Unreval XL and its variants

Enrico Thomae, Christopher Wolf

Horst Görtz Institute for IT-security
Faculty of Mathematics
Ruhr-University of Bochum, 44780 Bochum, Germany
{enrico.thomae,christopher.wolf}@ruhr-uni-bochum.de

First Version: 2010-11-23

Abstract. Systems of non-linear multivariate equations are at the heart of many cryptographic algorithms, in particular in the public key setting. This paper investigates some algorithms to solve such systems. Usually, computing the Gröbner basis of the corresponding ideal is the best choice in this context. The best known and also most efficient algorithms for this task are F_4 and F_5. Another strategy to solve such systems is called *eXtended Linearization (XL)* from Eurocrypt 2000. For two reasons this is not as popular as Gröbner bases. First it is believed that its running time is worse than F_4 and second it is not as well understood as Gröbner bases. This contribution challenges both.
First we revisit recent results of the analysis of XL by Moh, Diem, Yang *et al.* and connect them into one framework. Thereby we close some gaps in understanding XL. Second we use this knowledge to give a theoretical analysis of one of the most promising XL derivates, called *MutantXL*. Adapting our results on the $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic ($\mathcal{MQ}$) signature scheme Unbalanced Oil and Vinegar (UOV) shows that MutantXL *can* actually lead to more efficient attacks than methods based on Gröbner bases. We confirm in a theoretical way what Buchmann *et al.* observed on the connection between F_4 and MutantXL on the $\mathcal{MQ}$-system Hidden Field Equations (HFE), *i.e.* in some cases MutantXL is faster than F_4, respectively F_5.

Keywords: Multivariate Cryptography, Algebraic Cryptanalysis, eXtended Linearization, XL, MutantXL, Unbalanced Oil and Vinegar Signature Scheme

1 Introduction

This article deals with $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic systems of equations over (small) finite fields. Solving these equations is difficult as they are $\mathcal{NP}$-complete and also hard on average.
In this article, we will concentrate on the so-called 'eXtended Linearization' technique. In a nutshell, XL produces algebraic dependent, but linearly independent equations by multiplying the initial set of equations with all possible combinations of monomials up to a certain degree D. Next, the new system is viewed as a *linear* system of equations, *i.e.* treated as a matrix. When this matrix has a sufficiently high rank, XL succeeds. While this method will work in all practical cases for a high enough degree D, it is thought to be rather inefficient. In particular, Gröbner basis methods such as F_4 and F_5 have been described in the same fashion. Still, algebraic methods gave rise to a number of attacks, in particular on stream ciphers and block ciphers *e.g.* [Cou02, CP02, AK03, ACG+06]. For the first, 'algebraic immunity' has become an accepted design criterion [FM07], while for the latter, it is still unclear if algebraic attacks on real-world ciphers are actually more efficient than previously known methods. However, the methodology has also been applied in the area of hash functions [SKPI07], and coding based crypto systems [FOPT10]. Moreover, as any cryptographic system can be expressed as a system of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equations over a finite field, any major progress in this area could endanger at least some areas of cryptography. Hence, studying the *average* difficulty of $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic systems of equations is important for the security of cryptographic systems. We want to note that the AES seems to be particularly vulnerable to algebraic attacks, although no specific attack is known so far [MR02]. Still, a clarification of the attack complexity of concrete algorithms is beneficial for cryptography as a whole. In particular, a slight variation called *MutantXL* exploits the ideas of XL to the fullest and is hence far more efficient than earlier versions. In this article, we provide a theoretical framework to analyse XL and its derivates, also including *MutantXL*. The theoretical results are backed up with empirical studies. In particular, we were able to derive the central formulae both analytically and empirically.

1.1 Related Work

XL was initially proposed under the name 'relinearization' at Crypto 1999, and then renamed 'eXtended Linearization' one year later [KS99, CKPS00]. The main observation was that overdetermined systems of equations, *i.e.* systems with more equations than variables could be solved surprisingly easy using the linearization technique. The underdetermined case (more variables than equations) was tackled in [CGMT02]. In all cases, systems of equations are interpreted as matrix-vector equations and the aim is to find a matrix with as many (linearly independent) rows as columns. To this aim, the initial set of equations is *expanded* by generating algebraically trivial, but nevertheless valid and linearly independent equations. The final step consists of treating all monomials as independent variables and then solving a purely linear system of equations. Soon it was pointed out that the method was already known and performed for a small number of variables by geometers [Moh00]. Using it with much larger systems greatly helped to develop its understanding. Unfortunately the initial papers did not provide a deep analysis of the method and many claims showed to be overly ambiguous. At least since Courtois and Pieprzyk claimed to have broken AES [CP02] using an XL derivate called XSL and were disproved by Cid and Leurent [CL05] only a few years later, the community of cryptographers became increasingly reserved against this method. But thanks to Moh [Moh00], Diem [Die04], Yang and Chen [YC04a] and others, XL and variants are understood quit well today.

A second line of research are Gröbner bases. They use a more symbolic approach and eliminate monomials from the set of equations. To this aim, pairs of equations are formed and (hopefully) monomials eliminated. However, in most cases the computation is in vain as no useful elimination occurs. Since the algorithm F_4 [Fau99], there is a strong connection with linear algebra, too: In contrast to deal with *pairs* of equations, F_4 selects whole sets and tries to minimise the amount of useless computations by treating them in matrix-fashion. Its successor F_5 uses some even cleverer book-keeping to bring down the number of useless computations even further [Fau02b]. It is considered the fastest algorithm to compute Gröbner bases. And in fact, F_5 and its variants have an impressive track record in bringing down cryptographic systems and challenges [Fau02a, Fau03a, Fau03b, FJ03, FA03, BFP09, FOPT10].

A natural question to ask is whether XL or Gröbner are the preferred choice for cryptographic problems. Until now, the situation was quite clear: At Asiacrypt 2004 it was shown that XL actually is a sub-case of Gröbner algorithms and that we hence can expect that Gröbner algorithms are always faster than XL [AFI$^+$04, Die04].

A possible testbed for this question is the 'Unbalanced Oil and Vinegar scheme': In 1997 Patarin designed a new signature scheme called 'Oil and Vinegar' [Pat97], based on $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equations. After Kipnis and Shamir broke the balanced case in [KS98] the 'Unbalanced Oil and Vinegar' signature scheme, short UOV, was proposed [KPG99]. Even if most of the proposed schemes of the class of multivariate cryptosystems, like MIA, HFE, SFlash are broken in most of their variants, UOV is still believed to be secure. We can say that UOV is one of the most popular multivariate cryptosystem. Even newer schemes like Rainbow or enhanced TTS use the idea of UOV as trapdoor [DS05, YC05]. A study of the security of UOV was published by Braeken, Wolf and Preneel in 2005 [BWP05]. The best known attack against UOV until now uses Gröbner bases and is described in [BFP09]. In a nutshell, they use ordinary Gröbner basis computation, but guess some variables beforehand. Therefore, they either introduce contradictions in the system of equations, or they solve a system in less variables.

1.2 Organisation and Achievement

The contributions of this paper are manifold. *First*, we start by revisiting the well known technique of relinearization, introduced by Kipnis and Shamir at Crypto 1999 [KS99] and show in an easy way, that it is a subcase of XL. This was already hinted by Courtois *et al.* in [CKPS00], but not as clear and formal as one would deem necessary.
Second, we improve the constant ϵ in the ratio $m \geq \epsilon n^2$ for the number of variables n and the number of equations m from the initial value of $\epsilon = \frac{1}{10}$ [KS99] to $\frac{1}{12}$ for the corresponding XL of degree 2, therefore showing that far more pairs (n, m) are solvable with only moderate workload than previously suggested. In particular, this result is obtained using analytic methods, not empirical ones. Still, we have verified the theory empirically and found both in sync.
Third, we clarify the relationship between XL with homogeneous and inhomogeneous input. While the difference is subtle in most cases it becomes important for analysing MutantXL. We do so both by analytical and empirical methods.
Fourth, we show that certain sets of parameters for UOV get in reach for an improved version of MutantXL. These parameter sets were previously out of reach, in particular for algorithms using Gröbner bases techniques such as F_5.
Fifth, this raises the question if the cryptographic community was right in condemning XL for all possible application domains. While empirical evidence suggested already previously that this might be the case, we give a clear and theoretically sound analysis why this might be the case.
This paper starts with introducing some notation and the UOV system (section 1). After this, relinearization and XL are introduced and analysed in section 2. Based on this, we deepen our analysis of XL, using both theoretical and empirical methods (section 3). Variants of XL are introduced in section 4 and used to cryptanalyse UOV. Conclusions are given in section 5. Further results on the complexity of F_5, XL, and MutantXL can be found in the appendix.

1.3 Notation

Solving non-linear systems of m equations and n unknowns is a difficult problem in general. Restricting to the seemingly *easy* case of degree 2 equations is still difficult. Actually this problem is also known as $\mathcal{MQ}$-problem which is proven to be NP-hard [GJ79].

Let $P : \mathbb{F}_q^n \to \mathbb{F}_q^m$ be an $\mathcal{MQ}$ system of the form

$$\begin{aligned} p^{(1)}(x_1, \ldots, x_n) &= 0 \\ p^{(2)}(x_1, \ldots, x_n) &= 0 \\ &\vdots \\ p^{(m)}(x_1, \ldots, x_n) &= 0, \end{aligned} \tag{1}$$

with

$$p^{(k)}(x_1, \ldots, x_n) := \sum_{1 \le i \le j \le n} \gamma_{ij}^{(k)} x_i x_j + \sum_{1 \le i \le n} \beta_i^{(k)} x_i + \alpha^{(k)}. \tag{2}$$

We call equation $p^{(k)} = 0$ with $p^{(k)}$ defined by (2) inhomogeneous. The homogeneous case consists only of quadratic terms and is thus defined by

$$p^{(k)}(x_1, \ldots, x_n) := \sum_{1 \le i \le j \le n} \gamma_{ij}^{(k)} x_i x_j. \tag{3}$$

We need the classification into homogeneous and inhomogeneous later on, because results are different and it is not always easy to see that they are equal after transforming an inhomogeneous system in a homogeneous one.
Let $\pi^{(k)}$ be the coefficient vector of $p^{(k)}(x_1, \ldots, x_n)$ in lexicographic order, *i.e.*

$$\pi^{(k)} = (\gamma_{11}^{(k)}, \gamma_{12}^{(k)}, \ldots, \gamma_{1n}^{(k)}, \gamma_{22}^{(k)}, \gamma_{23}^{(k)}, \ldots, \gamma_{nn}^{(k)}, \beta_1^{(k)}, \ldots, \beta_n^{(k)}, \alpha^{(k)}).$$

Let Π be the corresponding coefficient matrix

$$\Pi := \begin{pmatrix} \pi^{(1)} \\ \vdots \\ \pi^{(m)} \end{pmatrix}.$$

Note that the problem of solving non-linear equations becomes easier if m exceeds n. In a sense, each equation encodes information about the solution vector $(x_1, \ldots, x_n) \in \mathbb{F}^n$. Obviously, having more information will guide the equation solver to find this solution—as long as the equation is independent from the previously known ones. The naive algorithm is to solve (1) by linearization, *i.e.* to substitute every monomial in $p^{(k)}$ by a new variable and to solve the obtained linear system of equations Π with Gaussian elimination. This will lead to the correct solution if we have $m \ge \frac{n(n+1)}{2} + n$ linearly independent equations, *i.e.* if the number of linearly independent equations is equal to the number of monomials. With the technique of relinearization, introduced in [KS99], we can solve P (asymptotically) if we have $m \ge 0.09175 \cdot n^2$ linearly independent equations. Lowering the trivial factor of $\frac{1}{2}$ to roughly $\frac{1}{10}$ was a big leap. We are able to further improve this to a factor of $\frac{1}{12}$ in the inhomogeneous case of XL (Degree 2), cf. Section 3.1.

