Cryptanalysis of Block Ciphers with Overdefined Systems of Equations

Nicolas T. Courtois[1,⋆] and Josef Pieprzyk[2]

[1] CP8 Crypto Lab, SchlumbergerSema,
36-38, rue de la Princesse, BP 45, 78430 Louveciennes Cedex, France,
courtois@minrank.org
[2] Center for Advanced Computing – Algorithms and Cryptography,
Department of Computing, Macquarie University, Sydney, NSW 2109, Australia,
josef@ics.mq.edu.au

Abstract. Several recently proposed ciphers, for example Rijndael and Serpent, are built with layers of small S-boxes interconnected by linear key-dependent layers. Their security relies on the fact, that the classical methods of cryptanalysis (e.g. linear or differential attacks) are based on probabilistic characteristics, which makes their security grow exponentially with the number of rounds N_r.

In this paper we study the security of such ciphers under an additional hypothesis: the S-box can be described by an overdefined system of algebraic equations (true with probability 1). We show that this is true for both Serpent (due to a small size of S-boxes) and Rijndael (due to unexpected algebraic properties). We study general methods known for solving overdefined systems of equations, such as XL from Eurocrypt'00, and show their inefficiency. Then we introduce a new method called XSL that uses the sparsity of the equations and their specific structure.

The XSL attack uses only relations true with probability 1, and thus the security does not have to grow exponentially in the number of rounds. XSL has a parameter P, and from our estimations is seems that P should be a constant or grow very slowly with the number of rounds. The XSL attack would then be polynomial (or subexponential) in N_r, with a huge constant that is double-exponential in the size of the S-box. The exact complexity of such attacks is not known due to the redundant equations. Though the presented version of the XSL attack always gives always more than the exhaustive search for Rijndael, it seems to (marginally) break 256-bit Serpent. We suggest a new criterion for design of S-boxes in block ciphers: they should not be describable by a system of polynomial equations that is too small or too overdefined.

Key Words: Block ciphers, AES, Rijndael, Square, Serpent, Camellia, multivariate quadratic equations, MQ problem, overdefined systems of multivariate equations, XL algorithm, Gröbner bases, sparse multivariate polynomials, Multivariate Cryptanalysis.

⋆ Partially written when visiting the Department of Computing, Macquarie University in Sydney, Australia.

Y. Zheng (Ed.): ASIACRYPT 2002, LNCS 2501, pp. 267–287, 2002.

1 Introduction

On October 2nd, 2000, NIST has selected Rijndael as the Advanced Encryption Standard. Serpent was second in the number of votes [1].

In the famous paper from 1949, Claude E. Shannon states that breaking a good cipher should require "as much work as solving a system of simultaneous equations in a large number of unknowns of a complex type", see [25]. This seemed very easy to achieve so far, as solving systems of equations can become intractable very easily. Though every cipher can be described in terms of solving multivariate equations over $GF(2)$, it does not mean that it can be broken. In [8] the whole AES is represented by one single equation with 2^{50} terms. Such a big equation has undoubtedly no consequences whatsoever on the security of AES. Recently however surprising attacks appeared in public key cryptography: the cryptanalysis of Matsumoto-Imai cryptosystem [17] by Patarin and the attack on the basic version of HFE cryptosystem by Courtois [6]. In these attacks the security collapses suddenly after discovering the existence of additional multivariate equations, that are not obvious and have not been anticipated by the designers. The subject of this paper is to see if such a weakness can compromise the security of a block cipher. For example, we show that the cryptanalysis of Rijndael and Serpent reduces to solving a big system of Multivariate Quadratic equations (a.k.a. MQ problem). Unlike in [8], MQ is a problem already known in cryptography that underlies the security of multivariate public key schemes such as HFE [19]. In [22,23] Shamir *et al.* show that though MQ is NP-hard, its complexity drops substantially when the MQ becomes overdefined (more equations than unknowns).[1]. In this paper we show that if the MQ is sparse and have a regular structure, it becomes even much easier. Such will be the MQ systems we will write for Rijndael and Serpent ciphers.

Since the pioneering work of Luby-Rackoff [12], there were many developments on the security of top-level schemes of block ciphers. The state of art in both security proofs and generic attacks for Feistel ciphers can be found in [15] and [18]. However, Rijndael is not a Feistel cipher and a more powerful theory has been developed by Vaudenay [26], to make security proofs against a large class of attacks including linear and differential cryptanalysis, for an arbitrary type of cipher. From this theory Moriai and Vaudenay developed security proofs for idealized versions of several AES candidates [27]. The outcome for Rijndael was somewhat strange: the cipher should have ≥ 384 rounds in order to make sure it was secure. Similar results were obtained for Serpent. Therefore, it is not completely unreasonable to believe, that the structure of Rijndael and Serpent could allow attacks with complexity growing slowly with the number of rounds. In this paper, it seems that we have found such an attack. It depends however more on algebraic properties of the S-boxes than on the structure of the cipher, and potentially, it can probably be extended to any block cipher.

The paper is organized as follows: First we describe a general class of ciphers that includes Rijndael and Serpent. Then we explore algebraic properties of

[1] Solving MQ in the opposite case of **under**defined systems, has been studied in [5].

their S-boxes and show that they can be described by an overdefined system of equations. Consequently, we formulate their cryptanalysis in terms of solving an overdefined system of quadratic equations. Though the general XL attack fails, we will present an improved method called XSL. It does not yet break Rijndael or Serpent but it gives definite conclusions about the design of block ciphers.

2 Substitution-Affine Ciphers, Rijndael and Serpent

According to Shannon's paradigm [25], a cipher is combination of some confusion and diffusion components. For example, SP-networks [7,10] are combinations of S-boxes with permutations of bits. More generally, we may allow linear or affine functions of bits, not only permutations of wires. We will call it a SA-cipher. In [21] Shamir and Biryukov study general top-level structural attacks against the SA-ciphers. These attacks will not depend on particular S-boxes used.

In the present paper we use specific properties of the S-boxes. We specify a restricted class of SA-ciphers called XSL-ciphers. Though our attacks are designed for XSL-ciphers, it is obvious that they can be easily extended to all SA-ciphers, and even to other block ciphers (including Feistel ciphers), provided that they use (only) somewhat "bad" S-boxes and have a regular periodic structure.

2.1 XSL-Ciphers and the Notations We Use

By definition, an XSL-cipher is a composition of N_r similar rounds:

X Before the first round, the input is XOR-ed with the key K_0. Let $i = 1$.
S Then a layer of B bijective S-boxes, on s bits each, is applied in parallel.
L Then a linear diffusion layer is applied.
X The result is XOR-ed with another session key K_i.
.. If $i = N_r$, the final result is produced.
Otherwise i is incremented and the process goes to the step S.

