Joan Daemen
Vincent Rijmen (Eds.)

LNCS 2365

Fast Software Encryption

9th International Workshop, FSE 2002
Leuven, Belgium, February 2002
Revised Papers

Springer

Joan Daemen Vincent Rijmen (Eds.)

Fast Software Encryption

9th International Workshop, FSE 2002
Leuven, Belgium, February 4-6, 2002
Revised Papers

Springer

Volume Editors

Joan Daemen
Proton World
Zweefvliegtuigstraat 10
1130 Brussel, Belgium
E-mail: joan.daemen@protonworld.com

Vincent Rijmen
Cryptomathic
Lei 8A
3000 Leuven, Belgium
E-mail: vincent.rijmen@cryptomathic.com

Cataloging-in-Publication Data applied for

Die Deutsche BBibliothek - CIP-Einheitsaufnahme

Fast software encryption : 9th international workshop ; revised papers / FSE 2002, Leuven, Belgium, February 2002. Joan Daemen ; Vincent Rijmen (ed.). - Berlin ; Heidelberg ; New York ; Barcelona ; Hong Kong ; London ; Milan ; Paris ; Tokyo : Springer, 2002
(Lecture notes in computer science ; Vol. 2365)
ISBN 3-540-44009-7

CR Subject Classification (1998): E.3, F.2.1, E.4, G.4

ISSN 0302-9743
ISBN 3-540-44009-7 Springer-Verlag Berlin Heidelberg New York

Typesetting: Camera-ready by author, data conversion by PTP-Berlin, Stefan Sossna e.K.
Printed on acid-free paper SPIN: 10870318 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 2365
Edited by G. Goos, J. Hartmanis, and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Tokyo

Preface

This Fast Software Encryption workshop was the ninth in a series of workshops started in Cambridge in December 1993. The previous workshop took place in Yokohama in April 2001. It concentrated on all aspects of fast primitives for symmetric cryptography: secret key ciphers, the design and cryptanalysis of block and stream ciphers, as well as hash functions and message authentication codes (MACs).

The ninth Fast Software Encryption workshop was held in February 2002 in Leuven, Belgium and was organized by General Chair Matt Landrock (Cryptomathic Belgium), in cooperation with the research group COSIC of K.U. Leuven. This year there were 70 submissions, of which 21 were selected for presentation and publication in this volume.

We would like to thank the following people. First of all the submitting authors and the program committee for their work. Then Markku-Juhani O. Saarinen, Orr Dunkelman, Fredrik Jönsson, Helger Lipmaa, Greg Rose, Alex Biryukov, and Christophe De Canniere, who provided reviews at the request of program committee members. Bart Preneel for letting us use COSIC's Web-review software in the review process and Wim Moreau for all his support. Finally we would like to thank Krista Geens of Cryptomathic for her help in the registration and the practical organization.

May 2002 Joan Daemen and Vincent Rijmen

Fast Software Encryption 2002

February 4–6, 2002, Leuven, Belgium

Sponsored by the
International Association for Cryptologic Research

General Chair

Matt Landrock, Cryptomathic, Belgium

Program Co-chairs

Joan Daemen, Proton World, Belgium
Vincent Rijmen, Cryptomathic, Belgium

Program Committee

Ross Anderson	Cambridge University, UK
Eli Biham	Technion, IL
Don Coppersmith	IBM, USA
Cunshen Ding	Hong Kong University of Science and Technology, HK
Thomas Johansson	Lund University, SE
Mitsuru Matsui	Mitsubishi Electric, JP
Willi Meier	Fachhochschule Aargau, CH
Kaisa Nyberg	Nokia, FI
Bart Preneel	Katholieke Universiteit Leuven, BE

Table of Contents

New Results on Boomerang and Rectangle Attacks*

Eli Biham[1], Orr Dunkelman[1], and Nathan Keller[2]

[1] Computer Science Department, Technion.
Haifa 32000, Israel
{biham,orrd}@cs.technion.ac.il

[2] Mathematics Department, Technion.
Haifa 32000, Israel
nkeller@tx.technion.ac.il

Abstract. The boomerang attack is a new and very powerful cryptanalytic technique. However, due to the adaptive chosen plaintext and ciphertext nature of the attack, boomerang key recovery attacks that retrieve key material on both sides of the boomerang distinguisher are hard to mount. We also present a method for using a boomerang distinguisher, which enables retrieving subkey bits on both sides of the boomerang distinguisher. The rectangle attack evolved from the boomerang attack.In this paper we present a new algorithm which improves the results of the rectangle attack.

