A Practical Related-Key Boomerang Attack for the Full MMB Block Cipher

Tomer Ashur[1] and Orr Dunkelman[2]

[1] Department of Electrical Engineering and iMinds
ESAT/COSIC, KU Leuven, Belgium
tashur@esat.kuleuven.be
[2] Computer Science Department
University of Haifa
Haifa 31905, Israel
orrd@cs.haifa.ac.il

Abstract. The MMB block cipher (Modular Multiplication-based Block cipher) is an iterative block cipher designed by Daemen, Govaerts, and Vandewalle in 1993 as an improvement of the PES and IPES ciphers.
In this paper we present several new related-key differential characteristics of MMB. These characteristics can be used to form several related-key boomerangs to attack the full MMB. Using 2^{20} adaptive chosen plaintexts and ciphertexts we recover all key bits in 2^{35} time for the full MMB. Our attack was experimentally verified, and it takes less than 15 minutes on a standard Intel i5 machine to recover the full MMB key.
After showing this practical attack on the full key of the full MMB, we present partial attacks on extended versions of MMB with up to 9 rounds (which is three more rounds than in the full MMB). We recover 62 out of the 128-bit key in time of $2^{29.2}$ for 7-round MMB, using 2^{20} adaptive chosen plaintexts and ciphertexts encrypted under 4 related-keys, and time of 2^{29} for 8-round MMB using 2^{20} adaptive chosen plaintexts and ciphertexts, encrypted under 6 related-keys. We show how an adversary can recover 31 out of the 128-bit key for the 9-round MMB in time of $2^{27.8}$ using 2^{19} adaptive chosen plaintexts and ciphertexts, encrypted under only 2 related-keys. We also show how the time complexity of all attacks can be reduced by partially precomputing the difference distribution table of MMB's components.

Key words: MMB, Differential Cryptanalysis, Related-Key Boomerang Attack.

1 Introduction

The MMB block cipher (Modular Multiplication-based Block cipher) is an iterative block cipher designed by Daemen, Govaerts, and Vandewalle [5] as an improvement of the PES and IPES ciphers [10,11]. The cipher works with blocks of 128 bits and an equal key length. The cipher's non-linearity comes from multiplication mod $2^{32}-1$ (hence the cipher's name). The cipher consists of 6 rounds without any initialization or finalization steps.

Previously published work on MMB includes two papers in the single-key model [7, 13]. Both papers were able to recover the full key of the full MMB. In [13] Wang et al. use a 5-round differential in a 1R attack in $2^{95.91}$ time, 2^{118} chosen plaintexts, and 2^{65} 32-bit memory words to break the full MMB. In [7] Jia et al. present several attacks, the best of which is a sandwich attack using $2^{13.4}$ time, 2^{40} adaptive chosen plaintexts and ciphertexts, and $2^{20.6}$ 32-bit memory words. We summarize these results in Table 2.

In this paper we present a related-key attack that allows an adversary to recover all key bits in time of 2^{35} using 2^{20} adaptive chosen plaintexts and ciphertexts encrypted under 4 related-keys. We first present two related-key differential characteristics of two and three rounds, respectively, and use them to construct two boomerangs covering 5 rounds of MMB. We then use these 5-round boomerangs to attack the full (6 rounds) MMB. Each of the boomerangs can be used to recover 31 bits of the key. The 62 recovered bits are then further used to recover another 31 bits of the key using a 1R related-key differential attack. The remaining 32 bits are then found by a simple exhaustive search.

To verify our results experimentally, we implemented the attack on the full (6-round) MMB using a C program. The program generates the required data, encrypts and decrypts it through the presented related-key boomerangs, identifies the right quartets, and recovers the key bits in about 15 minutes on a home PC.

After presenting our results, we show that even if MMB was extended to 7 or 8 rounds, it would still be insecure. To prove this claim, we extend the first phase of our attack to extended 7-round and 8-round variants of MMB with similar complexity. In other words, we show that using 2^{20} adaptive chosen plaintexts and ciphertexts, encrypted under 4 related keys for the 7-round variant, and 6 related keys for the 8-round variant, in time of about 2^{29} encryptions, an adversary can recover 62 bits out of the 128-bit key.

This paper is organized as follows: In Section 2 we give a brief description of the MMB block cipher; Section 3 describes some of the previous work done to analyze MMB; in Section 4 we describe the cryptanalytic techniques we use in the paper. In Section 5 we describe the related-key differential characteristics we use and how we use them to create the related-key boomerangs; Section 6 explains how to use the related-key boomerangs to recover the entire key of the full MMB; Section 7 discusses an extended variants of MMB with 7 and 8 rounds and how to attack them, and Section 8 concludes the paper.

2 A Brief Description of MMB and Our Notations

As mentioned before, MMB is an iterative block cipher with a 128-bit block and a 128-bit key. The message and key are each divided into four 32-bit words x_0, x_1, x_2, x_3, and k_0, k_1, k_2, k_3, respectively. In each round, four operations, $\sigma[k^j], \gamma, \eta$, and θ are performed over the state words. Three of the four operations, namely, $\sigma[k^j], \eta$, and θ are involutions (i.e., they are their own inverse).

The key injection operation, $\sigma[k^j]$, XORs the subkey into the message such that $\sigma[k^j](x_0, x_1, x_2, x_3) = (x_0 \oplus k_0^j, x_1 \oplus k_1^j, x_2 \oplus k_2^j, x_3 \oplus k_3^j)$ where $\oplus$ denotes the exclusive-or operation and j denotes the round number. The key injection operation is done 7 times, once at the beginning of the each round and once more after the last round.

The modular multiplication operation, γ, is the only non-linear operation in the cipher. In each encryption round, each of the 32-bit words is multiplied by a fixed constant such that the result y_i is

$$y_i = \begin{cases} x_i & \text{if } x_i = 2^{32} - 1 \\ x_i \otimes G_i & \text{if } x_i \neq 2^{32} - 1 \end{cases}$$

Where the operator $\otimes$ is the modular multiplication operator (i.e., $a \otimes b = (a * b)\ mod\ (2^{32} - 1)$) and $G_0 = 025F1CDB_x, G_1 = 2 \otimes G_0 = 04BE39B6_x, G_2 = 8 \otimes G_0 = 12F8E6D8_x$, and $G_3 = 128 \otimes G_0 = 2F8E6D81_x$. The result of the γ operation is therefore $(y_0, y_1, y_2, y_3) = \gamma(x_0, x_1, x_2, x_3)$.

Inverting γ is done by multiplying the ciphertext with G_i^{-1} such that

$$x_i = \begin{cases} y_i & \text{if } y_i = 2^{32} - 1 \\ y_i \otimes G_i^{-1} & \text{if } y_i \neq 2^{32} - 1 \end{cases}$$

where $G_0^{-1} = 0DAD4694_x, G_1^{-1} = 06D6A34A_x, G_2^{-1} = 81B5A8D2_x$ and $G_3^{-1} = 281B5A8D_x$.

For every word entering γ, the trivial differential transition $0 \rightarrow 0$ holds with probability 1. Another interesting property that was mentioned in [5] is that the differential transition $FFFFFFFF_x \rightarrow FFFFFFFF_x$ through γ also holds with probability 1. The use of these transitions is described in Section 5.

The η operation is a data-dependent operation on the leftmost and rightmost words of the state. If the LSB of the word is 1 it XORs a predefined constant δ into the word, otherwise it does nothing. Namely, $\eta(x_0, x_1, x_2, x_3) = (x_0 \oplus (lsb(x_0) \cdot \delta), x_1, x_2, x_3 \oplus (lsb(x_3) \cdot \delta))$ where $\delta = 2AAAAAAA_x$.

The diffusion between words comes from the θ operation that mixes the round's words such that every change in any word affects three words in the output. Namely, $\theta(x_0, x_1, x_2, x_3) = (x_0 \oplus x_1 \oplus x_3, x_0 \oplus x_1 \oplus x_2, x_1 \oplus x_2 \oplus x_3, x_0 \oplus x_2 \oplus x_3)$.