We denote the key bits used in an XSL-cipher by the variables $K_{i\ j}$ with $i = 0..N_r$ and $j = 1..Bs$. There are $N_r + 1$ session keys, K_0 is the first and K_{N_r} is the last. The number of key bits before expansion is H_k, the number of key bits after expansion is E_k, and the number of expanded key bits that are linearly independent is L_k. If we pick some subset of L_k key variables $K_{i\ j}$ that form a basis, then we will denote by $[K_{i\ j}]$ a linear expression of this bit $K_{i\ j}$ as a sum of the other $K_{k\ l}$ that are in the basis.

We call $X_{i\ j}$ the jth bit of the input of ith round S-boxes layer, step S (taken after the previous XOR with the session key X). We denote by $Y_{i\ j}$ the jth bit of the input of the linear part L of ith round (taken after the S-box application S). Similarly let $Z_{i\ j}$ be the jth bit of the output of the step L (before the next key XORing step X). Consequently we will denote the plaintext by Z_0 and the ciphertext by X_{N_r+1}, however these are constants, not variables. To summarize we have:

Step:	X	S	L	X	...	...	S	L	X	
Values:	Z_0	X_1	Y_1	Z_1	X_2	...	X_{N_r}	Y_{N_r}	Z_{N_r}	X_{N_r+1}

With these notations we obtain $X_{i+1\ j} = Z_{i\ j} \oplus K_{i\ j}$ for all $i = 0..N_r$.

2.2 Top-Level Structure of Rijndael

Rijndael specified in [4], is a special type of XSL-cipher with $s = 8$ and $B = 4N_b$. We will not give a full description of it, but will recall all the essential facts when necessary. Rijndael has $N_r = 10 \ldots 14$ rounds. The data in Rijndael is represented as rectangular "states" that are composed of N_b columns, each having the size of 4 S-boxes ($4s = 32$ bits). We have either $N_b = 4$, 6 or 8, which gives block sizes of $32N_b = 128$, 192 and 256 bits respectively. The encryption in Rijndael is performed as follows:

- X The input sequence $Z_{0\ j}$ is XOR-ed with the session key $K_{0\ j}$. Let $i = 1$.
- S Then resulting sequence $X_{i\ j}$ is transformed by $B = 4N_b$ S-boxes on $s = 8$ bits each.
- L Then resulting sequence $Y_{i\ j}$ is then subject to a composition of two linear transformations. First we have a permutation of bytes called ShiftRow, then four linear transformations MixColumn:$GF(256)^4 \rightarrow GF(256)^4$ applied in parallel for each of N_b columns.
  If $i = N_r$ (the last round) we have only ShiftRow, the MixColumn is omitted.
- X Then resulting sequence $Z_{i\ j}$ is XOR-ed with another session key K_i producing, either the ciphertext (if $= N_r$), or the process increments i and goes to step S.

The (unexpanded) key length is $H_k = 32N_k$ bits with $N_k = 4$, 6 or 8, thus again 128, 192 and 256 bits respectively. It is then expanded to $E_k = (N_r + 1)Bs = (N_r + 1)N_b \cdot 32$ bits.

2.3 Top-Level Structure of Serpent

Serpent described in [1] is an XSL-cipher with $s = 4$, $B = 32$ and $N_r = 32$. The block size is always 128 bits. The key length can be $H_k = 128$, 192 or 256 bits, and is also expanded to $E_k = (N_r + 1)Bs = 1056$ bits.

3 S-boxes and Overdefined Algebraic Equations

The only non-linear part of XSL-ciphers are the S-boxes. Let the function $F : GF(2)^s \rightarrow GF(2)^s$ be such an S-box, given an input $x = (x_1..x_s)$ we obtain an output $y = (y_1..y_s) = F(x)$. In Rijndael and Serpent, like for all other "good" block ciphers, the S-boxes are build with "good" boolean functions. Among the known criteria on cryptographically "good" boolean functions, we know that y_i should have a high algebraic degree in the x_i. However, this does not assure that there is no other "implicit" multivariate equations of the form

$P(x_1, \ldots, x_s, y_1, \ldots, y_s)$ that are of low algebraic degree. We will show that for Rijndael, and for Serpent, for very different reasons, a great number of such equations exist. We are interested in the actual number r of such equations $P(x_1, \ldots, x_s, y_1, \ldots, y_s)$, being of low degree d, e.g. $d \leq 2$. Unlike for "explicit" equations $y_i = f(x_1, .., x_s)$, this number r can be bigger than s. We are also interested in the total number of monomials t that appear in these equations, counted including the constant term. With these notations:

- In general, $t \approx \binom{s}{d}$. If $t \ll \binom{s}{d}$, we say that the equations are **sparse**.
- When $r \approx s$, the equations give enough information about the S-box, and the system will be sufficiently **defined** to yield about 1 solution x, given $y = F(x)$.

Consequently, when $r \gg s$, the system is said to be **overdefined**.

3.1 Quality of S-boxes and Random S-boxes

When r is close to t, we may eliminate most of the terms by linear elimination, and obtain simpler equations that are sparse and maybe even linear. For this reason, it is possible to measure the quality of our system of equations by the ratio $t/r \geq 1$. If t/r is close to 1, the S-box is considered as "bad". From this point of view, both overdefined systems (big r) and sparse systems (small t) will be "bad". Otherwise, if the system is not overdefined and not sparse, $t/r \approx \mathcal{O}(s^{d-1})$, and such an S-box will be "good" (unless s is very small). We will see that the actual contribution of the S-boxes to the complexity of the attacks described in this paper is approximatively $\Gamma = ((t-r)/s)^{\lceil (t-r)/s \rceil}$. It is possible to show that for a random S-box, the smallest value of Γ that can be achieved will be double-exponential in s. However it will be still relatively small for Serpent ($s = 4$). For different reasons, the Serpent and Rijndael S-boxes can both be described by overdefined systems with quite a small Γ.

3.2 Overdefined Equations on the Serpent S-box

We show that 4-bit S-boxes always give an overdefined system of quadratic equations. Consider a 16×37 matrix containing in each row the values of the $t = 37$ monomials $\{1, x_1, .., x_4, y_1, .., y_4, x_1x_2, .., x_1y_1, .., y_3y_4\}$ for each of the $2^s = 16$ possible entries $x = (x_1, .., x_4)$. The rank of this matrix is at most 16, therefore whatever is the S-box, there will be at least $r \geq 37 - 16 = 21$ quadratic equations. This is a very overdefined system since $21 \gg 4$. We have $t/r \approx 1.75$ and $\Gamma = ((t-r)/s)^{\lceil (t-r)/s \rceil} = 2^{8.0}$. We note that a smaller t/r would be achieved with cubic equations on this S-box, but Γ would be much bigger then. It is also possible to consider bi-affine equations. In this case we have $t = 25$ and $r \geq 25 - 16 = 9$ which is still overdefined, it gives the same value of $\Gamma = 2^{8.0}$.