Using these improvements we can attack 3.5-round SC2000 with 2^{67} adaptive chosen plaintexts and ciphertexts, and 10-round Serpent with time complexity of $2^{173.8}$ memory accesses (which are equivalent to $2^{165.3}$ Serpent encryptions) with data complexity of $2^{126.3}$ chosen plaintexts.

1 Introduction

Differential cryptanalysis [3] is based on studying the propagation of differences through an encryption function. Since its introduction many techniques based on it were introduced. Some of these techniques, like the truncated differentials [11] and the higher order differentials [2,11], are generalizations of the differential attack. Some other techniques like differential-linear attack [14] and the boomerang attack [18] use the differential attack as a building block.

The boomerang attack is an adaptive chosen plaintext and ciphertext attack. It is based on a pair of short differential characteristics used in a specially built quartet. In the attack a pair of plaintexts with a given input difference are encrypted. Their ciphertexts are used to compute two other ciphertexts according to some other difference, these new ciphertexts are then decrypted, and the difference after the decryption is compared to some (fixed known) value.

* The work described in this paper has been supported by the European Commission through the IST Programme under Contract IST-1999-12324.

The boomerang attack was further developed in [10] into a chosen plaintext attack called the amplified boomerang attack. Later, the amplified boomerang attack was further developed into the rectangle attack [7].

In the transition from the boomerang attack to the rectangle attack the probability of the distinguisher is reduced (in exchange for easing the requirements from adaptive chosen plaintext and ciphertext attack to a chosen plaintext attack). The reduction in the distinguisher's probability results in higher data complexity requirements. For example, the data requirements for distinguishing a 2.5-round SC2000 [17] from a random permutation using a rectangle distinguisher is $2^{84.6}$ chosen plaintext blocks, whereas only $2^{39.2}$ adaptive chosen plaintext and ciphertext blocks are required for the boomerang distinguisher.

In this paper we present a method to retrieve more subkey bits in key recovery boomerang attacks. We also present a better algorithm to perform rectangle attacks. These improvements result in better key recovery attacks which require less data or time (or both) and are more effective. The improvement to the generic rectangle attack reduces the time complexity of attacking 10-round Serpent from 2^{217} memory accesses[1] to $2^{173.8}$ memory accesses which are equivalent to about $2^{166.3}$ 10-round Serpent encryptions. We also prove that these key recovery attacks succeed (with very high probability) assuming that the distinguishers are successful.

The paper is organized as follows: In Section 2 we briefly describe the boomerang and the rectangle attacks. In Section 3 we present our new optimized generic rectangle attack and analyze its application to generic ciphers and to SC2000 and Serpent. In Section 4 we present our optimized generic boomerang attack and analyze its application to both a generic cipher and real blockciphers like SC2000 and Serpent. Section 5 describes a new technique to transform a boomerang distinguisher into a key recovery attack that retrieves more subkey material. We summarize this paper and our new results in Section 6.

2 Introduction to Boomerang and Rectangle Attacks

2.1 The Boomerang Attack

The boomerang attack was introduced in [18]. The main idea behind the boomerang attack is to use two short differentials with high probabilities instead of one differential of more rounds with low probability. The motivation for such an attack is quite apparent, as it is easier to find short differentials with a high probability than finding a long one with a high enough probability.

We assume that a block cipher $E : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$ can be described as a cascade $E = E_1 \circ E_0$, such that for E_0 there exists a differential $\alpha \to \beta$ with probability p, and for E_1 there exists a differential $\gamma \to \delta$ with probability q. The boomerang attack uses the first characteristic ($\alpha \to \beta$) for E_0 with respect to the pairs (P_1, P_2) and (P_3, P_4), and uses the second characteristic

[1] In [7] it was claimed to be 2^{205} due to an error that occurred in the analysis.

$(\gamma \to \delta)$ for E_1 with respect to the pairs (C_1, C_3) and (C_2, C_4). The attack is based on the following boomerang process:

- Ask for the encryption of a pair of plaintexts (P_1, P_2) such that $P_1 \oplus P_2 = \alpha$ and denote the corresponding ciphertexts by (C_1, C_2).
- Calculate $C_3 = C_1 \oplus \delta$ and $C_4 = C_2 \oplus \delta$, and ask for the decryption of the pair (C_3, C_4). Denote the corresponding plaintexts by (P_3, P_4).
- Check whether $P_3 \oplus P_4 = \alpha$.

