A New Attack Against Khazad

Frédéric Muller

DCSSI Crypto Lab, 18 rue du Docteur Zamenhof
F-92131 Issy-les-Moulineaux Cedex, France
Frederic.Muller@m4x.org

Abstract. Khazad is a new block cipher initially proposed as a candidate to the NESSIE project. Its design is very similar to Rijndael, although it is a 64-bit block cipher. In this paper, we propose a new attack that can be seen as an extension of the Square attack. It takes advantage of redundancies between the round key derivation and the round function, and also exploits some algebraic observations over a few rounds. As a result, we can break 5 rounds of Khazad faster than exhaustive key search. This is the best known cryptanalytic result against Khazad.

1 Introduction

Many recent block ciphers are built using an iterative Substitution Permutation Network (SPN). This includes in particular Shark [14], Square [5], Rijndael [6], Anubis [1] or Khazad [2]. These ciphers are generally designed to be immune against differential and linear cryptanalysis. However, a new powerful class of attack has emerged recently, the "Square" attack which was initially a dedicated attack [5] against the Square block cipher. It takes advantage of the bijectivity of most components of these ciphers (S-box, round key addition, ...), without analyzing their precise behavior. More generally, this class of high-level attacks can be seen as a dual technique to differential and linear cryptanalysis since it is based on the propagation of distributions along the cipher for a large set of plaintexts, rather than on statistical properties for a single plaintext (or a pair of plaintexts).

Since then, this technique has been successfully applied to many other block ciphers (see [3] and [8]). Currently, one of the best known attacks against Rijndael is Gilbert-Minier's collision attack on 7-rounds [9] which can also be seen as an extension of the "Square" attack. Besides, a more generic name for this technique, namely the "integral" attack has been recently proposed [10]. We use this terminology in the present paper.

Khazad is a 64-bit SPN block cipher with 8 rounds. It offers several interesting features. First, it achieves full diffusion over one round using an MDS matrix layer. Furthermore, all components are involution, so the only difference between encryption and decryption lies in the key scheduling. Thus the same security is expected in both directions.

Khazad was initially proposed as a NESSIE [11] candidate for 64 bits block cipher. However, it was not selected due to his low security margin [12]. In the Section 2, we provide some background about Khazad. Then, in Section 3 we present new observations about this cipher that we later exploit to mount a 5-round attack.

2 Some background about Khazad

Khazad is a byte-oriented cipher. Indeed, all operations handle bytes of data: S-box, linear application over GF(2^8), ... Like most word-oriented ciphers, Khazad may thus be subject to integral attacks. First, we describe quickly the main components of Khazad, then we present previously known cryptanalytic results against this cipher.

2.1 Description of the cipher

We only give a short overview of Khazad. More details can be found in [2]. During encryption, it iterates 8 times a SP round function. Throughout this paper, we denote by P its linear layer and S its S-box layer. Thus, each round consists, in this particular order, of

- a S layer where a fixed S-box is applied to each byte of the current state.
- a P layer which consists of a square matrix in $GF(2^8)$ of size 8
- a XOR layer, using the corresponding round subkey. This layer is called X_i at round number i.

A first XOR layer is applied prior to the first round. Besides, the last round does not include a matrix layer. Thus the full encryption function can be written as

$$(X_8 \circ S) \circ (X_7 \circ P \circ S) \circ \cdots \circ (X_1 \circ P \circ S) \circ X_0$$

By convention, the notation "0.5 round" denotes either the first two layers of the round (the S and P layers together), or the XOR layer alone.

2.2 Previous Results about Khazad

All cryptographic results concerning Khazad are summarized in Table 1. The best known attack so far was the straightforward application of the integral attack, originally proposed by the designers of the cipher [2]. Indeed, if a set of 256 plaintexts is introduced, such that the first byte takes all 256 possible values while other bytes have constant values, distributions for each byte can be easily described over 2 rounds (see Figure 1). For each byte, the following notations are used to represent these distributions over this set of 256 plaintexts

- A represents bytes where "All" possible values are represented exactly once.
- C represents bytes which have a "Constant" value
- S represents bytes where the "Sum" of all values is 0.