3.3 Overdefined Equations on the Rijndael S-box

For Rijndael we have $s = 8$. It is easy to see that with the method described above in Section 3.2, a random S-box on 8 bits will give $r = 0$ because $2^s = 256$

is bigger than the number 137 of possible quadratic terms. We see that $s = 8$ is quite big compared to Serpent: there are $(2^8)! \approx 2^{1684}$ bijective S-boxes on 8 bits, compared with only $(2^4)! \approx 2^{44}$ for $s = 4$. We don't expect any useful properties to happen by chance. Still, the design of the Rijndael S-box induces a lot of algebraic structure, see [4,2]. This yields very special properties.

Rijndael S-box is a composition of the "patched" inverse in GF(256) with 0 mapped on itself, with a multivariate affine transformation $GF(2)^8 \rightarrow GF(2)^8$. Following [4] we call these functions g and f respectively, and denote $S = f \circ g$. Let x be an input value and $y = g(x)$ the corresponding output value. We also note $z = S(x) = f(g(x)) = f(y)$. According to the definition of the S-box:

$$\forall x \neq 0 \qquad 1 = xy$$

This equation gives, in turn, 8 multivariate bi-linear equations in 8 variables and this leads to 8 bi-affine equations between the x_i and the z_j. It is possible to see that 7 of these equations are true with probability 1, and the 8th is true with probability 255/256. The existence of these equations for g and S is obvious. Surprisingly, much more such equations exist. For example we have:

$$x = y * x^2$$

Since $x \mapsto x^2$ is linear, if written as a set of 8 multivariate functions, the above equation gives 8 bi-affine equations between the x_i and the y_j, and, in turn, between the x_i and the z_j. Adding the fact that the above equation is symmetric with respect to the exchange of x and y, we get 16 bi-affine equations true with probability 1 between the x_i and the z_j.

From the above we have 23 quadratic equations between x_i and the z_j that are true with probability 1. We have explicitly computed these equations (see the extended version of this paper), verified that they are all linearly independent, and also that there are no more such bi-affine equations[2]. The number of terms present in these equations is $t = 81$. These terms are: $\{1, x_1, \ldots, x_8, z_1, \ldots, z_8, x_1z_1, \ldots, x_8z_8\}$, and there is no terms x_ix_j or z_iz_j. We get $t/r \approx 3.52$ and $\Gamma \approx 2^{22.9}$, much more than for Serpent.

An Additional 24th Equation: We observe that in Rijndael S-box, if x is always different than 0, there 24 linearly independent quadratic equations. For one S-box, the probability of this 24th equation to be true is 255/256. We are interested in probability that it is true for **all** S-boxes in the execution of Rijndael (i.e. we have $x \neq 0$ everywhere). As it has been already pointed out by the authors of [8], this probability is quite big. It is in fact:

$$(255/256)^{4 \cdot N_b N_r + 4 \cdot \lceil \frac{N_b(N_r+1)-N_k}{N_k} \rceil + 4 \cdot 1_{N_k=8} \cdot \lceil \frac{N_b(N_r+1)-N_k-4}{N_k} \rceil}$$

[2] If we square the equation $x = x^2 * y$ we obtain successively $x^2 = x^4 * y^2, \ldots, x^{128} = x * y^{128}$. It can be seen that each of them also gives 8 bi-affine equations. However, since the square is multivariate linear, each of them produces **the same** 8 equations, modulo a linear combination.

This gives between 1/2 for the smallest Rijndael 128 bits and about 1/9 for the biggest 256-bit version. Therefore, if an attack works better with 24 equations, and uses only one (or two) executions of the cipher it will be interesting to use $r = 24$ and repeat the whole attack a few times. Otherwise, we use $r = 23$.

Fully Quadratic Equations: It is possible to see that if we consider fully quadratic equations, not only bi-affine, for each S-box of Rijndael there are $r = 39$ quadratic equations with $t = 137$. The additional 16 equations come from the following two equations:

$$\begin{cases} x^4 y = x^3 \\ y^4 x = y^3 \end{cases}$$

However, when $r = 39$, $t = 137$, we have $\Gamma \approx 2^{47.0}$ instead of $2^{22.9}$ and we always obtained worse results in our attacks, than with $r = 23$, $t = 81$,

About Inverse-Based S-boxes: In general, it is easy to see that if the S-box on s bits is an affine transformation of the inverse function in $GF(2^s)$, then it will give $3s - 1$ bi-affine equations true with probability 1, and one additional equation true with probability $1 - \frac{1}{2^s}$. We conjecture for all s there are no more such equations (we verified this for several s). Up till now, it seemed a very good idea to use such S-boxes: the inverse function (and its affine equivalents) has meaningful **optimality** results with regard to linear, differential and high-order differential attacks, see [2,16]. However in our computer simulations, done for many permutations including all the possible powers in $GF(2^s)$, the inverse (and its equivalents) was always the **worse** in terms of the number of such bi-affine equations. It is an open problem to find any other non-linear function $GF(2^s) \to GF(2^s)$ that admits so many equations, for some $s > 0$. Therefore, we do not advocate to use such S-boxes even if they are probably still very secure.

Related Work: The equations we have found for the Rijndael S-box are exactly of the same type and of very similar origin, as the equations that Jacques Patarin have discovered in 1988 for the Matsumoto-Imai cryptosystem [17]. The existence of such equations for Rijndael S-boxes have been first discovered (but not published) by Courtois, Goubin and Patarin, as soon as Rijndael have been proposed as AES in 2000. Recently, in [14], Murphy and Robshaw pointed out that it is more interesting to manipulate equations over $GF(256)$. It leads to systems that are identical (or very similar) in terms of the number of equations and number of variables involved. However, the number of different monomials t present is lower, which is expected to give better results for our attacks.

4 MQ Attacks on Block Ciphers

Given an SA-cipher with S-boxes that can be described in terms of some algebraic equations, recovering the key can be written as a problem of solving a system of

such equations. If these equations are multivariate quadratic, we call this "the MQ attack". Such equations exist for Rijndael and Serpent, as shown above in Sections 3.3 and 3.2, respectively.

4.1 Attack Scenarios

There are many ways in which the MQ attack can be applied. The system of equations should be constructed in such a way that it has exactly one solution. A system that has one solution on average is sufficient in practice, and if there are a few solutions, prior to the solving stage, we would guess and fix a few bits.

First (General) Attack Ignoring the Key Schedule. This attack is designed for any XSL-cipher, whatever is the key schedule. For simplification we only consider the known plaintext attack. There are $(N_r + 1)$ keys K_i that are of the same size as a plaintext, and we need enough equations to determine them uniquely. Hence we need $(N_r + 1)$ known plaintexts. This attack scenario will be used in Section 6.

Second (Specific) Attack Using the Key Schedule. This attack is less general and relies on the fact that the key schedules in Rijndael and Serpent are very similar to the ciphers themselves: they use a combination of affine transformations and (the same) S-boxes. Due to the lack of space, we only study the first (more general) scenario and the second will be studied in a separate paper.