1.4 Unbalanced Oil and Vinegar

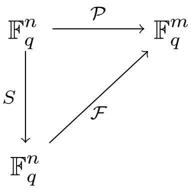
The public key in UOV is a vector $\mathcal{P} \in \mathcal{MQ}(\mathbb{F}^n, \mathbb{F}^m)$ of multivariate quadratic polynomials defined in (2)

$$\mathcal{P} := \begin{pmatrix} p^{(1)}(x_1, \ldots, x_n) \\ \vdots \\ p^{(m)}(x_1, \ldots, x_n) \end{pmatrix}.$$

Denote the number of oil variables by $o \in \mathbb{N}$, the number of vinegar variables by $v \in \mathbb{N}$ and set $n := o + v$. Let $V := \{1, \ldots, v\}$ and $O := \{v + 1, \ldots, n\}$ denote the sets of indices of vinegar and oil variables. The private key $\mathcal{F} := (f^{(1)}(u), \ldots, f^{(m)}(u))$ is defined by

$$f^{(k)}(u) := \sum_{i \in V, j \in O} \gamma_{ij}^{(k)} u_i u_j + \sum_{i,j \in V, i \leq j} \gamma_{ij}^{(k)} u_i u_j + \sum_{i \in V \cup O} \beta_{ij}^{(k)} u_i + \alpha^{(k)}. \quad (4)$$

It is important for finding a preimage that the variables in $f^{(k)}$ are not completely mixed, *i.e.* oil variables are only multiplied by vinegar variables and never by oil variables. This construction leads to an easy way to invert $f^{(k)}$. If we assign arbitrary values to the vinegar variables and if we set $m = o$ we obtain a system of o linear equations in o variables. It is very likely that this provides a solution. If not we try again. In the public key $\mathcal{P}$, the central map $\mathcal{F}$ is hidden by composing it with a linear map $S : \mathbb{F}_q^n \to \mathbb{F}_q^n$, *i.e.* $\mathcal{P} := \mathcal{F} \circ S$.



Typical values for UOV are field-size $q = 256$, number of variables $n = 78$, and number of equations $m = 26$ [BFP09]. We will use these to compare MutantXL with F_5 in section 4.

2 Relinearization vs XL

2.1 Relinearization

In [KS99] Kipnis and Shamir used relinearization to cryptanalyse HFE. The idea is very clear and simple. Given a random $\mathcal{MQ}$-system P we first linearise, *i.e.* introduce new variables $y_k := x_i x_j$. For simplicity of the analysis we assume P to be homogeneous. That means the number of unknowns $x_i x_j$ is $\binom{n+1}{2} = \frac{n(n+1)}{2}$. Notice that this is no restriction for asymptotic analysis and that we can express any non-homogeneous system in form of a homogeneous system by introducing one more variable. For random systems it is very likely that all of the m equations are linearly independent, cf. Section 3.1. This underdetermined system of linear equations is solved by Gaussian elimination, see figure 1 for illustration. As we can see, we obtain an exponential number $q^{\frac{n(n+1)}{2} - m}$ of parasitic solutions in $y_{m+1}, \ldots, y_{\frac{n(n+1)}{2}}$.
After linearization both $y_1 := x_1 x_1$ and $y_2 := x_1 x_2$ are two independent linear variables. But from an algebraic point of view this is not true as y_1 as well as y_2 depend on x_1. Relinearization exploits this structure to eliminate parasitic

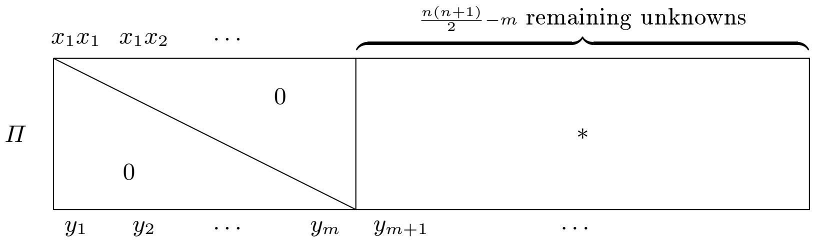


Fig. 1. Coefficient Matrix Π of P after Gaussian elimination

solutions, *i.e.* to fix the remaining variables $y_{m+1}, \dots, y_{\frac{n(n+1)}{2}}$ implicitly via new equations. The following equations are trivially true and linearly independent for some $y_a = x_i x_j$:

$$\begin{aligned} & x_i x_j x_k x_l = x_i x_k x_j x_l = x_i x_l x_j x_k \\ \Leftrightarrow \quad & y_{i_1} y_{i_2} = \quad y_{i_3} y_{i_4} \quad = y_{i_5} y_{i_6} \end{aligned} \tag{5}$$

Kipnis and Shamir required $i < j < k < l$ in the above equation. There are $\binom{n}{4}$ possibilities for $x_i x_j x_k x_l$ and thus we get $2\binom{n}{4}$ linear independent equations by (5). If this is larger than the number of unknowns in the remaining y's we are done and can solve the system, *i.e.* for

$$2\binom{n}{4} \geq \binom{\frac{n(n+1)}{2} - m + 1}{2}.$$

For m in the same magnitude as n this is not the case in general. For $m = \varepsilon n^2$ and only considering the n^4 part, we get the following asymptotic equation

$$0 \leq -\varepsilon^2 + \varepsilon - \frac{1}{12}$$

and hence $\varepsilon \geq 0.09175$.

Note, for inhomogeneous equations the overall analysis is the same but with a bigger number of unknowns. By

$$2\binom{n}{4} \geq \binom{\frac{n(n+1)}{2} + n - m + 1}{2},$$

we obtain the same asymptotic result. But later in the exact analysis we will need to distinguish between these two cases, as relinearization in the homogeneous case will be exactly the same as XL of degree 2.

The idea of XL (of degree 2) is simpler but not as easy to analyse. We multiply the coefficient matrix Π shown in figure 1 by every quadratic monomial $x_i x_j$

with $i \leq j$ and $i,j \in \{1,\dots,n\}$. This way we obtain $m\binom{n+1}{2}$ equations in $\binom{n+3}{4}$ monomials of degree 4. For $m = \varepsilon n^2$ the number of equations is asymptotically larger than the number of monomials for $\varepsilon \geq \frac{1}{12}$. The crucial question is if all produced equations are linearly independent. This question was not paid much attention by Courtois *et al.* in [CKPS00]. We will look at this in section 3. First let us define the XL algorithm in a rigorous way.

2.2 The XL algorithm

Note that each $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic equation can be rewritten into a $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic polynomial $p^{(k)}$ and the (implicit) equation $p^{(k)} = 0$. Hence, we will only concentrate on polynomials in the remainder of this text.

Definition 1. *Let $P^{\boldsymbol{inh}} := \{p^{(k)} \mid 1 \leq k \leq m\}$ be the set of inhomogeneous quadratic polynomials p as defined in (2) and $P^{\boldsymbol{hom}} := \{p^{(k)} \mid 1 \leq k \leq m\}$ the set of homogeneous quadratic polynomials p defined in (3). We define the set of all monomials of degree D by*

$$\mathtt{Mon}_D := \{\prod_{j=1}^{D} x_{i_j} \mid 1 \leq i_1 \leq i_2 \leq \ldots \leq i_D \leq n\}.$$

Multiplying $P^{\boldsymbol{inh}}$ by all monomials of degree D is described by the set

$$\mathtt{Blow}_D^{\boldsymbol{inh}} := \{ab \mid a \in \mathtt{Mon}_D \text{ and } b \in P^{\boldsymbol{inh}}\}.$$

The set $\mathtt{Blow}_D^{\boldsymbol{hom}}$ is defined analogous. The following set defines what we use as XL algorithm of degree D.

$$\mathtt{XL}_D^{\boldsymbol{inh}} := \bigcup_{i=1}^{D} \mathtt{Blow}_i^{\boldsymbol{inh}} \cup P^{\boldsymbol{inh}}.$$

Some authors also speak of XL of degree D meaning $\mathtt{XL}_{D-2}^{\mathtt{inh}}$. In this case D means the highest degree of all polynomials used for multiplication and not the degree of the extension. In our opinion, the latter is more general. Notice that defining $\mathtt{XL}_D^{\mathtt{hom}}$ analogous would not make any sense, because $\mathtt{Blow}_D^{\mathtt{hom}}$ only produces monomials of degree $D+2$ and thus there is no need to use the sets of lower degrees.

Definition 2 (XL algorithm). *First we generate $\mathtt{XL}_D^{\boldsymbol{inh}}$ and check if the number of linearly independent equations I is equal to the number of produced monomials T subtracted by $D+2$. In this case we linearise the system and solve it by Gaussian elimination. Notice, if $T - I \leq D+2$ we can choose the order of the monomials such that we obtain a univariate equation after linearization, which can be solved,* e.g. *by Berlekamp's algorithm. If $T - I > D+2$ we set $D := D+1$ and try again.*

2.3 Complexity Considerations

We discuss complexity considerations for algorithms of the XL-type. With minor modifications, they also apply to modern Gröbner basis algorithms. In both cases, we deal with a large matrix $\Pi \in \mathbb{F}^{M \times N}$ over a ground field $\mathbb{F}$ and M rows and N columns. Usually, $\mathbb{F}$ is very small (8 or 16 bit), so we can exclude it from our analysis. The number of columns N depends on the number of unknowns and is roughly $\binom{n+D+2}{D+2}$. It may vary a bit depending on the version of XL chosen. The number of rows M must be at least as big as the number of columns N. Otherwise, our linear system does not permit a unique solution. The overall complexity is therefore determined by 1.) building the matrix Π and 2.) finding a solution for the underlying system. We start with the first step: Here, we start with a dense polynomial $p \in P$ and multiply it with a single monomial $a \in \mathtt{Mon}_D$. The overall workload is therefore

$$|\mathtt{Mon}_D|\binom{n+2}{2}$$

multiplications and memory access for building the matrix Π. Note that each row in Π has $\binom{n+D+2}{D+2}$ but only $\binom{n+2}{2}$ non-zero elements. It is therefore extremely sparse. This can be exploited as we do not need to store $M \cdot N$ but only $M\binom{n+2}{2}$ elements.
Secondly, we consider solving the linear equation depending on the coefficient matrix Π. In a nutshell, we can upper-bound this by $O(M^\omega)$ for $2 \leq \omega \leq 3$ in general and $\omega = 2 + \epsilon$ for sparse equations. As we saw above, this is the case for XL. If we can avoid linear dependent equations in the intermediate steps, we have $M = N$ and can therefore bring down complexity. We see that the complexity of (2) clearly outperforms (1). Therefore, it is enough to consider M^2 in the sequel.