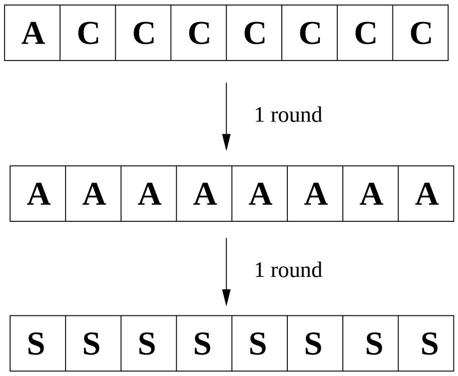

Fig. 1. The straightforward integral attack

From Figure 1, it appears that all bytes have a S distribution after 2 rounds. Such balanced distributions provide a 2-round distinguisher. Since there is no matrix layer in the last round, the corresponding subkey bytes can be guessed separately to mount a 3-rounds attack against Khazad. The resulting complexity is roughly 2^{16} S-box lookups and 2^{9} chosen plaintexts. Besides, this attack can be directly extended to 4 rounds by guessing one additional subkey. This increase the time complexity by a factor 2^{64}.

Other attacks have been examined throughout the NESSIE evaluation process. An impossible differential attack exists on 3 rounds of Khazad but its complexity is larger than the integral attack [12]. The design rationale apparently prevents differential and linear attacks since a large number of S-boxes is activated at each round. Besides, Gilbert-Minier's attack on Rijndael does not apply very well here, since it requires partial collision. New ideas to attack involutional ciphers have been recently proposed [4]. Indeed the cycle structure of 5-rounds Khazad presents some surprising properties. However these observations do not result yet on a concrete attack. Finally, the only cryptanalytic result on 5-rounds Khazad is the class of 2^{64} weak keys identified in [4] which can be broken with 2^{43} steps of analysis using 2^{38} encryption blocks.

Type of Attack	*Rounds*	*Time*	*Data*
integral attack [2]	3	2^{16}	2^{9}
impossible differential [12]	3	2^{64}	2^{13}
integral attack [2]	4	2^{80}	2^{9}
weak keys [4]	5	2^{43}	2^{38}
improved integral (this paper)	5	2^{91}	$\simeq 2^{64}$

Table 1. Summary of Known Attacks Against Khazad

3 New Observations on Khazad

In this Section, we investigate some properties of Khazad. Our first observations concern the key scheduling, which is a Feistel network based on the round function. We show some surprising weaknesses resulting from this redundancy. Then, we describe some algebraic properties of the cipher over a reduced number of rounds.

3.1 Redundancies in the Round Key Derivation

In order to speed up Khazad while keeping the same security, its designers adopted a key scheduling that inherits many properties of the round function. While this key scheduling can be viewed as a simple Feistel network, the derivation of the i-th round key K_i is basically just one round of encryption applied to K_{i-1}, with a particular XOR layer. Initially, the actual 128 bits of secret key are splitted into K_{-1} and K_{-2} which serve as initial values. Thus, we have the following relation, for $0 \leq i \leq 8$,

$$K_i = P \circ S(K_{i-1}) \oplus C_i \oplus K_{i-2} \tag{1}$$

where the C_i's are round constants. The use of the round function during key scheduling creates surprising cascade eliminations during encryption. To illustrate this, we consider encryption of the plaintext

$$Plain = K_0 \oplus S \circ P(0)$$

which depends only on the first subkey K_0. After 1 round, the internal value (denoted as Iv_1) is

$$\begin{aligned} Iv_1 &= K_1 \oplus P \circ S(Plain \oplus K_0) \\ &= K_1 \oplus P \circ S \circ S \circ P(0) \\ &= K_1 \end{aligned}$$

since P and S are involution. Then, after the second round, the internal value Iv_2 is

$$Iv_2 = K_2 \oplus P \circ S(Iv_1) = K_2 \oplus P \circ S(K_1) = K_0 \oplus C_2$$

because of relation (1). Thus, Iv_2 is basically known when K_0 is known, independently of K_1 and K_2. The following S and P layers can also be included, so we obtain an internal value depending only on K_0 after 2.5 rounds.