Stronger Attack Scenarios. If attacks based on MQ are possible, i.e. there are efficient methods to solve quadratic equations, then they allow to attack block ciphers in very strong scenarios. For example ciphertext-only attacks will be possible if the attacker is able to characterize the redundancy of the plaintext in terms of quadratic equations.

4.2 Direct MQ Attack on Rijndael and Serpent:

It can be seen that in the second attack scenario, the problem of recovering the key of the 128-bit Rijndael, amounts to solving a system of 8000 quadratic equations with 1600 variables. See Appendix A for details. Similarly, the 128-bit Serpent would give a system of $(N_r + 1)Br + N_rBr = 43680$ equations with $(N_r + 1)Bs + (N_r - 1)Bs = 8192$ variables.

In the remaining part of the paper we study solving such (and similar) systems of equations. Our results are given in Sections 5.2 and 7.

5 Generic Methods for Solving Multivariate Quadratic Equations

MQ is known to be an NP-hard problem [23]. Several public key cryptosystems are based on MQ, for example HFE [19]. However, little is known about the actual hardness of MQ in practice. From the above it is clear that if this problem was very easy for 1600 variables, then Rijndael would be broken. With current attacks, factoring a 1600-bit RSA modulus provides a security level slightly lower than 2^{128} [24]. Therefore, MQ should be at least as hard as factoring.

5.1 Solving MQ with the XL Algorithm

In [22] Shamir and Kipnis made an important discovery about the MQ problem: solving it should be much easier for overdefined systems[3]. This idea has been developed and consolidated in [23]. An algorithm called XL is developed for this problem. It seems that for a random system of quadratic equations over $GF(2)$ (or one that looks random) that has a unique solution, the XL method should always work (but maybe not for some very special systems). In [13] T.T. Moh states that "From the theory of Hilbert-Serre, we may deduce that the XL program will work for many interesting cases for D large enough". From [23] it appears that XL would be polynomial for very overdefined systems, and it seems that a variant of XL might even be subexponential in general (not only for overdefined systems). However, very little is known about the actual behaviour of XL for very big systems of equations and one can only talk about conjectured complexities.

5.2 First Attempt to Cryptanalyse Rijndael with XL

For the 128-bit Rijndael with 128-bit key, following the Theorem A.3.1, we get a system of $m = 8000$ equations with $n = 1600$ variables. Following the complexity evaluation of XL from [23], the complexity would be about $\binom{n}{n/\sqrt{m}}^{\omega} \approx 2^{330}$, assuming $\omega = 2.376$, the best known Gaussian reduction exponent, see [3].

This attack fails because for a random system of quadratic $R_{ini} = m = 8000$ equations with $n = 1600$ variables, we have about $T_{ini} \approx n^2/2 \approx 2^{20}$ terms. This gives $R_{ini}/T_{ini} \approx 2^{-7.3}$ that is very small and the XL algorithm has to do extensive work in order to achieve an expanded system with $R/T \approx 1$. It is easy to see that in our whole system $T_{ini} \approx (8 \cdot 32 + 8 \cdot 32 + 8 + 32 + 8)(N_r \cdot 4 \cdot N_b)$ and this gives only $R_{ini}/T_{ini} \approx 2^{-3.5}$. Therefore there **should** be a better attack. In the next Section 6.2 we will write the quadratic equations in a different way in order to achieve an even higher value of R_{ini}/T_{ini}.

6 XSL Attack on Block Ciphers

In this section we will write a system of equations that describe uniquely the secret key of the cipher, following the first attack scenario from Section 4.1, that

[3] In this paper we will show that if the MQ is sparse, it is even much easier to solve.

does not depend on the key schedule. In order to solve these equations, we are going to introduce an improved version of the XL approach from [23], that takes advantage of their specific structure and sparsity. We call it "the XSL algorithm" where XSL stands for: "e**X**tended **S**parse **L**inearization" or "multiply(**X**) by **S**elected monomials and **L**inearize". In the XL algorithm, we would multiply each of the equations by all possible monomials of some degree $D-2$, see [23]. Instead we will only multiply them by carefully selected monomials. It seems that the best thing to do, is to use products of monomials, that already appear in other equations.

6.1 Final Step and Working Condition of the XSL Attacks

In [23], when $R \geq T$, we have as many equations as the number of terms that appear in these equations and the big system is expected to be solved by adding a new variable for each term, and solving a linear system (doing this is known as linearization). There is no need to have R much bigger than T. because obviously, the number of linearly independent equations (denoted later by $Free$, cannot exceed T. In the original paper about XL [23], the system was solved when $T - Free$ was a small number. Still it is easy to see that both XL and XSL algorithms can be extended to the case when $T - Free$ is very big (!).

Let x_1 be a variable, and let T' be the number of terms that can be multiplied by x_1 and still belong to the set of T terms. Now we assume that $Free \geq T - T' + C$ with a small C. We apply the following algorithm called "the T' method".

1. By one single gaussian elimination we bring the system to a form in which each term is a known linear combination of the terms in T'.
2. We do the same pre-computation two times, for example with T' defined for x_1 and separately for x_2.
3. In each of the two systems, we have a subsystem of C equations that contain only terms of T'. These new equations are probably **not** of the same kind that the initial equations generated in XL-like attacks: only combining all the equations one can obtain some information about the solution.
4. In each of the two subsystems of exceeding C equations, we multiply each equation by x_1 and x_2, respectively. Then we substitute the expressions from point 1 in these to get some **other** equations that contain only terms of T', but for the other variable. These equations are expected to be new and different. First because the equations from point 2 are believed to contain "some information" about the solution, and moreover if we are over $GF(2)$ we will interact with the equation of the field GF(2) that is not necessarily done elsewhere. We have done some computer simulations that show that this heuristic works very well. See also Appendix B for an example.
5. Thus, if at the beginning $Free \geq C + T - T'$ we can "grow" the number of equations. For now we expect to up to $2C$ additional equations.
6. We expect that the number of new equations grows at exponential rate[4].

[4] However, even if it grows by 1 each time, the attack will work as predicted.

7. If the initial system had a unique solution, we expect that we will end up with $Free = T$ or $Free = T - 1$.
8. For each equation containing only terms in T', the cost to compute a derived additional equation will be about T'^2. Since there are T' equations missing, we expect to do about T'^3 additional operations in the attack, which can probably be reduced to T'^ω and thus will be smaller than T^ω.
9. If the whole attack fails one should try with another couple of variables instead of x_1 and x_2, or use three variables from the start (and three systems). We conjecture that three variables should always be sufficient. The number of possibilities grows very fast with the number of variables, a new equation obtained with one variable can be immediately transformed and expanded with **all** the other variables.

For example, in our attack on Rijndael 256 bits given in Section 7.1, we have $T \approx 2^{125}$ and $T' \approx 2^{114}$. The attack is expected to work as long as $Free > T - T'$.