The j^{th} round of MMB over the block $X = (x_0, x_1, x_2, x_3)$ is: $\rho[k^j](X) = \theta(\eta(\gamma(\sigma[k^j](X))))$. A full description of MMB with plaintext P is:

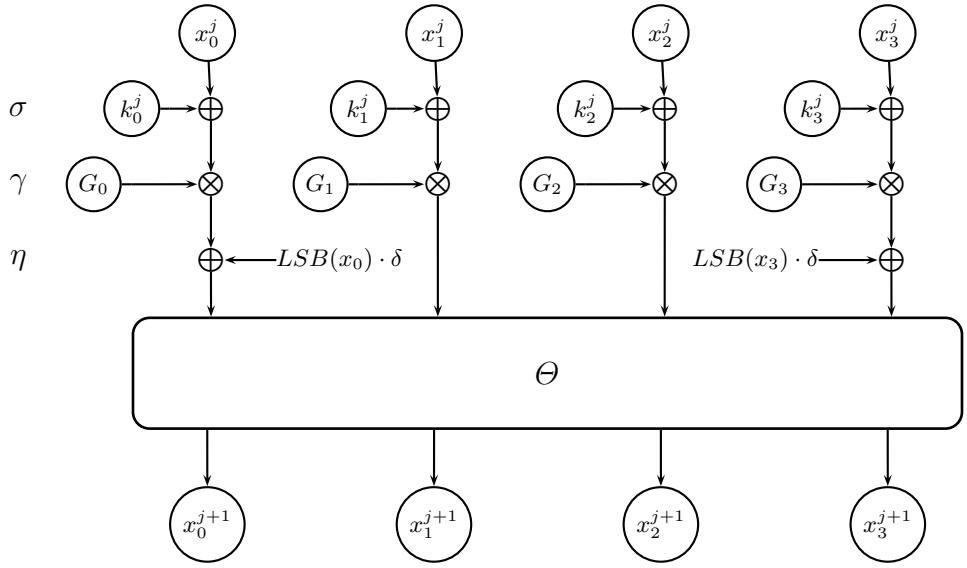
$$\sigma[k^6](\rho[k^5](\rho[k^4](\rho[k^3](\rho[k^2](\rho[k^1](\rho[k^0](P))))))$$

A schematic view of MMB's round function can be found on Figure 1.

2.1 Key Schedule

The original version of MMB used a simple key schedule algorithm that rotates the key words one position to the left (e.g. the key for round 0 is (k_0, k_1, k_2, k_3), the key for round 1 is (k_1, k_2, k_3, k_0) etc.). The key schedule is cyclic and repeats

Fig. 1. MMB's Round Function in Round j.



every 4 rounds [5]. To avoid exploitable symmetry properties a new version of MMB was published where in each round, in addition to the position change, each key word is XORed with a round-dependent constant. Therefore, the key word i for round j is $k_i^j = k_{i+j \ mod \ 4} \oplus (2^j \cdot B)$ with $B = DAE_x$ [4].[1]

2.2 Notations

The notations used throughout the paper are described in Table 1 for the readers' convenience.

3 Previous Attacks on MMB

Wang et al. identified for MMB a 2-round differential characteristic with probability 1 [13]. This 2-round differential characteristic, described in Equation (1) was extended into a 5-round differential characteristic with probability of 2^{-110}. This 5-round differential characteristic can be used in an attack that recovers

[1] We note that the change in the key schedule algorithm does not affect our attack which is differential in nature. In other words, all the attacks reported in this paper work for both key schedules, i.e., the original one and the tweaked one.

Symbol	Meaning/Value
$\oplus$	Exclusive-or
$\otimes$	Multiplication modulo $2^{32}-1$
$X \rightarrow Y$	Differential transition from X to Y
0	00000000_x
$\bar{0}$	$FFFFFFFF_x$
δ	$2AAAAAAA_x$
$\bar{\delta}$	$\delta \oplus FFFFFFFF_x$
G_0	$025F1CDB_x$
G_1	$04BE39B6_x$
G_2	$12F8E6D8_x$
G_3	$2F8E6D81_x$

Table 1. Notations Used Throughout this Paper

all of MMB's key bits with data complexity of 2^{118} chosen plaintexts, time complexity of $2^{95.91}$ encryptions, and memory requirements of 2^{65} 32-bit blocks. We note that the time complexity described in [13] does not take into account the fact that the time required to encrypt 2^{118} plaintexts cannot be less than 2^{118}.

$$(0,\bar{0},\bar{0},0) \xrightarrow{\sigma[\mathrm{k}^0]} (0,\bar{0},\bar{0},0) \xrightarrow{\gamma} (0,\bar{0},\bar{0},0) \xrightarrow{\eta} (0,\bar{0},\bar{0},0) \xrightarrow{\theta} (\bar{0},0,0,\bar{0}) \qquad (1)$$
$$\xrightarrow{\sigma[\mathrm{k}^1]} (\bar{0},0,0,\bar{0}) \xrightarrow{\gamma} (\bar{0},0,0,\bar{0}) \xrightarrow{\eta} (\bar{\delta},0,0,\bar{\delta}) \xrightarrow{\theta} (0,\bar{\delta},\bar{\delta},0)$$

Jia et al. [7] improved Wang's analysis to build a 5-round sandwich distinguisher (an extension of the boomerang distinguisher) with probability 1. This attack exploits the 2-round differential characteristic identified in [13] to construct a 5-round sandwich that is then used to recover the full key of the full MMB with 2^{40} adaptive plaintexts and ciphertexts, $2^{13.4}$ time, and 2^{16} memory bytes. They also showed how to transform their attack into a rectangle-like sandwich that can recover the full key of MMB in 2^{64} time, $2^{66.5}$ memory, and $2^{70.5}$ chosen plaintexts.

Table 2 summarizes all previous results on MMB and compares them with ours. Table 3 compares the amount of work it takes to recover the first 31 key bits for all the attacks presented in this paper.

4 Cryptanalytic Techniques for Block Ciphers Used in This Paper

4.1 Differential Cryptanalysis

One of the most notable techniques in cryptanalysis is differential cryptanalysis. Developed by Biham and Shamir [3], differential cryptanalysis examines the evolution of differences between two inputs. An input difference is the difference between two inputs entering a cryptosystem, usually with respect to

Rounds	Attack	Time*	Data	Memory (bytes)	Number of Keys	Number of Recovered Bits	Source
6	Differential Cryptanalysis	$2^{95.9}$	2^{118} CP**	2^{66}	1	128	[13]
6	Rectangle-like sandwich	2^{64}	$2^{66.5}$ CP	$2^{72.5}$	1	128	[7]
6	Sandwich attack	$2^{13.4}$	2^{40} ACPC***	2^{18}	1	128	[7]
6	Related-key boomerang	$2^{29.4}$	2^{20} ACPC	$2^{21.3}$	4	62	Section 6
7	Related-key boomerang	$2^{29.2}$	2^{20} ACPC	$2^{21.3}$	4	62	Section 7
8	Related-key boomerang	2^{29}	2^{20} ACPC	$2^{21.3}$	6	62	Section 7
6	Related-key boomerang	2^{35}	2^{20} ACPC	$2^{21.3}$	4	128	Section 6

* The reported time is the analysis time (not including the time needed for data generation).

** Chosen plaintexts.

*** Adaptive chosen plaintexts and ciphertexts.

Table 2. Summary of the Attacks on MMB

Rounds	Time	Data	Memory	Keys
6	$2^{28.4}$	2^{19}	$2^{21.3}$	4
7	$2^{28.2}$	2^{19}	$2^{21.3}$	4
8	2^{28}	2^{19}	$2^{21.3}$	4
9	$2^{27.8}$	2^{19}	$2^{21.3}$	2

Table 3. Comparison of our Attacks for Recovering the First 31 Bits.

the exclusive-or operation. The output difference is the difference between the outputs of two such inputs. We say that an input difference Δ can cause an output difference Δ^* under the function f with probability p if a portion p of the possible pairs of messages having a difference Δ result in outputs having a difference Δ^* after applying f. If these conditions hold we write that $\Delta \xrightarrow{f} \Delta^*$ with probability p.

