials and the ciphertext. Next step is the import of the just saved data into the separate instance of the program in which the Differential Attack works on the ciphertext.

For the implementation we used SageMath [12] on a Linux laptop. The code of the implementation can be seen at:

`https://github.com/StroblLund/PollyCracker`

In comparison to [7] who used $\mathbf{F}_2[x]$ for the experiments, we choose $\mathbf{F}_p[x]$ for different values of p and a varying number of variables. We also decided to calculate $\Delta(c)$ explicitly contrary to [7], who just guessed them. It is unclear if [7] had a strict condition on c' such that the number of terms c' < number of terms c or if c' was just chosen in a way such that it is smallest among all possible c'. We decided to experiment with the later. For our experiments, we think it is not too much to assume that the constant part of our ciphertext differs from our message.

An example in which the Differential attack works on a 2-monomial is the following:

Example 9. Let Alice's zero be $\sigma = (40, 1000, 321)$ and $\mathbf{F}_{9973}[x, y, z]$.

$$\begin{aligned}
q_1 :&= x^8z^{10} + y^{10}z \\
q_2 :&= x^2y^{15}z^{15} + x^{13}y^3z^8 + xy^{17}z^4 + x^{10}y^6 + 2617 \\
h_1 :&= 2073x^{22}y^{30}z^{16} + 4557x^{15}y^9z^5 \\
h_2 :&= 7900x^9y^{37}z^9 + 7900x^{28}y^{15}z^{11} + 1322xy^3z^{11} + 5618 \\
c :&= 7900x^{11}y^{52}z^{24} + 7900x^{41}y^{18}z^{19} + 7900x^{10}y^{54}z^{13} + 7900x^{29}y^{32}z^{15} \\
&+ 7900x^{19}y^{43}z^9 + 7900x^{38}y^{21}z^{11} + 271x^9y^{37}z^9 + 271x^{28}y^{15}z^{11} \\
&+ 4557x^{23}y^9z^{15} + 1322x^3y^{18}z^{26} + 4557x^{15}y^{19}z^6 + 1322x^{14}y^6z^{19} \\
&+ 1322x^2y^{20}z^{15} + 5618x^2y^{15}z^{15} + 1322x^{11}y^9z^{11} + 5618x^{13}y^3z^8 \\
&+ 5618xy^{17}z^4 + 5618x^{10}y^6 + 9016xy^3z^{11} + 2130
\end{aligned}$$

You are welcome to check that sent message $m = 26$.

During the experiments it became visible that it may not be the case that we end up with only a constant term after the differential attack halts or loops between two alternating sets of potential hidden monomials. Never the less, in many cases, the constant term in the final simplified ciphertext c' turned out to be the sent message.

Example 10. Let Alice zero for next following example be $\sigma = (9, 0)$ over the polynomial ring $\mathbf{F}_{79}[x, y, z]$

$$\begin{aligned}
q_1 &:= x^8y^7 + x^6y^4 + x^8 - 16 \\
q_2 &:= 6x^6y^9 \\
h_1 &:= 4x^{14}y^{12} - 14x^{12}y^{13} - 27 \\
h_2 &:= -24x^{14}y^{11} - 24x^{12}y^8 - 24x^{14}y^4 + 10x^5y^8 - 11x^6y^4 - 12 \\
m &:= 26 \\
c &:= 4x^{22}y^{19} + 4x^{20}y^{16} + 4x^{22}y^{12} - 19x^{11}y^{17} + 15x^{14}y^{12} \\
&\quad - 27x^8y^7 + 7x^6y^9 - 27x^6y^4 - 27x^8 - 16
\end{aligned}$$

The algorithm finds during the first step the potential monomials: $4x^{14}y^{12}, 52$ due to the terms in $\Delta(c) \cap \Delta(q_1)$. The set $\Delta(c) \cap \Delta(q_2)$ is empty, so there will be no potential terms for reduction with h_2. Reducing the message with the first leads to: $c' := 60x^{11}y^{17} + 52x^8y^7 + 7x^6y^9 + 52x^6y^4 + 52x^8 + 63$. In the next step the intersection only includes 52, which then reduces the ciphertext to: $60x^{11}y^{17} + 7x^6y^9 + 26$

This did not only work for monomials but also for 2-monomials.

Example 11. Like in the previous example $\sigma = (9, 0)$ and $\mathbf{F}_{9973}[x, y, z]$.

$$\begin{aligned}
q_1 &:= x^7y^9 + x^7y^4 + x^5y^5 + x^4y^6 + x^5 + xy - 36 \\
q_2 &:= 29x^5y^4 + 10y^2 \\
h_1 &:= -20x^12y^14 + 32x^15y^10 - 32 \\
h_2 &:= 2x^{19}y^{16} + 2x^{16}y^{18} + 47x^{14}y^{19} \\
&\quad + 47x^{12}y^{15} + 2x^{13}y^{13} + 47x^{12}y^{10} \\
&\quad + 35x^{10}y^7 + 46x^7y^{10} + 31 \\
c &:= -21x^{24}y^{20} - 21x^{21}y^{22} + 32x^{22}y^{19} + 32x^{22}y^{14} + 32x^{20}y^{15} \\
&\quad + 32x^{19}y^{16} - 21x^{18}y^{17} - 4x^{14}y^{21} + 32x^{20}y^{10} - 4x^{12}y^{17} + 32x^{16}y^{11} \\
&\quad - 12x^{15}y^{11} + 33x^{15}y^{10} - 4x^{12}y^{12} + 34x^{10}y^9 - 14x^7y^{12} \\
&\quad - 32x^7y^9 - 32x^7y^4 - 32x^5y^5 - 32x^4y^6 + 30x^5y^4 - 32x^5 - 32xy - 6y^2 - 17
\end{aligned}$$

In the majority of the tested cases, the algorithm was not successful with reducing the ciphertext down to the message, but the constant part of the remaining polynomial was equal to the sent message.