6.2 Core of the XSL Attacks

In this version of the XSL attack we assume that the system of equations for each S-box is overdefined and $r \geq 2s$. Let S be the total number of S-boxes in our attack. Since we are going to use the most general attack scenario described in 4.1 that ignores the key schedule of the cipher, we will have to consider $N_r + 1$ executions of the cipher, see Section 4.1. S will be equal to

$$S = B \cdot N_r(N_r + 1).$$

Equations on the S-boxes and Their Multiples

Let A be an S-box of a XSL-cipher, called "active S-box". We write:

$$0 = \sum \alpha_{ijk} X_{i\ j} Y_{i\ k} + \sum \beta_{ij} X_{i\ j} + \sum \gamma_{ij} Y_{i\ j} + \delta.$$

The total number of terms (i.e. all monomials including the constant) that appear in these equations is small, only t (most of them of the form $X_{i\ j} Y_{i\ k}$). For this reason (unlike in Appendix A) we use both the original variables $X_{i\ j}$ and $Y_{i\ k}$.

We will not use these equations directly, but we will, from these equations, and separately for each S-box, choose some $t - r$ terms as a basis, and write the expression of each of the remaining r terms as a linear combination of the $(r - t)$ terms for the same S-box. We will choose a basis such that all the terms $X_{i\ j}$ and $Y_{i\ j}$ are not in the basis and such that 1 is in the basis. This is possible because $r \geq 2s$. Each time, in the attack we want to use one of the other r terms, we will directly write them as the linear combination of the elements of the basis. We define $[X_{i\ j}]$ and $[Y_{i\ j}]$ as precisely these linear combinations of the $(t - r)$ elements of the basis.

Note: This can be called "a compact version of the first XSL attack." A different approach is possible that uses all the t terms for each S-box (and later their products). This gives different results and will be studied in a separate paper.

Products of Terms

The critical parameter of our attack will be $P \in \mathbb{N}$. We will manipulate products of up to P terms that come from P different S-boxes. The total number of terms used in the attack is about:

$$T \approx (t-r)^P \cdot \binom{S}{P}$$

Moreover, we have (see the definition of T' given in Section 6.1 above):

$$T' \approx t'(t-r)^{P-1} \cdot \binom{S-1}{P-1}$$

with $t' < t$ being the number of terms in the basis for one S-box, that can be multiplied by some fixed variable $X_{i\ j}$, and are still in the basis. For example for Rijndael we use $r = 23$, $t = 81$, and get $t' = 9$.

6.3 Equations on the Diffusion Layers

We will construct a set of equations in such a way that they can be multiplied by many products of terms, and that all the resulting product can be written using only the products of up to P terms, that are taken in the respective bases we have chosen for P different S-boxes. We will get equations that are linear combinations of the T monomials, as defined above. It seems that the best way to attack our problem is to completely eliminate all the key variables and write all possible equations of the form:

$$[X_{i\ j}] \oplus \sum \alpha_j [Y_{i-1\ j}] = [X'_{i\ j}] \oplus \sum \alpha_j [Y'_{i-1\ j}] = [X''_{i\ j}] \oplus \sum \alpha_j [Y''_{i-1\ j}] = \dots$$

The expressions $[X_{i\ j}]$ and $[Y_{i\ j}]$ have been defined above, they are linear combinations of quadratic terms that are the elements of the basis.

We have $N_r(N_r+1)(Bs)$ such equations. Each of these equations, called "active equation", will be multiplied by products of terms for some $(P-1)$ "passive" S-boxes. Here we need to exclude the terms for a few neighbouring S-boxes (i.e. that have common variables with the active equation), it does not change a lot the number of equations generated. The number of new equations is called R. It is approximatively:

$$R \approx N_r(N_r+1)(Bs) \cdot (t-r)^{P-1} \cdot \binom{S}{P-1} \approx S \cdot s\,(t-r)^{P-1} \cdot \binom{S}{P-1}$$

6.4 Expected Complexity of the XSL Attack

The goal of the attack is to obtain $R > T - T'$. It gives:

$$Ss\,(t-r)^{P-1} \binom{S}{P-1} > (t-r)^P \binom{S}{P} - t'\,(t-r)^{P-1} \binom{S-1}{P-1}$$

$$\frac{S^2 s}{S-P+1} > \frac{(t-r)S}{P} - t'$$

We will assume that $P \ll S$ (S is usually quite big) and thus $S-P+1 \approx S$.

$$s > \frac{(t-r)}{P} - \frac{t'}{S}$$

$$s + \frac{t'}{S} > \frac{(t-r)}{P}$$

We see that this condition can always be satisfied, if P is sufficiently big. We get that:

$$P \geq \frac{(t-r)}{s + \frac{t'}{S}} \quad (\#)$$

Note: From this it might seem that the XSL attack will work for $r = 0$, however we have previously assumed that $r \geq 2s$. Therefore $r = 0$ is not possible.

Let T^ω, be the complexity of the Gaussian reduction, the complexity of the attack is about:

$$WF \;=\; T^\omega \;\approx\; (t-r)^{\omega P}\binom{S}{P}^{\omega} \;\approx\; (t-r)^{\omega P}\left(B \cdot N_r^2\right)^{\omega P} \;\approx\; \left(\frac{t-r}{s} \cdot Bs \cdot N_r^2\right)^{\omega P}$$

Now let us apply the estimation (#). In practice the value $\frac{t'}{S}$ will be very small, and vanishes for big ciphers (big N_r or big B). Therefore we assume that $P = \lceil \frac{t-r}{s} \rceil + o(1)$. It gives the following (rough) estimation of the complexity of the XSL attack on block ciphers, again assuming that $r \geq 2s$:

$$WF \;\approx\; \left(\frac{t-r}{s}\right)^{\omega\left\lceil \frac{t-r}{s}\right\rceil + o(1)} \cdot (Bs \cdot N_r^2)^{\omega\left\lceil \frac{t-r}{s}\right\rceil + o(1)}$$

$$WF = \Gamma^\omega \cdot (\text{Block size})^{\omega \frac{t-r}{s}} \, (\text{Number of rounds})^{2\omega \frac{t-r}{s}}$$

This is polynomial in the block size and the number of rounds. The constant part depends on Γ that depends only on the parameters of the S-box used in the cipher, and is in general double-exponential in s, see Section 3.1. For a given cipher the constant part Γ^ω in the complexity of XSL will be fixed (but usually very, very big).

6.5 Actual Complexity of the XSL Attacks

In the above derivation we assumed that all the equations in R are linearly independent and this implies that for some fixed P the attack will always work for any number of rounds. From our early simulations[5] it seems that the attack works for many rounds and then it fails. Thus P would rather increase (but slowly) with the number of rounds.

[5] See, the second XSL attack, preliminary version, . `http://eprint.iacr.org/2002/044/`

If P were constant, for a fixed S-box that have many overdefined equations, the XSL attack will be polynomial in the number of rounds. Even if P grows slowly, and XSL is subexponential, it would be already an important breakthrough, as the complexity of the classical attacks on block ciphers, such as linear or differential cryptanalysis, grows exponentially in the number of rounds (and so does the number of required plaintexts).