We call these steps (encryption, XOR by δ and then decryption) a δ–shift.

For a random permutation the probability that the last condition is satisfied is 2^{-n}. For E, however, the probability that the pair (P_1, P_2) is a right pair with respect to the first differential $(\alpha \to \beta)$ is p. The probability that both pairs (C_1, C_3) and (C_2, C_4) are right pairs with respect to the second differential is q^2. If all these are right pairs, then they satisfy $E_1^{-1}(C_3) \oplus E_1^{-1}(C_4) = \beta = E_0(P_3) \oplus E_0(P_4)$, and thus, with probability p also $P_3 \oplus P_4 = \alpha$. Therefore, the probability of this quartet of plaintexts and ciphertexts to satisfy the boomerang conditions is $(pq)^2$. Therefore, $pq > 2^{-n/2}$ must hold for the boomerang distinguisher (and the boomerang key recovery attacks) to work.

The attack can be mounted for all possible β's and γ's simultaneously (as long as $\beta \neq \gamma$), thus, a right quartet for E is built with probability $(\hat{p}\hat{q})^2$, where:

$$\hat{p} = \sqrt{\sum_{\substack{\beta \\ \alpha \to \beta}} \Pr{}^2[\alpha \to \beta]} \quad \text{and} \quad \hat{q} = \sqrt{\sum_{\substack{\gamma \\ \gamma \to \delta}} \Pr{}^2[\gamma \to \delta]}.$$

We refer the reader to [18,7] for the complete description and the analysis.

2.2 The Rectangle Attack

Converting adaptive chosen plaintext and ciphertext distinguishers into key recovery attacks pose several difficulties. Unlike the regular known plaintext, chosen plaintext, or chosen ciphertext distinguishers, using the regular methods of [3,15,11,4,5,14] to use adaptive chosen plaintext and ciphertext distinguishers in key recovery attacks fail, as these techniques require the ability to directly control either the input or the output of the encryption function.

In [10] the amplified boomerang attack is presented. This is a method for eliminating the need of adaptive chosen plaintexts and ciphertexts. The amplified boomerang attack achieves this goal by encrypting many pairs with input difference α, and looking for a quartet (pair of pairs) for which, $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$ when $P_1 \oplus P_2 = P_3 \oplus P_4 = \alpha$. Given the same decomposition of E as before, and the same basic differentials $\alpha \to \beta, \gamma \to \delta$, the analysis shows that the probability of a quartet to be a right quartet is $2^{-(n+1)/2}pq$.

The reason for the lower probability is that no one can guarantee that the γ difference (in the middle of the encryption; needed for the quartet to be a right boomerang quartet) is achieved. The lower probability makes the additional problem (already mentioned earlier) of finding and identifying the right quartets even more difficult.

The rectangle attack [7] shows that it is possible to count over all the possible β's and γ's, and presents additional improvements over the amplified boomerang attack. The improvements presented in the rectangle attack improve the probability of a quartet to be a right rectangle quartet to $2^{-n/2}\hat{p}\hat{q}$.[2]

3 Improving the Rectangle Attack

The main problem dealt in previous works is the large number of possible quartets. Unlike in the boomerang attack, in which the identification of possible quartets is relatively simple, it is hard to find the right quartets in the rectangle attacks since the attacker encrypts a large number of pairs (or structures) and then has to find the right quartets through analysis of the ciphertexts. As the number of possible quartets is quadratic in the number of pairs[3], and as the attacker has to test all the quartets, it is evident that the time complexity of the attack is very large.

In this section we present an algorithm which solves the above problem by exploiting the properties of a right quartet, and which tests only a small part of the possible quartets. The new algorithm is presented on a generic cipher with the following parameters: Let E be a cipher which can be described as a cascade $E = E_f \circ E_1 \circ E_0 \circ E_b$, and assume that E_0 and E_1 satisfy the properties of E_0 and E_1 presented in Section 2 (i.e., there exist differentials $\alpha \to \beta$ with probability p of E_0 and $\gamma \to \delta$ with probability q of E_1). An outline of such an E is presented in Figure 1. We can treat this E as composed of $E' = E_1 \circ E_0$ (for which we have a distinguisher) surrounded by the additional rounds of E_b and E_f. As mentioned in Section 2, for sufficiently high probabilities $\hat{p}, \hat{q}$, we can distinguish $E_1 \circ E_0$ from a random permutation using either a boomerang or a rectangle distinguisher. However, we also like to mount key recovery attacks on the full E.