A differential characteristic that describes a single encryption round is called a 1-round differential characteristic. Biham and Shamir showed that two or more differential characteristics can be concatenated to form a longer differential characteristics if the output difference of one differential characteristic is the input difference of the other differential characteristic.

Once a good long differential characteristic is identified, the adversary tries to find a pair of messages that satisfies it. By examining many plaintext pairs, the adversary tries to distinguish the wrong pairs (i.e., those pairs which do not satisfy the differential characteristic) from the right pairs (i.e., those pairs which satisfy the differential characteristic). The amount of data needed to find a right pair is proportional to the inverse of the probability of the differential characteristic used and can be somewhat reduced by various techniques. Once a right pair is found, it can be used to recover the keys used in the cryptosystem by examining which keys cause the messages to satisfy the required differences.

4.2 Related-Key Differential Attack

Since its publication in 1990, differential cryptanalysis received a great deal of attention in the cryptographic community. Several researchers published extensions for the core technique. One of these extensions is the related-key differential attack published by Kelsey et al. in 1997 [8]. In a related-key differential attack the adversary is allowed, in addition to examining the evolution of differences between inputs, to introduce differences to the key. Namely, in the attack, two plaintexts are encrypted using two keys that have some difference chosen by the adversary. This difference is injected into the intermediate encryption values by the key injection operation and sometimes cancel previous differences. Modulo some small technical issues, the remainder of the attack is the same as in regular differential attacks.

4.3 The Boomerang Attack

Another extension to differential cryptanalysis is the boomerang attack suggested by Wagner in 1999 [12]. A boomerang attack uses two differential characteristics of relatively small number of rounds n and m with probabilities p and q, respectively, to construct a distinguisher for $m + n$ rounds.

A boomerang is composed of two differential characteristics $\triangle \rightarrow \triangle^*$ for n rounds and $\bigtriangledown^* \rightarrow \bigtriangledown$ for m rounds with probabilities p and q, respectively. The adversary chooses two plaintexts P_1 and P_2 such that $P_1 \oplus P_2 = \triangle$ and asks for their respective values C_1 and C_2 after $m + n$ encryption rounds. The adversary then XORs these ciphertexts with $\bigtriangledown$ to obtain the ciphertexts C_3 and C_4, respectively, and asks for their decrypted values P_3 and P_4. The boomerang suggests that $P_1 \oplus P_2 = P_3 \oplus P_4 = \triangle$ with probability $p^2 \cdot q^2$.

4.4 Related-Key Boomerang Attack

The related-key boomerang attack is an extension of the boomerang attack first suggested in 2004 by Kim et al. [2,6,9]. The idea of a related-key boomerang is to use two related-key differentials to construct the boomerang. After constructing this boomerang, the attack is then carried in the same way as with regular boomerangs (again, modulo a few small differences).

5 A Related-Key Boomerang attack for the Full MMB

Before we describe the related-key differential characteristics used to construct the boomerangs we observe that for any plaintext, and any operation, the trivial differential transition $0 \rightarrow 0$ holds with probability 1. Another interesting property which is described in [5] is that an input difference $FFFFFFFF_x$ between two input words to $\otimes$ cause an output difference of $FFFFFFFF_x$ with probability 1 (independent of G_i).

Another point worth mentioning is that if the difference between the leftmost or the rightmost words entering η is $\bar{0}$, the output difference must be

$\delta \oplus FFFFFFFF_x$. The η operation XORs the constant $\delta = 2AAAAAAA_x$ to the leftmost and rightmost words if their least significant bit is 1. In the event that the difference between two input words is $FFFFFFFF_x$, one of them must have 1 as its least significant bit while the other must have 0, thus, δ is XORed only to one of them, causing the transition.

We present three related-key differentials: The 3-round related key differential $\triangle \rightarrow \triangle^*$ with input difference $(0, 0, \bar{0}, \bar{0})$ and key difference $(0, 0, \bar{0}, \bar{0})$. This differential is an extension of Equation (1) where we use the key difference to control the propagation of the difference. The related-key differential

$$\begin{aligned}\triangle = (0,0,\bar{0},\bar{0}) &\xrightarrow[(0,0,\bar{0},\bar{0})]{\sigma[\mathrm{k}^1]} (0,0,0,0) \xrightarrow{\gamma} (0,0,0,0) \xrightarrow{\eta} (0,0,0,0) \xrightarrow{\theta} (0,0,0,0)\\ &\xrightarrow[(0,\bar{0},\bar{0},0)]{\sigma[\mathrm{k}^2]} (0,\bar{0},\bar{0},0) \xrightarrow{\gamma} (0,\bar{0},\bar{0},0) \xrightarrow{\eta} (0,\bar{0},\bar{0},0) \xrightarrow{\theta} (\bar{0},0,0,\bar{0})\\ &\xrightarrow[(\bar{0},\bar{0},0,0)]{\sigma[\mathrm{k}^3]} (0,\bar{0},0,\bar{0}) \xrightarrow{\gamma} (0,\bar{0},0,\bar{0}) \xrightarrow{\eta} (0,\bar{0},0,\bar{\delta}) \xrightarrow{\theta} (\delta,\bar{0},\delta,\bar{\delta}) = \triangle^*\end{aligned}$$

holds with probability 1. We can extend this related-key differential by prepending an additional round

$$(\bar{X},\bar{0},0,\bar{0}) \xrightarrow[(\bar{0},0,0,\bar{0})]{\sigma[\mathrm{k}^0]} (X,\bar{0},0,0) \xrightarrow{\gamma} (\bar{\delta},\bar{0},0,0) \xrightarrow{\eta} (\bar{0},\bar{0},0,0) \xrightarrow{\theta} (0,0,\bar{0},\bar{0}) = \triangle, \tag{2}$$

where $\bar{X}$ is some undetermined difference satisfying $\bar{X} \xrightarrow{\oplus \mathrm{k}_0^0} X$ and $X \xrightarrow{\otimes \mathrm{G}_0} \bar{\delta}$.

The second related-key differential we use is a 4-round related-key differential $\bigtriangledown^* \rightarrow \bigtriangledown$ with input difference $(0, 0, \bar{0}, 0)$ and key difference $(0, 0, \bar{0}, 0)$

$$\begin{aligned}\bigtriangledown^* = (0,0,\bar{0},0) &\xrightarrow[(0,0,\bar{0},0)]{\sigma[\mathrm{k}^1]} (0,0,0,0) \xrightarrow{\gamma} (0,0,0,0) \xrightarrow{\eta} (0,0,0,0) \xrightarrow{\theta} (0,0,0,0)\\ &\xrightarrow[(0,\bar{0},0,0)]{\sigma[\mathrm{k}^2]} (0,\bar{0},0,0) \xrightarrow{\gamma} (0,\bar{0},0,0) \xrightarrow{\eta} (0,\bar{0},0,0) \xrightarrow{\theta} (\bar{0},\bar{0},\bar{0},0)\\ &\xrightarrow[(\bar{0},0,0,0)]{\sigma[\mathrm{k}^3]} (0,\bar{0},\bar{0},0) \xrightarrow{\gamma} (0,\bar{0},\bar{0},0) \xrightarrow{\eta} (0,\bar{0},\bar{0},0) \xrightarrow{\theta} (\bar{0},0,0,\bar{0})\\ &\xrightarrow[(0,0,0,\bar{0})]{\sigma[\mathrm{k}^4]} (\bar{0},0,0,0) \xrightarrow{\gamma} (\bar{0},0,0,0) \xrightarrow{\eta} (\bar{\delta},0,0,0) \xrightarrow{\theta} (\bar{\delta},\bar{\delta},0,\bar{\delta}) = \bigtriangledown\end{aligned}$$

that also holds with probability 1. This differential can also be extended by prepending an additional round:

$$(0,\bar{0},\bar{0},\bar{Y}) \xrightarrow[(0,0,0,\bar{0})]{\sigma[\mathrm{k}^0]} (0,\bar{0},\bar{0},Y) \xrightarrow{\gamma} (0,\bar{0},\bar{0},\bar{\delta}) \xrightarrow{\eta} (0,\bar{0},\bar{0},\bar{0}) \xrightarrow{\theta} (0,0,\bar{0},0) = \bigtriangledown^*, \tag{3}$$

where like in the case of $\bar{X}$, $\bar{Y}$ is an undetermined difference satisfying $\bar{Y} \xrightarrow{\oplus k_3^0} Y$ and $Y \xrightarrow{\otimes G_3} \bar{\delta}$. We list the most probable values of Y's and X's (with their probability) in Appendix A. In Section 6 we show how to use the birthday paradox to construct pairs which satisfy these differences regardless of the exact probabilities.