Many similar eliminations can be obtained with chosen plaintext or ciphertext once a round key is known (or guessed). Obviously, this observation suggests guessing K_0 to obtain a known intermediate value and then trying to extend these observations over the last rounds. We describe such an attack in Section 4.

3.2 Algebraic Properties of 2.5 rounds of Khazad

In this section, we consider the algebraic properties of Khazad. More precisely, we show that the last 2.5 rounds of Khazad can be expressed using a reduced number of algebraic relations. We also show that this system can be used to retrieve several subkey bits, once a few intermediate values and their corresponding ciphertexts are known.

As it was originally argued in [10], interpolation and algebraic attacks are good candidates to be combined with an integral attack which usually provides known intermediate values (or linear relations between these values). In the case of Khazad, we consider the following situation, encountered later in Section 4.

- We know 256 full intermediate values: $Y_1, \ldots, Y_{256}$.
- 2.5 rounds of encryption remain unknown. The 3 corresponding round keys are denoted as K, K' and K''.
- The resulting ciphertexts $Z_1, \ldots, Z_{256}$ are known. The last 2.5 rounds of encryption can be expressed as

$$Z_i = K'' \oplus S(K' \oplus P \circ S(K \oplus Y_i))$$

or, equivalently,

$$S(Z_i \oplus K'') = K' \oplus P \circ S(K \oplus Y_i)$$

for $1 \leq i \leq 256$. Then, the first byte of $S(K \oplus Y_i)$, later referred to as w_i can be obtained by just guessing the first byte of K. Let $P_1(x)$ denotes the linear function that returns the first byte of $P(x)$ for any 64 bits input x. We have the following relation

$$P_1 \circ S(Z_i \oplus K'') = P_1(K') \oplus w_i \qquad (2)$$

In addition, if we guess the byte $P_1(K')$, we obtain, for each i, a condition on K'' of the form

$$P_1 \circ S(known \oplus K'') = known \qquad (3)$$

While it is not straightforward to solve such a non linear system, we apparently obtain enough conditions to retrieve the value of K''.

Suppose we replace, in relation (3), the S-box by its exact algebraic interpolation over GF(2). From the left hand side of (3), one sees that only a reduced number of monomials in the bits of K'' appear. Indeed, S operates on the bytes of K'', thus the monomials are those involving bits of K'' that "belong" to the same byte. For instance, representing K'' as $(k_1, \ldots, k_{64})$, all monomials over $(k_1, \ldots, k_8)$ may appear while no monomial involving simultaneously k_8 and k_9 can appear.

Thus, (3) can be seen as a system of 8 relations over GF(2) involving $8 \times 2^8 = 2^{11}$ monomials in the bits of K''. Since 256 such relations are known (one for each $i = 1, \ldots, 256$), we obtain a system with 2^{11} unknown monomials and 2^{11} relations. Two bytes of round keys have been guessed to build this system.

3.3 Properties of the Linear System

In the previous Section, we built a linear system over GF(2) of 2^{11} relations involving 2^{11} unknowns, which are monomials over the 64 bits of the last round subkey. This can be summarized as

$$b = M\ x$$

where x and b are vectors of 2^{11} bits and M is a square matrix obtained after replacing the S-box by its algebraic expression over GF(2). x contains the unknown monomials and, for $i = 0, \dots, 256$, each relation (3) is turned into eight conditions on the bits of x, that correspond to bits $b_{8i}, \dots, b_{8i+7}$ of b. More precisely, from relation (2), we see that, for all i,

$$b_i = c_i \oplus \{P_1(K')\}_i$$

where $\{P_1(K')\}_i$ denotes the bit number (i mod 8) of $P_1(K')$, and c_i depends only on the intermediate values and the first byte of K.

In the general case, one could expect to solve this system by inverting the matrix M. However, M is built from an S-box interpolation, thus it is not a random matrix. It turns out that rows of M cannot have full rank for two reasons:

- The algebraic degree of the Khazad S-box is 7, therefore the 8 columns of M corresponding to monomials of degree 8 necessarily contain only zeroes.
- The coefficients of M corresponding to degree 7 monomials are independent of the plaintext.