Recall that the rectangle distinguisher parameters are α, δ, $\hat{p}$, and $\hat{q}$. Given these parameters, the rectangle distinguisher of the cipher $E' = E_1 \circ E_0$ can easily be constructed.

Before we continue we introduce some additional notations: Let X_b be the set of all plaintext differences that may cause a difference α after E_b. Let V_b be the space spanned by the values in X_b. Note that usually $n - r_b$ bits are set to 0 for all the values in V_b. Let $r_b = \log_2 |V_b|$ and $t_b = \log_2 |X_b|$ (r_b and t_b are not necessarily integers). Let m_b be the number of subkey bits which enter E_b and affect the difference of the plaintexts by decrypting pairs whose difference after E_b is α, or formally

[2] This is a lower bound for the probability. For further analysis see [7].

[3] In the rectangle attack the quartet $[(x, y), (z, w)]$ differs from the quartet $[(x, y), (w, z)]$.

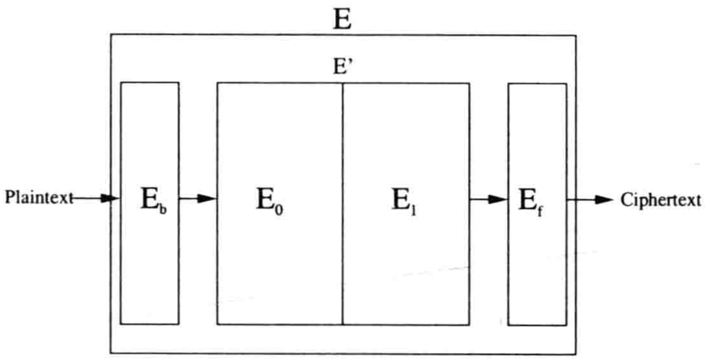

Fig. 1. Outline of E

$$m_b = \left| \left\{ K' \middle| w(K') = 1 \text{ and } \exists K, x : \begin{array}{l} E_{b_K}^{-1}(x) \oplus E_{b_K}^{-1}(x \oplus \alpha) \neq \\ E_{b_{K \oplus K'}}^{-1}(x) \oplus E_{b_{K \oplus K'}}^{-1}(x \oplus \alpha) \end{array} \right\} \right|$$

where $w(x)$ denotes the hamming weight of x.

Similarly, let X_f is the set of all ciphertext differences that a difference δ before E_f may cause. Let V_f denote the space spanned by the values of X_f and denote $r_f = \log_2 |V_f|$. Let $t_f = \log_2 |X_f|$. Let m_f be the number of subkey bits which enter E_f and affect the difference when encrypting a pair with difference δ or formally

$$m_f = \left| \left\{ K' \middle| w(K') = 1 \text{ and } \exists K, x : \begin{array}{l} E_{f_K}(x) \oplus E_{f_K}(x \oplus \alpha) \neq \\ E_{f_{K \oplus K'}}(x) \oplus E_{f_{K \oplus K'}}(x \oplus \alpha) \end{array} \right\} \right| .$$

We outline all these notations in Figure 2.

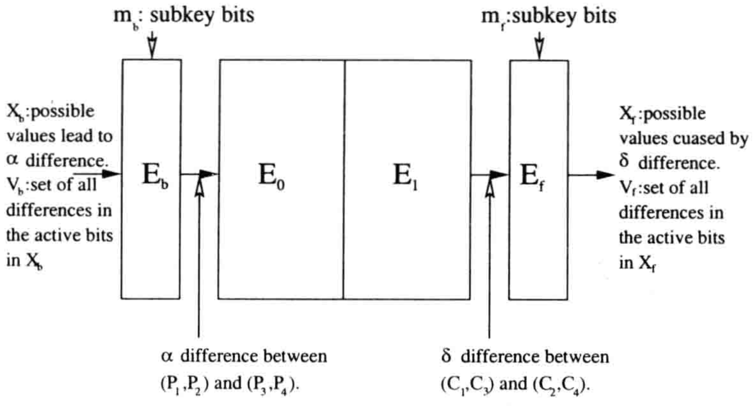
m_b: subkey bits
m_f:subkey bits
X_b:possible values lead to α difference. V_b:set of all differences in the active bits in X_b
E_b
E_0
E_1
E_f
X_f:possible values cuased by δ difference. V_f:set of all differences in the active bits in X_f
α difference between (P_1,P_2) and (P_3,P_4).
δ difference between (C_1,C_3) and (C_2,C_4).