The third related-key differential we use is a 2-round related-key differential $\tau \to \tau^*$ with input difference $(0,0,0,\bar{0})$ and key difference $(0,0,0,\bar{0})$ that holds with probability[2] 1:

$$\tau = (0,0,0,\bar{0}) \xrightarrow[(0,0,0,\bar{0})]{\sigma[k^4]} (0,0,0,0) \xrightarrow{\gamma} (0,0,0,0) \xrightarrow{\eta} (0,0,0,0) \xrightarrow{\theta} (0,0,0,0)$$
$$\xrightarrow[(0,0,\bar{0},0)]{\sigma[k^5]} \xrightarrow{\gamma} (0,0,\bar{0},0) \xrightarrow{\eta} (0,0,\bar{0},0) \xrightarrow{\theta} (0,\bar{0},\bar{0},\bar{0}) = \tau^*$$

We construct two boomerangs. The first 5-round related-key boomerang is the concatenation of $\tau \to \tau$ after $\triangle \to \triangle^*$ without the additional round presented in Equation (2). This boomerang has probability 1 and can only be used as a distinguisher. Prepending one more round (as specified in Equation (2)) to $\triangle \to \triangle^*$ forms a 6-round related-key boomerang we denote by B_0. This boomerang is depicted in Figure 2.

The second boomerang, which we denote by B_1 is constructed by concatenating the first round of $\triangle \to \triangle^*$ after $\bigtriangledown^* \to \bigtriangledown$ to form a 5-round boomerang with probability 1. We then prepend one more round (as specified in Equation (3)) to $\bigtriangledown \to \bigtriangledown^*$ to form a 6-round boomerang that can be used in a 1R attack. The second boomerang is depicted in Figure 3.

6 Description of the Key Recovery Attack

In this section we describe our related-key boomerang attack on MMB and the key recovery phase that is used to recover 62 bits out of the 128-bit key. We then show how to efficiently recover the remaining 66 key bits given the knowledge of the previous 62, for the full MMB. We conclude the section with a description of our experimental verification of this attack.

6.1 Related-Key Boomerang Attack

We recall that the 128-bit key is composed of four 32-bit key words (k_0, k_1, k_2, k_3). We recover each of these words separately. The first 31 key bits (those of k_0) are recovered using the boomerang B_0 and the last 31 key bits (those of k_3) are recovered using the boomerang B_1.

In order to use B_0 we need 4 related-keys. Two of them, namely

$$K^1 = (k_0, k_1, k_2, k_3); K^2 = K^1 \oplus (\bar{0}, 0, 0, \bar{0})$$

[2]Note that the key difference for the differential for $\bigtriangledown^* \to \bigtriangledown$ is the same as for the differential $\tau \to \tau^*$

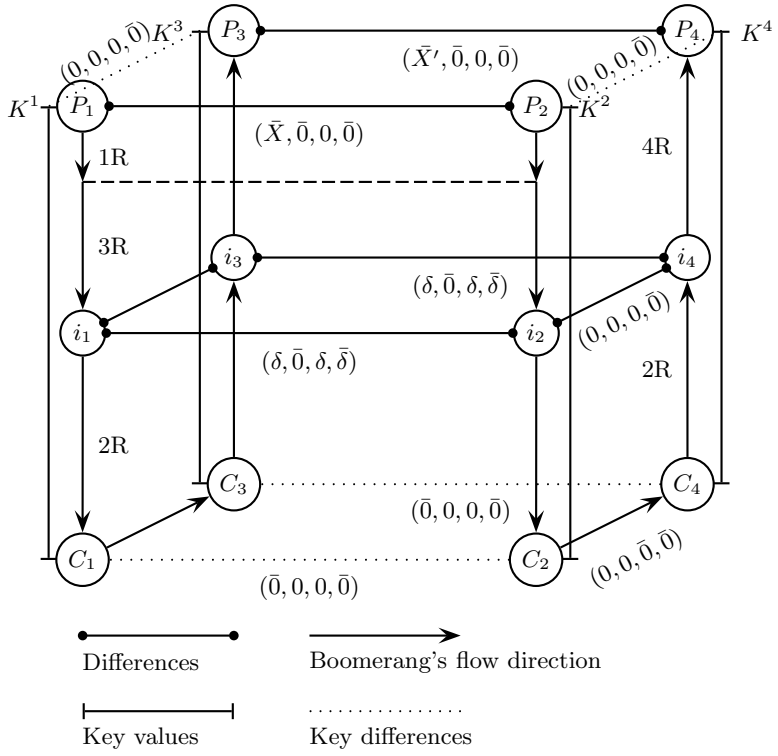


Fig. 2. The Description of B_0.

are used for encryption and the other two, namely

$$K^3 = K^1 \oplus (0,0,0,\bar{0}); K^4 = K^2 \oplus (0,0,0,\bar{0}) = K^1 \oplus (\bar{0},0,0,0),$$

are used for decryption.

We pick a set of 2^{17} random plaintexts $\mathcal{P}^1 = \{P_0^1, ..., P_{2^{17}-1}^1\}$ all having the same value in bits 32–127 and different values in bits 0–31. Then, we generate another set of 2^{17} plaintexts $\mathcal{P}^2 = \{P_0^2, ..., P_{2^{17}-1}^2\}$ where $P_i^2 = P_i^1 \oplus (0,\bar{0},0,\bar{0})$. We then ask for the encryption of all the values in $\mathcal{P}^1$ under K^1 to obtain the set of respective ciphertexts, $\mathcal{C}^1 = \{C_0^1, ..., C_{2^{17}-1}^1\}$, and ask for the encryption of all values in $\mathcal{P}^2$ under K^2 to obtain the respective set of ciphertexts $\mathcal{C}^2 = \{C_0^2, ..., C_{2^{17}-1}^2\}$.

We XOR all values of $\mathcal{C}^1$ and $\mathcal{C}^2$ with $(0,0,\bar{0},\bar{0})$ to obtain $\mathcal{C}^3 = \mathcal{C}^1 \oplus (0,0,\bar{0},\bar{0}) = \{C_0^3, ..., C_{2^{17}-1}^3\}$ and $\mathcal{C}^4 = \mathcal{C}^2 \oplus (0,0,\bar{0},\bar{0}) = \{C_0^4, ..., C_{2^{17}-1}^4\}$. We ask for the decryption of the ciphertexts in $\mathcal{C}^3$ under K^3 to obtain a set of plaintexts

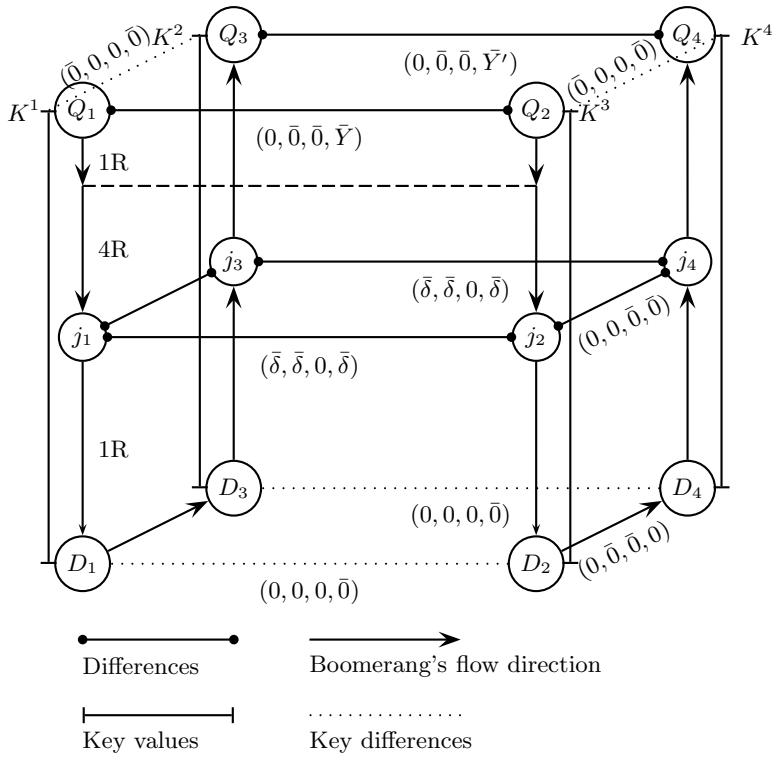


Fig. 3. The Description of B_1

$\mathcal{P}^3 = \{P_0^3, ..., P_{2^{17}-1}^3\}$, and the decryption of the ciphertexts in $\mathcal{C}^4$ under K^4 to obtain a set of plaintexts $\mathcal{P}^4 = \{P_0^4, ..., P_{2^{17}-1}^4\}$.

