Block Ciphers and Systems of Quadratic Equations[⋆]

Alex Biryukov and Christophe De Cannière[⋆⋆]

Katholieke Universiteit Leuven, Dept. ESAT/SCD-COSIC,
Kasteelpark Arenberg 10,
B–3001 Leuven-Heverlee, Belgium
{alex.biryukov, christophe.decanniere}@esat.kuleuven.ac.be

Abstract. In this paper we compare systems of multivariate polynomials, which completely define the block ciphers Khazad, Misty1, Kasumi, Camellia, Rijndael and Serpent in the view of a potential danger of an algebraic re-linearization attack.

Keywords: Block ciphers, multivariate quadratic equations, linearization, Khazad, Misty, Camellia, Rijndael, Serpent.

1 Introduction

Cryptanalysis of block ciphers has received much attention from the cryptographic community in the last decade and as a result several powerful methods of analysis (for example, differential and linear attacks) have emerged. What most of these methods have in common is an attempt to push statistical patterns through as many iterations (rounds) of the cipher as possible, in order to measure non-random behavior at the output, and thus to distinguish a cipher from a truly random permutation. A new generation of block-ciphers (among them the Advanced Encryption Standard (AES) Rijndael) was constructed with these techniques in mind and is thus not vulnerable to (at least a straightforward application of) these attacks. The task of designing ciphers immune to these statistical attacks is made easier by the fact that the complexity of the attacks grows exponentially with the number of rounds of a cipher. This ensures that the data and the time requirements of the attacks quickly become impractical.

A totally different generic approach is studied in a number of recent papers [5, 7], which attempt to exploit the simple algebraic structure of Rijndael. These papers present two related ways of constructing simple algebraic equations that completely describe Rijndael. The starting point is the fact that the only non-linear element of the AES cryptosystem, the S-box, is based on an inverse

[⋆] The work described in this paper has been supported in part by the Commission of the European Communities through the IST Programme under Contract IST-1999-12324 and by the Concerted Research Action (GOA) Mefisto.

[⋆⋆] F.W.O. Research Assistant, sponsored by the Fund for Scientific Research – Flanders (Belgium) function (chosen for its optimal differential and linear properties). This allows to find a small set of quadratic multivariate polynomials in input and output bits that completely define the S-box. Combining these equations, an attacker can easily write a small set of sparse quadratic equations (in terms of intermediate variables) that completely define the whole block-cipher. Building on recent progress in re-linearization techniques [4, 8] which provide sub-exponential algorithms to solve over-defined systems of quadratic (or just low degree) equations, Courtois and Pieprzyk [5] argue that a method called XSL might provide a way to effectively solve this type of equations and recover the key from a few plaintext-ciphertext pairs. The claimed attack method differs in several respects from the standard statistical approaches to cryptanalysis: (a) it requires only few known-plaintext queries; (b) its complexity doesn't seem to grow exponentially with the number of rounds of a cipher. However, no practical attack of this type was demonstrated even on a small-scale example, so far. Research on such attacks is still at a very early stage, the exact complexity of this method is not completely understood and many questions concerning its applicability remain to be answered.

In this paper we will not try to derive a full attack or calculate complexities. Our intention is merely to compare the expected susceptibility of different block ciphers to a hypothetical algebraic attack over $GF(2)$ and $GF(2^8)$. For this purpose we will construct systems of equations for the 128-bit key ciphers[1] Khazad [3], Misty1 [9], Kasumi [10], Camellia-128 [2], Rijndael-128 [6] and Serpent-128 [1] and compute some properties that might influence the complexity of solving them.

2 Constructing Systems of Equations

The problem we are faced with is to build a system of multivariate polynomials which relates the key bits with one or two (in the case of the 64-bit block ciphers) plaintext-ciphertext pairs and which is as simple as possible. The main issue here is that we have to define what we understand by simple.

Since we do not know the most efficient way of solving such systems of equations, our simplicity criterion will be based on some intuitive assumptions:

1. Minimize the total number of *free* terms (free monomials). This is the number of terms that remain linearly independent when considering the system as a linear system in monomials. For example, adding two linearly independent equations which introduce only one new monomial will reduce the number of free terms by one. In order to achieve this, we will try to:
   (a) Minimize the degree of the equations. This reduces the total number of possible monomials.
   (b) Minimize the difference between the total number of terms and the total number of (linearly independent) equations. This is motivated by the fact that each equation can be used to eliminate a term.

[1] For ciphers which allow different key sizes, we will denote the 128-bit key version by appending "128" to the name of the cipher.

2. Minimize the size of individual equations. This criterion arises from the observation that sparse systems are usually easier to solve. Note that point 1 already assures the "global sparseness" of the system and that point 2 adds some local sparseness if it is possible.

Another criterion, which is used in [8] and [4], is to minimize the ratio between the total number of terms and the number of equations. This is equivalent to the criterion above when the system involves all terms up to a certain degree (as would be the case for a random quadratic system, for example). We believe, however, that this criterion is less natural in cases where the number of terms can be reduced, which is the case for the systems considered in this paper.

The most straightforward way of constructing a system of equations for a block cipher is to derive equations for each individual component and to insert them in a single system. In the next subsections we will briefly discuss the contribution of each component.

