A Practical-Time Attack on the A5/3 Cryptosystem Used in Third Generation GSM Telephony

Orr Dunkelman, Nathan Keller, and Adi Shamir

Faculty of Mathematics and Computer Science
Weizmann Institute of Science
P.O. Box 26, Rehovot 76100, Israel
{orr.dunkelman,nathan.keller,adi.shamir}@weizmann.ac.il

Abstract. The privacy of most GSM phone conversations is currently protected by the 20+ years old A5/1 and A5/2 stream ciphers, which were repeatedly shown to be cryptographically weak. They will soon be replaced in third generation networks by a new A5/3 block cipher called KASUMI, which is a modified version of the MISTY cryptosystem. In this paper we describe a new type of attack called a *sandwich attack*, and use it to construct a simple distinguisher for 7 of the 8 rounds of KASUMI with an amazingly high probability of 2^{-14}. By using this distinguisher and analyzing the single remaining round, we can derive the complete 128 bit key of the full KASUMI by using only 4 related keys, 2^{26} data, 2^{30} bytes of memory, and 2^{32} time. These complexities are so small that we have actually simulated the attack in less than two hours on a single PC, and experimentally verified its correctness and complexity. Interestingly, neither our technique nor any other published attack can break MISTY in less than the 2^{128} complexity of exhaustive search, which indicates that the changes made by the GSM Association in moving from MISTY to KASUMI resulted in a much weaker cryptosystem.

1 Introduction

The privacy and security of GSM cellular telephony is protected by the A5 family of cryptosystems. The first two members of this family, A5/1 (developed primarily for European markets) and A5/2 (developed primarily for export markets) were designed in the late 1980's in an opaque process and were kept secret until they were reverse engineered in 1999 from actual handsets [13]. Once published, it became clear that A5/2 provided almost no security, and A5/1 could be attacked with practical complexity by a variety of techniques (e.g., [2, 11, 15]). The most recent attack was announced in December 2009, when a team of cryptographers led by Karsten Nohl [1] published a 2 terabyte rainbow table for A5/1, which makes it easy to derive the session key of any particular conversation with minimal hardware support.

In response to these developments, the GSM Association had stated in [23] that they might speed up their transition to a new algorithm called A5/3, and

they plan to discuss this matter in a meeting that will be held in February 2010. The A5/3 algorithm was developed for third generation GSM telephony in 2002, and its specifications were published in 2003 [21]. It is already implemented in about 40% of the three billion available handsets, but very few of the 800 mobile carriers in more than 200 countries which currently use GSM cellular telephony have switched so far to the new standard. Once adopted, A5/3 will become one of the most widely used cryptosystems in the world, and its security will become one of the most important practical issues in cryptography.

The A5/3 cryptosystem is based on the MISTY block cipher which was published at FSE 1997 by Matsui [20]. It has 64 bit blocks, 128 bit keys, and a complex recursive Feistel structure with 8 rounds, each one of which consists of 3 rounds, each one of which has 3 rounds of nonlinear SBox operations. MISTY has provable security properties against various types of attacks, and no attack is known on its full version. The best published attack can be applied to a 6-round reduced variant of the 8-round MISTY, and has a completely impractical time complexity of more than 2^{123} [14]. However, the designers of A5/3 decided to make MISTY faster and more hardware-friendly by simplifying its key schedule and modifying some of its components, calling the new variant *KASUMI*. In [22], the designers provide a rational for each one of these changes, and in particular they analyze the resistance of KASUMI against related key attacks by stating that "removing all the FI functions in the key scheduling part makes the hardware smaller and/or reduces the key set-up time. We expect that related key attacks do not work for this structure". The best attack found by the designers and external evaluators of KASUMI is described as follows: "There are chosen plaintext and/or related key attacks against KASUMI reduced to 5 rounds. We believe that with further analysis it might be possible to extend some attacks to 6 rounds, but not to the full 8 round KASUMI".

The existence of better related key attacks on the full KASUMI was already shown in [7, 19]. Their attack had a data complexity of $2^{54.6}$ and time complexity of $2^{76.1}$, which are impractical but better than exhaustive search. In this paper we develop a new attack, which requires only 4 related keys, 2^{26} data, 2^{30} bytes of memory, and 2^{32} time. Since these complexities are so low, we could verify our attack experimentally, and our unoptimized implementation on a single PC recovered about 96 key bits in a few minutes, and the complete 128 bit key in less than two hours. Careful analysis of our attack technique indicates that it can not be applied against the original MISTY, since it exploits a sequence of coincidences and lucky strikes which were created when MISTY was changed to KASUMI by the GSM Association. This calls into question both the design of KASUMI and its security evaluation against related key attacks.

We use a new type of attack which is an improved version of the boomerang attack introduced in [24]. We call it a "sandwich attack", since it uses a distinguisher which is divided into three parts: A thick slice ("bread") at the top, a thin slice ("meat") in the middle, and a thick slice ("bread") at the bottom. The top and bottom parts are assumed to have high probability differential characteristics, which can be combined into consistent quartet structures by the

standard boomerang technique. However, in our case they are separated by the additional middle slice, which can significantly reduce the probability of the resulting boomerang structure. Nevertheless, as we show in this paper, careful analysis of the dependence between the top and bottom differentials allows us in some cases to combine the two properties above and below the middle slice with an enhanced probability. In particular, we show that in the case of KASUMI we can use top and bottom 3-round differential characteristics with an extremely high probability of 2^{-2} each, and combine them via a middle 1-round slice in such a way that the "price in probability" of the combination is 2^{-6}, instead of the 2^{-32} we would expect from a naive analysis. This increases the probability of our 7-round distinguisher from 2^{-40} to 2^{-14}, and has an even bigger impact on the amount of data and the time complexity of the attack due to the quadratic dependence of the number of cases we have to sample on the distinguishing probability. Such a three level structure was used in several previous attacks such as [9, 10] (where it was called the "Feistel switch" or the "middle round S-box trick"), but to the best of our knowledge it was always used in the past in simpler situations in which the transition probability through the middle layer (in at least one direction) was 1 due to the structural properties of a single Feistel round, or due to the particular construction of a given SBox. Our sandwich attack is the first nontrivial application of such a structure, and the delicacy of the required probabilistic analysis is demonstrated by the fact that a tiny change in the key schedule of KASUMI (which has no effect on the differential probabilities of the top and bottom layers) can change the probability of the combined distinguisher from the surprisingly high value of 2^{-14} to 0.

This paper is organized as follows: Section 2 describes the new sandwich attack, and discusses the transition between the top and bottom parts of the cipher through the middle slice of the sandwich. Section 3 describes the KASUMI block cipher. Section 4 describes our new 7-round distinguisher for KASUMI which has a probability of 2^{-14}, and demonstrates its extreme sensitivity to tiny structural modifications. In Section 5 we use the new distinguisher to develop a practical-time key recovery attack on the full A5/3 cryptosystem.

2 Sandwich Attacks

In this section we describe the technique used in our attacks on KASUMI. We start with a description of the basic (related-key) boomerang attack, and then we describe a new framework, which we call a (related-key) *sandwich attack*, that exploits the dependence between the underlying differentials to obtain a more accurate estimation of the probability of the distinguisher. Finally, we describe the chosen plaintext variant of the attack, which we call (related-key) *rectangle-like sandwich attack*. We note that the idea of using dependence between the differentials in order to improve the boomerang distinguisher was implicitly proposed by Wagner [24], and was also used in some simple scenarios in [9, 10]. Therefore, our framework can be considered as a formal treatment and generalization of the ideas proposed in [9, 10, 24].

2.1 The Basic Related-Key Boomerang Attack

The related-key boomerang attack was introduced by Kim et al. [18, 16], and independently by Biham et al. [6], as a combination of the boomerang attack [24] and the related-key differential attack [17]. In this attack, the cipher is treated as a cascade of two sub-ciphers $E = E_1 \circ E_0$, and related-key differentials of E_0 and E_1 are combined into an adaptive chosen plaintext and ciphertext distinguisher for E.