In fact there is another way to see that there is a risk that the problem to break Rijndael might be subexponential when the number of rounds grows. Indeed, in this paper we show how to write Rijndael as an overdefined system of quadratic equations, with size that is linear in N_r, see Appendix A. The problem of solving such a system of quadratic equations over $GF(2)$ might already be subexponential using the original XL algorithm from [23]. See Section A.4 for more comments on this. Finally, our equations from Appendix A are also overdefined, sparse and have a lot of structure, which also should help the attacker.

7 Consequences of the XSL Attack

7.1 Application to Rijndael

We consider the 128-bit Rijndael with 256 bit keys. We have $N_b = 4, N_k = 8, N_r = 14, s = 8, r = 23, t = 81, t' = 9, S = 3360$, then for $P = (t - r)/(s + t'/S) = 8$, computed following (#), we get $T \approx 2^{125}, T' \approx 2^{114}, R \approx 2^{125}$ with $\frac{R}{T-T'} = 1.106$. The result is:

$$T^{\omega} \approx 2^{298}$$

This version of the XSL attack fails also for other variants of AES. We expect that much better results should be obtained with the combination of the second XSL attack[5], with equations over $GF(256)$ as proposed by Murphy and Robshaw [14]. It is not excluded that even AES-128 could be broken: for $N_b = 4, N_k = 4, N_r = 10, s = 8, r = 24, t = 41, t' = 4, S = 201$, our early estimation gives that for $P = 3$ we have $T \approx 2^{36}, T' \approx 2^{27}, R \approx 2^{36}, R' \approx 2^{33}$, and $\frac{R+R'}{T-T'} = 1.01$. If this attack worked as well as expected[5], the resulting complexity would be $T^{\omega} \approx 2^{87}$.

7.2 Application to Serpent

For 256-bit Serpent, we have $N_r = 32, s = 4, r = 21, t = 37, t' = 5, S = 33792$. Then for $P = (t - r)/(s + t'/S) = 5$, we get $T \approx 2^{88}, T' = 2^{74}, R = 2^{88}, \frac{R}{T-T'} = 1.25$. The result is:

$$T^{\omega} \approx 2^{210}$$

It seems that the XSL attack breaks 256 bit Serpent. Though it is obtained with the fairly theoretical $\omega = 2.376$ from [3], using Strassen's exponent we still get 2^{245}. It is however not proven that the attack will work as predicted for $P = 5$. Though XSL attacks will probably always work for some P, we considered the minimum value P for which $\frac{R}{T-T'} \geq 1$. This condition is necessary, but probably not sufficient. A small change (e.g. increase by 1 or 2) in P would lead to a dramatic overload in the complexity, going beyond the exhaustive search.

7.3 Consequences for the Design of Block Ciphers

There are two complementary approaches in the block cipher design that could be seen in the AES contest. Either a cipher is designed with a very small number of rounds that are very complex (for example in DFC), or it has a large number of rounds that are very simple (for example in Serpent). In [27] the authors warn that: "an attack against Serpent may hold for any set of (random) S-boxes". It seems that we have found such an attack and using many layers of very simple S-boxes is maybe not such a very good idea. Still, a correct choice of parameters will prevent the attacks.

For different reasons, the XSL attack is also applicable to all ciphers in which the only non-linear part is the inverse function in $GF(2^s)$, with a small s. Therefore, ciphers such as Rijndael and Camellia should either use s that is sufficiently large, maybe $s > 8$, or consider different S-boxes. This last possibility should give new optimal designs of S-boxes, not only close to optimal in terms of linear and differential attacks, but also incorporating our new criterion, i.e. having a big value of Γ, for example $\Gamma > 2^{32}$.

Even if the attacks of the present paper have not yet been tested on really big examples, they are an important threat for ciphers such as Rijndael, Serpent and Camellia. We propose that all block ciphers should apply the following criterion (due originally to Shannon [25]): The attacker should not be able to write a system of algebraic equations of simple type and of reasonable size, that completely characterizes the secret key. It can be achieved if one uses at least a few (relatively) big randomly generated S-boxes.

8 Conclusion

In this paper we point out an unexpected property of Rijndael and Serpent: they can be described as a system of overdefined and sparse quadratic equations over $GF(2)$. It was known from [23] that solving such systems is easier if they are overdefined, and the problem might even be subexponential (conjectured) for small fields such as $GF(2)$. It is therefore possible that the security of Rijndael and Serpent would not grow exponentially with the number of rounds.

A direct application of the XL attack from Eurocrypt'00 is extremely inefficient. Knowing that the equations are not only overdefined, but also sparse and structured, we have introduced a new method called XSL. If the XSL attack works as well predicted, it might (marginally) break Serpent 256 bits. With equations over GF(2) we do not get an efficient attack for AES. However a different version of XSL combined with equations over GF(256) is expected to give much better results. In order to prevent such attacks, we propose that at least a few S-boxes in a cipher should not be described by a small system of overdefined multivariate equations.

Acknowledgments

The following people have contributed to this paper: Mehdi-Laurent Akkar, Don Coppersmith, Louis Goubin, Philip Hawkes, Jacques Patarin, Pham Thi Minh Tam, and the anonymous reviewers of Asiacrypt.