2.1 S-boxes

In most block ciphers, the S-boxes are the only source of nonlinearity and the equations describing them will be the main obstacle that prevents the system from being easily solved.

For any S-box of practical size, one can easily generate a basis of linearly independent multivariate polynomials that spans the space of all possible equations between the input and the output bits. This is illustrated for a small example in Appendix A.1. In this space we would like to find a set of equations that is as simple as possible (according to our criterion), but still completely defines the S-box. In some cases, this optimal set of equations might be an over-defined system.[2]

Performing an exhaustive search over all possible sets of equations is infeasible, even for small S-boxes. In this paper, we will therefore restrict our search to systems consisting only of equations from the basis. It appears that this restriction still produces sufficiently simple systems for small S-boxes, although the results rapidly deteriorate when the size of the S-boxes increases. Fortunately, many large S-boxes used in practice are derived from simple algebraic functions, and this usually directly leads to simple polynomial systems (see Sect. 3.2, for example). Nothing guarantees however that these systems are optimal and the results derived in this paper should therefore be considered as an approximation only. An efficient way of finding optimal systems for arbitrary S-boxes is still an interesting open problem.

[2] In this paper, we do not consider "over-definedness" to be a criterion on itself. The reason is that it is not clear whether an over-defined system with a lot of free terms should be preferred over a smaller, defined system with less free terms. We note however that the systems of all S-boxes studied below can easily be made over-defined, should the solving algorithm require it.

2.2 *FL*-blocks

Both Misty1 and Camellia-128 contain an additional nonlinear component called FL-block. It is a function of an input word X and a key word K and it is defined as

$$Y_R = X_R \oplus [(X_L \cap K_L) \lll s] \tag{1}$$
$$Y_L = X_L \oplus (Y_R \cup K_R) \tag{2}$$

with X, Y and K $2w$-bit words. The constant s is 0 for Misty1 and 1 for Camellia-128 and the word size w is 16 and 32 for Misty1 and Camellia-128 respectively. The definition above can directly be translated into a system of quadratic equations in $GF(2)$:

$$y_{R,i} = x_{R,i} + y_{L,j} \cdot k_{L,j} \tag{3}$$
$$y_{L,i} = x_{L,i} + y_{R,i} + k_{R,i} + y_{R,i} \cdot k_{R,i} \tag{4}$$

for

$$0 \le i < w \tag{5}$$
$$j = i - s \mod w \tag{6}$$

What should be remembered is that the $2w$ nonlinear equations of this system contain $6w$ linear and $2w$ quadratic terms.

2.3 Linear layers

The linear layers of a block cipher consist of both linear diffusion layers and key additions. Writing a system of equations for these layers is very straightforward. The number of equations and the number of terms in such a linear system are fixed and the only property we can somewhat control is the density of the individual equations, for example by combining equations that have many terms in common.

In cases where a linear layer only consists of bit permutations, we will not insert separate equations, but just rename the variables accordingly.

3 Comparison of Block Ciphers

The block ciphers we intend to compare are Khazad, Misty1, Kasumi, Camellia-128, Rijndael-128 and Serpent-128. The characteristics of these ciphers are summarized in Table 1 and Table 2. For each of them we will construct a system of equations as described in the previous section, and compute the total number of terms and the number of equations. The final results are listed in Table 3.

Table 1. Characteristics of the ciphers (without key schedule).

	Khazad	Misty1	Camellia-128	Rijndael-128	Serpent-128
Block size	64 bit	64 bit	128 bit	128 bit	128 bit
Key size	128 bit	128 bit	128 bit	128 bit	128 bit
Rounds	8	8	18	10	32
S-boxes	384/64	$24 + 48$	144	160	1024
S-box width	4/8 bit	7 bit and 9 bit	8 bit	8 bit	4 bit
Nonlinear order[a]	3/7	3 and 2	7	7	2
FL-blocks	-	10	4	-	-
FL-block width	-	32 bit	64 bit	-	-

[a] The non-linear order of an S-box is the minimal degree required to express a linear combination of output bits as a polynomial function of input bits.

Table 2. Characteristics of the key schedules.

	Khazad	Misty1	Camellia-128	Rijndael-128	Serpent-128
Key size	128 bit	128 bit	128 bit	128 bit	128 bit
Expanded key	576 bit	256 bit	256 bit	1408 bit	4224 bit
S-boxes	432/72	$8 + 16$	32	40	1056
S-box width	4/8 bit	7 bit and 9 bit	8 bit	8 bit	4 bit

3.1 Khazad

Khazad [3] is a 64-bit block cipher designed by P. Barreto and V. Rijmen, which accepts a 128-bit key. It consists of an 8-round SP-network that makes use of an 8-bit S-box and a 64-bit linear transformation based on an MDS code. The 8-bit S-box is in turn composed of 3 times two 4-bit S-boxes called P and Q, which are connected by bit-permutations. The key schedule is a 9-round Feistel iteration based on the same round function as in the cipher itself.

For both P and Q we can generate 21 linearly independent quadratic equations in 36 terms (excluding '1'). As discussed before, we want to find a subset of these equations which forms a system that is as simple as possible. A possible choice can be found in Appendix A. The system describing P consists of 4 equations in 16 terms. For Q, the best set we found is an over-defined system of 6 equation in 18 terms.