Let us assume that there exists a related-key differential $\alpha \rightarrow \beta$ for E_0 under key difference ΔK_{ab} with probability p. (i.e., $\mathbf{Pr}[E_{0(K)}(P) \oplus E_{0(K \oplus K_{ab})}(P \oplus \alpha) = \beta] = p$, where $E_{0(K)}$ denotes encryption through E_0 under the key K). Similarly, we assume that there exists a related-key differential $\gamma \rightarrow \delta$ for E_1 under key difference ΔK_{ac} with probability q. The related-key boomerang distinguisher requires encryption/decryption under the secret key K_a, and under the related-keys $K_b = K_a \oplus \Delta K_{ab}$, $K_c = K_a \oplus \Delta K_{ac}$, and $K_d = K_c \oplus \Delta K_{ab} = K_b \oplus \Delta K_{ac}$. The algorithm of the distinguisher is extremely simple:

1. Choose M plaintexts at random, and initialize a counter C to zero. For each plaintext P_a, perform the following:
   (a) Ask for the ciphertexts $C_a = E_{K_a}(P_a)$ and $C_b = E_{K_b}(P_b)$ where $P_b = P_a \oplus \alpha$.
   (b) Ask for the plaintexts $P_c = E^{-1}_{K_c}(C_c)$ and $P_d = E^{-1}_{K_d}(C_d)$ where $C_c = C_a \oplus \delta$ and $C_d = C_b \oplus \delta$.
   (c) If $P_c \oplus P_d = \alpha$, increment the counter C by 1.
2. If $C > Threshold$, output "E". Otherwise, output "Random Permutation".

A quartet constructed by the boomerang attack is depicted on the left side of Figure 1.

For a random permutation the probability that the last condition is satisfied is 2^{-n}, where n is the block size. For E, the probability that the pair (P_a, P_b) is a right pair with respect to the first differential (i.e., the probability that the intermediate difference after E_0 equals β, as predicted by the differential) is p. Assuming independence, the probability that both pairs (C_a, C_c) and (C_b, C_d) are right pairs with respect to the second differential is q^2. If all these are right pairs, then $E_1^{-1}(C_c) \oplus E_1^{-1}(C_d) = \beta = E_0(P_c) \oplus E_0(P_d)$. Thus, with probability p, $P_c \oplus P_d = \alpha$. Hence, the total probability of this quartet of plaintexts and ciphertexts to satisfy the condition $P_c \oplus P_d = \alpha$ is at least $(pq)^2$. Therefore, if $pq \gg 2^{-n/2}$, the algorithm above allows to distinguish E from a random permutation given $O((pq)^{-2})$ adaptively chosen plaintexts and ciphertexts.

The distinguisher can be improved by considering multiple differentials of the form $\alpha \rightarrow \beta'$ and $\gamma' \rightarrow \delta$ (for the same α and δ). We omit this improvement here since it is not used in our attack on KASUMI, and refer the reader to [6]. For a rigorous treatment of the related-key boomerang attack, including a discussion of the independence assumptions the attack relies upon, we refer the interested reader to [19].

The standard way to use the related-key boomerang distinguisher in a key-recovery attack is to add a round before the distinguisher and retrieve key material in the first round. We use the dual technique of adding a round after the distinguisher, applying the attack in a chosen ciphertext/adaptive chosen plaintext manner, and retrieving key material in the last round.

2.2 Related-Key Sandwich Attacks

In this framework we consider the cipher as a cascade of three sub-ciphers: $E = E_1 \circ M \circ E_0$. Our assumptions are the same as in the basic attack: We assume that there exists a related-key differential $\alpha \rightarrow \beta$ for E_0 under key difference ΔK_{ab} with probability p, and a related-key differential $\gamma \rightarrow \delta$ for E_1 under key difference ΔK_{ac} with probability q. The attack algorithm is also exactly the same as in the basic attack (ignoring the middle sub-cipher M). However, the analysis is more delicate and requires great care in analyzing the dependence between the various distributions.

The main idea behind the sandwich attack is the transition in the middle. In the basic boomerang attack, if the pair (P_a, P_b) is a right pair with respect to the first differential, and both pairs (C_a, C_c) and (C_b, C_d) are right pairs with respect to the second differential, then we have

$$(X_a \oplus X_b = \beta) \wedge (X_a \oplus X_c = \gamma) \wedge (X_b \oplus X_d = \gamma), \tag{1}$$

where X_i is the intermediate encryption value of P_i, and thus

$$X_c \oplus X_d = (X_c \oplus X_a) \oplus (X_a \oplus X_b) \oplus (X_b \oplus X_d) = \beta \oplus \gamma \oplus \gamma = \beta, \tag{2}$$

resulting in $P_c \oplus P_d = \alpha$ with probability p (see Figure 1).

In the new sandwich framework, instead of condition (1), we get

$$(X_a \oplus X_b = \beta) \wedge (Y_a \oplus Y_c = \gamma) \wedge (Y_b \oplus Y_d = \gamma). \tag{3}$$

Therefore, the probability of the three-layer related-key boomerang distinguisher is $p^2 q^2 r$, where

$$r = \mathbf{Pr}\left[(X_c \oplus X_d = \beta) \middle| (X_a \oplus X_b = \beta) \wedge (Y_a \oplus Y_c = \gamma) \wedge (Y_b \oplus Y_d = \gamma)\right]. \tag{4}$$

Without further assumptions on M, r is expected to be very low (close to 2^{-n}), and thus the distinguisher is expected to fail. However, as observed in [9, 10, 24], in some cases the differentials in E_0 and E_1 can be chosen such that the probability penalty r in going through the middle sub-cipher (in at least one direction) is 1, which is much higher than expected.

An example of this phenomenon, introduced in [24] and described in [10] under the name "Feistel switch", is the following. Let E be a Feistel cipher, decomposed as $E = E_1 \circ M \circ E_0$, where M consists of one Feistel round (see Figure 2). Assume that the differentials $\alpha \rightarrow \beta$ (for E_0) and $\gamma \rightarrow \delta$ (for E_1) have

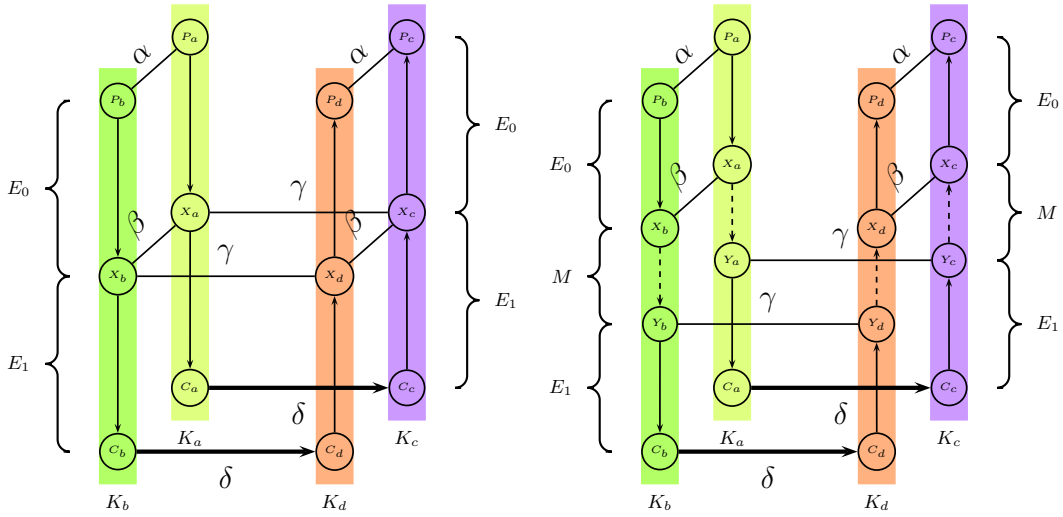


Fig. 1. Related-Key Boomerang and Sandwich Quartets

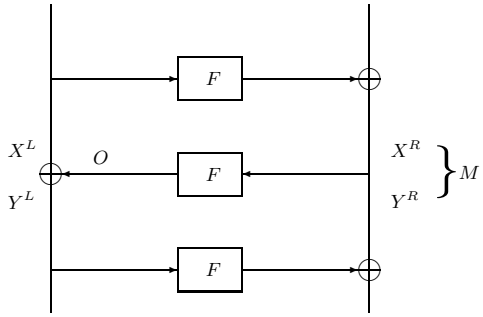


Fig. 2. A Feistel construction. M is the second round.

no key difference (i.e., $\Delta K_{ab} = \Delta K_{ac} = 0$), and satisfy $\beta^R = \gamma^L$ (i.e., the right half of β equals the left half of γ). We would like to compute the value of r.