2.4 Relinearization as subcase of XL

Moh analysed relinearization for $i \leq j \leq k \leq l$ [Moh00]. Asymptotically he obtains the same result as Kipnis and Shamir. To compare relinearization with XL we also need the smaller terms and therefore we use the exact analysis by Moh. For $i \leq j \leq k \leq l$ we get

$$2\binom{n}{4} + \frac{n(n-1)(n-2)}{2} + \frac{n(n-1)}{2} = 2\binom{n}{4} + 3\binom{n}{3} + \binom{n}{2}$$

equations by relinearization, instead of $2\binom{n}{4}$ in the case $i < j < k < l$. Figure 2 illustrate the given situation. To allow to distinguish cases we assume m to be of the form $\sum_{i=0}^{\gamma-1}(n-i) = \gamma n + \frac{\gamma-\gamma^2}{2}$ for $\gamma = \varepsilon n$ and thus $m = (\varepsilon - \frac{\varepsilon^2}{2})n^2 + \frac{\varepsilon}{2}n$. Through this $y_{m+1} = x_{\gamma+1}x_{\gamma+1}$ holds and due to the graded lexicographical order for all indices of not specified monomials x_ix_j in the $*$ block, see figure 2,

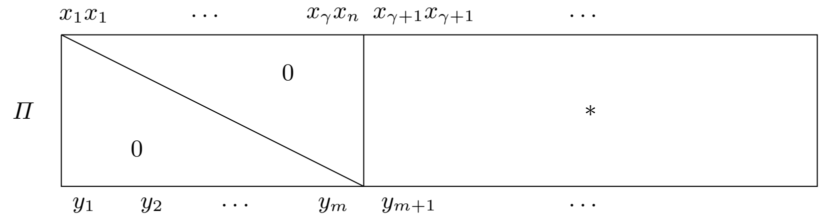


Fig. 2. Coefficient Matrix Π of P after Gaussian elimination

it holds $i, j > \gamma$. This allows us to analyse $x_i x_j x_k x_l$ in the two cases $i \leq \gamma$ and $i > \gamma$.

We want to show that multiplying by special monomials is exact the same as relinearization. Due to the choice of m we can distinguish two cases.

Case 1, $i \leq \gamma$. For $i \leq \gamma$ relinearization behaves exactly as XL.

$$\underbrace{x_i x_j}_{y_{i_1}} x_k x_l = \underbrace{x_i x_k}_{y_{i_2}} x_j x_l = \underbrace{x_i x_j}_{y_{i_3}} x_j x_k \text{ with } i_1, i_2, i_3 \in \{1, \ldots, m\} \tag{6}$$

Equations (6) used by relinearization can be produced in XL by multiplying the row of y_{i_1} by $x_k x_l$.

Case 1.1, $i < j < k < l$. There are $\sum_{i=1}^{\gamma} \binom{n-i}{3}$ possibilities for $x_i x_j x_k x_l$, as well as for $x_i x_k x_j x_l$ and $x_i x_l x_j x_k$ for $i < j < k < l$. So we produce $3 \sum_{i=1}^{\gamma} \binom{n-i}{3}$ equations with XL by multiplying y_i by $x_k x_l$. But we also produce $\sum_{i=1}^{\gamma} \binom{n-i}{3}$ new monomials containing variables x_i with $i \leq \gamma$ and so the number of remaining new equations is $2 \cdot \sum_{i=1}^{\gamma} \binom{n-i}{3}$.

Case 1.2, ($j = 1$ and $k \neq l$) or ($j = k$ and $i \neq l$) or ($k = l$ and $i \neq j$). In the case of two equal and two different indices we have the following 3 possibilities of monomials: $x_i x_i x_k x_l$, $x_i x_j x_j x_l$ and $x_i x_j x_k x_k$. Any of them produces 3 equations $x_i x_i x_k x_l = x_i x_k x_i x_l = x_i x_l x_i x_k$. Notice that the last equality is not used by relinearization, because it is trivial. So we ignore them too. Since $x_i x_i x_k x_l$ introduce a new monomial, only $x_i x_k x_i x_l$ gives us a new equation. So we have $3 \cdot \sum_{i=1}^{\gamma} \binom{n-i}{2}$ new equations in total.

Case 1.3, $i = j$ and $k = l$ and $i \neq k$. In this case relinearization uses $x_i x_i x_k x_k = x_i x_k x_i x_k$. The left monomial produces new monomials in XL and the right monomial produces $\sum_{i=1}^{\gamma}(n-i)$ new equations. To sum up all cases, we produced

$$2\cdot\sum_{i=1}^{\gamma}\binom{n-i}{3}+3\cdot\sum_{i=1}^{\gamma}\binom{n-i}{2}+\sum_{i=1}^{\gamma}(n-i)$$

new equations by adapting relinearization to XL. Notice that we produced more equations than this, but used them to eliminate the newly introduced monomials of degree 4 with variables x_i and $i\leq\gamma$. So the number of unknowns in XL is only the number of degree 4 monomials containing variables x_i with $i>\gamma$, *i.e.* $\binom{n-\gamma+3}{4}$.

Case 2, $i>\gamma$. For $i>\gamma$ relinearization uses the equations

$$x_ix_jx_kx_l=x_ix_kx_jx_l=x_ix_lx_jx_k.$$

This equations cannot be produced by XL, because they are trivially true. The difference between both methods is that relinearization produce more variables after the second linearization step and XL does not. So we do not need these equations for XL because they are only needed in relinearization to eliminate variables we do not have in XL.

The following equations sum up the number of unknowns and equations in both methods. The left terms are the number of unknowns and the right terms are the number of equations.

Relinearization:

$$\frac{(\binom{n+1}{2}-m)(\binom{n+1}{2}-m+1)}{2}\leq 2\binom{n}{4}+3\binom{n}{3}+\binom{n}{2}$$

$$\uparrow\qquad\qquad\qquad\qquad\uparrow$$

$$\Delta_1\qquad\qquad\qquad\qquad\Delta_2$$

$$\downarrow\qquad\qquad\qquad\qquad\downarrow$$

XL:

$$\binom{n-\gamma+3}{4}\leq 2\cdot\sum_{i=1}^{\gamma}\binom{n-i}{3}+3\cdot\sum_{i=1}^{\gamma}\binom{n-i}{2}+\sum_{i=1}^{\gamma}(n-i)$$

To show that both are equal, we have to show that the difference Δ_1 between the left terms is equal to the difference Δ_2 of the right terms. We us $m=\gamma n+\frac{\gamma-\gamma^2}{2}$ $(*)$ and the following equality for $k\in\mathbb{N}_{>0}$

$$\binom{n}{k}-\binom{n-\gamma}{k}=\sum_{i=1}^{\gamma}\binom{n-i}{k-1}$$

We get

$$\begin{aligned}
\Delta_1 &= \frac{(\binom{n+1}{2} - m)(\binom{n+1}{2} - m + 1)}{2} - \binom{n-\gamma+3}{4} \\
&\overset{(*)}{=} 2\binom{n-\gamma}{4} + 3\binom{n-\gamma}{3} + \binom{n-\gamma}{2} \\
&= 2\left(\binom{n-\gamma}{4} - \binom{n}{4} + \binom{n}{4}\right) \\
&\quad +3\left(\binom{n-\gamma}{3} - \binom{n}{3} + \binom{n}{3}\right) \\
&\quad +\binom{n-\gamma}{2} - \binom{n}{2} + \binom{n}{2} \\
&= 2\binom{n}{4} + 3\binom{n}{3} + \binom{n}{2} \\
&\quad -2 \cdot \sum_{i=1}^{\gamma} \binom{n-i}{3} - 3 \cdot \sum_{i=1}^{\gamma} \binom{n-i}{2} - \sum_{i=1}^{\gamma} (n-i) \\
&= \Delta_2 \qquad \square
\end{aligned}$$

To conclude, if we use the XL method and multiply not by all quadratic monomials, but by special ones we do the same as relinearization does, and thus relinearization is a subcase of XL. Now we want to show that it is equal in the homogeneous case of degree two.

Relinearization is equal to $\texttt{Blow}_2^{\texttt{hom}}$

In section 3.1 we will show that the number of linearly independent equations produced by $\texttt{Blow}_2^{\texttt{hom}}$ is $m\binom{n+1}{2} - \binom{m}{2}$. Using this we can analyse if XL outperforms relinearization or not. In the homogeneous case the following must hold for $\texttt{Blow}_2^{\texttt{hom}}$ to obtain a solution.

$$m\binom{n+1}{2} - \binom{m}{2} - \binom{n+3}{4} \geq -D - 2 \tag{7}$$

The following must hold for relinearization to obtain a solution.

$$2\binom{n}{4} + 3\binom{n}{3} + \binom{n}{2} - \binom{\frac{(n+1)n}{2} - m + 1}{2} \geq -D - 2 \tag{8}$$

Because of following equality, inequations (7) and (8) are equal.

$$\begin{aligned}
&m\binom{n+1}{2} - \binom{m}{2} - \binom{n+3}{4} \\
&= \frac{n^4}{24} + \frac{n^3}{4} - \frac{n^2 m}{2} + \frac{11n^2}{24} - \frac{nm}{2} + \frac{n}{4} + \frac{m^2}{2} - \frac{m}{2} \\
&= 2\binom{n}{4} + 3\binom{n}{3} + \binom{n}{2} - \binom{\frac{(n+1)n}{2} - m + 1}{2}
\end{aligned}$$

In the inhomogeneous case, $\texttt{Blow}_2^{\texttt{inh}}$ is slightly better than relinearization. As depicted in section 3.2 table 5 we get a factor of $\frac{1}{12}$ instead of 0.09175 in the asymptotic analysis. We can also derive this from the inequations above. If we homogenise the inhomogeneous system we have to substitute n by $(n+1)$ in inequation (7). Relinearization does not depend on the question whether equations are homogeneous or not, *i.e.* inequation (8) stays the same and thus both are not longer equal.

3 Analysis of XL

3.1 The number of linearly independent equations

The crucial point by using XL is to determine the number of linearly independent equations produced by $\texttt{Blow}_D$ or $\texttt{XL}_D^{\texttt{inh}}$. This is needed to calculate D and therefore implies the complexity of the whole algorithm. For random equation systems we will revisit the formulas derived theoretically by Moh [Moh00], Yang and Chen [YC04a] or by experiments for D between 0 and 5 over $\mathbb{F}_2$ by Courtois and Patarin [CP03]. Notice that the formulas are independent of the ground field $\mathbb{F}_q$. The field has only impact on the number of unknowns if we can reduce them by the field equations $x^q - x$. This is only the case for $D \geq q$. Our experiments were performed independently of previously known results. In addition, we also considered the homogeneous case.