We expect, due to the birthday paradox, that two plaintexts P_i^1 and P_j^2, taken from $\mathcal{P}^1$ and $\mathcal{P}^2$, respectively, will collide (i.e., have a zero difference) in bits 0–31 after a single round of $\sigma[k^0], \gamma$, and η with a non-negligible probability.[3] Such two colliding plaintexts form a pair with input difference $\triangle$ as required by the differential characteristic $\triangle \rightarrow \triangle^*$ (the difference in the remaining words is set with probability 1). Since both differentials used in the boomerang hold with probability 1, the encryption, XOR by $(0,0,\bar{0},\bar{0})$ and 5-round decryption of it will inevitably result with a difference of Δ causing its respective P_i^3 and P_j^4 to also have a difference of the form $(\bar{X}', \bar{0}, 0, \bar{0})$ after the decryption.

[3] As we discuss later, we actually expect four such pairs. Given that the actual number of such pairs follows a Poisson distribution with a mean value of 4, we expect at least one such pair to exist with probability of 98.2%.

Analyzing the expected number of right pairs is straightforward using the birthday paradox framework. The values of $\mathcal{P}_1$ occupy 2^{17} bins out of the 2^{32} possible bins. Therefore, each of the 2^{17} possible values of $\mathcal{P}_2$ has a chance of $\frac{2^{17}}{2^{32}} = 2^{-15}$ to collide with a value from $\mathcal{P}_1$. Hence, the expected number of right pairs (which lead to right quartets with probability 1) is $2^{17} \cdot \frac{2^{17}}{2^{32}} = 4$. In Section 6.4 we test this prediction empirically.

We store all values of $\mathcal{P}_3$ in a hash table using bits 32–127 as the hash key. Then, once obtaining the values of $\mathcal{P}_4$ we search for "collisions" in these bits (taking into account the expected difference between them) to identify a candidate pair (and thus, a candidate quartet). The probability that among all the possible 2^{34} pairs, two plaintexts form a wrong pair (i.e., agreeing on bits 32–127 without following the boomerang) is $2^{34} \cdot 2^{-96} = 2^{-62}$. Thus, we can safely assume that all candidate quartets are right quartets. Note that we do not need to store the plaintexts with their respective ciphertexts, hence, reducing the memory complexity.

Once we identify the four plaintexts forming a right quartet, $((P_i^1, P_j^2), (P_i^3, P_j^4))$, we try all the 2^{32} possible values for $k_0^1 = k_0^3$ and $k_0^2 = k_0^4 = k_0^1 \oplus \bar{0}$ (the first 32 bits of K^1, K^2, K^3, and K^4) to see which of them causes both pairs to have a zero difference in the first word after one round. These 2^{32} trials suggest two possible values as the key word, either k_0^1 or $\bar{k_0^1}$ (corresponding to 31-bit of key material). Note that usually in related-key attacks we expect **one** solution for these cases. However, in the specific case of $\otimes$, complementing the entire input necessarily complements the entire output. Hence, if the two inputs to $\otimes$ are x and x', and a 32-bit key word k satisfies $((x \oplus k) \otimes G_0) \oplus (x' \oplus \bar{k}) \otimes G_0 = \bar{0}$, then $\bar{k}$ also satisfies this relation, as both results are complemented when the value of k_0 is complemented. For this reason, it is enough to try only 2^{31} values for each key word, and for every value that satisfies the requirements, also add its complement value to the set of possible key words. At the last part of the attack, we encrypt a plaintext using all key combinations to determine which value is the right key and which value is its complementary.

To recover bits 96–127 of the key we use the same method. We pick 2^{17} random plaintexts $\mathcal{Q}^1 = \{Q_0^1, ..., Q_{2^{17}-1}^1\}$ all having the same value in bits 0–95 and different values in bits 96–127. Then, we generate another 2^{17} plaintexts $\mathcal{Q}^2 = \{Q_0^2, ..., Q_{2^{17}-1}^2\}$, where $Q_i^2 = Q_i^1 \oplus (0, \bar{0}, \bar{0}, 0)$ and use the same algorithm to encrypt the plaintexts under K^1 and K^3, XOR the ciphertexts with $\triangle^*$ and decrypt them under K^2 and K^4, respectively. The key word k_3 is then recovered by 2^{31} trials in a similar way to the one described for recovering k_0.

6.2 Recovering the Remaining Key Bits

Recovering Bits 32–63 of the Key Once we obtained key bits 0–31 and 96–127, we use an extension of the related-key differential $\bigtriangledown^* \rightarrow \bigtriangledown$ to recover key bits 32–63 with a simple 1R attack. The 4-round related-key differential characteristic

$$\begin{aligned}
\bigtriangledown^* = (0,0,\bar{0},0) &\xrightarrow[(0,0,\bar{0},0)]{\sigma[\mathrm{k}^1]} (0,0,0,0) \xrightarrow{\gamma} (0,0,0,0) \xrightarrow{\eta} (0,0,0,0) \xrightarrow{\theta} (0,0,0,0) \\
&\xrightarrow[(0,\bar{0},0,0)]{\sigma[\mathrm{k}^2]} (0,\bar{0},0,0) \xrightarrow{\gamma} (0,\bar{0},0,0) \xrightarrow{\eta} (0,\bar{0},0,0) \xrightarrow{\theta} (\bar{0},\bar{0},\bar{0},0) \\
&\xrightarrow[(\bar{0},0,0,0)]{\sigma[\mathrm{k}^3]} (0,\bar{0},\bar{0},0) \xrightarrow{\gamma} (0,\bar{0},\bar{0},0) \xrightarrow{\eta} (0,\bar{0},\bar{0},0) \xrightarrow{\theta} (\bar{0},0,0,\bar{0}) \\
&\xrightarrow[(0,0,0,\bar{0})]{\sigma[\mathrm{k}^4]} (\bar{0},0,0,0) \xrightarrow{\gamma} (\bar{0},0,0,0) \xrightarrow{\eta} (\bar{\delta},0,0,0) \xrightarrow{\theta} (\bar{\delta},\bar{\delta},0,\bar{\delta}) = \bigtriangledown
\end{aligned}$$

holds with probability 1. With the extension in Equation (3) it forms a 5-round related-key differential characteristic for MMB. i.e., once we know a quartet of values which satisfy Equation (3) (for example, as part of a right quartet in B_1) we can use it as two right pairs with respect to this 5-round related-key differential characteristic.