Constructing the system

Cipher

Variables. The variables of the system are the inputs and outputs of the 4-bit S-boxes. After combining six 4-bit S-boxes into a 8-bit S-box and renaming the variables according to the bit-permutations, we obtain a system in 32 variables. This is repeated for each of the 64 8-bit S-boxes.

Linear equations. The S-box layers are connected to each other and to the (known) plaintext and ciphertext by 9 64-bit linear layers, each including a key addition. Their contributions in the system are 9 times 64 linear equations.

Nonlinear equations. The only nonlinear equations are the S-box equations. For each 8-bit S-box we obtain a system of 30 equations in 32 linear and 54 quadratic terms. Note that a better system may be obtained if one optimizes the sets of the equations, used in the neighboring 4-bit S-boxes, trying to maximize the number of common terms between the two S-boxes.

Key schedule

Variables. In order to take the key schedule into account, we need to include extra variables for the first 64 bits of the key (called K^{-2}) and for the inputs and outputs of the 4-bit S-boxes in the key schedule (which will include the 64 variables for K^{-1}).

Linear equations. The linear layers between the 9 S-box layers in the key schedule define 8 times 64 linear equations.

Nonlinear equations. Again, the nonlinear equations are just the S-box equations.

The number of equations and terms in the final system are listed in Table 3. Note that we need to build a system for two different 64-bit plaintext-ciphertext pairs in order to be able to solve the system for the 128-bit key.

3.2 Misty1 and Kasumi

Misty1 [9] is a 64-bit block and 128-bit key block cipher designed by M. Matsui. It is an 8-round Feistel-network based on a 32-bit nonlinear function. Additionally, each pair of rounds is separated by a layer of two 32-bit FL-blocks. The nonlinear function itself has a structure that shows some resemblances with a Feistel-network and contains three 7-bit S-boxes S_7, and six 9-bit S-boxes S_9. The key schedule derives an additional 128-bit key K' from the original key K by applying a circular function which contains 8 instances of S_7 and 16 of S_9.

The 9-bit S-box S_9 is designed as a system of 9 quadratic equations in 54 terms (derived from the function $y = x^5$ in $GF(2^9)$) and this is probably the best system according to our criterion. The 7-bit S-box S_7 is defined by 7 cubic equations in 65 terms (see Appendix A), but it appears that its input and output bits can also be related by a set of 21 quadratic equations in 105 terms. In this set, our search program is able to find a subset of 11 equations in 93 terms which completely defines the S-box.

The substitution S_7 was not selected at random, however, as it was designed to be linearly equivalent with the function $y = x^{81}$ in $GF(2^7)$. This information allows to derive a much simpler system of quadratic equations: raising the previous expression to the fifth power, we obtain $y^5 = x^{405} = x^{24}$, which can be written as $y \cdot y^4 = x^8 \cdot x^{16}$. This last equation is quadratic, since y^4, x^8 and x^{16}

are linear functions in $GF(2^7)$. Next, we express x and y as a linear function of the input and output bits of S_7 and obtain a system of 7 quadratic equations in 56 terms over $GF(2)$.[3] This is a considerable improvement compared to what our naive search program found.

Kasumi [10] is a 64-bit block and 128-bit key block cipher derived from Misty1. It is built from the same components and uses S-boxes which are based on the same power functions and are equivalent up to an affine transform. Its structure is slightly different: relevant for the derivation of the equations is the fact that it uses 24 more instances of the 7-bit S-boxes, that the placement of its 8 FL-blocks is different, and that the key schedule is completely linear.

Constructing the system for Misty1

Cipher

Variables. The variables of the system are chosen to be the inputs and outputs of the S-boxes, the output bits of the first pair of FL-blocks, both the input and the output bits of the 6 inner FL-blocks and the input of the last pair of FL-blocks. We expect however that the variables for some of the output bits of the first FL-blocks can be replaced by constants, as about 3/8 of them will only depend on the (known) plaintext bits. This is also true for the output bits of the last FL-blocks.

Linear equations. These include all linear relations between different S-boxes and between FL-blocks and S-boxes. Moreover, 3/8 of the equations of the outer FL-blocks will turn out to be linear. The total number of linear equations can be found in Table 3.

Nonlinear equations. Apart from the nonlinear equations of the S-boxes, we should also include the equations of the inner FL-blocks and equations of the outer FL-blocks that remained nonlinear (about 1/4).

Key schedule

Variables. The additional variables in the key schedule are the input and output bits of the S-boxes (which include K) and the 128 bits of K'.

Linear equations. The linear equations are formed by the additions after each S-box.

Nonlinear equations. These are just the S-box equations.

In the same manner, we can derive a system for Kasumi. The final results are listed in Table 3 and here again we need two plaintext/ciphertext pairs if we want to be able to extract the 128-bit key.

[3] Fourteen more quadratic equations can be derived from the relations $x^{64} \cdot y = x^2 \cdot x^{16}$ and $x \cdot y^{64} = y^2 \cdot y^4$, but these would introduce many new terms.