Fig. 2. The Notations Used in This Paper

Our new algorithm for using rectangle distinguisher in a key recovery attack is as follows:

1. Create $Y = \lceil 2^{n/2+2-r_b}/\hat{p}\hat{q} \rceil$ structures of 2^{r_b} plaintexts each. In each structure choose P_0 randomly and let $L = P_0 \oplus V_b$ be the set of plaintexts in the structure.
2. Initialize an array of $2^{m_b+m_f}$ counters. Each counter corresponds to a different guess of the m_b subkey bits of E_b and the m_f subkey bits of E_f.
3. Insert the $N = Y \cdot 2^{r_b}$ ciphertexts into a hash table according to the $n - r_f$ ciphertext bits that are set to 0 in V_f. If a pair agrees on these $n - r_f$ bits, check whether the ciphertext difference is in X_f.
4. For each collision (C_1, C_2) which remains, denote C_i's structure by S_{C_i} and attach to C_1 the index of S_{C_2} and vice versa.
5. In each structure S we search for two ciphertexts C_1 and C_2 which are attached to some other S'. When we find such a pair we check that the $P_1 \oplus P_2$ (the corresponding plaintexts) is in X_b, and check the same for the plaintexts which P_1 and P_2 are related to.
6. For all the quartets which passed the last test denote by (P_1, P_2, P_3, P_4) the plaintexts of a quartet and by (C_1, C_2, C_3, C_4) the corresponding ciphertexts. Increment the counters which correspond to all subkeys K_b, K_f (actually their bits which affect the α and δ differences, respectively) for which $E_{b_{K_b}}(P_1) \oplus E_{b_{K_b}}(P_2) = E_{b_{K_b}}(P_3) \oplus E_{b_{K_b}}(P_4) = \alpha$ and $E^{-1}_{f_{K_f}}(C_1) \oplus E^{-1}_{f_{K_f}}(C_3) = E^{-1}_{f_{K_f}}(C_2) \oplus E^{-1}_{f_{K_f}}(C_4) = \delta$.
7. Output the subkey with maximal number of hits.

The data complexity of the attack is $N = 2^{r_b}Y = 2^{r_b}\lceil 2^{n/2+2-r_b}/\hat{p}\hat{q} \rceil$ chosen plaintexts. The time complexity of Step 1 (the data collection step) is N encryptions. The time complexity of Step 2 is $2^{m_b+m_f}$ memory accesses in a trivial implementation and only one memory access using a more suitable data structures (like B-trees).

Step 3 requires N memory accesses for the insertion of the ciphertexts into a hash table (indexed by the $n - r_f$ bits which are set to 0 in V_f). The number of colliding pairs is about $N^2 \cdot 2^{r_f-n}/2$ as there are N plaintexts divided into 2^{n-r_f} bins (each bin correspond to a value of the $n - r_f$ bits). Note that we not necessarily use all the bins due to large memory requirements (i.e., we can hash only according the first 30 bits set to 0 in V_f). For each collision we check whether the difference of the ciphertexts of the colliding pair belongs to X_f. We keep all the 2^{t_f} values of X_f in a hash table, and thus, the check requires one memory access for each colliding pair. Out of the 2^{r_f} possible differences for a colliding pair, only 2^{t_f} differences are in X_f (i.e., can occur in a right quartet), and thus, about $N^2 \cdot 2^{t_f-n-1}$ pairs remain. The time complexity of this step is $N + N^2 \cdot 2^{r_f-n-1}$ memory accesses on average.

Step 4 requires one memory access for each pair which passes the filtering of Step 3. In a real implementation it is wiser to implement Step 4 as part of Step 3, but we separate these steps for the sake of simpler analysis. As there are

$N^2 \cdot 2^{t_f-n-1}$ such pairs, the time complexity of this step is on average $N^2 \cdot 2^{t_f-n-1}$ memory accesses.