Table 1. Number of linearly independent equations produced by $\texttt{Blow}_D^{\texttt{hom}}$, experimentally derived.

D	Number of linearly independent equations
0	m
1	mn
2	$m\binom{n+1}{2} - \binom{m}{2}$
3	$m\binom{n+2}{3} - \binom{m}{2}n$
4	$m\binom{n+3}{4} - \binom{m}{2}\binom{n+1}{2} + \binom{m}{3}$
5	$m\binom{n+4}{5} - \binom{m}{2}\binom{n+2}{3} + \binom{m}{3}n$

Experimental setup and connection between homogeneous and inhomogeneous case
As you can see in table 1 and 2 the formulas of $\texttt{Blow}_D^{\texttt{hom}}$ and $\texttt{Blow}_D^{\texttt{inh}}$ are slightly different. These equations were obtained experimentally by a total of several 10,000 experiments and later verified theoretically. All experiments were performed on a Intel Xeon X33502.66GHz (Quadcore) with 8 GB of RAM using only one core and the software system Magma V2.16-1 [MAG]. Parameters were running for various tuples (n, m, D) in the range $3 \leq n \leq 15$, $3 \leq m \leq 50$,

Table 2. Number of linearly independent equations produced by $\mathtt{Blow}_D^{\mathtt{inh}}$, experimentally derived.

D	Number of linearly independent equations
0	m
1	mn
2	$m\binom{n+1}{2}$
3	$m\binom{n+2}{3} - \binom{m-1}{3}$
4	$m\binom{n+3}{4} - \binom{m-1}{3}n + \binom{m-1}{4}$
5	$m\binom{n+4}{5} - \binom{m-1}{3}\binom{n+1}{2} + \binom{m-1}{4}n - \binom{m-1}{5} + \binom{m-1}{4}$
6	$m\binom{n+5}{6} - \binom{m-1}{3}\binom{n+2}{3} + \binom{m-1}{4}\binom{n+1}{2} - \binom{m-1}{5}n + \binom{m-1}{4}n + \binom{m-1}{6} - \binom{m-1}{5}$

Table 3. Number of linearly independent equations produced by $\mathtt{XL}_D^{\mathtt{inh}}$, experimentally derived.

D	Number of linearly independent equations
0	m
1	$m + mn$
2	$m + mn + m\binom{n+1}{2} - \binom{m}{2}$
3	$m\binom{n+3}{3} - \binom{m}{2}(n+1)$
4	$m\binom{n+4}{4} - \binom{m}{2}\binom{n+2}{2} + \binom{m}{3}$
5	$m\binom{n+5}{5} - \binom{m}{2}\binom{n+3}{3} + \binom{m}{3}(n+1)$

$1 \leq D \leq 8$. First, all data-points were fitted with an automated polynomial fitter (multivariate equations in two or three variables). In a second, semi-automated step, these polynomials were expressed in form of binomials.

Hence we showed experimentally that we obtain

$$m + mn + m\binom{n+1}{2} - \binom{m}{2}$$

linearly independent equations for $\mathtt{XL}_2^{\mathtt{inh}}$, i.e if we join the $\mathtt{Blow}_i^{\mathtt{inh}}$ for $i = 0 \ldots 2$ there are new linear dependencies. And thus we get the same result by homogenising an inhomogeneous system and using $\mathtt{Blow}_2^{\mathtt{hom}}$ and by using $\mathtt{XL}_2^{\mathtt{inh}}$ itself. Note that we have to substitute n by $n+1$ in the formula of $\mathtt{Blow}_2^{\mathtt{hom}}$ and that the number of variables is $\binom{n+4}{4} - 1$ because we know x_{n+1}^4 by the choice of $x_{n+1} = 1$ for homogenisation. Thus we get the following.

$$
\begin{aligned}
&\texttt{Blow}_2^{\texttt{hom}}\\
&: m\binom{n+2}{2} - \binom{m}{2} - \binom{n+4}{4} + 1\\
&= m\binom{n+1}{2} + m\binom{n+1}{1} - \binom{m}{2} - \binom{n+3}{4} - \binom{n+3}{3} + 1\\
&= m + mn + m\binom{n+1}{2} - \binom{m}{2} - \binom{n+3}{4} - \binom{n+2}{3} - \binom{n+2}{2} + 1\\
&= m + mn + m\binom{n+1}{2} - \binom{m}{2} - \binom{n+3}{4} - \binom{n+2}{3} - \binom{n+1}{2} - n\\
&: \texttt{XL}_2^{\texttt{inh}}
\end{aligned}
$$

The above is also true for arbitrary D. If you choose m high enough, you may wonder if the number of linearly independent equations for inhomogeneous systems becomes less than 0. Note that all equations first reach the maximum number of linear independent equations, *i.e.* $\binom{n+D+1}{D+2} + \binom{n+D}{D+1} + \binom{n+D-1}{D} - D - 2$, the number of unknowns for $\texttt{Blow}_D^{\texttt{inh}}$ subtracted by $D + 2$. If the number of equations is higher than the number we need to solve the system, the formulae do no longer fit.

3.2 Asymptotic analysis

For an asymptotic analysis we choose $m = \varepsilon n^2$. We cannot hope to get m in the order of n because then $\mathsf{P} = \mathsf{NP}$ would become very likely. But even if m stays in the order of n^2 the factor ε may be small enough for the cryptanalysis of small parameters. We see from section 2.1 and table 4, XL of degree 2 is asymptotically the same as relinearization.

Table 4. Asymptotic analysis of $\texttt{Blow}_D^{\texttt{hom}}$ and $\texttt{XL}_D^{\texttt{inh}}$.

Degree	$* \leq 0$	ε
0	$\frac{1}{2} - \varepsilon$	$\frac{1}{2}$
1	$\frac{1}{6} - \varepsilon$	$\frac{1}{6}$
2	$\frac{1}{24} - \frac{1}{2}\varepsilon + \frac{1}{2}\varepsilon^2$	$0,09175$
3	$\frac{1}{120} - \frac{1}{6}\varepsilon + \frac{1}{2}\varepsilon^2$	$0,06125$
4	$\frac{1}{720} - \frac{1}{24}\varepsilon + \frac{1}{4}\varepsilon^2 - \frac{1}{6}\varepsilon^3$	$0,04525$

Something unexpected happens in table 5. For $D = 2$ using $\texttt{Blow}_D^{\texttt{inh}}$ is asymptotically better than using $\texttt{XL}_D^{\texttt{inh}}$. But for $D > 2$ there is no asymptotic solution for $\texttt{Blow}_D^{\texttt{inh}}$ at all!

Table 5. Asymptotic analysis of $\mathtt{Blow}_D^{\mathtt{inh}}$.

Degree	$* \le 0$	ε
0	$\frac{1}{2} - \varepsilon$	$\frac{1}{2}$
1	$\frac{1}{6} - \varepsilon$	$\frac{1}{6}$
2	$\frac{1}{24} - \frac{1}{2}\varepsilon$	$\frac{1}{12}$

3.3 XL of high degrees D

Courtois *et al.* claimed in [CKPS00] that every $\mathcal{MQ}$-system could be solved by XL in sub-exponential time, if we chose D high enough. Well, this is not true in the inhomogeneous case $m = n$, as shown by Yang in [YC04b]. More precisely, there is a upper bound on D off which the number of new equations equals the number of new monomials. Remember XL needs the difference between the number of monomials T and the number of linearly independent equations I to be less or equal to $D + 2$. So after reaching the upper bound of D, XL can only solve the problem, if we increase D up to this difference. It is obvious that this is not efficient any more. We want to show this fact for the homogeneous case. The inhomogeneous case is analogous.
First let us consider the case $D = 2k$. The number of linearly independent equations subtracted by the number of monomials is given by

$$\sum_{i=0}^{k}(-1)^i\binom{m}{i+1}\binom{n+2(k-i)-1}{n-1} - \binom{n+2k+1}{n-1}$$
$$= -\sum_{i=0}^{2k+2}(-1)^i\binom{m-n}{i}\binom{m}{2k-i+2}. \quad (9)$$

In the special case $m = n$ inhomogeneous, *e.g.* $m + 1 = n$ homogenised, (9) does not further increase if we choose $2k + 2$ bigger than m, *i.e.* $D > m - 2$, and thus $k = \frac{m-2}{2}$ is an upper bound. We get the following.

$$\sum_{i=0}^{m}(-1)^i\binom{-1}{i}\binom{m}{m-i}$$
$$= \sum_{i=0}^{m}\binom{m}{i}$$
$$= 2^m.$$

We used $\binom{-1}{i} = (-1)^i\binom{1+i-1}{i} = (-1)^i$. So the number of linearly independent equations subtracted by the number of monomials is $T - I = 2^m$. XL succeed, if we raise $D+2$ up to 2^m, because $I - T \ge -D-2$ must hold. But for $m = n+1$ inhomogeneous equations, this become much better and D gets polynomial in

m. For $D = m - 2$ we always obtain a solution since

$$\sum_{i=0}^{m}(-1)^i\binom{m}{i} = 0.$$

Let $m = n+a$ and $a \in \mathbb{N}_{>1}$. The upper bound for D to solve the system is given by $D = 2m-n-1 = n+2a-1$. The term $\binom{a}{i}$ becomes 0 for $i = a+1, \ldots, n+2a+1$ and the term $\binom{n+a}{n+2a+1-i}$ for $i = 0, \ldots, a$. Thus it hold

$$-\sum_{i=0}^{n+2a+1}(-1)^i\binom{a}{i}\binom{n+a}{n+2a+1-i} = 0.$$

3.4 Theoretical analysis

Lemma 1. *If P^{hom} contains random equations then the number of linearly independent equations produced by $\mathtt{Blow}_D^{hom}$ is upper bounded by*

$$D = 2k: \quad \sum_{i=0}^{k}(-1)^i\binom{m}{i+1}\binom{n+2(k-i)-1}{2(k-i)} \tag{10}$$

$$D = 2k+1: \quad \sum_{i=0}^{k}(-1)^i\binom{m}{i+1}\binom{n+2(k-i)}{2(k-i)+1}.$$

This bound holds with very high probability.

Before proving this lemma at the end of this section, we need some intermediate results.
Equations (10) on $\mathtt{Blow}_D^{\mathtt{inh}}$ and $\mathtt{XL}_D^{\mathtt{inh}}$ was given and proven inductively by Moh [Moh00]. We want to formulate this proof in more detail and give a good intuition were the systematic linear dependencies come from. First we concentrate on $\mathtt{Blow}_2^{\mathtt{hom}}$ and search for the $\binom{m}{2}$ linear dependent equations out of all $m\binom{n+1}{2}$ produced equations. Let f, g be two $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic polynomials in n variables each. Denote $\mathtt{Mon}_f, \mathtt{Mon}_g$ the set of monomials in f and g, respectively. Assume the existence of some admissible ordering for multivariate polynomials f, g, *e.g. degrev-lex* or *lex*.