3.3 Camellia-128

The 128-bit block cipher Camellia-128 [2] was designed by K. Aoki *et al.* It is based on an 18-round Feistel-network with two layers of two 64-bit FL-blocks after the 6th and the 12th round. The 64-bit nonlinear function used in the Feistel iteration is composed of a key addition, a nonlinear layer of eight 8-bit S-boxes and a linear transform. An additional 128-bit key K_A is derived from the original key K_L by applying four Feistel iterations during the key scheduling.

Camellia-128 uses four different S-boxes which are all equivalent to an inversion over $GF(2^8)$ up to an affine transform. As pointed out in [5], the input and output bits of such S-boxes are related by 39 quadratic polynomials in 136 terms. By taking the subset of equations that does not contain any product of two input or two output bits, one can build a system of 23 quadratic equations in 80 terms (excluding the term '1'), which is probably the optimal system.

Constructing the system

Cipher

Variables. The variables of the system will be the inputs and outputs of all S-boxes and FL-blocks.

Linear equations. All linear layers between different S-boxes and between FL-blocks and S-boxes will insert linear equations in the system. Their exact number is given in Table 3.

Nonlinear equations. The only nonlinear equations are the equations of the S-box and the FL-block.

Key schedule

Variables. The key schedule adds variables for the inputs and outputs of its 32 S-boxes and for K_L and K_A.

Linear equations. These are all linear layers between different S-boxes or between an S-box and the bits of K_L and K_A.

Nonlinear equations. Again, the only nonlinear equations included in the system are the S-box equations.

3.4 Rijndael-128

Rijndael-128 [6] is a 128-bit block cipher designed by J. Daemen and V. Rijmen and has been adopted as the Advanced Encryption Standard (AES). It is a 10-round SP-network containing a nonlinear layer based on a 8-bit S-box. The linear diffusion layer is composed of a byte permutation followed by 4 parallel 32-bit linear transforms. The 128-bit key is expanded recursively by a key scheduling algorithm which contains 40 S-boxes.

The 8-bit S-box is an affine transformation of the inversion in $GF(2^8)$, and as mentioned previously, it is completely defined by a system of 23 quadratic equations in 80 terms.

Constructing the system

Cipher

Variables. As variables we choose the input and output bits of of each of the 160 S-boxes.

Linear equations. Each of the 11 linear layers (including the initial and the final key addition) correspond to a system of 128 linear equations.

Nonlinear equations. These are just the S-box equations.

Key schedule

Variables. The variables in the key schedule are the inputs and outputs of the S-boxes (which include the bits of W_3) and the bits of W_0, W_1 and W_2.

Linear equations. The linear equations are all linear relations that relate the input of the S-boxes to the output of another S-boxes or to W_0, W_1 and W_2.

Nonlinear equations. The S-box equations.

3.5 Serpent-128

Serpent-128 [1] is a 128-bit block cipher designed by R. Anderson, E. Biham and L. Knudsen. It is a 32-round SP-network with nonlinear layers consisting of 32 parallel 4-bit S-boxes and a linear diffusion layer composed of 32-bit rotations and XORs. In the key schedule, the key is first linearly expanded to 33 128-bit words and then each of them is pushed through the same nonlinear layers as the ones used in the SP-network.

Serpent-128 contains 8 different S-boxes and for each of them we can generate 21 linearly independent quadratic equations in 36 terms. The best subsets of equations we found allow us to build systems of 4 equations in 13 terms for S_0, S_1, S_2 and S_6. S_4 and S_5 can be described by a system of 5 equations in 15 terms, and S_3 and S_7 by a system of 5 equations in 16 terms. As an example, we included the systems for S_0 and S_1 in Appendix A.

Constructing the system

Cipher

Variables and equations The system for the SP-network of Serpent-128 is derived in exactly the same way as for Rijndael-128. The results are shown in Table 3.

Key schedule

Variables. The variables of the system are the inputs and outputs of the S-boxes, i.e. the bits of the linearly expanded key and the bits of the final round keys.

Linear equations. These are the linear equations relating the bits of linearly expanded key.

Nonlinear equations. These are the S-box equations.

4 Equations in $GF(2^8)$ for Rijndael and Camellia

In [7], S. Murphy and M. Robshaw point out that the special algebraic structure of Rijndael allows the cipher to be described by a very sparse quadratic system over $GF(2^8)$. The main idea is to embed the original cipher within an extended cipher, called BES (Big Encryption System), by replacing each byte a by its 8-byte *vector conjugate* in $GF(2^8)^8$:

$$\left(a^{2^0}, a^{2^1}, a^{2^2}, a^{2^3}, a^{2^4}, a^{2^5}, a^{2^6}, a^{2^7}\right) \qquad (7)$$

The advantage of this extension is that it reduces all transformations in Rijndael to very simple $GF(2^8)$ operations 8-byte vectors, regardless of whether these transformations were originally described in $GF(2^8)$ or in $GF(2)^8$. The parameters of the resulting system in $GF(2^8)$ are listed in Table 4. Note that the results are in exact agreement with those given in [7], though the equations and terms are counted in a slightly different way.