Assume that condition (3) holds. In this case, by the Feistel construction, $X_i^R = Y_i^L$ for all i, we have

$$X_a^R \oplus X_b^R = \beta^R = \gamma^L = X_a^R \oplus X_c^R = X_b^R \oplus X_d^R, \tag{5}$$

and thus,

$$(X_a^R = X_d^R) \qquad \text{and} \qquad (X_b^R = X_c^R). \tag{6}$$

Therefore, the output values of the F-function in the Feistel round represented by M, denoted by (O_a, O_b, O_c, O_d), satisfy

$$(O_a = O_d) \qquad \text{and} \qquad (O_b = O_c).$$

Since by the Feistel construction, $X_i^L = Y_i^R \oplus O_i$ and by condition (3), $Y_a \oplus Y_b \oplus Y_c \oplus Y_d = 0$, it follows that

$$X_a \oplus X_b \oplus X_c \oplus X_d = 0,$$

which by condition (3) implies $X_c \oplus X_d = \beta$. Thus, in this case we get

$$r = \mathbf{Pr}\left[(X_c \oplus X_d = \beta) \middle| (X_a \oplus X_b = \beta) \wedge (Y_a \oplus Y_c = \gamma) \wedge (Y_b \oplus Y_d = \gamma)\right] = 1,$$

independently of the choice of the F-function used.

Other examples of the same phenomenon are considered in [9] (under the name "middle round S-box trick"), and in [10] (under the names "ladder switch" and "S-box switch").

Our attack on KASUMI is the first non-trivial example of this phenomenon in which a careful analysis shows that r is smaller than 1, but much larger than its expected value under the standard independence assumptions. In our attack, the cipher E (7-round KASUMI) is a Feistel construction, M consists of a single round, and $\beta = \gamma$. However, the argument presented above cannot be applied directly since there is a non-zero key difference in M, and thus a zero input difference to the F-function does not imply zero output difference. Instead, we analyze the F-function thoroughly and show that in this case, $r = 2^{-6}$ (instead of 2^{-32}, which is the expected value for a random Feistel round in a 64-bit cipher).

Remark 1. We note that our treatment of the sandwich distinguisher allows us to specify the precise independence assumptions we rely upon. Since r is defined as a conditional probability, the only independence assumptions we use are between the differentials of E_0 and E_1, and thus the formula p^2q^2r relies on exactly the same assumptions as the ordinary boomerang attack. Moreover, in our case the assumptions seem more likely to hold since the insertion of M in the middle decreases the potential dependencies between the differentials for E_0 and the differentials for E_1. In [9, 10, 24], this situation was treated as a "trick" allowing to increase the probability of the distinguisher, or in other words, as a failure of the formula p^2q^2 in favor of the attacker. This approach is problematic since once we claim that the entire formula does not hold due to dependencies, we cannot rely on independence assumptions in other places where such dependencies were not found yet.

2.3 A Rectangle-Like Sandwich Attack

The transformation of the (related-key) boomerang distinguisher into a chosen plaintext rectangle attack relies on standard birthday-paradox arguments. The division into sub-ciphers and the assumptions are the same as in the (related-key)

boomerang distinguisher. The key idea behind the transformation is to encrypt many plaintext pairs with input difference α, and to look for quartets (i.e., pairs of pairs) that happen to conform to the requirements of the boomerang process. In other words, the adversary considers quartets of plaintexts of the form $((P_a, P_b = P_a \oplus \alpha), (P_c, P_d = P_c \oplus \alpha))$ encrypted under the related-keys K_a, K_b, K_c, and K_d, respectively, and a quartet is called a "right quartet" if the following conditions are satisfied:

1. $E_{0(K_a)}(P_a) \oplus E_{0(K_b)}(P_b) = \beta = E_{0(K_c)}(P_c) \oplus E_{0(K_d)}(P_d)$.
2. $E_{0(K_a)}(P_a) \oplus E_{0(K_a)}(P_c) = \gamma$ (which leads to $E_{0(K_b)}(P_b) \oplus E_{0(K_d)}(P_d) = \gamma$ if this condition holds along with the previous one).
3. $C_a \oplus C_c = \delta = C_b \oplus C_d$.

The probability of a quartet to be a right quartet is a lower bound on the probability of the event

$$C_a \oplus C_c = \delta = C_b \oplus C_d. \tag{7}$$

The usual assumption is that each of the above conditions is independent of the rest, and hence the probability that a given quartet $((P_1, P_2), (P_3, P_4))$ is a right quartet is $p^2 \cdot 2^{-n} \cdot q^2$. Since for a random permutation, the probability of condition (7) is 2^{-2n}, the rectangle process can be used to distinguish E from a random permutation if $pq \gg 2^{-n/2}$ (the same value as in the standard boomerang distinguisher).

However, the data complexity of the distinguisher is $O(2^{n/2}(pq)^{-1})$, which is much higher than the complexity of the boomerang distinguisher. The higher data complexity follows from the fact that the event $E_{0(K_a)}(P_a) \oplus E_{0(K_c)}(P_c) = \gamma$ occurs with a "random" probability of 2^{-n} (in fact, this is the birthday-paradox argument behind the construction). The identification of right quartets is also more complicated than in the boomerang case, as instead of checking a condition on pairs, the adversary has to go over all the possible quartets. At the same time, the chosen plaintext nature allows using stronger key recovery techniques. An optimized method of finding the right rectangle quartets is presented in [5].

The transformation of the (related-key) sandwich framework into the (related-key) rectangle-like sandwich framework is performed similarly. The algorithm of the distinguisher remains the same, and the probability of a quartet to be a right quartet is $p^2 \cdot 2^{-n} \cdot r' \cdot q^2$, where

$$r' = \mathbf{Pr}\left[(Y_b \oplus Y_d = \gamma) \middle| (X_a \oplus X_b = \beta) \wedge (X_c \oplus X_d = \beta) \wedge (Y_a \oplus Y_c = \gamma)\right]. \tag{8}$$

It follows from symmetry arguments that in the case where E is a Feistel cipher, M consists of a single round, and $\beta^R = \gamma^L$, we have $r' = r$. Thus, in our attack on KASUMI we will be able to use the computation of r in the sandwich framework to find also the probability of the related-key rectangle-like sandwich distinguisher in the case of a 3-slice sandwich structure.

3 The KASUMI Block Cipher

KASUMI [21] is a 64-bit block cipher with 128-bit keys. It has a recursive Feistel structure, following its ancestor MISTY. The cipher has eight Feistel rounds, where each round is composed of two functions: the FO function which is in itself a 3-round 32-bit Feistel construction, and the FL function that mixes a 32-bit subkey with the data in a linear way. The order of the two functions depends on the round number: in the even rounds the FO function is applied first, and in the odd rounds the FL function is applied first.

The FO function also has a recursive structure: its F-function, called FI, is a four-round Feistel construction. The FI function uses two non-linear S-boxes $S7$ and $S9$ (where $S7$ is a 7-bit to 7-bit permutation and $S9$ is a 9-bit to 9-bit permutation), and accepts an additional 16-bit subkey, which is mixed with the data. In total, a 96-bit subkey enters FO in each round — 48 subkey bits are used in the FI functions and 48 subkey bits are used in the key mixing stages.

The FL function accepts a 32-bit input and two 16-bit subkey words. One subkey word affects the data using the OR operation, while the second one affects the data using the AND operation. We outline the structure of KASUMI and its parts in Fig. 3.

The key schedule of KASUMI is much simpler than the original key schedule of MISTY, and the subkeys are linearly derived from the key. The 128-bit key K is divided into eight 16-bit words: $K_1, K_2, \ldots, K_8$. Each K_i is used to compute $K'_i = K_i \oplus C_i$, where the C_i's are fixed constants (we omit these from the paper, and refer the intrigued reader to [21]). In each round, eight words are used as the round subkey (up to some in-word rotations). Hence, each 128-bit round subkey is a linearly modified version of the secret key. We summarize the details of the key schedule of KASUMI in Table 1.