Let $Q_i^1 \in \mathcal{Q}^1$ and $Q_j^2 \in \mathcal{Q}^2$ be two plaintexts forming a right pair, and let $D_i^1 = (w_6 \oplus k_0^6, x_6 \oplus k_1^6, y_6 \oplus k_2^6, z_6 \oplus k_3^6)$ and $D_j^2 = (w_6^* \oplus k_0^6, x_6^* \oplus k_1^6, y_6^* \oplus \bar{k_2^6}, z_6^* \oplus k_3^6)$ be their respective ciphertexts. We observe that each of the words $w, x, y, z, w^*, x^*, y^*$, and z^* is the result of the θ operation which XORs three intermediate values. We denote these intermediate values as $a, b, c, d, a^*, b^*, c^*$, and d^*, i.e.,

$$w_6 = a_6 \oplus b_6 \oplus d_6; w_6^* = a_6^* \oplus b_6^* \oplus d_6^*$$

$$x_6 = a_6 \oplus b_6 \oplus c_6; x_6^* = a_6^* \oplus b_6^* \oplus c_6^*$$

$$y_6 = b_6 \oplus c_6 \oplus d_6; y_6^* = b_6^* \oplus c_6^* \oplus d_6^*$$

$$z_6 = a_6 \oplus c_6 \oplus d_6; z_6^* = a_6^* \oplus c_6^* \oplus d_6^*.$$

To recover k_2 we simply XOR the first three words of each ciphertext

$$w_6 \oplus k_0^6 \oplus x_6 \oplus k_1^6 \oplus y_6 \oplus k_2^6 = a_6 \oplus b_6 \oplus d_6 \oplus k_0^6 \oplus a_6 \oplus b_6 \oplus c_6 \oplus k_1^6 \oplus b_6 \oplus c_6 \oplus d_6 \oplus k_2^6 = b_6 \oplus k_0^6 \oplus k_1^6 \oplus k_2^6$$

and

$$w_6^* \oplus k_0^6 \oplus x_6^* \oplus k_1^6 \oplus y_6^* \oplus \bar{k_2^6} = a_6^* \oplus b_6^* \oplus d_6^* \oplus k_0^6 \oplus a_6^* \oplus b_6^* \oplus c_6^* \oplus k_1^6 \oplus b_6^* \oplus c_6^* \oplus d_6^* \oplus \bar{k_2^6} = b_6^* \oplus k_0^6 \oplus k_1^6 \oplus \bar{k_2^6}$$

where the values of k_0^6 and k_1^6 are the 64 key bits previously recovered.[4] The adversary then searches for the values of k_2^6 and $\bar{k_2^6}$ that satisfy the equation $((b^* \oplus \bar{k_2^6} \oplus k_0^6 \oplus k_1^6) \otimes G_2^{-1}) \oplus ((b \oplus k_2^6 \oplus k_0^6 \oplus k_1^6) \otimes G_2^{-1}) = \bar{\delta}$. Taking the second pair of a right boomerang quartet allows discarding a few more of the remaining wrong options.

[4]Recall that only 62 bits of key material were recovered. However, for the execution of this phase, any of the 4 possibilities for the suggested key suffices.

Recovering Bits 64–95 of the Key After recovering k_0, k_2, and k_3, the remaining k_1 (32 bits) is recovered by exhaustive search (i.e., brute force) which is repeated 8 times to recover the missing key bit for each of k_0, k_2, and k_3.

Analysis of the Full Attack The first part of the attack requires encryptions/decryptions of 2^{19} plaintexts for each boomerang, then a single round encryption/decryption for each of the 2^{31} possible keys, per message. Therefore, the time complexity it takes to recover 62 key bits in the first phase is: $2 \cdot (4 \cdot 2^{17} + \frac{1}{6} \cdot 2^{31}) = 2^{29.4}$ (two boomerangs, $4 \cdot 2^{17}$ messages are encrypted and decrypted for each of them, then, each message is passed through $\frac{1}{6}$ of the full encryption for each of the 2^{31} possible key words). The memory complexity is $2^{21.3}$ bytes,[5] and the data complexity is 2^{20} adaptive chosen plaintexts and ciphertexts, using 4 related keys. The second part of the attack requires running 2^{31} round operations (which are about $\frac{1}{6} \cdot 2^{31}$=$2^{29.4}$ full MMB encryptions) with no additional memory and data requirements to recover another 31 bits of key material. The third part of the attack requires running $8 \cdot 2^{32}$ =2^{35} full MMB encryptions, again, with no additional memory and data requirements. Thus, the overall complexity of this attack is 2^{35} time, $2^{21.3}$ memory, and 2^{20} adaptive chosen plaintexts and ciphertexts encrypted under 4 related-keys.

6.3 Experimental Verification

The low time, data, and memory complexities of the attack allow verifying it experimentally. The implementation of the attack uses two programming languages: C and Python. The C program was used to implement the cryptographic parts of the attack (i.e. the boomerangs and the key search). Python was used to invoke different modules of the attack and collect data for statistical analysis.

The C program was compiled and ran on a Debian Linux machine using GCC 4.4.5 with the -O3 optimization flag. The program starts by generating a random 128-bit plaintext and a random 128-bit key. It then forks into two processes, one implementing B_0 and the other implementing B_1. The first process generates a second plaintext and a second key with the appropriate differences and replaces the first word of both plaintexts with a random one. It then saves the two plaintexts and encrypts them under the related-keys to obtain their respective ciphertexts. The ciphertexts and the keys are then XORed with the appropriate values and decrypted to obtain new plaintexts. For each such new plaintext, the program stores it for later use. Once all plaintexts are decrypted, the program

[5] We alert the reader that in each boomerang we need to store 2^{17} 128-bit plaintexts from P^3, and 2^{17} 32-bit representations of the plaintexts from P^1. The ciphertexts themselves are not used in the key recovery part, and thus are not stored. The plaintexts of P_2 and P_4 are only used online to search the hash table for collisions, and thus do not need to be stored.

searches for right quartets. This is done by searching for pairs in which bits 32–127 of the decrypted plaintexts have difference of $(\bar{0}, 0, \bar{0})$.[6]

Once a right quartet is found, the key recovery is done by trying all possible values as the key for the first plaintext word in both pairs and checking which value leads to a zero difference after a single round of $\sigma[k^0], \gamma$, and η. All such values are written into the output file as possible keys. This process is repeated for all quartets satisfying the conditions (i.e., the candidate quartets). The second process does the same with the minor change that it searches for decrypted pairs in which bits 0–95 has difference of $(0, \bar{0}, \bar{0})$ and searches for the forth key word instead of the first.

The Python program was written in Python 2.6.6 over GCC 4.4.5. Once the C program finishes its execution the Python program reads the two output files and invokes another C program that uses the results of the previous phase to recover key bits 32–63 by iterating over all possible key values which satisfy the conditions in Subsection 6.2. The python program then runs another C program that exhaustively searches for the last key word. The program tries in parallel all 8 possible key words combinations with all 2^{32} possible values for the remaining key word. Once the full key is identified in one of the subprograms, the program outputs it and terminates.

6.4 Results of the Experimental Verification

Our experiment included running the program 100 times. Out of these 100 trials, recovering k_0 was successful 98 times (98%), Recovering k_3 was successful 98 times (98%). In 98 of the trials (98%), both k_0 and k_3 were recovered successfully. The key word k_2 was recovered successfully 98 times (98%), i.e., whenever k_0 and k_3 were both recovered, so was k_2. We consider the experiment to be successful in recovering a key word when the Python program returns exactly 2 possible values for that word: the correct one and its complement.

We also tested the actual amount of quartets. Out of the 100 trials, the program found on average 4.06 candidate quartets for B_0 and 4.01 candidate quartets for B_1. This result is perfectly aligned with the calculation we presented in Section 6.1.

The average running time of the program on an i5 personal computer with 4 GBs RAM, running Debian Linux is 196.56 seconds for the first phase and 106.38 seconds for the second phase with standard deviations of 61.47 seconds and 52.19 seconds, respectively. Executing 2^{32} encryptions of the full MMB requires 341.57 seconds. When parallelized over an i5 CPU with 4 cores and terminated on key detection, the average running time of this stage is 504.40 seconds with a standard deviation of 329.01 seconds. Hence, the average total time required for the recovery of the full key is 13.5 minutes with a standard deviation of 4.19 minutes.

[6] Although an implementation using a hash-table is faster in theory, we found out that in practice, the required bookkeeping induces higher overhead than a simple list of values.