Indeed, every output bit s_j of the S-box can be represented by a relation of the form

$$s_j = \sum_{\alpha_1 + \dots + \alpha_8 \leq 7} \beta_j^{(\alpha_1, \dots, \alpha_8)} i_1^{\alpha_1} \cdots i_8^{\alpha_8} \tag{4}$$

for some coefficients $\beta_j^{(\alpha_1, \dots, \alpha_8)}$, with $(i_1, \dots, i_8)$ denoting the input bits.

However, M is obtained by applying several times the S-box to inputs of the form

$$(i_1, \dots, i_8) = (c_1 \oplus k_1, \dots, c_8 \oplus k_8)$$

where the c_i's are ciphertext bits and the k_i's are subkey bits. Substituting these values in (4), it is clear that terms of degree 7 in the subkey bits are independent of the c_i's. Therefore, for all $i = 0, \dots, 256$, relation (3) always provides the same coefficients for degree 7 monomials. The 64 corresponding columns of M are not free (and in fact have rank 8).

Moreover, when computing $b_t \oplus b_{8i+t}$ for $i = 1, \dots, 255$ and $t = 0, \dots, 7$, monomials of degree 7 are eliminated. Hence, we can obtain $8 \times 255 = 2040$ new relations of degree 6, thus involving only $2^{11} - 8 - 64 = 1976$ monomials. This result on a new matrix M' having 2040 lines and 1976 columns. The initial system

$$b = M\ x$$

can be rewritten as

$$b' = M'\ x'$$

where the vector b' contains 2040 bits of the form $b_t \oplus b_{8i+t}$ and x' contains the 1976 monomials of degree 6 or less. Besides, b' does not depends on K', whose bits get eliminated when computing $b_t \oplus b_{8i+t}$. A direct application of the gauss algorithm on M' provides at least 64 conditions on its rows, thus conditions on the bits of b'. These conditions must be satisfied when the correct byte of K has been guessed.

Therefore, we do not have to solve the initial system. From the interpolation matrix M, we can build $2040 - 1976 = 64$ linear conditions and thus detect the correct guess for the corresponding 8 bits of K. We programmed this algebraic step using the NTL library [13]. It turns out from our experiment that the kernel of M' has always rank 64 (although it would be no problem if its dimension was larger). Thus we obtain easily enough linear conditions to verify the correct guess.

To summarize, we have shown that the last 2.5 rounds of Khazad can be expressed with a low degree algebraic system, after guessing a reduced number of bits. 64 linear conditions can be used to discard wrong guesses without actually solving this system.

4 An attack against 5 rounds of Khazad

In this Section, we develop the previous observations on Khazad to mount a new attack against 5 rounds of this cipher. The sketch of this attack works as follows

4.1 Sketch of the Attack

- Guess all 64 bits of K_0
- Guess 8 bits of K_1
- Introduce 256 chosen plaintexts in order to
  - apply the integral attack, starting from the end of the X_0 layer
  - obtain known intermediate values after 2.5 rounds as in Section 3.1
- Build the interpolation matrix as described in Section 3.2.
- Build 64 linear conditions from the matrix.
- Guess the first byte of K_3
  - Verify the linear conditions.
  - Discard wrong guesses.
- A large portion of guess of K_0 and K_1 are also discarded through the absence of a matching K_3
- Recover the whole secret key.

Most elements in this attack have been developed previously. Additional elements needed to connect all together are described in the following section.

4.2 Strengthening the integral attack

Once K_0 has been guessed, we can choose the plaintext $Plain$ to obtain any intermediate value after 0.5 round, since this value is equal to $P \circ S(Plain \oplus K_0)$ (and also to $K_1 \oplus Iv_1$). We consider an integral attack starting from there. Let us consider the set of 256 plaintexts such that $Iv_1 \oplus K_1$ takes all values on its first byte and has constant value equal to 0 on its other bytes. This can be represented as

$$Iv_1 = K_1 \oplus (i, 0, 0, 0, 0, 0, 0, 0)$$

for $0 \leq i \leq 255$. As in the classical integral attack, we obtain A distributions after 1.5 round and S distributions after 2.5 rounds. Besides, since the first byte of K_1 is guessed, we have, for all i

$$Iv_2 = K_2 \oplus P \circ S(K_1 \oplus (i, 0, \ldots, 0)) = K_0 \oplus C_2 \oplus P(\Delta_i)$$

where Δ_i is known. Hence, we obtain 256 known intermediate values after 2.5 rounds. Moreover these values are balanced as in the integral attack of [2] though we do not specifically use this property.