Round	$KL_{i,1}$	$KL_{i,2}$	$KO_{i,1}$	$KO_{i,2}$	$KO_{i,3}$	$KI_{i,1}$	$KI_{i,2}$	$KI_{i,3}$
1	$K_1 \lll 1$	K'_3	$K_2 \lll 5$	$K_6 \lll 8$	$K_7 \lll 13$	K'_5	K'_4	K'_8
2	$K_2 \lll 1$	K'_4	$K_3 \lll 5$	$K_7 \lll 8$	$K_8 \lll 13$	K'_6	K'_5	K'_1
3	$K_3 \lll 1$	K'_5	$K_4 \lll 5$	$K_8 \lll 8$	$K_1 \lll 13$	K'_7	K'_6	K'_2
4	$K_4 \lll 1$	K'_6	$K_5 \lll 5$	$K_1 \lll 8$	$K_2 \lll 13$	K'_8	K'_7	K'_3
5	$K_5 \lll 1$	K'_7	$K_6 \lll 5$	$K_2 \lll 8$	$K_3 \lll 13$	K'_1	K'_8	K'_4
6	$K_6 \lll 1$	K'_8	$K_7 \lll 5$	$K_3 \lll 8$	$K_4 \lll 13$	K'_2	K'_1	K'_5
7	$K_7 \lll 1$	K'_1	$K_8 \lll 5$	$K_4 \lll 8$	$K_5 \lll 13$	K'_3	K'_2	K'_6
8	$K_8 \lll 1$	K'_2	$K_1 \lll 5$	$K_5 \lll 8$	$K_6 \lll 13$	K'_4	K'_3	K'_7

$(X \lll i)$ — X rotated to the left by i bits

Table 1. KASUMI's Key Schedule Algorithm

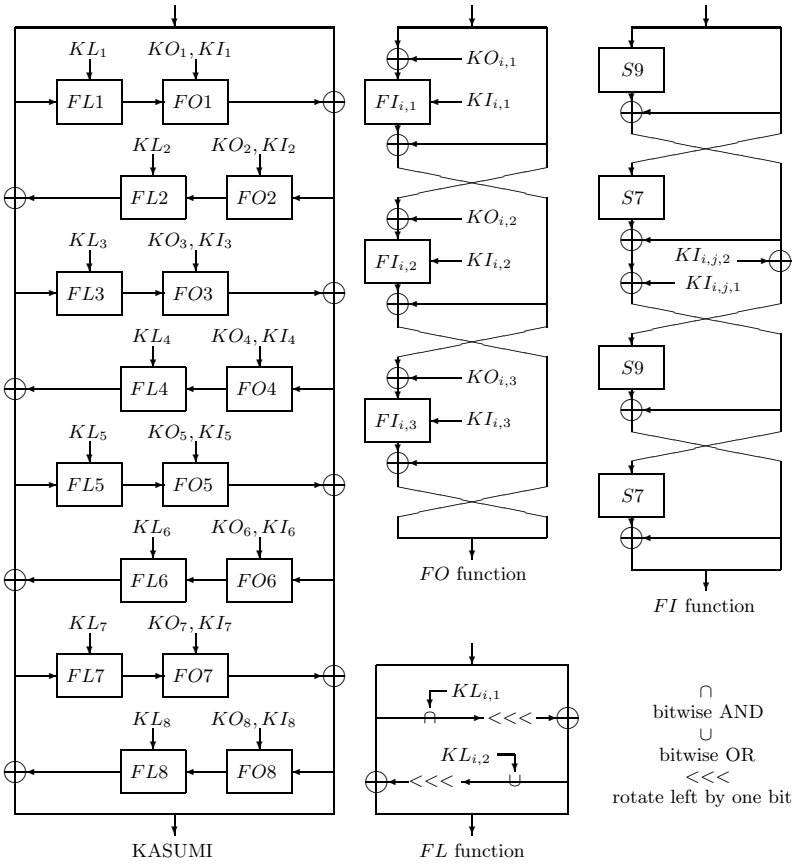


Fig. 3. Outline of KASUMI

4 A Related-Key Sandwich Distinguisher for 7-Round KASUMI

4.1 The New Distinguisher

In our distinguisher, we treat rounds 1–7 of KASUMI as a cascade $E = E_1 \circ M \circ E_0$, where E_0 consists of rounds 1–3, M consists of round 4, and E_1 consists of rounds 5–7. The related-key differential we use for E_0 is a slight modification of the differential characteristic presented in [12], in which

$$\alpha = (0_x, 0010\ 0000_x) \rightarrow (0_x, 0010\ 0000_x) = \beta.$$

The corresponding key difference is $\Delta K_{ab} = (0, 0, 8000_x, 0, 0, 0, 0, 0)$, i.e., only the third key word has the single bit difference $\Delta K_3 = 8000_x$. This related-key differential is depicted in Figure 4. The related-key differential we use for E_1 is

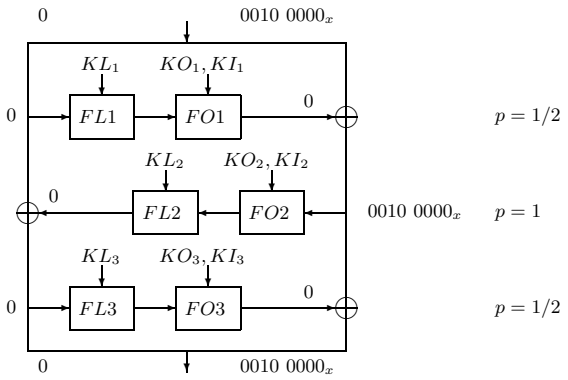


Fig. 4. 3-Round Related-Key Differential Characteristic of KASUMI

the same differential shifted by four rounds, in which the data difference is the same, but the key difference is $\Delta K_{ac} = (0, 0, 0, 0, 0, 0, 8000_x, 0)$ (to handle the different subkeys used in these rounds).

As shown in [12], the probability of each one of of these 3-round differential characteristics is $1/4$. In order to find the probability of the related-key sandwich distinguisher, we have to compute the probability

$$\mathbf{Pr}\left[(X_c \oplus X_d = \beta)\middle|(X_a \oplus X_b = \beta) \wedge (Y_a \oplus Y_c = \gamma) \wedge (Y_b \oplus Y_d = \gamma)\right], \quad (9)$$

where (X_a, X_b, X_c, X_d) and (Y_a, Y_b, Y_c, Y_d) are the intermediate values before and after the middle slice of the sandwich during the encryption/decryption of the quartet (P_a, P_b, P_c, P_d) (see the right side of Figure 1). This computation, which is a bit complicated, spans the rest of this subsection.

Consider a quartet (P_a, P_b, P_c, P_d) for which the condition

$$(X_a \oplus X_b = \beta) \wedge (Y_a \oplus Y_c = \gamma) \wedge (Y_b \oplus Y_d = \gamma) \quad (10)$$

is satisfied. As explained in Section 2, since M is a single Feistel round, this implies that

$$(X_a^R = X_d^R) \wedge (X_b^R = X_c^R), \quad (11)$$

where X_i^R denotes the right half of X_i that enters the function $FO4$. Moreover, as the right half of the differences $\beta = \gamma$ is zero, we have

$$X_a^{RR} = X_b^{RR} = X_c^{RR} = X_d^{RR}, \quad (12)$$

where X_i^{RR} denotes the right half (i.e., the 16 right bits) of X_i^R.