The fact that Camellia uses the same S-box as Rijndael (up to a linear equivalence) suggests that a similar system can be derived for this cipher as well. The only complication is the construction of a system in $GF(2^8)$ for the FL-blocks. This can be solved as follows: first we multiply each 8-byte vector at the input of the FL-blocks by an 8×8-byte matrix $[b_{i,j}]^{-1}$ where $b_{i,j} = x^{i \cdot 2^j}$ are elements of $GF(2^8)$ written as polynomials. The effect of this multiplication is that vectors constructed according to (7) are mapped to an 8-byte vector consisting only of 0's and 1's, corresponding to the binary representation of the original byte a. This implies that we can reuse the quadratic equations given in Section 2.2, but this time interpreted as equations in $GF(2^8)$. Finally, we return to vector conjugates by multiplying the results by the matrix $[b_{i,j}]$. Note that these additional linear transformations before and after the FL-blocks do not introduce extra terms or equations, as they can be combined with the existing linear layers of Camellia.

Table 4 compares the resulting system with the one obtained for Rijndael and it appears that both systems have very similar sizes. We should note, however, that certain special properties of BES, for example the preservation of algebraic curves, do not hold for the extended version of Camellia, because of the FL-blocks.

5 Interpretation of the Results, Conclusions

In this section we analyze the results in Tables 3 and 4, which compare the systems of equations generated by the different ciphers. Each table is divided into three parts: the first describes the structure of the cipher itself, the second describes the structure of the key-schedule and the third part provides the total count. Each part shows the number of variables, the number of equations (we provide separate counts for non-linear and linear equations) and the number of terms (with separate counts for quadratic and linear terms). The bottom line

Table 3. A comparison of the complexities of the systems of equations in $GF(2)$

	Khazad	Misty1	Kasumi	Camellia-128	Rijndael-128	Serpent-128
Cipher	(×2)	(×2)	(×2)			
Variables	**2048**	**1664**	**2004**	**2816**	**2560**	**8192**
Linear eqs.	576	904	1068	1536	1408	4224
Nonlinear eqs.	1920	824	1000	3568	3680	4608
Equations	**2496**	**1728**	**2068**	**5104**	**5088**	**8832**
Linear terms	2048	1664	2004	2816	2560	8192
Quadratic terms	3456	2960	3976	9472	10240	6400
Terms	**5504**	**4624**	**5980**	**12288**	**12800**	**14592**
Key schedule						
Variables	**2368**	**528**	**256**	**768**	**736**	**8448**
Linear eqs.	512	200	128	384	288	4096
Nonlinear eqs.	2160	200	0	736	920	4752
Equations	**2672**	**400**	**128**	**1120**	**1208**	**8848**
Linear terms	2368	528	256	768	736	8448
Quadratic terms	3888	912	0	2048	2560	6600
Terms	**6256**	**1440**	**256**	**2816**	**3296**	**15048**
Total						
Variables	**6464**	**3856**	**4264**	**3584**	**3296**	**16640**
Linear eqs.	1664	2008	2264	1920	1696	8320
Nonlinear eqs.	6000	1848	2000	4304	4600	9360
Equations	**7664**	**3856**	**4264**	**6224**	**6296**	**17680**
Linear terms	6464	3856	4264	3584	3296	16640
Quadratic terms	10800	6832	7952	11520	12800	13000
Terms	**17264**	**10688**	**12216**	**15104**	**16096**	**29640**
Free terms	**9600**	**6832**	**7952**	**8880**	**9800**	**11960**

provides the number of free terms, which is the difference between the total number of terms and the total number of equations.

Note that for 64-bit block ciphers we have to take two plaintext-ciphertext queries in order to be able to find a solution for the 128-bit key. For such ciphers the numbers provided in the "Cipher" part of the table have to be doubled, in order to get the complete number of equations, terms and variables. Note however, that one does not have to double the number of equations for the key-schedule, which are the same for each encrypted block. The total for 64-bit block ciphers shows the proper total count: twice the numbers of the cipher part plus once the numbers from the key-schedule part.

After comparing systems of equations for different ciphers we notice that the three ciphers in the 128-bit category (Camellia-128, Rijndael-128, Serpent-128) don't differ drastically in the number of free terms. Camellia-128 and Rijndael-128 in particular have very similar counts of variables, equations and terms, both in $GF(2)$ and $GF(2^8)$ which is in part explained by the fact that they use the same S-box and have approximately equiv-

Table 4. A comparison of the complexities of the systems of equations in $GF(2^8)$.

	Camellia-128	Rijndael-128
Cipher	**2816**	**2560**
Linear eqs.	1536	1408
Nonlinear eqs.	1408	1280
Equations	**2944**	**2688**
Linear terms	2816	2560
Quadratic terms	1408	1280
Terms	**4224**	**3840**
Key schedule		
Variables	**768**	**736**
Linear eqs.	384	288
Nonlinear eqs.	256	320
Equations	**640**	**608**
Linear terms	768	736
Quadratic terms	256	320
Terms	**1024**	**1056**
Total		
Variables	**3584**	**3296**
Linear eqs.	1920	1696
Nonlinear eqs.	1664	1600
Equations	**3584**	**3296**
Linear terms	3584	3296
Quadratic terms	1664	1600
Terms	**5248**	**4896**
Free terms	**1664**	**1600**

alent number of rounds (Rijndael-128 has 10 SPN rounds and Camellia-128 has 18 Feistel rounds which is "equivalent" to 9 SPN rounds). On the other hand Serpent-128 has 3–4 times more SPN rounds than Camellia-128 and Rijndael-128, and this explains why its system of equations has approximately 3–4 times more variables (at least in the cipher part). The number of quadratic terms in Serpent-128 is much smaller due to the 4-bit S-boxes, and this keeps the number of free terms within the range of the other two ciphers.