Then, we are exactly in the situation described in Section 3.2 with 256 known intermediate values and the corresponding ciphertexts, with 2.5 rounds inbetween. We have seen that a matrix can be built and 64 linear conditions derived from this matrix. Using them, we can guess then verify the value of the first byte of K_3. Since there are 64 conditions, many wrong guess on K_0 and K_1 can even be filtered out by the absence of a matching value for K_3. The number of remaining guesses afterwards is only

$$2^{64} \times 2^8 \times 2^8 \times 2^{-64} = 2^{16}$$

thus we can guess the 56 remaining bits of K_1 and deduce the full secret key - which is equivalent to (K_0, K_1) - for a total complexity of 2^{80} basic operations.

In fact, the linear algebra step has a larger complexity. The matrix we build is independent of the 8 guessed bits of K_1, so we need to built it 2^{64} times, and then apply the Gaussian algorithm in each case. This algorithm has complexity of $(2^{11})^3$ binary operations. Using 32 bits instructions, it can be fasten up to obtain a complexity equivalent to 2^{28} S-box lookups. Thus, this step is roughly equivalent to $2^{64} \times 2^{28} = 2^{92}$ S-box lookups.

On the other hand, building the matrix of interpolation has a much smaller complexity since it can be largely precomputed (it is just a collection of smaller matrix blocks, each depending on 8 bits of ciphertext). Besides, the cost of verifying linear conditions corresponds in average to $2^{72} \times 2^8 \times 2 = 2^{81}$ evaluations of linear conditions on 2^{11} bits long vectors, each costing roughly 2×2^{11} bitwise operations. Using 32 bits instruction, this is roughly equivalent to

$$2^{81} \times 2^{11} \times 2 \times 2^{-5} = 2^{88}$$

S-box lookups. Therefore, the dominant cost in our attack is the linear algebra step.

To summarize, our attack against 5 rounds Khazad recovers the full 128 bits secret key with time complexity equivalent on average to 2^{91} S-box lookups and using basically the complete dictionary of 2^{64} plaintexts.

4.3 Overview of cryptanalytic results against Khazad

In Table 1, we have summarized all known cryptanalytic results against reduced-round versions of Khazad. Our improved integral attack is the best cryptanalytic result against Khazad. However, its data complexity represents in average the complete dictionary of 2^{64} plaintexts. Indeed, we need to encrypt 256 plaintexts for each guess of the first subkey. It is possible that the correct subkey is identified early, however, in average, all possible plaintexts will have been encrypted by the time we find the correct K_0. We did not manage to find a technique to guess the subkeys in a better order, or to trade data complexity for time complexity. This is a topic for further research.

In practice this huge data complexity will make the attack infeasible, although it is significantly faster than exhaustive key search. Furthermore, it is widely considered that recent block ciphers should resist key recovery attack even when the full dictionary is known. Therefore, we consider this new attack is a significant step forward in the analysis of Khazad. Whether it can be extended to 6 or more rounds remains an open question that should be further investigated.

5 Possible Extensions

The attack we have described in Section 4 does not depend in depth from the components of this cipher. Concerning the S-box, the only property we use is its algebraic degree of 7. Concerning the MDS matrix, no property is specifically used. Therefore, Khazad cannot be strengthen by changing these components, and our attack depends only on the high-level structure of the cipher.