Consider now the computation depicted in Figure 5. The function $FO4$ is a 3-round Feistel construction whose 32-bit values after round j are denoted by $(X_a^j, X_b^j, X_c^j, X_d^j)$, and the function FI is a 4-round Feistel construction whose 16-bit values after round j are denoted by $(I_a^j, I_b^j, I_c^j, I_d^j)$. Note that the key differences ΔK_{ab} and ΔK_{ac} affect in round 4 the subkeys $KI_{4,3}$ and $KI_{4,2}$, respectively, and in particular, there is no key difference in the first round of $FO4$. As a result, Equation (11) implies that

$$(X_a^1 = X_d^1) \wedge (X_b^1 = X_c^1). \tag{13}$$

Furthermore, there is no key difference in the pairs corresponding to (P_a, P_b) and (P_c, P_d) in the second round of $FO4$, and thus Equation (12) implies

$$(I_a^2 = I_b^2) \wedge (I_c^2 = I_d^2). \tag{14}$$

Combining equations (13) and (14), we get the following relation in the right half of the intermediate values after round 3 of $FO4$:

$$X_a^{3R} \oplus X_b^{3R} \oplus X_c^{3R} \oplus X_d^{3R} = 0. \tag{15}$$

In the F-function of round 3 of $FO4$ we consider the pairs corresponding to (P_a, P_d) and (P_b, P_c). Since the key difference in these pairs (that equals $K_{ab} \oplus K_{ac}$) affects only the subkey $KI_{4,3,1}$, Equation (13) implies

$$I_a^{3R} \oplus I_b^{3R} \oplus I_c^{3R} \oplus I_d^{3R} = 0 \tag{16}$$

in the right hand side of the output. In the left hand side of the output, the XOR of the four values is not necessarily equal to zero, due to the subkey difference that affects the inputs to the second $S7$ in $FI_{4,3}$. However, if these 7-bit inputs, denoted by (J_a, J_b, J_c, J_d), satisfy one of the conditions:

$$((J_a = J_b) \wedge (J_c = J_d)) \qquad \text{or} \qquad ((J_a = J_c) \wedge (J_b = J_d)), \tag{17}$$

then Equation (16) implies

$$I_a^{3L} \oplus I_b^{3L} \oplus I_c^{3L} \oplus I_d^{3L} = 0. \tag{18}$$

Since we have $J_a \oplus J_d = J_b \oplus J_c$ (both are equal to the subkey difference in $KI_{4,3,1}$), each one of the two conditions in Equation (17) is expected to hold[1] with probability 2^{-7}. Therefore, combining Equations (15), (16), and (18) we get that the condition

$$X_a^3 \oplus X_b^3 \oplus X_c^3 \oplus X_d^3 = 0 \tag{19}$$

holds with probability 2^{-6}.

[1] This estimate is based on a randomness assumption that could be inaccurate in our case due to dependence between the differential characteristics. However, the experiments presented below verify that this probability is indeed as expected.

Finally, since the FL function is linear for a given key and there is no key difference in $FL4$, we can conclude that whenever Equation (19) holds, the outputs of the F-function in round 4 (denoted by $(O_a^4, O_b^4, O_c^4, O_d^4)$) satisfy

$$O_a^4 \oplus O_b^4 \oplus O_c^4 \oplus O_d^4 = 0 \tag{20}$$

with probability 2^{-6}. Since by condition (10),

$$Y_a^L \oplus Y_b^L \oplus Y_c^L \oplus Y_d^L = 0,$$

it follows that

$$X_a^L \oplus X_b^L \oplus X_c^L \oplus X_d^L = 0 \tag{21}$$

also holds with probability 2^{-6}. Combining it with Equation (11) yields

$$\mathbf{Pr}\left[(X_c \oplus X_d = \beta)\middle|(X_a \oplus X_b = \beta) \wedge (Y_a \oplus Y_c = \gamma) \wedge (Y_b \oplus Y_d = \gamma)\right] = 2^{-6}. \tag{22}$$

Therefore, the overall probability of the related-key sandwich distinguisher is

$$(1/4)^2 \cdot (1/4)^2 \cdot 2^{-6} = 2^{-14}, \tag{23}$$

which is much higher than the probability of $(1/4)^2 \cdot (1/4)^2 \cdot 2^{-32} = 2^{-40}$ which is expected by the naive analysis of the sandwich structure.

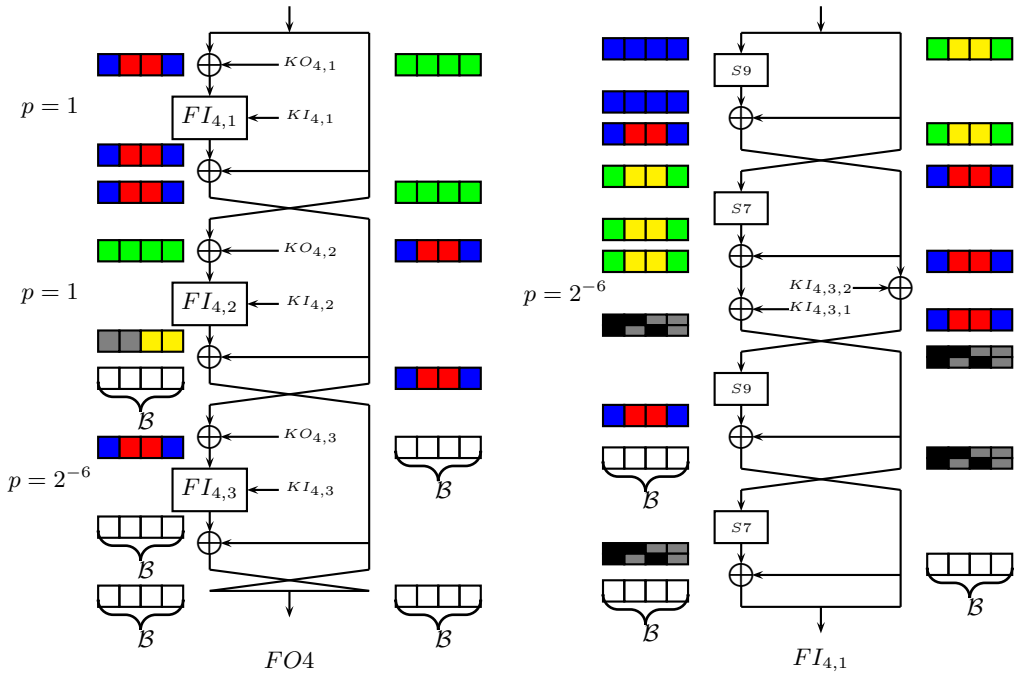
4.2 Experimental Verification

To verify the properties of the new distinguisher, we used the official code available as an appendix in [21]. The verification experiment was set up as follows: In each test we randomly chose a key quartet satisfying the required key differences. We then generated 2^{16} quartets by following the boomerang procedure described above. We utilized a slight improvement of the first differential suggested in [12] that increases its probability in the encryption direction by a factor of 2 by fixing the value of two plaintext bits. Hence, we expect the number of right quartets in each test to be distributed according to a Poisson distribution with a mean value of $2^{16} \cdot 2^{-14} \cdot 2 = 8$. We repeated the test 100,000 times, and obtained a distribution which is extremely close to the expected distribution. The full results are summarized in Table 2.

4.3 A Counter Example

In this subsection we present an example which demonstrates the extremely delicate nature of such probability estimations. In this example, we make a tiny change in the key schedule of KASUMI, which does not seem to have any effect on the differential probabilities of any one of its three sub-ciphers. However, for this example, the probability of the distinguisher is zero!

The only change we make in KASUMI is that in the first subkey, the roles of the key words K_5' and K_8' are interchanged. The rest of the key schedule is



Values marked by the same color are equal. Values marked by $\mathcal{B}$ are balanced (i.e., the XOR of all four values is 0). The values in $FI_{4,3}$ which are either gray or black suggest one of two possible cases.

Fig. 5. The Development of Differences in FO_4 and in $FI_{4,3}$

modified according to the rule of the original KASUMI, i.e., the key words are rotated cyclically in each round (e.g., $KI_{2,1} = K'_1, KI_{3,3} = K'_7$).

Since our change affects only the subkeys used in $KI_{i,1}$ and $KI_{i,3}$ in each round, the differentials used in our distinguisher on KASUMI remain exactly the same for the new variant (with the same input/output differences, the same key differences and the same probabilities). However, we claim that in this case,

$$r = \mathbf{Pr}\left[(X_c \oplus X_d = \beta) \middle| (X_a \oplus X_b = \beta) \wedge (Y_a \oplus Y_c = \gamma) \wedge (Y_b \oplus Y_d = \gamma)\right] = 0, \tag{24}$$

and thus the probability of the distinguisher is zero. In all the computations below, the notations are the same as in the original distinguisher above. The impossible transition is depicted in Figure 6.