When one switches to $GF(2^8)$ for Camellia-128 and Rijndael-128, the equations for the S-boxes are much sparser and the number of free terms is reduced. It is not clear however how to make a fair comparison between systems in $GF(2)$ and $GF(2^8)$. The number of free terms in Table 4 is several times smaller, but then again, working in a larger field might increase the complexity of the solving algorithm.

Comparing Khazad and Misty1, both of which consist of 8 rounds, one may notice that Khazad has approximately twice more variables and equations (due to the fact that we write equations for the intermediate layers of the S-box[4]). On the other hand the number of quadratic terms per nonlinear equation is considerably higher for MISTY1 due to its larger S-boxes.

Finally note, that the results presented in this paper are very sensitive to the criteria used for the choice of the equations. Our main criterion was to minimize the number of free terms, for example we did not aim to find the most over-defined systems of equations. At the moment of this writing one can only speculate what would be the criteria for a possible algebraic attack (if such attack will be found to exist at all).

Acknowledgements. We wish to thank the anonymous referees, whose comments helped to improve this paper. We would also like to thank Jin Hong for pointing out an error in a previous version of this paper.

References

[1] R. Anderson, E. Biham, and L. Knudsen, "Serpent: A proposal for the advanced encryption standard." Available from `http://www.cl.cam.ac.uk/~rja14/serpent.html`.

[2] K. Aoki, T. Ichikawa, M. Kanda, M. Matsui, S. Moriai, J. Nakajima, and T. Tokita, "Camellia: A 128-bit block cipher suitable for multiple platforms." Submission to NESSIE, Sept. 2000. Available from `http://www.cryptonessie.org/workshop/submissions.html`.

[3] P. Barreto and V. Rijmen, "The KHAZAD legacy-level block cipher." Submission to NESSIE, Sept. 2000. Available from `http://www.cryptonessie.org/workshop/submissions.html`.

[4] N. Courtois, A. Klimov, J. Patarin, and A. Shamir, "Efficient algorithms for solving overdefined systems of multivariate polynomial equations," in *Proceedings of Eurocrypt'00* (B. Preneel, ed.), no. 1807 in Lecture Notes in Computer Science, pp. 392–407, Springer-Verlag, 2000.

[5] N. Courtois and J. Pieprzyk, "Cryptanalysis of block ciphers with overdefined systems of equations," in *Proceedings of Asiacrypt'02* (Y. Zheng, ed.), no. 2501 in Lecture Notes in Computer Science, Springer-Verlag, 2002. Earlier version available from `http://www.iacr.org`.

[6] J. Daemen and V. Rijmen, "AES proposal: Rijndael." Selected as the Advanced Encryption Standard. Available from `http://www.nist.gov/aes`.

[7] S. Murphy and M. Robshaw, "Essential algebraic structure within the AES," in *Proceedings of Crypto'02* (M. Yung, ed.), no. 2442 in Lecture Notes in Computer Science, pp. 17–38, Springer-Verlag, 2002. `NES/DOC/RHU/WP5/022/1`.

[8] A. Shamir and A. Kipnis, "Cryptanalysis of the HFE public key cryptosystem," in *Proceedings of Crypto'99* (M. Wiener, ed.), no. 1666 in Lecture Notes in Computer Science, pp. 19–30, Springer-Verlag, 1999.

[9] E. Takeda, "Misty1." Submission to NESSIE, Sept. 2000. Available from `http://www.cryptonessie.org/workshop/submissions.html`.

[4] There are no quadratic equations for the full 8-bit S-box of KHAZAD and for the original S-box (before the tweak). There are many cubic equations, which is true for any 8-bit S-box.

[10] Third Generation Partnership Project, "3GPP KASUMI evaluation report," tech. rep., Security Algorithms Group of Experts (SAGE), 2001. Available from `http://www.3gpp.org/TB/other/algorithms/KASUMI_Eval_rep_v20.pdf`.

A Equations

A.1 Constructing a System – Example

This appendix illustrates how a set of linearly independent S-box equations can be derived for a small example.

The n-bit S-box considered here is a 3-bit substitution defined by the following lookup table: $[7, 6, 0, 4, 2, 5, 1, 3]$. In order to find all linearly independent equations involving a particular set of terms (in this example all input and output bits x_i and y_j together with their products x_iy_j), we first construct a matrix containing a separate row for each term. Each row consists of 2^n entries, corresponding to the different values of the particular term for all possible input values (in this case from 0 to 7). The next step is to perform a Gaussian elimination on the rows of the matrix and all row operations required by this elimination are applied to the corresponding terms as well (see below). This way, a number of zero rows might appear and it is easy to see that the expressions corresponding to these rows are exactly the equations we are looking for. Note also that this set of equations forms a linearly independent *basis* and that any linear combination is also a valid equation.