References

1. Ross Anderson, Eli Biham and Lars Knudsen: *Serpent: A Proposal for the Advanced Encryption Standard.* Available from `http://www.cl.cam.ac.uk/~rja14/serpent.html`
2. Anne Canteaut, Marion Videau: *Degree of composition of highly nonlinear functions and applications to higher order differential cryptanalysis;* Eurocrypt 2002, LNCS 2332, Springer.
3. Don Coppersmith, Shmuel Winograd: "Matrix multiplication via arithmetic progressions"; J. Symbolic Computation (1990), 9, pp. 251-280.
4. Joan Daemen, Vincent Rijmen: *AES proposal: Rijndael;* The latest revised version of the proposal is available on the internet, `http://csrc.nist.gov/encryption/aes/rijndael/Rijndael.pdf`
5. Nicolas Courtois, Louis Goubin, Willi Meier, Jean-Daniel Tacier: *Solving Underdefined Systems of Multivariate Quadratic Equations;* PKC 2002, LNCS 2254, Springer, pp. 211-225.
6. Nicolas Courtois: *The security of Hidden Field Equations (HFE)*; Cryptographers' Track Rsa Conference 2001, San Francisco 8-12 April 2001, LNCS2020, Springer-Verlag, pp. 266-281.
7. Horst Feistel: *Cryptography and computer privacy;* Scientific American, vol. 228, No. 5, pp. 15-23, May 1973.
8. Niels Ferguson, Richard Schroeppel and Doug Whiting: *A simple algebraic representation of Rijndael;* SAC'01, page 103, LNCS 2259, Springer.
9. Niels Ferguson, John Kelsey, Stefan Lucks, Bruce Schneier, Mike Stay, David Wagner, Doug Whiting: Improved Cryptanalysis of Rijndael, FSE 2000, Springer.
10. J.B. Kam and G.I. Davida: *Structured design of substitution-permutation encryption networks;* IEEE Trans. on Computers, Vol. C-28, 1979, pp.747-753.
11. Lars R. Knudsen, Vincent Rijmen: *On the Decorrelated Fast Cipher (DFC) and its Theory*; FSE'99, Springer, LNCS 1636, pp. 81-94.
12. Michael Luby, Charles W. Rackoff, *How to construct pseudorandom permutations from pseudorandom functions;* , SIAM Journal on Computing, vol. 17, n. 2, pp. 373-386, April 1988.
13. T.T. Moh: *On The Method of XL and Its Inefficiency Against TTM*, available at `http://eprint.iacr.org/2001/047/`.
14. S. Murphy, M. Robshaw: *Essential Algebraic Structure within the AES,* Crypto 2002, Springer.
15. Moni Naor and Omer Reingold: *On the construction of pseudo-random permutations: Luby-Rackoff revisited;* Journal of Cryptology, vol 12, 1999, pp. 29-66.
16. Kaisa Nyberg: *Differentially Uniform Mappings for Cryptography;* Eurocrypt'93, LNCS 765, Springer, pp. 55-64.
17. Jacques Patarin: *Cryptanalysis of the Matsumoto and Imai Public Key Scheme of Eurocrypt'88*; Crypto'95, Springer-Verlag, pp. 248-261.
18. Jacques Patarin: *Generic Attacks on Feistel Schemes* ; Asiacrypt 2001, LNCS 2248, Springer, pp. 222-238.

19. Jacques Patarin: *Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): two new families of Asymmetric Algorithms;* in Eurocrypt'96, Springer Verlag, pp. 33-48.
20. Jacques Patarin, Nicolas Courtois, Louis Goubin: *Improved Algorithms for Isomorphism of Polynomials*; Eurocrypt 1998, Springer-Verlag.
21. Adi Shamir, Alex Biryukov: *Structural Cryptanalysis of SASAS;* Eurocrypt 2001, LNCS 2045, Springer, pp. 394-405.
22. Adi Shamir, Aviad Kipnis: *Cryptanalysis of the HFE Public Key Cryptosystem*; In Advances in Cryptology, Proceedings of Crypto'99, Springer-Verlag, LNCS.
23. Adi Shamir, Jacques Patarin, Nicolas Courtois, Alexander Klimov, *Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations*, Eurocrypt'2000, LNCS 1807, Springer, pp. 392-407.
24. Robert D. Silverman: *A Cost-Based Security Analysis of Symmetric and Asymmetric Key Lengths;* RSA Lab. report, `www.rsasecurity.com/rsalabs/bulletins/bulletin13.html`.
25. Claude Elwood Shannon: *Communication theory of secrecy systems;* , Bell System Technical Journal 28 (1949), see in patricular page 704.
26. Serge Vaudenay: *Provable Security for Block Ciphers by Decorrelation;* Technical Report LIENS-98-8, ENS, France, available at `http://lasecwww.epfl.ch/query.msql?ref=Vau98b`.
27. Serge Vaudenay, Shiho Moriai: *On the Pseudorandomness of Top-Level Schemes of Block Ciphers;* Asiacrypt 2000, LNCS 1976, Springer, pp. 289-302.

A Direct MQ Attack on Rijndael

It is interesting to know how to describe Rijndael as a system of quadratic equations with a minimum number of variables and maximum number of equations. We are in the second attack scenario with one or a few known plaintexts (see Section 4.1).

A.1 Minimizing the Number of Variables for Rijndael

For each round i, we know that there are $4r \cdot N_b$ quadratic equations between the $(Z_{i-1\ j} + K_{i-1\ j})$ and the $(Z_{i\ k})$. They are of the following form:

$$0 = \sum \alpha_{ijk} Z_{i-1\ j} Z_{i\ k} + \sum \alpha_{ijk} [K_{i-1\ j}] Z_{i\ k} + \sum \beta_{ij} Z_{i\ j} + \sum \beta_{ij} [K_{i\ j}] + \gamma.$$

Exception is made for the first round, for which the Z_0 being known, they are of the form:

$$0 = \sum \alpha_{ij} [K_{0\ i}] Z_{1\ j} + \sum \beta_i Z_{1\ i} + \sum \gamma_i [K_{0\ i}] + \delta.$$

Finally, for the last round, the $X_{N_r\ k}$ will be expressed as a sum of the known ciphertext $Z_{N_r+1\ k}$ and $[K_{N_r\ k}]$, giving the equations of the form:

$$0 = \sum \alpha_{ij} Z_{N_r-1\ i} [K_{N_r\ j}] + \sum \alpha_{ij} [K_{N_r-1\ i}][K_{N_r\ j}] + \sum \beta_i Z_{N_r-1\ i} +$$
$$+ \sum \beta_i [K_{N_r-1\ i}] + \sum \gamma_i [K_{N_r\ i}] + \delta.$$

In all we will get $4 \cdot r \cdot N_r \cdot N_b$ quadratic equations over $GF(2)$. The number of variables $Z_{i\ j}$ is only $4s \cdot (N_r - 1) N_b$.

A.2 Using the Key Schedule

We have:

$$X_{i+1\ j} = Z_{i\ j} \oplus [K_{i\ j}] \quad \text{for all } i = 0..N_r. \tag{1}$$

In order to define what are the $[K_{i\ j}]$ we need to choose a basis for the $K_{i\ j}$. From the key schedule [4] it is obvious that one may take as "true key variables" all the N_k variables from the first round, then all the first columns of each consecutive key states, and if $N_k = 8$, also the 5th columns. By inspection we see that the number of "true key variables" is:

$$L_k = \begin{cases} 32 \cdot (N_k + \lceil (N_r \cdot N_b + N_b - N_k)/N_k \rceil) & \text{if } N_k \neq 8 \\ 32 \cdot (N_k + \lceil (N_r \cdot N_b + N_b - N_k)/4 \rceil) & \text{if } N_k = 8 \end{cases}$$

For example, for 128-bit Rijndael with $H_k = 128$ we have $L_k = 32 \cdot (4+10) = 448$.