Since the differentials are the same as in the original distinguisher, we have

$$(X_a^R = X_d^R) \wedge (X_b^R = X_c^R)$$

and

$$X_a^{RR} = X_b^{RR} = X_c^{RR} = X_d^{RR}.$$

Right Quartets	0	1	2	3	4	5	6	7
Theory ($Poi(8)$)	34	268	1,073	2,863	5,725	9,160	12,214	13,959
Experiment	32	259	1,094	2,861	5,773	9,166	12,407	13,960
Right Quartets	8	9	10	11	12	13	14	15
Theory ($Poi(8)$)	13,959	12,408	9,926	7,219	4,813	2,962	1,692	903
Experiment	13,956	12,230	9,839	7,218	4,804	3,023	1,672	859
Right Quartets	16	17	18	19	20	21	22	23
Theory ($Poi(8)$)	451	212	94	40	16	6	2	0.8
Experiment	472	219	89	39	13	12	2	0
Right Quartets	24	25						
Theory ($Poi(8)$)	0.26	0.082						
Experiment	0	1						

Table 2. The Number of Right Quartets in 100,000 Experiments

Also, since the second round of $FO4$ is unchanged, we have

$$(I_a^2 = I_b^2) \wedge (I_c^2 = I_d^2).$$

Therefore,

$$X_a^{3R} \oplus X_b^{3R} \oplus X_c^{3R} \oplus X_d^{3R} = I_a^1 \oplus I_b^1 \oplus I_c^1 \oplus I_d^1.$$

In the first round of $FO4$ we have a difference between the modified variant and the original KASUMI, as in the modified variant there is subkey difference in the pairs corresponding to (P_a, P_b) and to (P_c, P_d) in the MSB of the subkey $KI_{4,1,1}$. Let us analyze the function $FI_{4,1}$.

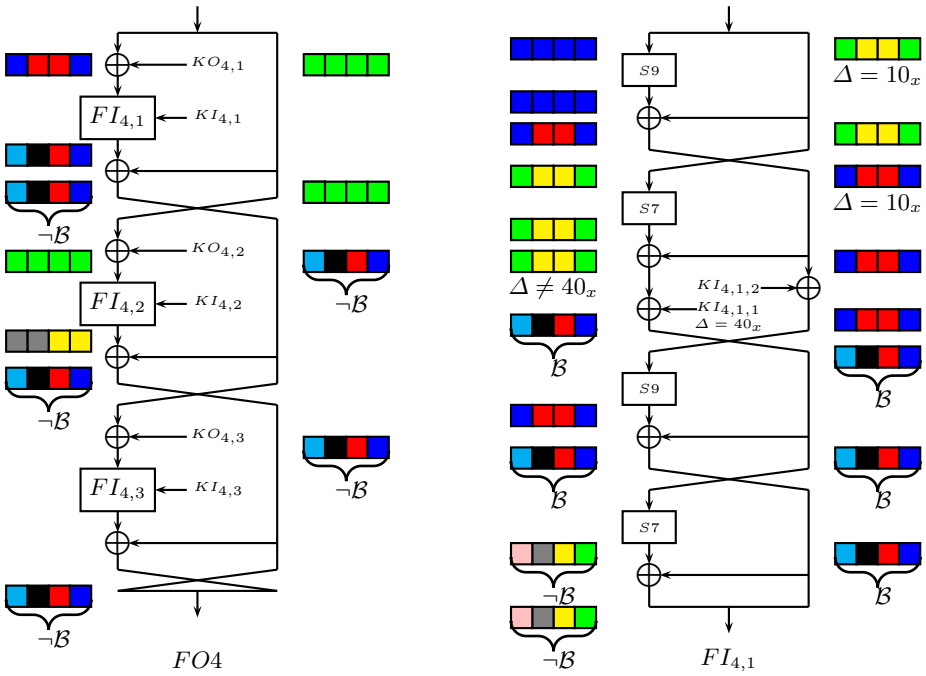
By the structure of the differential, its inputs are of the form

$$(X_a^R \oplus KO_{4,1}, X_b^R \oplus KO_{4,1}, X_c^R \oplus KO_{4,1}, X_d^R \oplus KO_{4,1}) = (t, t \oplus 0010_x, t \oplus 0010_x, t).$$

After the first round of $FI_{4,1}$ the values remain in the form $(t', t' \oplus 0010_x, t' \oplus 0010_x, t')$. After the second round the values are of the form u, v, v, u. We claim that $u \oplus v \neq MSB$. Indeed, if we had $u \oplus v = MSB$, then the 7-bit outputs of the S-box $S7$ had to be of the form $(u', u' \oplus 1000000_2 \oplus 0010000_2, u' \oplus 1000000_2 \oplus 0010000_2, u')$. However, the differential $(0010000_2 \rightarrow 1010000_2)$ is impossible for $S7$, and thus this event cannot occur.

We now claim that the four 7-bit intermediate values after the XOR with the subkey $KI_{4,1,1}$ are different. Indeed, these values are of the form $(u \oplus k, v \oplus k \oplus MSB, v \oplus k, u \oplus k \oplus MSB)$, and these are all different since $u \neq v$ and $u \neq v \oplus MSB$.

Finally, we consider the S-box $S7$ in the fourth round of $FI_{4,1}$. Its four inputs are all different, and can be divided into two pairs $(u \oplus k, v \oplus k \oplus MSB)$ and



Values marked by the same color are equal. Values marked by $\neg\mathcal{B}$ are not balanced (i.e., the XOR of all four values is not 0). The values in $FI_{4,1}$ with difference $\Delta =??$ below them suggest the difference. Note that in S7 of KASUMI, $10_x \not\to 50_x$.

Fig. 6. The Development of Differences in FO_4 and in $FI_{4,1}$ in the Modified KASUMI

$(v \oplus k, u \oplus k \oplus MSB)$ with the same difference. Since $S7$ is an *almost perfect nonlinear permutation*, this implies that the two corresponding pairs of outputs have distinct differences, and thus, the XOR of the four output values is necessarily non-zero. Since the XOR of the output values in the right half is zero, we have

$$I_a^1 \oplus I_b^1 \oplus I_c^1 \oplus I_d^1 \neq 0,$$

and hence,

$$X_a^{3R} \oplus X_b^{3R} \oplus X_c^{3R} \oplus X_d^{3R} \neq 0.$$

Therefore, the XOR of the four outputs of $FO4$ is non-zero with probability 1, and since FL is linear, this implies that the XOR of the four outputs of FL is non-zero with probability 1. This proves that

$$\mathbf{Pr}\left[(X_c \oplus X_d = \beta) \middle| (X_a \oplus X_b = \beta) \wedge (Y_a \oplus Y_c = \gamma) \wedge (Y_b \oplus Y_d = \gamma)\right] = 0,$$

and thus the distinguisher fails in this variant of KASUMI, as asserted.

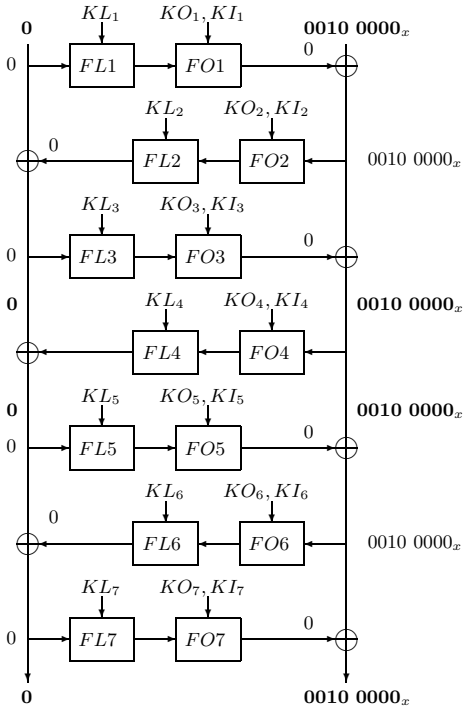


Fig. 7. The 7-Round Related-Key Sandwich Distinguisher of KASUMI

For the sake of completeness, we also implemented this variant of KASUMI, and verified experimentally that the number of right quartets with the desired sandwich property was always zero.[2]

5 Related-Key Sandwich Attack on the Full KASUMI

Our attack on the full KASUMI (depicted in Figure 7) applies the distinguisher presented in Section 4 to rounds 1–7, and retrieves subkey material in round 8. Let $\Delta K_{ab} = (0, 0, 8000_x, 0, 0, 0, 0, 0)$ and $\Delta K_{ac} = (0, 0, 0, 0, 0, 0, 8000_x, 0)$, and let K_a, $K_b = K_a \oplus \Delta K_{ab}$, $K_c = K_a \oplus \Delta K_{ac}$, and $K_d = K_c \oplus \Delta K_{ab}$ be the unknown related keys we wish to retrieve.

The attack algorithm is as follows:

1. **Data Collection Phase:**

[2] We used 100 keys, and for each of them we generated 2^{24} quartets. We first verified using the official key schedule that indeed right quartets are encountered, and then modified the key schedule accordingly. None of the experiments yielded a right quartet.