$$\left[\begin{array}{l|l}
1 & 1\,1\,1\,1\,1\,1\,1\,1 \\
x_0 & 0\,1\,0\,1\,0\,1\,0\,1 \\
x_1 & 0\,0\,1\,1\,0\,0\,1\,1 \\
x_2 & 0\,0\,0\,0\,1\,1\,1\,1 \\
y_0 & 1\,0\,0\,0\,0\,1\,1\,1 \\
y_1 & 1\,1\,0\,0\,1\,0\,0\,1 \\
y_2 & 1\,1\,0\,1\,0\,1\,0\,0 \\
x_0y_0 & 0\,0\,0\,0\,0\,1\,0\,1 \\
x_0y_1 & 0\,1\,0\,0\,0\,0\,0\,1 \\
x_0y_2 & 0\,1\,0\,1\,0\,1\,0\,0 \\
x_1y_0 & 0\,0\,0\,0\,0\,0\,1\,1 \\
x_1y_1 & 0\,0\,0\,0\,0\,0\,0\,1 \\
x_1y_2 & 0\,0\,0\,1\,0\,0\,0\,0 \\
x_2y_0 & 0\,0\,0\,0\,0\,1\,1\,1 \\
x_2y_1 & 0\,0\,0\,0\,1\,0\,0\,1 \\
x_2y_2 & 0\,0\,0\,0\,0\,1\,0\,0
\end{array}\right]
\rightarrow
\left[\begin{array}{l|l}
1 & 1\,1\,1\,1\,1\,1\,1\,1 \\
x_0 & 0\,1\,0\,1\,0\,1\,0\,1 \\
x_1 & 0\,0\,1\,1\,0\,0\,1\,1 \\
1+x_0+x_1+y_0 & 0\,0\,0\,1\,1\,1\,1\,0 \\
x_2 & 0\,0\,0\,0\,1\,1\,1\,1 \\
1+x_1+y_1 & 0\,0\,0\,0\,0\,1\,0\,1 \\
1+x_0+x_1+y_0+y_1+y_2 & 0\,0\,0\,0\,0\,0\,1\,1 \\
x_0+x_0y_2 & 0\,0\,0\,0\,0\,0\,0\,1 \\
\hline
x_2+y_0+y_1+x_0y_1 & 0\,0\,0\,0\,0\,0\,0\,0 \\
1+x_1+y_1+x_0y_0 & 0\,0\,0\,0\,0\,0\,0\,0 \\
1+x_0+x_1+y_0+y_1+y_2+x_1y_0 & 0\,0\,0\,0\,0\,0\,0\,0 \\
x_0+x_0y_2+x_1y_1 & 0\,0\,0\,0\,0\,0\,0\,0 \\
1+x_1+x_2+y_0+x_0y_2+x_1y_2 & 0\,0\,0\,0\,0\,0\,0\,0 \\
y_0+y_2+x_0y_2+x_2y_0 & 0\,0\,0\,0\,0\,0\,0\,0 \\
x_0+x_2+y_0+y_2+x_2y_1 & 0\,0\,0\,0\,0\,0\,0\,0 \\
1+x_0+x_1+y_1+x_0y_2+x_2y_2 & 0\,0\,0\,0\,0\,0\,0\,0
\end{array}\right]$$

A.2 Khazad

P: 4 quadratic equations in 16 terms:

$$\begin{aligned}
x_0y_1 + y_1 + x_0y_2 + x_0y_3 + x_2y_3 &= x_0 + x_2 + x_3 + 1\\
x_3y_1 + x_0y_2 + y_2 + x_0y_3 &= x_0 + x_1 + x_2 + x_3\\
x_0y_1 + x_3y_2 + y_2 + x_0y_3 + y_3 &= x_2 + x_3\\
y_0 + x_0y_2 + x_0y_3 + y_2y_3 + y_3 &= x_0x_3 + x_1 + 1
\end{aligned}$$

Q: 6 quadratic equations in 18 terms:

$$\begin{aligned}
y_0 + x_0y_1 + y_1 + x_0y_2 &= x_0x_2 + x_1x_2 + x_3 + 1\\
x_1y_0 + y_1 + x_0y_2 + y_3 &= x_0x_1 + x_0x_2 + x_2 + x_3 + 1\\
x_1y_0 + y_0 + x_0y_1 + y_2 + x_3y_3 &= x_0x_1 + x_0 + x_2\\
y_0y_1 + y_0 + x_0y_1 + x_0y_2 + y_3 &= x_0x_2 + x_0 + x_1 + x_2 + x_3\\
y_0y_2 + y_1 + y_2 + y_3 &= x_0x_2 + x_1 + x_2 + x_3 + 1\\
y_0 + y_1y_2 + y_1 + y_2 &= x_0x_1 + x_1 + x_3 + 1
\end{aligned}$$

A.3 Misty1

Table 5 shows the original systems of equations which completely define the S-boxes S_7 and S_9 of Misty1. We omit the quadratic system of 7 equations in 56 terms used for S_7 in Section 3.2 because of its complexity.