Lemma 2. *Let f, g be a pair of linearly independent, $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic polynomials. Moreover, let $F := \{bf : b \in \mathtt{Mon}_g\}$ and $G := \{ag : a \in \mathtt{Mon}_f\}$ be the sets of cross-wise monomial multiplication of f and g, respectively. Then these two sets produce at most $|F| + |G| - 1$ linearly independent equations.*

Proof. We denote our two polynomials by $f := \sum_{i=1}^{\sigma}\alpha_i a_i$ and $g := \sum_{i=1}^{\tau}\beta_i b_i$ for non-zero field elements $\alpha_i, \beta_j \in \mathbb{F}^*$ and monomials a_i, b_j for $1 \le i \le \sigma$ and $1 \le j \le \tau$. All monomials have degree 2, *i.e.* we have $\deg(a_i), \deg(b_i) = 2$. The important property of the two sets F, G is that each monomial ab for $a \in \mathtt{Mon}_f$ and $b \in \mathtt{Mon}_g$ exists twice, namely once in $bf \in F$ and once in $ag \in G$. The following equation shows that adding all equations of F multiplied by coefficients β_i is equal to adding all equations of G multiplied by coefficients α_i and thus the set $F \cup G$ is linear dependent.

$$\sum_{i=1}^{\tau} \beta_i b_i f = \sum_{i=1}^{\tau} \beta_i b_i \sum_{i=1}^{\sigma} \alpha_i a_i = \sum_{i=1}^{\sigma} \alpha_i a_i \sum_{i=1}^{\tau} \beta_i b_i = \sum_{i=1}^{\sigma} \alpha_i a_i g$$

Clearly this construction fails if we delete one equation in $F \cup G$. □

Corollary 1. *The set $\mathtt{Blow}_2^{hom}$ contains at most $\binom{n+1}{2}m - \binom{m}{2}$ linearly independent equations.*

Proof. By its definition, we have at most $\binom{n+1}{2}m$ elements in $\mathtt{Blow}_2^{\mathtt{hom}}$. This explains the first part of the sum and also gives an upper bound. Considering all pairs $(f, g) \in P \times P$ with $f < g$ and also Lemma 2, we obtain $\binom{m}{2}$ linear dependencies. □

Corollary 2. *The set $\mathtt{XL}_2^{inh}$ contains at most $\binom{n}{2}m + nm + m - \binom{m}{2}$ linearly independent equations.*

Proof. This corollary works similar to corollary 1. By its definition, we have at most $\binom{n}{2}m + nm + m$ elements in $\mathtt{XL}_2^{\mathtt{inh}}$. This explains the first part of the sum and also gives an upper bound. Considering all pairs $(f, g) \in P \times P$ with $f < g$ and also Lemma 2, we obtain $\binom{m}{2}$ linear dependencies. □

Lemma 3. *Let f, g be a pair of linearly independent, homogeneous $\mathcal{M}$ultivariate $\mathcal{Q}$uadratic polynomials. For $n \geq k > 2$, the set $\mathtt{Blow}_k^{hom} = \{\mu f, \mu g : \mu \in \mathtt{Mon}_k\}$ contains at most $2\binom{n+k-1}{k} - \binom{n+k-3}{k-2}$ linearly independent equations.*

Proof. The first part of the sum is a result of the $\binom{n+k-1}{k}$ choices of the monomial μ. We fix some monomial $v \in \mathtt{Mon}_{k-2}$ and study the two sets $F_v := \{vfb : b \in \mathtt{Mon}_g\}$ and $G_v := \{vga : a \in \mathtt{Mon}_f\}$. For a given pair F_v, G_v, we can now apply lemma 2. We have $|\mathtt{Mon}_{k-2}| = \binom{n+k-3}{k-2}$ individual choices for v. □

Extending this lemma from pairs to sets is kind of tricky, because since $D \geq 4$ we obtain new linear dependencies between 3 and more equations. Thus we are counting linear dependencies twice if we only consider pairs f, g. To count all equations only once, we need a property (equation (11)) which follows if the system of equations is pairwise coprime. First we show that this occurs with very high probability. Then we show that if the system is pairwise coprime the upper bound of lemma 1 is tight.

Corollary 3. *Two randomly chosen $\mathcal{MQ}$-equations f and g are not coprime with probability*

$$\frac{q + 2(q-1)(q^{n+1} - 1)}{q^{\binom{n+2}{2}}}.$$

Proof. Two randomly chosen quadratic polynomials f and g are not coprime iff they share a common factor. Per definition $\gcd(f, a) = 1$ for all $a \in \mathbb{F}_q$ and thus the common factor have to be a polynomial of degree one. Let $g = ab$ and $f = cd$ with $a, b, c, d \in \mathbb{F}[x_1, \ldots, x_n]$ and $\deg(a) = \deg(b) = \deg(c) = \deg(d) = 1$. We choose g arbitrary and count the number of f with a common factor. In case 1 $f = \lambda g$ with $\lambda \in \mathbb{F}_q$ gives q possibilities. In case 2 we assume w.l.o.g. $d \neq \lambda a, \lambda b$. Furthermore $\lambda \neq 0$ as we count this in case 1. We can choose $c = \lambda a$ or $c = \lambda b$ with $d \neq 0$ arbitrary. This give $2(q-1)(q^{n+1} - 1)$ possibilities.
The total number of choices of f is $q^{\binom{n+2}{2}}$ and thus the probability of not being coprime is $\frac{q+2(q-1)(q^{n+1}-1)}{q^{\binom{n+2}{2}}}$. □

The probability of a $\mathcal{MQ}$-system to be pairwise coprime is simply one minus $\binom{m+1}{2}$ times the probability of lemma 3. Note that the probability of a $\mathcal{MQ}$-system to be pairwise coprime increase if q, m or n increase. Already for the small parameters $q = 4$ and $m = n = 9$ it is greater than $1 - 2^{-80}$. We have also verified this experimentally (cf. Section 3.1). Note for fixed q the probability increase exponentially in n.
Denote with $\mathrm{Lin}(S, k)$ the *linear closure* of degree k of a polynomial f or a set S, respectively, as

$$\mathrm{Lin}(f, k) := \{a + b : a, b \in \{\varphi\mu f : \varphi \in \mathbb{F}, \mu \in \mathtt{Mon}_k\}\}$$

$$\mathrm{Lin}(S, k) := \{a + b : a, b \in \{\varphi\mu s : \varphi \in \mathbb{F}, \mu \in \mathtt{Mon}_k, s \in S\}\}.$$

We can also think of $\mathrm{Lin}(\cdot, k)$ as possible rows in the corresponding coefficient matrix Π for S or f. Moreover, denote with $|\mathrm{Lin}(S, k)|$ the *number of its elements* and with $\#\mathrm{Lin}(S, k)$ the *number of linear independent equations* in $\mathrm{Lin}(S, k)$. The latter can also be viewed as the rank of the corresponding coefficient matrix. *Assumption:* Let $f \notin \mathrm{Lin}(S, 0)$ be a quadratic polynomial, $\mathcal{S} := \{g_1, \ldots, g_m\}$ a set of $m \in \mathbb{N}$ linearly independent quadratic polynomials, also to f, and $k \geq 0$ some extension degree. Then we have

$$\#(Lin(\mathcal{S}, k) \cap Lin(f, k)) = \#Lin(\mathcal{S}, k-2) \tag{11}$$

with very high probability. For the special case $m = 1$ condition (11) means that both polynomials are co-prime.
Before finishing the proof of lemma 1, we want to give some intuition behind the overall idea: in a nutshell, we will make use of the inclusion/exclusion principle for different dimensions k. The reason is that for some dimension k' we will count the same linear independent equation twice—which we have to correct at this level. For dimension $k' + 2$, there is an overcorrection, which has to be corrected again and so on. Hence, we end up with a sum in $(-1)^r$ and a count of the number of equations we have to correct. Recall that we wanted to count the number of linearly independent equations of $\mathtt{Blow}_D^{\mathtt{hom}}$ and hence deal with a polynomial system $P^{\mathtt{hom}}$.

Proof (lemma 1). First we reformulate the formula of lemma 1. The number of linearly independent equations $\#\mathrm{Lin}(P^{\mathtt{hom}}, k)$ there is given by

$$\sum_{0\le 2i\le D} (-1)^i \binom{m}{i+1}\binom{n+D-2i-1}{n-1}. \tag{12}$$

We proof this by induction via m. The case $m = 1$ is trivial.
Let us assume equation (12) holds for m. We have to show that it also holds for $m + 1$.

We have $P^{\mathtt{hom}}_{m+1} := P^{\mathtt{hom}}_m \cup \{p_{m+1}\}$ and write

$$\begin{aligned}\#\mathrm{Lin}(P^{\mathtt{hom}}_{m+1}, D) = {} & \#\mathrm{Lin}(P^{\mathtt{hom}}_m, D) + \#\mathrm{Lin}(p_{m+1}, D)\\ & - \#(\mathrm{Lin}(P^{\mathtt{hom}}_m, D) \cap \mathrm{Lin}(p_{m+1}, D)).\end{aligned}$$

The last term simplifies to $\#\mathrm{Lin}(P^{\mathtt{hom}}_m, D-2)$ using equation 11. Using the induction hypothesis we obtain the following formula for $\#\mathrm{Lin}(P^{\mathtt{hom}}_{m+1}, D)$.

$$\begin{aligned}& \sum_{0\le 2i\le D} (-1)^i \binom{m}{i+1}\binom{n+D-2i-1}{n-1}\\ & + \binom{n+D-1}{D}\\ & - \sum_{0\le 2i\le D-2} (-1)^i \binom{m}{i+1}\binom{n+D-2i-3}{n-1}\\ & = \sum_{0\le 2i\le D} (-1)^i \binom{m}{i+1}\binom{n+D-2i-1}{n-1}\\ & + \sum_{0\le 2i\le D} (-1)^i \binom{m}{i}\binom{n+D-2i-1}{n-1}\end{aligned} \tag{13}$$

Exploiting $\binom{m}{l} = \binom{m-1}{l} + \binom{m-1}{l-1}$ yields

$$(13) = \sum_{0\le 2i\le D} (-1)^i \binom{m+1}{i+1}\binom{n+D-2i-1}{n-1}$$

Since we have $\varepsilon > 0$, lemma 1 gives an upper bound of the number of linearly independent equations. But as we saw in corr. 3, the value ε is very small in practice, so this bound is tight for all practical cases. □

Lemma 1 only handles the homogeneous case. The proof for the inhomogeneous case is analogous. Actually there is a strong connection between the homogeneous and inhomogeneous case, because we reach the same results, if we homogenise non-homogeneous system, as we saw in section 3.1.