(a) Choose a structure of 2^{24} ciphertexts of the form $C_a = (X_a, A)$, where A is fixed and X_a assumes 2^{24} arbitrary different values. Ask for the decryption of all the ciphertexts under the key K_a and denote the plaintext corresponding to C_a by P_a. For each P_a, ask for the encryption of $P_b = P_a \oplus (0_x, 0010\ 0000_x)$ under the key K_b and denote the resulting ciphertext by C_b. Store the pairs (C_a, C_b) in a hash table indexed by the 32-bit value C_b^R (i.e., the right half of C_b).

(b) Choose a structure of 2^{24} ciphertexts of the form $C_c = (Y_c, A \oplus 0010\ 0000_x)$, where A is the same constant as before, and Y_c assumes 2^{24} arbitrary different values. Ask for the decryption of the ciphertexts under the key K_c and denote the plaintext corresponding to C_c by P_c. For each P_c, ask for the encryption of $P_d = P_c \oplus (0_x, 0010\ 0000_x)$ under the key K_d and denote the resulting ciphertext by C_d. Then, access the hash table in the entry corresponding to the value $C_d^R \oplus (0_x, 0010\ 0000_x)$, and for each pair (C_a, C_b) found in this entry, apply Step 2 on the quartet (C_a, C_b, C_c, C_d).

In the first step described above, the $(2^{24})^2 = 2^{48}$ possible quartets are filtered according to a condition on the 32 difference bits which are known (due to the output difference δ of the distinguisher), which leaves about 2^{16} quartets with the required differences.

In Step 2 we can identify the right quartets instantly using an extremely lucky property of the KASUMI structure. We note that a pair (C_a, C_c) can be a right quartet if and only if

$$C_a^L \oplus FL8(FO8(C_a^R)) = C_c^L \oplus FL8(FO8(C_c^R)), \tag{25}$$

since by the Feistel structure, this is the only case of which the difference after round 7 is the output difference of the sandwich distinguisher (i.e., $\delta = (0_x, 0010\ 0000_x)$). However, the values C_a^R and C_c^R are fixed for all the considered ciphertexts, and hence Equation (25) yields

$$C_a^L \oplus C_c^L = FL8(FO8(A)) \oplus FL8(FO8(A \oplus (0_x, 0010\ 0000_x))) = const. \tag{26}$$

Thus, the value $C_a^L \oplus C_c^L$ is equal for all the right quartets. This allows us to perform the following simple filtering:

2. **Identifying the Right Quartets:**

(a) Insert the approximately 2^{16} remaining quartets (C_a, C_b, C_c, C_d) into a hash table indexed by the 32-bit value $C_a^L \oplus C_c^L$, and apply Step 3 only to bins which contain at least three quartets.

Since the probability of a 3-collision in a list of 2^{16} random 32-bit values is lower than $\binom{2^{16}}{3} \cdot 2^{-64} \leq 2^{-18}$, with very high probability only the right quartets remain after this filtering.

In the following step, we treat all the remaining quartets as right quartets. Under this assumption, we know not only the actual inputs to round 8, but also the differences in the outputs of round 8.

OR — $KL_{8,2}$

	(X'_{bd}, Y'_{bd})			
(X'_{ac}, Y'_{ac})	(0,0)	(0,1)	(1,0)	(1,1)
(0,0)	{0,1}	—	1	0
(0,1)	—	—	—	—
(1,0)	1	—	1	—
(1,1)	0	—	—	0

AND — $KL_{8,1}$

	(X'_{bd}, Y'_{bd})			
(X'_{ac}, Y'_{ac})	(0,0)	(0,1)	(1,0)	(1,1)
(0,0)	{0,1}	—	0	1
(0,1)	—	—	—	—
(1,0)	0	—	0	—
(1,1)	1	—	—	1

* The two bits of the differences are denoted by (input difference, output difference): (X'_1, Y'_1) for one pair and (X'_2, Y'_2) for the other pair.

Table 3. Possible Values of $KL_{8,2}$ and $KL_{8,1}$

3. **Analyzing Right Quartets:**
   (a) For each remaining quartet (C_a, C_b, C_c, C_d), guess the 32-bit value of $KO_{8,1}$ and $KI_{8,1}$. For the two pairs (C_a, C_c) and (C_b, C_d) use the value of the guessed key to compute the input and output differences of the OR operation in the last round of both pairs. For each bit of this 16-bit OR operation of $FL8$, the possible values of the corresponding bit of $KL_{8,2}$ are given in Table 3. On average $(8/16)^{16} = 2^{-16}$ values of $KL_{8,2}$ are suggested by each quartet and guess of $KO_{8,1}$ and $KI_{8,1}$.[3] Since all the right quartets suggest the same key, all the wrong keys are discarded with overwhelming probability, and the attacker obtains the correct value of $(KO_{8,1}, KI_{8,1}, KL_{8,2})$.
   (b) Guess the 32-bit value of $KO_{8,3}$ and $KI_{8,3}$, and use this information to compute the input and output differences of the AND operation in both pairs of each quartet. For each bit of the 16-bit AND operation of $FL8$, the possible values of the corresponding bit of $KL_{8,1}$ are given in Table 3. On average $(8/16)^{16} = 2^{-16}$ values of $KL_{8,1}$ are suggested by each quartet and guess of $KO_{8,3}$, $KI_{8,3}$, and thus the attacker obtains the correct value of $(KO_{8,3}, KI_{8,3}, KL_{8,1})$.
4. **Finding the Right Key:** For each value of the 96 bits of ($KO_{8,1}$, $KI_{8,1}$, $KO_{8,3}$, $KI_{8,3}$, $KL_{8,1}$,$KL_{8,2}$) suggested in Step 3, guess the remaining 32 bits of the key, and perform a trial encryption.

The data complexity of the attack is 2^{25} chosen ciphertexts and 2^{25} adaptively chosen plaintexts encrypted/decrypted under one of four keys. The time complexity is dominated by the trial encryptions performed in step 4 to find the last 32 bits of the key, and thus it is approximately equal to 2^{32} encryptions. The probability of success is approximately 76% (this is the probability of having at least three right pairs in the data pool).

The memory complexity of the attack is also very moderate. We just need to store 2^{26} plaintext/ciphertext pairs, where each pair takes 16 bytes. Hence, the total amount of memory used in the attack is 2^{30} bytes, i.e., 1 GByte of memory.

[3] The simple proof of this claim is given in Section 4.3 of [7].

5.1 Experimental Verification

We performed two types of experiments to verify our attack. In the first experiment, we just generated the required data, and located the right quartets (thus verifying the correctness of our randomness assumptions). The second experiment was the application of the full attack (both with and without the final exhaustive search over the remaining 32 key bits). All our experiments were carried out on an Intel Core Duo 2 machine with a T7200 CPU (2 GHz, 4 MB L2 Cache, 2 GB RAM, Linux-2.6.27 kernel, with gcc 4.3.2 and standard optimization flags (`-O3`, `-fomit-frame-pointers`, `-funroll-loops`), single core, single thread).

The first experiment was conducted 1,000 times. In each test, we generated the data and found candidate quartets according to Steps 1 and 2 of the attack algorithm. Once these were found, we partially decrypted the quartets, and checked how many quartets were right ones. Table 4 details the outcome of these experiments, which follow the expected distribution.

Right Quartets	0/1/2	3	4	5	6	7	8	9	10	11	12
Theory ($Poi(4)$)	238	195	195	156	104	60	30	13	5	2	0.6
Experiment	247	197	180	167	112	52	30	7	4	3	1

Table 4. The Number of Identified Right Quartets in 1,000 tests

The second experiment simulated the full attack. We repeated it 100 times, and counted in each case how many times the final exhaustive search over 2^{32} possible keys would have been evoked. In 78 out of these 100 experiments, the key was found when 3 or more quartets were identified to be right ones (the expected number was 76.1).

About 50% of the tests were able to identify the right key by invoking either 2 or 4 exhaustive searches. As the first part of the attack (which identifies candidate quartets) takes about 8 minutes, and each exhaustive search (using the official KASUMI source code) takes about 26 minutes, we could find the full 128 bit key in about 50% of our tests in less than 112 minutes (using a single core). It is important to note that by increasing the running time, one can increase the success rate of the attack without increasing its data requirements.