For our experiments, we only relied on the Differential Attack which we ran repetitively. The results for the monomial case were positive.

1. For $p = 11166643$ we generated 100 ciphertexts ranging in length from 315 to 1571 terms in 5 variables. All of the attacks gave the correct constant part in the remaining ciphertext.

2. For $p = 12116604131$ we generated 27 ciphertexts with varying length between 136 and 1206 terms with 10 different variables. Also in this case all simplified c' ended up with the constant which we tried to hide.

The 2-monomial case gave the same results.

1. For $p = 11166643$ we generated 10 ciphertexts ranging in length from 496 to 1141 terms in 10 variables. The majority of the attacks were successful, revealing the correct message α, one ciphertext remained a polynomial with the correct constant value.
2. For $p = 12116604131$ we generated 10 ciphertexts with varying length between 280 and 644 terms in 10 different variables. In nine out of ten cases we could reduce the ciphertext down to the message α, in the remaining case we ended up with a simplified ciphertext whose constant part is the message α.

In order to explain our observation, we have to go back to the point at which Bob encrypts the message. Bob chooses arbitrary polynomials, which, like the public polynomials, need to contain constant parts in order to disguise his message. A public polynomial which is either a monomial or a 2-monomial does not take part in this process.

This lead over to the point at which we started thinking about which possible hidden monomials do actually matter to the attacking process, as during some of the tests, we couldn't reduce the simplified message to it's constant part, but the constant part turned out to be the message turned out to be the message. It seems that mainly the constant monomials matter, as after we stopped finding these, the constant part of the simplified ciphertext was our sent message. So it is sufficient to calculate c' until there are no possible hidden monomials of degree zero?

The observation should also carry over to the more general case in which the message space consists not only of constants but of normal words depending on the ideal. This raises the question if parts of the algorithm presented before can be altered even more to either make it generally possible to decrypt the message or decrease the computational effort.

3 Conclusion

The experimental results point in the direction that the Differential attack by Hofheinz and Steinwandt may also work for the case in which the set of public polynomials contains a monomial or 2-monomial. This would contradict the statement in [10]. Further work has to be done and more examples tested to give more solid evidence that this is the case. It remains open to check if the algorithm can be altered to achieve the result faster.

Bibliography

[1] David Cox, John Little, and Donald O'Shea. *Ideals, Varieties and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Springer, 1991.

[2] M. Brear. *Undergraduate Algebra: A Unified Approach*. Springer Undergraduate Mathematics Series. Springer International Publishing, 2019. ISBN: 9783030140533.

[3] Thomas Becker, Volker Weispfenning, and Heinz Kredel. *Gröbner Bases: A Computational Approach to Commutative Algebra*. Berlin, Heidelberg: Springer-Verlag, 1993. ISBN: 0387979719.

[4] Boo Barkee et al. "Why You Cannot Even Hope to use Gröbner Bases in Public Key Cryptography: An Open Letter to a Scientist Who Failed and a Challenge to Those Who Have Not Yet Failed". In: *Journal of Symbolic Computation* 18.6 (1994), pp. 497–501. ISSN: 0747-7171. DOI: `https://doi.org/10.1006/jsco.1994.1061`.

[5] Neal Koblitz and A. J. Menezes. *Algebraic aspects of cryptography*. Berlin: Springer, 1998.

[6] Michael Fellows and Neal Koblitz. "COMBINATORIAL CRYPTOSYSTEMS GALORE! 1". In: (Jan. 1994). DOI: `10.1090/conm/168/01688`.

[7] Dennis Hofheinz and Rainer Steinwandt. *A Differential Attack on Polly Cracker*. 2002.

[8] S.D. Galbraith. *Mathematics of Public Key Cryptography*. Cambridge University Press, 2012. ISBN: 9781107013926.

[9] Samuel Lundqvist. "Complexity of comparing monomials and two improvements of the BM-algorithm". In: (2008). DOI: `10.48550/ARXIV.0807.2370`. URL: `https://arxiv.org/abs/0807.2370`.

[10] Massimo Caboara, Fabrizio Caruso, and Carlo Traverso. "Lattice Polly Cracker cryptosystems". In: *Journal of Symbolic Computation* 46.5 (2011), pp. 534–549. DOI: `https://doi.org/10.1016/j.jsc.2010.10.004`.

[11] Massimo Caboara, Fabrizio Caruso, and Carlo Traverso. "Gröbner bases for public key cryptography". In: *ISSAC '08*. 2008.

[12] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 9.6)*. `https://www.sagemath.org`. 2022.

Master’s Theses in Mathematical Sciences 2022:E39
ISSN 1404-6342

LUNFMA-3130-2022

Mathematics
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lth.se/