4 Variants of XL

Inspired by Gröbner bases and some other observations there is a whole family of XL-like algorithms, which try to use some additional ideas to speed up the original XL algorithm. We revisit the most important ones and give some reasons if and under which circumstances they are useful. Some examples are FXL, XFL, XLF, XL', XL2 and XSL [CKPS00, BFP09, YC04a, Cou04, CP03].

FXL

FXL, or *fixing extended linearization*, was suggested in the original paper of Courtois *et al.* [CKPS00] and is nothing else than XL with guessing some variables beforehand. That this is quit a good idea is already shown for the Gröbner base algorithm in [BFP09]. That it is also a good idea for XL shows equation (9) in section 3.3. We saw that the case $m = n$ is exponential in D, but already the case $m = n + 1$ is polynomial, so it helps to guess at least one variable. The optimal number of guessed variables is discovered by Yang and Chen in [YC04a] section 5.2.

XFL

XFL is a variant of FXL. We choose f variables, but do not guess them right in the beginning. We choose the order of the monomials in a way that all monomials containing any of the f variables are eliminated last. Now we linearise the system and apply Gaussian elimination. Because the system was underdeterminend, we obtain no unique solution. To do so, we guess one of the f variables and apply Gaussian elimination again. Why is this stepwise guessing better than FXL in some case? First we have to do the most work, i.e. the first Gaussian elimination, only once. In FXL we have to do this after every wrong guessing. But notice, that there the number of monomials is smaller, so we carefully have to calculate the right tradeoff between the two variants. Second XFL may use dependencies among the f variables and thus succeed.

XLF

XLF just take the *field equations* $(x^q - x) = 0$ in $\mathbb{F}_q$ into account and was first mentioned in [Cou04]. XLF makes sense in the inhomogeneous case, if D get larger than $(q - 2)$. In this case the analysis becomes slightly different, because the number of produced monomials decrease, i.e. monomials x_i^D reduce to x_i which already exists. This means we need less linearly independent equations to succeed. Note that XLF is one of a handful variants which improve the inhomogeneous case, but not the homogeneous one. In the homogeneous case we only have monomials of degree $D + 2$. If we reduce them we get monomials of lower degree, but they did not exist before and thus the number of unknowns stay the same. Even if the formulas of the number of linearly independent equations in section 3.1 showed that the inhomogeneous and homogeneous case are equal using homogenization, this is not true any longer if we want to use some algebraic dependencies. By homogenization we grout the structure of the inhomogeneous equations we want to use by methods like XLF or MutantXL.

XL'

Introduced by Courtois and Patarin in [CP03] this variant solve the equation system by XL until there are only $\binom{r+D+2}{D+2}$ equations in r variables left. This remaining equation system is solved by brute force or other algorithms like Gröbner bases.

Lemma 4. *For practical purposes, FXL is better than XL'.*

Proof. We call FXL better than XL', *i.e.* FXL $\geq$ XL', if $(T-I)_{\text{FXL}}$ is smaller than $(T-I)_{\text{XL'}}$. With section 3.3 and $D = 2k$ we can write

$$
\begin{aligned}
(T-I)_{\text{FXL}} &= \binom{n-r+D+1}{D+2} - \sum_{i=0}^{k}(-1)^i \binom{m}{i+1}\binom{n-r+D-2i-1}{n-r-1} \\
&= \sum_{i=0}^{2k+2}(-1)^i \binom{m-n+r}{i}\binom{m}{2k-i+2}
\end{aligned}
$$

and

$$
\begin{aligned}
(T-I)_{\text{XL'}} &= \binom{n+D+1}{D+2} - \sum_{i=0}^{k}(-1)^i \binom{m}{i+1}\binom{n+D-2i-1}{n-1} - \binom{r+2k+2}{2k+2} \\
&= \sum_{i=0}^{2k+2}(-1)^i \binom{m-n}{i}\binom{m}{2k-i+2} - \binom{r+2k+2}{2k+2} + 1.
\end{aligned}
$$

If we would plot formula $(T-I)_{\text{XL'}} - (T-I)_{\text{FXL}}$ we would see that this is greater than zero, *i.e.* FXL is better than XL', for r less than some bound depending on k. For increasing k the bound on r decrease. It seems very hard to calculate this bound in an analytical way. But for real world parameter $k < 10$ and $r \ll n$ we are below this bound. W.l.o.g. we can assume $m = n$, otherwise we substitute r. See table 6 for the upper bound on r depending on m and k. With F_5 we can solve $\mathcal{MQ}$-systems up to $m = 20$ in 2^{66} operations, so we stopped the table at $m = 30$ for practical purpose. Even $k > 6$ is of no practical interest because the workload without considering guessing would be larger than $\binom{n+2k+2}{2k+2}^{\omega}$ for $2 \leq \omega \leq 3$. Note that the cases marked gray are always solvable by $\texttt{XL}_{2k}^{\texttt{inh}}$ without guessing. In all the other cases the bound on r is high enough to guess as many variables as we need to solve the equation system with FXL. So we claim that FXL is always better than XL' for practical purpose.

XSL

Courtois and Pieprzyk [CP02] published this method at Asiacrypt 2002 and claimed to have broken AES. This was disproofed in 2005 by Leurent and Cid

$m\backslash k$	1	2	3	4	5	6
5	1	0	0	0	0	0
10	6	3	1	0	0	0
15	11	8	6	5	1	1
20	15	13	12	10	8	6
25	20	18	17	15	12	10
30	25	23	22	19	17	15

Table 6. Upper bound on r.

[CL05]. The idea of XSL is to use the special structure of the equation system. If some equations are sparse you might introduce more new monomials by multiplying them by all monomials of a special degree. So in some case it might be better to multiply some equations only by some monomials. It is in no way clear how to do this. The idea of XSL is connected to Coppersmiths lattice based method to solve modular equations. Like in XL you multiply the equation by so called shift polynomials. Choosing the right shift set is a difficult problem. In the case of two unknowns, we can plot the Newton polytope and get an intuition. But in multivariate cryptography you deal with a lot more unknowns. So it is an important open problem to find the right shift set for some given equation.

MutantXL

One of the most efficient derivates of XL is called MutantXL. It was introduced in [MMD+08] and claims to be as fast as F_4 in some cases.

Let I be the number of linearly independent equations produced by $\texttt{XL}_D^{\texttt{inh}}$ and $T = \binom{n+D+2}{D+2}$ the number of degree $\leq D+2$ monomials. If $T - I > D+2$ this is not solvable by linearization and thus we would continue with $\texttt{XL}_{D+1}^{\texttt{inh}}$ in the original XL algorithm. MutantXL is a step in between. It uses equations that would be produced by $\texttt{XL}_{D+k}^{\texttt{inh}}$ with $k > 0$ but without introducing new monomials. To do so we use only polynomials of degree $< D+2$, so called mutants, that are produced in the Gaussian elimination step of $\texttt{XL}_D^{\texttt{inh}}$. For example multiplying these polynomials by all monomials of $\texttt{Mon}_1$ leads to new equations without generating new monomials. Note that this strategy is useful only for inhomogeneous equations. In the homogeneous case all monomials are of the same degree and thus mutants never occur. This is another example that we lose information by homogenising equation systems.

Definition 3. *Let* $f = \sum_{i=1}^{m} g_{j_i} h^{(i)}$ *with* $h^{(i)} \in P^{\boldsymbol{inh}}$ *and* g_{j_i} *some polynomial of degree* $\leq D$ *be a representation of* f*. This is not unique. The index set* J *denotes all representations and* $j \in J$*. The* level (lev) *of this representation is defined by*

$$lev\left(\sum_{i=1}^{m} g_{j_i} h^{(i)}\right) := max\left\{deg\left(g_{i_j} h^{(i)}\right) \mid 1 \leq i \leq m\right\}.$$

The level of g is defined by the minimum level of all its representations.

$$lev\,(g) := min\{lev\left(\sum_{i=1}^{m} g_{j_i} h^{(i)}\right) | j \in J\}$$

We call g a mutant if $deg(g) > lev(g)$.

The crucial question as always is how many equations produced by mutants are linearly independent from the known ones. We give two upper bounds on this number. We showed experimentally that the smaller bound is tight. We will give some theoretical explanation on that. To conclude we compare MutantXL to F_5 and show that indeed in some case it is faster.

Remark: To implement MutantXL correctly, we will introduce the term of *trivial mutants*. Using $\mathtt{XL}_D^{\mathtt{inh}}$ all equations produced by $\mathtt{Blow}_{<D}^{\mathtt{inh}}$ are mutants by definition. But all their multiples of certain degree are already contained in $\mathtt{XL}_D^{\mathtt{inh}}$ and thus are not linearly independent. We can reduce the computational workload if we only consider mutants produced by $\mathtt{Blow}_D^{\mathtt{inh}}$.

To avoid hiding the upper bounds behind formalism, we start with the case $|\mathtt{Mon}_{D+2}| \leq I_{\mathtt{XL}_D^{\mathtt{inh}}} \leq |\mathtt{Mon}_{D+2}| + |\mathtt{Mon}_{D+1}|$ illustrated in figure 3.

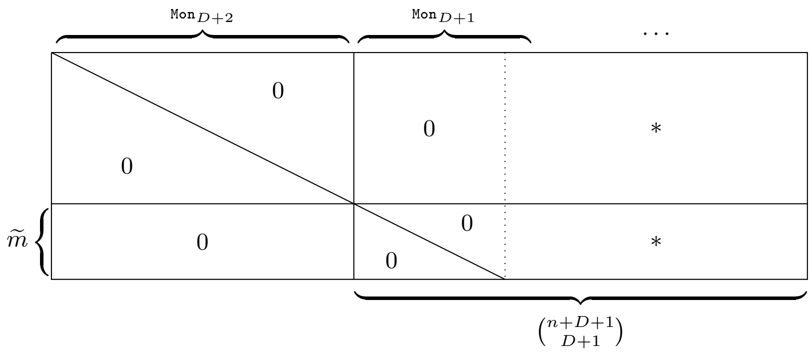


Fig. 3. Coefficient Matrix Π of $\mathtt{XL}_D^{\mathtt{inh}}$ after Gaussian elimination

The first upper bound is the number of equations produced by Mutants. In the above case $k = 1$ this is $n(I_{\mathtt{XL}_D^{\mathtt{inh}}} - |\mathtt{Mon}_{D+2}|)$ or $n\widetilde{m}$ using the notation of figure 3. Experiments for $2 \leq n \leq 7$ and $n \leq m \leq 9$ show that this trivial bound is way above the correct number of new linear independent equations. The second upper bound is a result of the fact that all $n\widetilde{m}$ equations produced by mutants are implicit equations of $\mathtt{XL}_{D+1}^{\mathtt{inh}}$. Exactly $I_{\mathtt{XL}_{D+1}^{\mathtt{inh}}} - I_{\mathtt{XL}_D^{\mathtt{inh}}}$ of them are linear independent to the previous ones. But they all contain monomials of $\mathtt{Mon}_{D+3}$. Equations produced by Mutants have maximal degree $D+2$ and thus first all $|\mathtt{Mon}_{D+3}|$ monomials have to be reduced. Therefor $I_{\mathtt{XL}^{\mathtt{inh}}_{D+1}} - I_{\mathtt{XL}^{\mathtt{inh}}_{D}} - |\mathtt{Mon}_{D+3}|$ is an upper bound on the number of linear independent equations produced by Mutants. Note that this bound was tight in all our experiments.