5.1 Key Scheduling Redundancy Attacks

In the case of Khazad, we have shown that re-using the round function inside the key scheduling has surprising effects. More generally, when a block cipher E uses in its key scheduling the same basic components as in the round function, a general problem is to consider the encryption of a chosen plaintext

$$Plain = \Phi(K_0)$$

for a well chosen function Φ of the first round key K_0. More precisely, one should investigate if a cascade elimination cannot occur and yield a predictable value of $E_K(Plain)$, or even a simple function of a subkey. For instance, if

$$E_K(Plain) = \Psi(K_i)$$

for some i and some function Ψ, one may recover K_i from the guess of K_0 with time and data complexity roughly equivalent to the size of the subkeys. If, in addition, the full secret key can be reconstructed from K_0 and K_i (which is sometimes the case for key scheduling based on Feistel networks), this can lead to an attack. This threat mostly concerns "small" block ciphers (like Khazad), where the round subkeys are smaller than the secret key.

In addition, an improvement would be to guess only a part of the first subkey, to obtain partially known intermediate values, in the case of SPN block ciphers that do not achieve full diffusion. This is the case of Anubis or Rijndael, though we did not manage to obtain any such observation against those. Furthermore, the existence of improved cascade eliminations on Khazad should also be further analyzed. More generally, using a key scheduling that is not too similar to the round function is probably a more reasonable thing.

5.2 Combining Integral and Interpolation Attacks

This idea of combining integral attacks with attacks based on the algebraic properties of a block cipher was originally introduced in [10]. However, no successful application has been reported since then. Our improved integral attack against Khazad is apparently the first successful combination of these two cryptanalytic techniques, although we also use additional properties of Khazad here.

The problem is that integral attacks generally end up providing some information concerning a balanced set of intermediate values. This type of property does not pass well across S-box layers, while the diffusion layers very quickly increase the number of monomials. Thus it is generally difficult to write simple algebraic relations, even after guessing some subkey bits as we did for Khazad. An other specific problem is that only algebraic relations where intermediate values are expressed as a function of subkey and ciphertext bits are generally useful for interpolation attacks. For instance, low degree algebraic relations from the inversion in $GF(2^8)$ cannot be used, at least in a straightforward manner. In spite of these problems, we believe such attacks combining different cryptanalytic techniques may be of interest in the future.

5.3 Other algebraic approaches

In Section 3.3, we obtained a large multivariate, non linear system over GF(2). We used the relinearization technique [15], that means replacing all monomials (which happen to be present in reduced number here) by new unknowns and apply usual linear algebra techniques. This technique is not the best method known to solve nonlinear multivariate systems. However, it turns out to be sufficient and quite successful since we can obtain simple conditions on subkey bits by reducing the underlying matrix. In fact, we do not even need to solve this system, to finish with.

In a very generic way, what we obtain, for each guess of K_0, is a system of low degree involving a few unknown subkey bits. We need either to solve this system, or to detect quickly if it has some solutions. Our attack uses the second strategy, and requires one application of the gauss algorithm on a 2^{11} bits square matrix. In the light of recent progress ([7], [16]), better techniques to directly solve the system could be considered. However, it seems unlikely the time complexity could be pushed below the length of the outside loop, namely 2^{64}. Thus any complexity gain would probably be limited. However, it would be interesting to find if a similar simple system over more than 2.5 rounds could be derived.

5.4 Exposure of Round Keys

It results from the previous observations that the security of 5 rounds of Khazad depends only on the secrecy of the first subkey. Indeed the complexity of our attack is quite high, especially the data complexity, however this is mostly due to the cost of guessing the first subkey K_0.

If, somehow, the first round subkey is exposed, then 5 rounds reduced Khazad becomes insecure. In this case, the complete secret key can be recovered by applying a few times the attack of Section 4, which has complexity of only 2^{28} S-box lookups and 256 chosen plaintexts when K_0 is known. We consider this property is quite undesirable. Indeed, information about the first round subkey may be obtained by other means than exhaustive search. For instance, side channel attack techniques may provide this kind of information.

6 Conclusion

We have proposed a new attack against the block cipher Khazad. This cipher is very interesting, because it constitutes a reduced and simplified version of Rijndael, so its analysis is very helpful in understanding the security of word-oriented SPN block ciphers, which are now largely used since the standardization of Rijndael as the AES. In particular, the new class of integral (aka Square) attacks which are (almost) independent of the S-boxes should be further investigated.