A.4 Serpent-128

S_0: 4 quadratic equations in 13 terms:

$$\begin{aligned}
y_3 &= x_0x_3 + x_0 + x_1 + x_2 + x_3\\
y_0 + y_1 &= x_0x_1 + x_1x_3 + x_2 + x_3\\
y_0y_3 + y_1y_3 + y_1 &= x_0 + x_2 + x_3 + 1\\
y_0y_3 + y_0 + y_2 + y_3 &= x_0x_1 + x_1 + x_3 + 1
\end{aligned}$$

S_1: 4 quadratic equations in 13 terms:

$$\begin{aligned}
x_3 &= y_0 + y_1y_3 + y_2 + y_3\\
x_0 + x_1 &= y_0y_1 + y_0y_3 + y_0 + y_2 + y_3 + 1\\
x_0x_3 + x_1x_3 + x_0 + x_3 &= y_1 + y_2 + y_3 + 1\\
x_0x_3 + x_2 + x_3 &= y_0y_3 + y_1 + y_3 + 1
\end{aligned}$$

Table 5. MISTY1: S_7 and S_9.

S_7: 7 cubic equations in 65 terms:

$$
\begin{aligned}
y_0 &= x_0 + x_1x_3 + x_0x_3x_4 + x_1x_5 + x_0x_2x_5 + x_4x_5 + x_0x_1x_6 + x_2x_6 + x_0x_5x_6 + x_3x_5x_6 + 1\\
y_1 &= x_0x_2 + x_0x_4 + x_3x_4 + x_1x_5 + x_2x_4x_5 + x_6 + x_0x_6 + x_3x_6 + x_2x_3x_6 + x_1x_4x_6 + x_0x_5x_6 + 1\\
y_2 &= x_1x_2 + x_0x_2x_3 + x_4 + x_1x_4 + x_0x_1x_4 + x_0x_5 + x_0x_4x_5 + x_3x_4x_5 + x_1x_6 + x_3x_6 + x_0x_3x_6 + x_4x_6 + x_2x_4x_6\\
y_3 &= x_0 + x_1 + x_0x_1x_2 + x_0x_3 + x_2x_4 + x_1x_4x_5 + x_2x_6 + x_1x_3x_6 + x_0x_4x_6 + x_5x_6 + 1\\
y_4 &= x_2x_3 + x_0x_4 + x_1x_3x_4 + x_5 + x_2x_5 + x_1x_2x_5 + x_0x_3x_5 + x_1x_6 + x_1x_5x_6 + x_4x_5x_6 + 1\\
y_5 &= x_0 + x_1 + x_2 + x_0x_1x_2 + x_0x_3 + x_1x_2x_3 + x_1x_4 + x_0x_2x_4 + x_0x_5 + x_0x_1x_5 + x_3x_5 + x_0x_6 + x_2x_5x_6\\
y_6 &= x_0x_1 + x_3 + x_0x_3 + x_2x_3x_4 + x_0x_5 + x_2x_5 + x_3x_5 + x_1x_3x_5 + x_1x_6 + x_1x_2x_6 + x_0x_3x_6 + x_4x_6 + x_2x_5x_6
\end{aligned}
$$

S_9: 9 quadratic equations in 54 terms:

$$
\begin{aligned}
y_0 &= x_0x_4 + x_0x_5 + x_1x_5 + x_1x_6 + x_2x_6 + x_2x_7 + x_3x_7 + x_3x_8 + x_4x_8 + 1\\
y_1 &= x_0x_2 + x_3 + x_1x_3 + x_2x_3 + x_3x_4 + x_4x_5 + x_0x_6 + x_2x_6 + x_7 + x_0x_8 + x_3x_8 + x_5x_8 + 1\\
y_2 &= x_0x_1 + x_1x_3 + x_4 + x_0x_4 + x_2x_4 + x_3x_4 + x_4x_5 + x_0x_6 + x_5x_6 + x_1x_7 + x_3x_7 + x_8\\
y_3 &= x_0 + x_1x_2 + x_2x_4 + x_5 + x_1x_5 + x_3x_5 + x_4x_5 + x_5x_6 + x_1x_7 + x_6x_7 + x_2x_8 + x_4x_8\\
y_4 &= x_1 + x_0x_3 + x_2x_3 + x_0x_5 + x_3x_5 + x_6 + x_2x_6 + x_4x_6 + x_5x_6 + x_6x_7 + x_2x_8 + x_7x_8\\
y_5 &= x_2 + x_0x_3 + x_1x_4 + x_3x_4 + x_1x_6 + x_4x_6 + x_7 + x_3x_7 + x_5x_7 + x_6x_7 + x_0x_8 + x_7x_8\\
y_6 &= x_0x_1 + x_3 + x_1x_4 + x_2x_5 + x_4x_5 + x_2x_7 + x_5x_7 + x_8 + x_0x_8 + x_4x_8 + x_6x_8 + x_7x_8 + 1\\
y_7 &= x_1 + x_0x_1 + x_1x_2 + x_2x_3 + x_0x_4 + x_5 + x_1x_6 + x_3x_6 + x_0x_7 + x_4x_7 + x_6x_7 + x_1x_8 + 1\\
y_8 &= x_0 + x_0x_1 + x_1x_2 + x_4 + x_0x_5 + x_2x_5 + x_3x_6 + x_5x_6 + x_0x_7 + x_0x_8 + x_3x_8 + x_6x_8 + 1
\end{aligned}
$$