Our implementation of the attack presented in Section 5 is available upon request from the authors (via the program chairs, to maintain anonymity). [1]

7 Attacking More Rounds of MMB

In this section we expand our attack to show that even if MMB was extended to 7 or 8 rounds our attack could still be used to recover 62 bits of the key, namely, k_0 and k_3. We first show how to extend the existing boomerangs to cover 6 rounds of MMB, and recover 62 key bits of the 7-round variant. Then, we use the same related-key differentials in different settings to construct related-key boomerangs for the 8-round variant of MMB. Both attacks have been verified experimentally, and can recover the key bits in only a few minutes using a home PC.

7.1 Attacking 7 Rounds of MMB

We start by showing that the related-key differential characteristic $\tau \rightarrow \tau^*$ can be extended by one more round and thus, B_0 can be extended to cover 7 rounds of MMB. This extended boomerang can be used to recover k_0 as before.

To attack the 7-round variant of MMB we reuse the previously used differential $\triangle \rightarrow \triangle^*$

$$\begin{aligned}\triangle = (0,0,\bar{0},\bar{0}) &\xrightarrow[(0,0,\bar{0},\bar{0})]{\sigma[\mathrm{k}^1]} (0,0,0,0) \xrightarrow{\gamma} (0,0,0,0) \xrightarrow{\eta} (0,0,0,0) \xrightarrow{\theta} (0,0,0,0)\\ &\xrightarrow[(0,\bar{0},\bar{0},0)]{\sigma[\mathrm{k}^2]} (0,\bar{0},\bar{0},0) \xrightarrow{\gamma} (0,\bar{0},\bar{0},0) \xrightarrow{\eta} (0,\bar{0},\bar{0},0) \xrightarrow{\theta} (\bar{0},0,0,\bar{0})\\ &\xrightarrow[(\bar{0},\bar{0},0,0)]{\sigma[\mathrm{k}^3]} (0,\bar{0},0,\bar{0}) \xrightarrow{\gamma} (0,\bar{0},0,\bar{0}) \xrightarrow{\eta} (0,\bar{0},0,\bar{\delta}) \xrightarrow{\theta} (\delta,\bar{0},\delta,\bar{\delta}) = \triangle^*\end{aligned}$$

which holds with probability 1 by

$$(\bar{X},\bar{0},0,\bar{0}) \xrightarrow[(\bar{0},0,0,\bar{0})]{\sigma[\mathrm{k}^0]} (X,\bar{0},0,0) \xrightarrow{\gamma} (\bar{\delta},\bar{0},0,0) \xrightarrow{\eta} (\bar{0},\bar{0},0,0) \xrightarrow{\theta} (0,0,\bar{0},\bar{0}) = \triangle$$

to form a 4-round related-key differential which is used as the basis of the boomerang. We also append one more round to the related-key differential characteristic $\tau \rightarrow \tau^*$ presented in Section 5 to form a 3-round related-key differential characteristic $\tau \rightarrow \tau_e^*$ with probability 1:

$$\begin{aligned}\tau = (0,0,0,\bar{0}) &\xrightarrow[(0,0,0,\bar{0})]{\sigma[\mathrm{k}^4]} (0,0,0,0) \xrightarrow{\gamma} (0,0,0,0) \xrightarrow{\eta} (0,0,0,0) \xrightarrow{\theta} (0,0,0,0)\\ &\xrightarrow[(0,0,\bar{0},0)]{\sigma[\mathrm{k}^5]} (0,0,\bar{0},0) \xrightarrow{\gamma} (0,0,\bar{0},0) \xrightarrow{\eta} (0,0,\bar{0},0) \xrightarrow{\theta} (0,\bar{0},\bar{0},\bar{0})\\ &\xrightarrow[(0,\bar{0},0,0)]{\sigma[\mathrm{k}^6]} (0,0,\bar{0},\bar{0}) \xrightarrow{\gamma} (0,0,\bar{0},\bar{0}) \xrightarrow{\eta} (0,0,\bar{0},\bar{\delta}) \xrightarrow{\theta} (\bar{\delta},\bar{0},\delta,\delta) = \tau_e^*.\end{aligned}$$

Using this extended differential, the extended B_0 (namely B_0^e) is constructed by appending the 3 rounds of $\tau \rightarrow \tau_e^*$ after the 3 rounds of $\triangle \rightarrow \triangle^*$ and prepending the additional input rounds of Equation (2) to form a 7-round boomerang with keys $K^1 = (k_0, k_1, k_2, k_3)$ and $K^2 = K^1 \oplus (\bar{0}, 0, 0, \bar{0})$, which are used for encryption, and $K^3 = K^1 \oplus (0, 0, 0, \bar{0})$ and $K^4 = K^2 \oplus (0, 0, 0, \bar{0}) = K^3 \oplus (\bar{0}, 0, 0, \bar{0})$ which are used for decryption.

The extended B_1 is constructed by appending the first 2 rounds of $\triangle \rightarrow \triangle^*$ after the 4 rounds of $\bigtriangledown^* \rightarrow \bigtriangledown$

$$\begin{aligned}
\bigtriangledown^* = (0,0,\bar{0},0) &\xrightarrow[(0,0,\bar{0},0)]{\sigma[\mathrm{k}^1]} (0,0,0,0) \xrightarrow{\gamma} (0,0,0,0) \xrightarrow{\eta} (0,0,0,0) \xrightarrow{\theta} (0,0,0,0) \\
&\xrightarrow[(0,\bar{0},0,0)]{\sigma[\mathrm{k}^2]} (0,\bar{0},0,0) \xrightarrow{\gamma} (0,\bar{0},0,0) \xrightarrow{\eta} (0,\bar{0},0,0) \xrightarrow{\theta} (\bar{0},\bar{0},\bar{0},0) \\
&\xrightarrow[(\bar{0},0,0,0)]{\sigma[\mathrm{k}^3]} (0,\bar{0},\bar{0},0) \xrightarrow{\gamma} (0,\bar{0},\bar{0},0) \xrightarrow{\eta} (0,\bar{0},\bar{0},0) \xrightarrow{\theta} (\bar{0},0,0,\bar{0}) \\
&\xrightarrow[(0,0,0,\bar{0})]{\sigma[\mathrm{k}^4]} (\bar{0},0,0,0) \xrightarrow{\gamma} (\bar{0},0,0,0) \xrightarrow{\eta} (\bar{\delta},0,0,0) \xrightarrow{\theta} (\bar{\delta},\bar{\delta},0,\bar{\delta}) = \bigtriangledown
\end{aligned}$$

and prepending the additional input round

$$(0,\bar{0},\bar{0},\bar{Y}) \xrightarrow[(0,0,0,\bar{0})]{\sigma[\mathrm{k}^0]} (0,\bar{0},\bar{0},Y) \xrightarrow{\gamma} (0,\bar{0},\bar{0},\bar{\delta}) \xrightarrow{\eta} (0,\bar{0},\bar{0},\bar{0}) \xrightarrow{\theta} (0,0,\bar{0},0) = \bigtriangledown^*,$$

thus, forming the 7-round boomerang B_1^e which uses K^1 and K^3 for encryption, and K^2 and K^4 for decryption.

We use the same method as in Section 6 to generate two sets of plaintexts of size 2^{17} each, that differ only in bits 0–31, and another two sets of plaintexts of size 2^{17} each, that differ only in bits 96–127. Then, we encrypt the plaintexts under the appropriate related-keys, XOR them with the required differences and decrypt under the appropriate keys to find right quartets with respect to B_0^e and B_1^e. As in Section 6 we expect two plaintexts, one of each set to collide with non-negligible probability, thus, satisfying the required input differences for $\triangle \rightarrow \triangle^*$ and $\bigtriangledown^* \rightarrow \bigtriangledown$. Two possible values for k_0 (corresponding to 31-bit key material) are then recovered by 2^{31} trials and another 31 bits for k_3 are recovered by another 2^{31} trials.

This attack uses an overall time of $2^{29.2}$, $2^{21.3}$ memory, and 2^{20} adaptive chosen plaintexts and ciphertexts encrypted under four related-keys.