Step 5 implements a search for possible quartets. In a right quartet both (P_1, P_2) and (P_3, P_4) must satisfy that $E_b(P_1) \oplus E_b(P_2) = E_b(P_3) \oplus E_b(P_4) = \alpha$, and thus any right quartet must be combined from some $P_1, P_2 \in S$ and $P_3, P_4 \in \tilde{S}$ where S and $\tilde{S}$ are two (not necessarily distinct) structures. Moreover, a right quartet satisfies that $E_f^{-1}(C_1) \oplus E_f^{-1}(C_3) = E_f^{-1}(C_2) \oplus E_f^{-1}(C_4) = \delta$, and thus C_1 is attached to S_{C_3} and C_2 is attached to S_{C_4} and as P_3, P_4 are from the same structure then – $S_{C_3} = S_{C_4}$. Therefore, in each structure S we search for colliding attachments, i.e., pairs of ciphertexts in S which are attached to the same (other) structure $\tilde{S}$. The $N^2 \cdot 2^{t_f-n-1}$ attachments (colliding pairs) are distributed over Y structures, and we get that approximately $(N \cdot 2^{t_f+r_b-n-1})^2/Y$ possible quartets are suggested in each structure (where a quartet corresponds to a pair of plaintexts from some structure attached to the same structure). We implement the test in the same manner as in Step 3, i.e., keeping a hash table H_S for each structure S and inserting each ciphertext C to H_{S_C} according to the index of the structure attached to C. Denoting the plaintexts of the suggested quartet by (P_1, P_2, P_3, P_4) and their corresponding ciphertexts by (C_1, C_2, C_3, C_4), we first check that $P_1 \oplus P_2 \in X_b$. This test requires one memory access for each possible quartet. The probability that the value $P_1 \oplus P_2$ is in X_b is $2^{t_b-r_b}$. A quartet which fails this test can be discarded immediately. Therefore, out of the $N^2 \cdot 2^{2t_f+2r_b-2n-2}$ possible quartets only $N^2 \cdot 2^{2t_f+r_b+t_b-2n-2}$ quartets remain. As stated before, this filtering requires one memory accesses for each candidate quartet, thus the algorithm requires $N^2 \cdot 2^{2t_f+2r_b-2n-2}$ memory accesses. We can discard more quartets by testing whether $P_3 \oplus P_4 \in X_b$. In total this step requires $N^2 \cdot 2^{2t_f+2r_b-2n-2} \cdot (1 + 2^{t_b-r_b})$ memory accesses and about $N^2 \cdot 2^{2t_f+2t_b-2n-2}$ quartets remain after this step.

In Step 6 we try to deduce the right subkey from the remaining quartets. Recall that a right quartet satisfies $E_b(P_1) \oplus E_b(P_2) = \alpha = E_b(P_3) \oplus E_b(P_4)$. Both pairs are encrypted by the same subkey, hence, a right quartet must agree on K_b (the m_b subkey bits which enter E_b and affect the output difference α). There are 2^{t_b} possible input differences that lead to α difference after E_b, therefore, $2^{m_b-t_b}$ subkeys on average take one of these values into the difference α. As each pair suggests $2^{m_b-t_b}$ subkeys, they agree on average on $(2^{m_b-t_b})^2/2(2^{m_b}) = 2^{m_b-2t_b-1}$ subkeys for E_b. We can find these options by keeping in a precomputed table either the possible values for any pair on its own, or for the whole quartet. Repeating the analysis for E_f, (C_1, C_3), and (C_2, C_4) we get about $2^{m_f-2t_f-1}$ subkeys suggestions from each quartet. Thus, each of the remaining quartets suggests $2^{m_b+m_f-2t_f-2t_b-2}$ possible subkeys. There are $2^{m_b+m_f}$ possible subkeys and $N^2 \cdot 2^{2t_f+2t_b-2n-2} \cdot 2^{m_b+m_f-2t_f-2_b-2} = N^2 \cdot 2^{m_b+m_f-2n-4}$ hits. The expected number of hits for a (wrong) subkey is about $N^2 \cdot 2^{-2n-4}$. Since $N \leq 2^n$ is the number of plaintexts the expected number of hits per wrong subkey is less than $2^{-4} = 1/16$, and we can conclude that the attack almost always succeeds in recovering subkey bits (since the number of expected hits for the right subkey is 4), or at least reduces the number of candidates for the right subkey. We