To gerneralize the above example let $k \in \mathbb{N} : \sum_{j=0}^{k-1} |\mathtt{Mon}_{D+2-j}| \leq I \leq \sum_{j=0}^{k} |\mathtt{Mon}_{D+2-j}|$.

Corollary 4. *The maximal number of equations produced by Mutants is given by*

$$\sum_{i=1}^{k-1} \binom{n+i-1}{i} |\mathit{Mon}_{D+2-i}| + \binom{n+k-1}{k} \left(I - \sum_{i=0}^{k-1} |\mathit{Mon}_{D+2-i}| \right).$$

Corollary 5. *A nontrivial upper bound on the number of linearly independent equations produced by Mutants is given by*

$$\sum_{i=1}^{k} I_{\mathit{XL}^{\mathit{inh}}_{D+i}} - I_{\mathit{XL}^{\mathit{inh}}_{D}} - \sum_{j=1}^{i} |\mathit{Mon}_{D+2+j}|.$$

We come back to the example in figure 3 to get an intuition on a lower bound, *i.e.* to show that corollary 5 is tight. In lemma 2 we saw that new linear dependent equations are produced block-wise, *i.e.* if we multiply f and g by all monomials of degree two, all equations are linearly independent besides one. Multiplying the mutants with degree one monomials we implicitly use equations of $\mathtt{Blow}^{\mathtt{inh}}_{D+1}$. If D was even, no new linear dependencies are produced.

Remark 1. MutantXL will hardly work in the case $m = n$. As seen in section 3.3 we need $D = 2^m$ to solve for case $m = n$. The reason was that the number of newly generated linearly independent equations obtained by increasing D equals the number of new monomials and thus the second bound on MutantXL will always yield zero.

Note: A further improvement of MutantXL called MXL_2 use ideas of XSL and is published in [MMDB08].

A comparison with the fastest known attack on UOV can be found in table 7. Given are the F_5 algorithm, a version were one or two variables are fixed before performing F_5 ('HybridF_5'), and the results from corollary 5 for this parameter set. We can see that MutantXL outperforms both variants of F_5 for challenges on UOV. See more detailed tables in the appendix A.

5 Conclusion

While Relinearization and XL seemed to be a magnificent tool for cryptanalysis in the beginning, their effectiveness was diminished in subsequent years. In

Table 7. Comparison between F_5, HybridF_5 and MutantXL in terms of workload in field operations over GF(q).

UOV	[log$_2$]		
	F_5	HybridF_5	this work
$m = 10$	41.36	37.75	**37**
$m = 20$	82.51	66.73	**63**

addition, existing Gröbner bases algorithms performed better in most cases, so XL came more and more out of focus.
Empirical evidence with (naturally) small values of n already suggested in the case of the $\mathcal{MQ}$-scheme HFE that Gröbner bases might not be as efficient as MutantXL [MMD$^+$08]. In this paper, we have shown that this is not a coincidence for small values of n, but a systematic finding which can be put on firm theoretical foundations. Hence, we showed that MutantXL can compete with F_5. It seems a matter of the right implementation which of the two is faster. In this context it is an important open question how to generate linear independent equations only. Up to now we need to produce all equations and eliminate the linear dependent ones by Gaussian elimination.
Taking a wider perspective, this result is not that surprising than it seems at first glance. Main reason is that XL computes only **one** solution for a given ground field $\mathbb{F}$. In contrast, Gröbner bases were designed to compute **all** solutions, moreover in the *algebraic closure* of $\mathbb{F}$. Obviously, the latter task is more general and hence computational more difficult. Still, using tricks like truncated Gröbner bases and field equations $(x^q - x)$ algorithms based on Gröbner basis computation were able to level the field and outperform XL. An additional reason might be that decades of research went into tuning GB-algorithms while barely 10 years have passed since XL and its variations were introduced to the cryptographic community. Hence, there might be more room for improving XL according to the needs of cryptography than in the case of GB-algorithms. In addition, in cryptography *one* solution is sufficient in most cases to solve a cryptographic problem rather than a huge set of them. Therefore, it was time to develop a theoretical framework to thoroughly analyse XL and its derivates, so running times and memory requirements can be predicted without relying on (possibly) noisy empirical evidence.
All in all, it may be a sensible course of action to spend further time to clarity the speed gap between Gröbner bases and (Mutant)XL to avoid further surprises in other cryptanalytic areas.

Acknowledgements

The authors were funded via an DFG (German Research Foundation) Emmy Noether grant. In addition, the work described in this paper has been supported in part by the European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II.

Bibliography

[ACG+06] Frederik Armknecht, Claude Carlet, Philippe Gaborit, Simon Künzli, Willi Meier, and Olivier Ruatta. Efficient computation of algebraic immunity for algebraic and fast algebraic attacks. In *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 147–164. Serge Vaudenay, editor, Springer, 2006. ISBN 3-540-34546-9.

[ACr04] Pil Joong Lee, editor. *Advances in Cryptology — ASIACRYPT 2004*, volume 3329 of *Lecture Notes in Computer Science*. Springer, 2004. ISBN 3-540-23975-8.

[AFI+04] Gwénolé Ars, Jean-Charles Faugère, Hideki Imai, Mitsuru Kawazoe, and Makoto Sugita. Comparison between xl and gröbner basis algorithms. In ACr [ACr04], pages 338–353.

[AK03] Frederik Armknecht and Matthias Krause. Algebraic attacks on combiners with memory. In Cr [Cr03], pages 162–175.

[BFP09] Luk Bettale, Jean-Charles Faugère, and Ludovic Perret. Hybrid approach for solving multivariate systems over finite fields. *In Journal of Mathematical Cryptology*, 3:177–197, 2009.

[BWP05] An Braeken, Christopher Wolf, and Bart Preneel. A study of the security of Unbalanced Oil and Vinegar signature schemes. In *The Cryptographer's Track at RSA Conference 2005*, volume 3376 of *Lecture Notes in Computer Science*. Alfred J. Menezes, editor, Springer, 2005. 13 pages, cf `http://eprint.iacr.org/2004/222/`.

[CGMT02] Nicolas Courtois, Louis Goubin, Willi Meier, and Jean-Daniel Tacier. Solving underdefined systems of multivariate quadratic equations. In *Public Key Cryptography — PKC 2002*, volume 2274 of *Lecture Notes in Computer Science*, pages 211–227. David Naccache and Pascal Paillier, editors, Springer, 2002.

[CKPS00] Nicolas T. Courtois, Alexander Klimov, Jacques Patarin, and Adi Shamir. Efficient algorithms for solving overdefined systems of multivariate polynomial equations. In *Advances in Cryptology — EUROCRYPT 2000*, volume 1807 of *Lecture Notes in Computer Science*, pages 392–407. Bart Preneel, editor, Springer, 2000. Extended Version: `http://www.minrank.org/xlfull.pdf`.

[CL05] Carlos Cid and GaÃńtan Leurent. An analysis of the xsl algorithm. In *Proceedings of Asiacrypt 2005, LNCS*, volume 3788 of *Lecture Notes in Computer Science*, pages 333–352. Bimal Roy, editor, Springer-Verlag, 2005. ISBN 3-540-30684-6.

[Cou02] Nicolas Courtois. Higher order correlation attacks, XL algorithm and cryptanalysis of Toyocrypt. In *ICISC*, volume 2587 of *Lecture Notes in Computer Science*, pages 182–199. Pil Joong Lee and Chae Hoon Lim, editors, Springer, 2002.

[Cou04] Nicolas Courtois. Algebraic attacks over $GF(2^k)$, application to HFE challenge 2 and Sflash-v2. In *Public Key Cryptography — PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 201–217. Feng Bao, Robert H. Deng, and Jianying Zhou (editors), Springer, 2004. ISBN 3-540-21018-0.

[CP02] Nicolas T. Courtois and Josef Pieprzyk. Cryptanalysis of block ciphers with overdefined systems of equations. In *Advances in Cryptology — ASIACRYPT 2002*, volume 2501 of *Lecture Notes in Computer Science*, pages 267–287. Yuliang Zheng, editor, Springer, 2002.

[CP03] Nicolas T. Courtois and Jacques Patarin. About the XL algorithm over GF(2). In *CT-RSA'03: Proceedings of the 2003 RSA conference on The cryptographers' track*, pages 141–157, Berlin, Heidelberg, 2003. Springer-Verlag.

[Cr03] Dan Boneh, editor. *Advances in Cryptology — CRYPTO 2003*, volume 2729 of *Lecture Notes in Computer Science*. Springer, 2003. ISBN 3-540-40674-3.

[Die04] Claus Diem. The XL-algorithm and a conjecture from commutative algebra. In ACr [ACr04]. ISBN 3-540-23975-8.

[DS05] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. In *Conference on Applied Cryptography and Network Security — ACNS 2005*, volume 3531 of *Lecture Notes in Computer Science*, pages 164–175. Springer, 2005.

[FA03] Jean-Charles Faugère and Gwénolé Ars. An algebraic cryptanalysis of nonlinear filter generators using Gröbner bases. Rapport de recherche 4739, February 2003. `www.inria.fr/rrrt/rr-4739.html`.

[Fas07] Alex Biryukov, editor. *Fast Software Encryption — FSE 2007*, volume 4593 of *Lecture Notes in Computer Science*. Springer, 2007. ISBN 978-3-540-74617-1.

[Fau99] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases (F_4). *Journal of Pure anNational Institute of Standards and Technologyd Applied Algebra*, 139:61–88, June 1999.

[Fau02a] Jean-Charles Faugère. HFE challenge 1 broken in 96 hours. Announcement that appeared in `news://sci.crypt`, 19^{th} of April 2002.

[Fau02b] Jean-Charles Faugère. A new efficient algorithm for computing Gröbner bases without reduction to zero (F_5). In *International Symposium on Symbolic and Algebraic Computation — ISSAC 2002*, pages 75–83. ACM Press, July 2002.

[Fau03a] Jean-Charles Faugère. Algebraic cryptanalysis of (HFE) using Gröbner bases. Technical report, Institut National de Recherche en Informatique et en Automatique, February 2003. `http://www.inria.fr/rrrt/rr-4738.html`, 19 pages.

[Fau03b] Jean-Charles Faugère. Fast Gröbner. Algebraic cryptanalysis of HFE and filter generators. In *Workshop on Coding and Cryptography 2003*, pages 175–176. Daniel Augot, Pascal Charpin, and

Grigory Kabatianski, editors, l'Ecole Supérieure et d'Appliction des Transmissions, 2003.