5.2 Related-Key Rectangle-Like Sandwich Attack on the Full KASUMI

The related-key sandwich distinguisher of 7-round KASUMI presented in Section 4 can be transformed in a standard way to a related-key rectangle-like sandwich distinguisher in which the probability of a quartet to be a right quartet is $(1/4)^2 \cdot (1/4)^2 \cdot 2^{-64} \cdot 2^{-6} = 2^{-78}$. It is worth noting that in a chosen

plaintext manner, one can ensure that the first round of the differential characteristic for E_0 is followed with probability 1 rather than 1/2, and hence the actual probability is 2^{-76}. This distinguisher can be used to mount a related-key rectangle-like sandwich attack on the full KASUMI. The attack is very similar to the attack presented in detail in [7], and hence we omit the full description here, and just mention the changes.

Instead of starting with 2^{51} plaintexts in each structure, the attacker can take 2^{39} plaintexts. These plaintexts contain 2^{78} possible quartets, and after the first filtering step only 2^{14} quartets remain. Then in Step 2(a) of the attack the attacker gets 2^{30} suggestions for 48 key bits (instead of 2^{54} as in [7]), and thus all the wrong suggestions can be discarded (since the right pairs suggest the same value). As a result, the time complexity of the following steps becomes negligible, and the overall time complexity is dominated by the time required to encrypt the 2^{41} chosen plaintexts. This is worse than the 2^{32} time complexity of our sandwich attack but still practical, and has the important advantage that it only requires chosen plaintexts rather than the chosen ciphertexts/adaptive chosen plaintexts of the sandwich attack.

As in the ordinary sandwich attack, the memory used during the rectangle-like sandwich attack is dominated by the storage of the plaintexts and the ciphertexts. By first encrypting the data under keys K_a and K_b, storing it in a sorted table, and then encrypting the data under K_c and K_d in a pair-by-pair manner, we have to store only 2^{40} plaintext/ciphertext pairs. Hence, the total memory complexity of the attack is about 16 terabytes (2^{44} bytes). Fortunately, this memory is accessed sequentially and can be relatively slow, so we only need a few hard disks to keep this data.

6 Summary

In this paper we develop a new sandwich attack on iterated block ciphers, and use it to reduce the time complexity of the best known attack on the full KASUMI from an impractical 2^{76} to the very practical 2^{32}. However, the new attack uses both related keys and chosen messages, and thus it might not be applicable to the specific way in which KASUMI is used as the A5/3 encryption algorithm in third generation GSM telephony. Our main point was to show that contrary to the assurances of its designers, the transition from MISTY to KASUMI led to a much weaker cryptosystem, which should be avoided in any application in which related key attacks can be mounted.

References

1. A5/1 Security Project, *Creating A5/1 Rainbow Tables*, 2009. Available online at `http://reflextor.com/trac/a51`.
2. Elad Barkan and Eli Biham, *Conditional Estimators: an Effective Attack on A5/1*, proceedings of SAC 2005, Lecture Notes in Computer Science 3897, pp. 1–19, Springer, 2006.

3. Eli Biham, *New Types of Cryptanalytic Attacks Using Related Keys*, Journal of Cryptology, Vol. 7, No. 4, pp. 229–246, Springer, 1994.
4. Eli Biham, Orr Dunkelman, and Nathan Keller, *The Rectangle Attack — Rectangling the Serpent*, Advances in Cryptology, proceedings of EUROCRYPT 2001, Lecture Notes in Computer Science 2045, pp. 340–357, Springer, 2001.
5. Eli Biham, Orr Dunkelman, and Nathan Keller, *New Results on Boomerang and Rectangle Attacks*, proceedings of Fast Software Encryption 2002, Lecture Notes in Computer Science 2365, pp. 1–16, Springer 2002.
6. Eli Biham, Orr Dunkelman, and Nathan Keller, *Related-Key Boomerang and Rectangle Attacks*, Advances in Cryptology, proceedings of EUROCRYPT 2005, Lecture Notes in Computer Science 3494, pp. 507–525, Springer, 2005.
7. Eli Biham, Orr Dunkelman, and Nathan Keller, *A Related-Key Rectangle Attack on the Full KASUMI*, Advances in Cryptology, proceedings of ASIACRYPT 2005, Lecture Notes in Computer Science 3788, pp. 443–461, Springer, 2005.
8. Alex Biryukov, *The Boomerang Attack on 5 and 6-Round Reduced AES*, proceedings of AES 2004, Lecture Notes in Computer Science 3373, pp. 11–15, Springer, 2005.
9. Alex Biryukov, Christophe De Cannière, and Gustaf Dellkrantz, *Cryptanalysis of SAFER++*, Advances in Cryptology, proceedings of CRYPTO 2003, Lecture Notes in Computer Science 2729, pp. 195–211, Springer, 2003.
10. Alex Biryukov and Dmitry Khovratovich, *Related-key Cryptanalysis of the Full AES-192 and AES-256*, Advances in Cryptology, proceedings of ASIACRYPT 2009, Lecture Notes in Computer Science 5912, pp. 1–18, Springer, 2009.
11. Alex Biryukov, Adi Shamir, and David Wagner, *Real Time Cryptanalysis of A5/1 on a PC*, proceedings of Fast Software Encryption 2000, Lecture Notes in Computer Science 1978, pp. 1–18, Springer, 2001.
12. Mark Blunden and Adrian Escott, *Related Key Attacks on Reduced Round KASUMI*, proceedings of Fast Software Encryption 2001, Lecture Notes in Computer Science 2355, pp. 277–285, Springer, 2002.
13. Marc Briceno, Ian Goldverg, and David Wagner, *A Pedagogical Implementation of the GSM A5/1 and A5/2 "voice privacy" encryption algorithms*, 1999. Available online at `http://cryptome.org/gsm-a512.htm`.
14. Orr Dunkelman and Nathan Keller, *An Improved Impossible Differential Attack on MISTY1*, Advances in Cryptology, proceedings of ASIACRYPT 2008, Lecture Notes in Computer Science 5350, pp. 441–454, Springer, 2008.
15. Patrik Ekdahl and Thomas Johansson, *Another Attack on A5/1*, IEEE Transactions on Information Theory Vol. 49, No. 1, pp. 284–289, IEEE, 2003.
16. Seokhie Hong, Jongsung Kim, Guil Kim, Sangjin Lee, and Bart Preneel, *Related-Key Rectangle Attacks on Reduced Versions of SHACAL-1 and AES-192*, proceedings of Fast Software Encryption 1999, Lecture Notes in Computer Science 3557, pp. 368–383, Springer, 2005.
17. John Kelsey, Bruce Schneier, and David Wagner, *Key Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES*, Advances in Cryptology, proceedings of CRYPTO 1996, Lecture Notes in Computer Science 1109, pp. 237–251, Springer, 1996.
18. Jongsung Kim, Guil Kim, Seokhie Hong, and Dowon Hong, *The Related-Key Rectangle Attack — Application to SHACAL-1*, proceedings of Australasian Conference on Information Security and Privacy 2004, Lecture Notes in Computer Science 3108, pp. 123–136, Springer, 2004.
19. Jongsung Kim, Seokhie Hong, Bart Preneel, Eli Biham, Orr Dunkelman, and Nathan Keller, *Related-Key Boomerang and Rectangle Attacks*, submitted, 2009.

20. Mitsuru Matsui, *Block encryption algorithm MISTY*, proceedings of Fast Software Encryption 1997, Lecture Notes in Computer Science 1267, pp. 64–74, Springer, 1997.
21. 3rd Generation Partnership Project, Technical Specification Group Services and System Aspects, 3G Security, *Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification*, V3.1.1, 2001.
22. 3rd Generation Partnership Project, Technical Specification Group Services and System Aspects, 3G Security, *Specification of the A5/3 Encryption Algorithms for GSM and ECSD, and the GEA3 Encryption Algorithm for GPRS; Document 4: Design and evaluation report*, V6.1.0, 2002.
23. TECHNEWSWORLD, *Hackers Jimmy GSM Cellphone Encryption*, published 29/12/2009. Available online at `http://www.technewsworld.com/rsstory/68997.html`.
24. David Wagner, *The Boomerang Attack*, proceedings of Fast Software Encryption 1999, Lecture Notes in Computer Science 1636, pp. 156–170, Springer, 1999.