In this paper, we break 5 rounds of Khazad (against 8 rounds for the full cipher) faster than exhaustive search: we use about 2^{64} chosen plaintexts and 2^{91} S-box lookups in average. Although the cryptanalytic techniques we exploit are not new, we combine them in a new and unexpected way to improve on known attacks. A very surprising improvement arises from some redundancies between the key scheduling and the round function of Khazad. Whether this attack can be improved or extended to 6 rounds remains a topic for further research.

References

1. P. Barreto and V. Rijmen. The Anubis Block Cipher. In *First Open NESSIE Workshop, KU-Leuven*, 2000. Submission to NESSIE.
2. P. Barreto and V. Rijmen. The Khazad Legacy-Level Block Cipher. In *First Open NESSIE Workshop, KU-Leuven*, 2000. Submission to NESSIE.

3. P. Barreto, V. Rijmen, J. Nakahara Jr., B. Preneel, J. Vadewalle, and H. Y. Kim. Improved SQUARE Attacks against Reduced-Round HIEROCRYPT. In M. Matsui, editor, *Fast Software Encryption – 2001*, volume 2355 of *Lectures Notes in Computer Science*, pages 165–173. Springer, 2001.
4. A. Biryukov. Analysis of Involutional Ciphers: Khazad and Anubis. In T. Johansson, editor, *Fast Software Encryption – 2003*, Lectures Notes in Computer Science. Springer, 2003. To appear.
5. J. Daemen, L. Knudsen, and V. Rijmen. The Block Cipher Square. In E. Biham, editor, *Fast Software Encryption – 1997*, volume 1267 of *Lectures Notes in Computer Science*, pages 149–165. Springer, 1997.
6. J. Daemen and V. Rijmen. AES Proposal: Rijndael. In *AES Round 1 Technical Evaluation CD-1: Documentation*. NIST, 1998.
7. J.C. Faugère and A. Joux. Algebraic Cryptanalysis of Hidden Field Equation (HFE) Cryptosystems Using Gröbner Bases. In D. Boneh, editor, *Advances in Cryptology – Crypto'03*, volume 2729 of *Lectures Notes in Computer Science*, pages 44–60. Springer, 2003.
8. N. Ferguson, J. Kelsey, S. Lucks, B. Schneier, M. Stay, D. Wagner, and D. Whiting. Improved Cryptanalysis of Rijndael. In B. Schneier, editor, *Fast Software Encryption – 2000*, volume 1978 of *Lectures Notes in Computer Science*, pages 213–230. Springer, 2000.
9. H. Gilbert and M. Minier. A Collision Attack on Seven Rounds of Rijndael. In *Third AES Conference*, pages 230–241. NIST, 2000.
10. L. Knudsen and D. Wagner. Integral Cryptanalysis. In J. Daemen and V. Rijmen, editors, *Fast Software Encryption – 2002*, volume 2365 of *Lectures Notes in Computer Science*, pages 112–127. Springer, 2002. Extended Abstract.
11. NESSIE - New European Schemes for Signature, Integrity and Encryption. `http://www.cryptonessie.org`.
12. NESSIE Security Report D20, version 2-0. Available at `http://www.cryptonessie.org`.
13. NTL library. Available at `http://www.shoup.net`.
14. V. Rijmen, J. Daemen, B. Preneel, A. Bosselaers, and E. De Win. The Cipher SHARK. In D. Gollmann, editor, *Fast Software Encryption – 1996*, volume 1039 of *Lectures Notes in Computer Science*, pages 99–112. Springer, 1996.
15. A. Shamir and A. Kipnis. Cryptanalysis of the HFE Public Key Cryptosystem. In M. Wiener, editor, *Advances in Cryptology – Crypto'99*, volume 1666 of *Lectures Notes in Computer Science*, pages 19–30. Springer, 1999.
16. A. Shamir, J. Patarin, N. Courtois, and A. Klimov. Efficient Algorithms for solving Overdefined Systems of Multivariate Polynomial Equations. In B. Preneel, editor, *Advances in Cryptology – Eurocrypt'00*, volume 1807 of *Lectures Notes in Computer Science*, pages 392–407. Springer, 2000.