[FJ03] Jean-Charles Faugère and Antoine Joux. Algebraic cryptanalysis of Hidden Field Equations (HFE) using Gröbner bases. In Cr [Cr03], pages 44–60.

[FM07] Simon Fischer and Willi Meier. Algebraic immunity of S-boxes and augmented functions. In Fast Software Encryption — FSE [Fas07], pages 366–381. ISBN 978-3-540-74617-1.

[FOPT10] Jean-Charles Faugère, Ayoub Otmani, Ludovic Perret, and Jean-Pierre Tillich. Algebraic cryptanalysis of McEliece variants with compact keys. In *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 279–298. Henri Gilbert, editor, Springer, 2010. ISBN 978-3-642-13189-9.

[GJ79] Michael R. Garey and David S. Johnson. *Computers and Intractability — A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, 1979. ISBN 0-7167-1044-7 or 0-7167-1045-5.

[KPG99] Aviad Kipnis, Jacques Patarin, and Louis Goubin. Unbalanced Oil and Vinegar signature schemes. In *Advances in Cryptology — EUROCRYPT 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 206–222. Jacques Stern, editor, Springer, 1999.

[KS98] Aviad Kipnis and Adi Shamir. Cryptanalysis of the oil and vinegar signature scheme. In *Advances in Cryptology — CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 257–266. Hugo Krawczyk, editor, Springer, 1998.

[KS99] Aviad Kipnis and Adi Shamir. Cryptanalysis of the HFE public key cryptosystem. In *Advances in Cryptology — CRYPTO 1999*, volume 1666 of *Lecture Notes in Computer Science*, pages 19–30. Michael Wiener, editor, Springer, 1999. `http://www.minrank.org/hfesubreg.ps` or `http://citeseer.nj.nec.com/kipnis99cryptanalysis.html`.

[MAG] Computational Algebra Group, University of Sydney. *The MAGMA Computational Algebra System for Algebra, Number Theory and Geometry.* `http://magma.maths.usyd.edu.au/magma/`.

[MMD+08] Mohamed Saied Mohamed, Wael Said Mohamed, Jintai Ding, Johannes Buchmann, Stefan Tohaneanu, Ralf-Philipp Weinmann, Daniel Carbarcas, and Dieter Schmidt. Mutantxl and mutant grÃűbner basis algorithm. In *SCC '08: Proceedings of the 1st International Conference on Symbolic Computation and Cryptography*, pages 16–22, 2008.

[MMDB08] Mohamed Saied Mohamed, Wael Said Mohamed, Jintai Ding, and Johannes Buchmann. MXL2: Solving polynomial equations over GF(2) using an improved Mutant strategy. In *PQCrypto '08: Proceedings of the 2nd International Workshop on Post-Quantum Cryptography*, pages 203–215, Berlin, Heidelberg, 2008. Springer-Verlag.

[Moh00] T. Moh. On the method of "XL" and its inefficiency to TTM, 2000.

[MR02] Sean Murphy and Matthew J.B. Robshaw. Essential algebraic structure within the AES. In *Advances in Cryptology — CRYPTO 2002*, volume 2442 of *Lecture Notes in Computer Science*, pages 1–16. Moti Yung, editor, Springer, 2002.

[Pat97] Jacques Patarin. The oil and vinegar signature scheme. Presented at the Dagstuhl Workshop on Cryptography, September 1997. transparencies.

[SKPI07] Makoto Sugita, Mitsuru Kawazoe, Ludovic Perret, and Hideki Imai. Algebraic cryptanalysis of 58-round SHA-1. In Fast Software Encryption — FSE [Fas07], pages 349–365. ISBN 978-3-540-74617-1.

[YC04a] Bo-Yin Yang and Jiun-Ming Chen. All in the XL family: Theory and practice. In *ICISC 2004*, pages 67–86. Springer, 2004.

[YC04b] Bo-Yin Yang and Jiun-Ming Chen. Theoretical analysis of XL over small fields. In *ACISP 2004*, volume 3108 of *LNCS*, pages 277–288. Springer, 2004.

[YC05] Bo-Yin Yang and Jiun-Ming Chen. Building secure tame-like multivariate public-key cryptosystems: The new TTS. In *ACISP 2005*, volume 3574 of *LNCS*, pages 518–531. Springer, July 2005.

A Complexity of F_5, XL and MutantXL

Complexity of F_5

We denote m the number of quadratic equations, $n = m$ the number of variables and r the number of guessed variables. Note that we used $\omega = 2$ as Bettale *et al.* did in [BFP09] to calculate the complexity of their hybrid approach. We obtain the same results as in [BFP09] table 4 for $m = 20$ and guessing one or two variables over $\mathbb{F}_{2^8}$, see table A. The values in the tables are rounded Log_2 complexities. The exact value for $m = 20$, $r = 1$ and $\mathbb{F}_{2^8}$ is 66,73 respectively 67,79 for $r = 2$.

$m \backslash r$	0	1	2	3	5
5	6	3	3	2	1
10	11	6	5	4	3
15	16	8	7	6	4
20	21	11	9	8	6
25	26	13	11	10	8
30	31	16	14	12	10

Table 8. Degree of Regularity d_{reg}

$m \backslash r$	0	1	2	3	5
5	20	14	13	11	5
10	41	31	28	25	22
15	62	44	41	38	32
20	83	60	54	51	44
25	103	72	66	63	56
30	123	88	82	75	69

Table 9. Complexity of F_5 over $\mathbb{F}_2$

$m \backslash r$	0	1	2	3	5
5	20	18	21	23	25
10	41	35	36	37	42
15	62	48	49	50	52
20	83	64	62	63	64
25	103	76	74	75	76
30	123	92	90	88	89

Table 10. Complexity of F_5 over $\mathbb{F}_{2^5}$

$m \backslash r$	0	1	2	3	5
5	20	21	27	32	40
10	41	38	42	46	57
15	62	51	55	59	67
20	83	67	68	72	79
25	103	79	80	84	91
30	123	95	96	96	104

Table 11. Complexity of F_5 over $\mathbb{F}_{2^8}$

Complexity of XL

First we assume $\binom{n+D+2}{D+2}^{\omega}$ to be the complexity of XL, *i.e.* we concentrate on the number of columns N, cf. section 2.3. The proof of lemma 1 showed that the linear dependent equations produced by XL are very systematic. So in case $D = 2$ it is no problem just to generate linear independent equations and thus derive the given complexity. We assume that this is also possible for $D > 2$. At least a description of how to generate only linear independent equations for

$D = 6$ would be sufficient for practical purpose.
Note that we only considered XL up to degree 9. Fields marked with '-' indicate that this is not enough to solve the corresponding systems of equation.

$m \backslash r$	0	1	2	3	5
5	-	3	1	0	0
10	-	8	4	3	1
15	-	-	6	5	3
20	-	-	9	7	5
25	-	-	-	9	7
30	-	-	-	-	9

Table 12. Degree D of XL.

$m \backslash r$	0	1	2	3	5
5	-	15	11	8	5
10	-	34	25	22	17
15	-	-	37	34	28
20	-	-	52	46	40
25	-	-	-	58	52
30	-	-	-	-	63

Table 13. Complexity of XL over $\mathbb{F}_2$

$m \backslash r$	0	1	2	3	5
5	-	19	19	20	25
10	-	38	33	34	37
15	-	-	45	46	48
20	-	-	60	58	60
25	-	-	-	70	72
30	-	-	-	-	83

Table 14. Complexity of XL over $\mathbb{F}_{2^5}$

$m \backslash r$	0	1	2	3	5
5	-	22	25	29	40
10	-	41	39	43	52
15	-	-	51	55	63
20	-	-	66	67	75
25	-	-	-	79	87
30	-	-	-	-	98

Table 15. Complexity of XL over $\mathbb{F}_{2^8}$

Now we assume $(m\binom{n+D}{D})^\omega$ to be the complexity of XL, *i.e.* we concentrate on the number of rows M, cf. section 2.3. This is a bad upper bound for the case that we produce all $m\binom{n+D}{D}$ equations and eliminate the linear dependent ones by Gaussian elimination. Note that this is always bigger than $\binom{n+D+2}{D+2}^\omega$ if XL succeed.

$m \backslash r$	0	1	2	3	5
5	-	16	12	8	5
10	-	37	27	23	19
15	-	-	39	36	29
20	-	-	55	48	41
25	-	-	-	61	54
30	-	-	-	-	66

Table 16. Complexity of XL over $\mathbb{F}_2$

$m \backslash r$	0	1	2	3	5
5	-	20	20	20	25
10	-	41	35	35	39
15	-	-	47	48	49
20	-	-	63	60	61
25	-	-	-	73	74
30	-	-	-	-	86

Table 17. Complexity of XL over $\mathbb{F}_{2^5}$

$m\backslash r$	0	1	2	3	5
5	-	23	26	29	40
10	-	44	41	44	54
15	-	-	53	57	64
20	-	-	69	69	76
25	-	-	-	82	89
30	-	-	-	-	101

Table 18. Complexity of XL over $\mathbb{F}_{2^8}$

Complexity of MutantXL

Note that we take $\binom{n+D+2}{D+2}^\omega$ and $\omega = 2$ to compute the complexity of MutantXL. This is a lower bound, because in practice one has to produce all equations produced by mutants and eliminate the linear dependent ones by Gaussian elimination. For example in the case $m = 20$, $r = 3$ and $\mathbb{F}_{2^8}$ the complexity of 2^{64} would raise up to 2^{66} and thus we are only slightly better than F_5. We think in practice it would be a matter of implementation which algorithm is the better one.

$m\backslash r$	0	1	2	3	5
5	-	2	0	0	0
10	-	7	3	2	1
15	-	-	5	4	2
20	-	-	8	6	4
25	-	-	-	8	6
30	-	-	-	-	8

Table 19. Degree of MutantXL.

$m\backslash r$	0	1	2	3	5
5	-	1	1	1	0
10	-	1	1	1	1
15	-	-	1	1	1
20	-	-	1	1	1
25	-	-	-	1	1
30	-	-	-	-	1

Table 20. k used by MutantXL

$m\backslash r$	0	1	2	3	5
5	-	13	9	8	5
10	-	32	23	20	14
15	-	-	34	31	25
20	-	-	49	43	36
25	-	-	-	55	48
30	-	-	-	-	60

Table 21. Complexity of MutantXL over $\mathbb{F}_2$.

$m\backslash r$	0	1	2	3	5
5	-	17	17	20	25
10	-	36	31	32	34
15	-	-	42	43	45
20	-	-	57	55	56
25	-	-	-	67	68
30	-	-	-	-	80

Table 22. Complexity of MutantXL over $\mathbb{F}_{2^5}$

$m\backslash r$	0	1	2	3	5
5	-	20	23	29	40
10	-	39	37	41	49
15	-	-	48	52	60
20	-	-	63	64	71
25	-	-	-	76	83
30	-	-	-	-	95

Table 23. Complexity of MutantXL over $\mathbb{F}_{2^8}$