Additional Equations. We call "redundant true variables" all the $L_k - H_k$ additional variables that are determined by some initial subset of H_k unexpanded variables. From the key schedule we see that for each of these $L_k - H_k$ "redundant true variables" we may write $r = 23$ (or 24) quadratic equations. Each of the "redundant true" key state columns is a XOR of one the previous columns, a parallel application of 4 S-boxes to another column, and of a constant. Thus these equations are of the form:

$$\sum \alpha_{ijkl} [K_{i\ j}][K_{k\ l}] + \sum \beta_{ij} [K_{i\ j}] + \gamma. \tag{2}$$

The number of these equations is:

$$r \cdot \frac{L_k - H_k}{s}$$

A.3 Summary of the Equations and Concrete Applications

Theorem A.3.1 (Reduction Rijndael $\rightarrow$ MQ). The problem of recovering the secret key of Rijndael given about one pair plaintext/ciphertext can be written as an overdefined system of

$$m = 4 \cdot r \cdot N_b \cdot N_r + r(L_k - H_k)/s$$

sparse quadratic equations with the number of unknowns being:

$$n = 4 \cdot s \cdot (N_r - 1)N_b + L_k.$$

Concrete Application to Rijndael: We will use fully quadratic equations obtained in Section 2. We have $r = 39$ and $t = 137$, however since this attack will only require 1 or 2 known plaintexts, we may assume $r = 40$ (see Section 2).

• Thus for the 128-bit Rijndael with 128-bit key, we can write the problem of recovering the key as a system of 8000 quadratic equations with 1600 variables.

• For the 256-bit Rijndael with 256-bit key, we get a system of 22400 quadratic equations with 4480 variables.

In general, no efficient algorithms are known to solve such big systems of equations. In fact however, they are sparse and have regular structure, see Section 5.2. In Section 6.2 we write quadratic equations in a different way, more suitable for our the XSL attacks.

A.4 Theoretical Consequences for Rijndael and AES

The above reduction has already some very important consequences for Rijndael and AES. We consider the security of some generalized version of Rijndael in which the number of rounds N_r increases and all the other parameters are fixed.

On one hand, in all general attacks previously known against such ciphers, for example in linear or differential attacks, the security grows exponentially with N_r. There are also combinatorial attacks such as square attack, but these will simply not work if N_r is sufficiently large. On the other hand, we observe that the number of variables (and the number of equations) in the reduction is **linear** in the number of rounds N_r. Therefore, if the MQ problem is subexponential, which seems possible from the XL paper [23], to break Rijndael would also be subexponential[6], i.e. the security would **not** grow exponentially with the number of rounds N_r.

Remark 1: It is important to see that the result would not be the same if the reduction were for example quadratic in N_r. In this case XL could be subexponential, for example in $n^{\sqrt{n}}$ but the Rijndael could still be fully exponential, for example in $(N_r^2)^{N_r}$.

Remark 2: It seems that the same remark will hold for any block cipher composed with rounds of fixed type: obviously each of them can always be written as a set of quadratic equations. However, in this case, the size of the system (even for one round) will be so huge that there will be no hope for any practical attacks.

B A Toy Example for the "T' Method"

This is a concrete working example for the final step of the XSL algorithm called the "T' method". It can also be applied to the XL algorithm.

We have $n = 5$ variables, and thus $T = 16$ and $T' = 10$. We start with a random system that has exactly one solution, and with $Free > T - T'$ and with 2 exceeding equations, i.e. $Free = T - T' + 2$. Here is a system in which T' is defined with respect to x_1:

[6] It also possible that XL is subexponential only on average, and AES gives very special systems.

$$\begin{cases} x_3x_2 = x_1x_3 + x_2 \\ x_3x_4 = x_1x_4 + x_1x_5 + x_5 \\ x_3x_5 = x_1x_5 + x_4 + 1 \\ x_2x_4 = x_1x_3 + x_1x_5 + 1 \\ x_2x_5 = x_1x_3 + x_1x_2 + x_3 + x_4 \\ x_4x_5 = x_1x_2 + x_1x_5 + x_2 + 1 \\ 0 = x_1x_3 + x_1x_4 + x_1 + x_5 \\ 1 = x_1x_4 + x_1x_5 + x_1 + x_5 \end{cases}$$

Here is the same system in which T' is defined with respect to x_2:

$$\begin{cases} x_1x_3 = x_3x_2 + x_2 \\ x_1x_4 = x_3x_2 + x_2 + x_1 + x_5 \\ x_1x_5 = x_2x_4 + x_3x_2 + x_2 + 1 \\ x_3x_5 = x_2x_4 + x_3x_2 + x_2 + 1 + x_4 + 1 \\ x_3x_4 = x_2x_4 + x_1 + 1 \\ x_4x_5 = x_1x_2 + x_2x_4 + x_3x_2 \\ 0 = x_1x_2 + x_2x_5 + x_3x_2 + x_2 + x_3 + x_4 \\ 0 = x_2x_4 \end{cases}$$

We have $rank = 8$. Now multiply the two exceeding equations of the first version of the system by x_1.

$$\begin{cases} 0 = x_1x_3 + x_1x_4 + x_1 + x_1x_5 \\ 0 = x_1x_4 \end{cases}$$

We have $rank = 10$. We get two new linearly independent equations.

We rewrite these equations, using the second system, only with terms that can be multiplied by x_2. Now we have 4 exceeding equations for the second system (two old and two new):

$$\begin{cases} 0 = x_1x_2 + x_2x_5 + x_3x_2 + x_2 + x_3 + x_4 \\ 0 = x_2x_4 \\ 0 = x_2x_4 + x_3x_2 + x_5 + x_2 + 1 \\ 0 = x_3x_2 + x_2 + x_1 + x_5 \end{cases}$$

We multiply these four equations by x_2.

$$\begin{cases} 0 = x_1x_2 + x_2x_5 + x_2x_4 + x_2 \\ 0 = x_2x_4 \\ 0 = x_2x_4 + x_3x_2 + x_5x_2 \\ 0 = x_3x_2 + x_2 + x_1x_2 + x_2x_5 \end{cases}$$

We are not lucky, the second equation is invariant by this transformation. Still we get three new linearly independent equations. We have $rank = 13$.

We rewrite, using the first system, the three new equations with terms that can be multiplied by x_1.

$$\begin{cases} 1 = x_1x_5 + x_2 + x_3 + x_4 \\ 1 = x_1x_2 + x_1x_3 + x_1x_5 + x_2 + x_3 + x_4 \\ 0 = x_3 + x_4 \end{cases}$$

Still $rank = 13$. Then we multiply the three new equations by x_1:

$$\begin{cases} 1 = x_1x_5 + x_1x_2 + x_1x_3 + x_1x_4 \\ 1 = x_1x_5 + x_1x_4 \\ 0 = x_3 + x_4 \end{cases}$$

We have $rank = 14$. We get one more linearly independent equation. The two other are redundant. Now we rewrite the first equation with terms that can be multiplied by x_2:

$$0 = x_1x_2 + x_2x_4 + x_3x_2 + x_1 + x_2 + x_5$$

We have still $rank = 14$. Then we multiply the new equation by x_2:

$$0 = x_2x_4 + x_3x_2 + x_2x_5 + x_2$$

We get another new linearly independent equation. We have $rank = 15$. The rank is the maximum that can be achieved, there are 15 non-zero monomials here, and $rank = 16$ can only be achieved for a system that is contradictory.