7.2 Attacking 8 Rounds of MMB

To attack the 8-round variant of MMB we use the related-key differentials in a different setting. We build another boomerang, B_2, which is constructed by

appending the 4 rounds of $\bigtriangledown^* \rightarrow \bigtriangledown$

$$\begin{aligned}\bigtriangledown^* = (0,0,\bar{0},0) &\xrightarrow[(0,0,\bar{0},0)]{\sigma[\mathrm{k}^1]} (0,0,0,0) \xrightarrow{\gamma} (0,0,0,0) \xrightarrow{\eta} (0,0,0,0) \xrightarrow{\theta} (0,0,0,0)\\ &\xrightarrow[(0,\bar{0},0,0)]{\sigma[\mathrm{k}^2]} (0,\bar{0},0,0) \xrightarrow{\gamma} (0,\bar{0},0,0) \xrightarrow{\eta} (0,\bar{0},0,0) \xrightarrow{\theta} (\bar{0},\bar{0},\bar{0},0)\\ &\xrightarrow[(\bar{0},0,0,0)]{\sigma[\mathrm{k}^3]} (0,\bar{0},\bar{0},0) \xrightarrow{\gamma} (0,\bar{0},\bar{0},0) \xrightarrow{\eta} (0,\bar{0},\bar{0},0) \xrightarrow{\theta} (\bar{0},0,0,\bar{0})\\ &\xrightarrow[(0,0,0,\bar{0})]{\sigma[\mathrm{k}^4]} (\bar{0},0,0,0) \xrightarrow{\gamma} (\bar{0},0,0,0) \xrightarrow{\eta} (\bar{\delta},0,0,0) \xrightarrow{\theta} (\bar{\delta},\bar{\delta},0,\bar{\delta}) = \bigtriangledown\end{aligned}$$

after the 3 rounds of $\bigtriangleup \rightarrow \bigtriangleup^*$

$$\begin{aligned}\bigtriangleup = (0,0,\bar{0},\bar{0}) &\xrightarrow[(0,0,\bar{0},\bar{0})]{\sigma[\mathrm{k}^1]} (0,0,0,0) \xrightarrow{\gamma} (0,0,0,0) \xrightarrow{\eta} (0,0,0,0) \xrightarrow{\theta} (0,0,0,0)\\ &\xrightarrow[(0,\bar{0},\bar{0},0)]{\sigma[\mathrm{k}^2]} (0,\bar{0},\bar{0},0) \xrightarrow{\gamma} (0,\bar{0},\bar{0},0) \xrightarrow{\eta} (0,\bar{0},\bar{0},0) \xrightarrow{\theta} (\bar{0},0,0,\bar{0})\\ &\xrightarrow[(\bar{0},\bar{0},0,0)]{\sigma[\mathrm{k}^3]} (0,\bar{0},0,\bar{0}) \xrightarrow{\gamma} (0,\bar{0},0,\bar{0}) \xrightarrow{\eta} (0,\bar{0},0,\bar{\delta}) \xrightarrow{\theta} (\delta,\bar{0},\delta,\bar{\delta}) = \bigtriangleup^*\end{aligned}$$

and prepend the extra input round

$$(\bar{X},\bar{0},0,\bar{0}) \xrightarrow[(\bar{0},0,0,\bar{0})]{\sigma[\mathrm{k}^0]} (X,\bar{0},0,0) \xrightarrow{\gamma} (\bar{\delta},\bar{0},0,0) \xrightarrow{\eta} (\bar{0},\bar{0},0,0) \xrightarrow{\theta} (0,0,\bar{0},\bar{0}) = \bigtriangleup.$$

The new boomerang, B_2, uses $K^1 = (k_0, k_1, k_2, k_3)$ and $K^2 = K^1 \oplus (\bar{0},0,0,\bar{0})$, for encryption, and $K^3 = K^1 \oplus (0,0,\bar{0},0)$ and $K^4 = K^2 \oplus (0,0,\bar{0},0) = K^3 \oplus (\bar{0},0,0,\bar{0})$ for decryption.

The second boomerang is the extension of B_1^e (namely, B_1^{ee}) where the 3 rounds of $\bigtriangleup \rightarrow \bigtriangleup^*$ are concatenated after the 4 rounds of $\bigtriangledown^* \rightarrow \bigtriangledown$, and the additional input round

$$(0,\bar{0},\bar{0},\bar{Y}) \xrightarrow[(0,0,0,\bar{0})]{\sigma[\mathrm{k}^0]} (0,\bar{0},\bar{0},Y) \xrightarrow{\gamma} (0,\bar{0},\bar{0},\bar{\delta}) \xrightarrow{\eta} (0,\bar{0},\bar{0},\bar{0}) \xrightarrow{\theta} (0,0,\bar{0},0) = \bigtriangledown^*,$$

is prepended. This boomerang uses K^1 and $K^5 = K^1 \oplus (0,0,0,\bar{0})$ for encryption, and K^2 and $K^6 = K^5 \oplus (\bar{0},0,0,\bar{0}) = K^1 \oplus (\bar{0},0,0,0)$ for decryption.

The same method as before is used when we generate two sets of plaintexts of size 2^{17} each, that differ only in bits 0–31, and another two sets of plaintexts of size 2^{17} each, that differ only in bits 96–127. Then, we encrypt the plaintexts under the appropriate related-keys, XOR them with the required differences and decrypt under the appropriate keys to find right quartets with respect to B_0^{ee} and B_1^{ee}. As in Section 6 we expect two plaintexts, one of each set to collide with non-negligible probability, thus, satisfying the required input differences for $\bigtriangleup \rightarrow \bigtriangleup^*$ and $\bigtriangledown^* \rightarrow \bigtriangledown$. Two possible values for k_0 (corresponding to 31-bit

key material) are then recovered by 2^{31} trials and another 31 bits for k_3 are recovered by another 2^{31} trials.

This part of the attack uses an overall time of 2^{29}, $2^{21.3}$ memory, and 2^{20} adaptive chosen plaintexts and ciphertexts encrypted under six related-keys.

7.3 Attacking 9 Rounds of MMB

In order to attack a 9-round variant of MMB, we append the 4-round differential characteristic $\bigtriangledown^* \rightarrow \bigtriangledown$ onto itself to construct an 8-round distinguisher with probability 1 for MMB. Then, by prepending the additional input round:

$$(0,\bar{0},\bar{0},\bar{Y}) \xrightarrow[(0,0,0,\bar{0})]{\sigma[\mathrm{k}^0]} (0,\bar{0},\bar{0},Y) \xrightarrow{\gamma} (0,\bar{0},\bar{0},\bar{\delta}) \xrightarrow{\eta} (0,\bar{0},\bar{0},\bar{0}) \xrightarrow{\theta} (0,0,\bar{0},0) = \bigtriangledown^*,$$

at the beginning of the distinguisher, an adversary can attack a variant of MMB with 9 rounds. This 9-round boomerang uses the same pair of related keys for both encryption and decryption, and thus, by using the exact same algorithm as before, the adversary can recover 31 key bits of k_3 in time of $2^{27.8}$, $2^{21.3}$ memory, and 2^{19} adaptive chosen plaintexts and ciphertexts encrypted under two related-keys.

8 Conclusions

In this paper we have used various techniques from the differential cryptanalysis family to break the MMB block cipher. By extending previous results along with a new related-key differential we discovered, we were able to identify three related-key differentials that allowed us to construct two 5-round related-key distinguishers with probability 1. We then used each of these distinguishers as the basis for a 6-round boomerang that is able to recover 31 key bits using 2^{19} data in $2^{28.4}$ time using four related keys. We then used the already recovered key bits to recover another 31 key bits using a simple 1R attack. The last 32 bits are recovered by exhaustive search. The suggested attack can recover all the key bits in 2^{35} time using 2^{20} adaptive chosen plaintexts and ciphertexts and $2^{21.3}$ memory.

We verified our results experimentally by writing a program that recovers the required key bits in about 15 minutes on a home PC. To the best of our knowledge, though it has been many years since MMB was presented, this is the first practical time attack that recovers its full key.

Finally, we showed that even if MMB had been extended to include 7 or 8 rounds an adversary can still recover half of its key bits using the same techniques, with similar time, data and memory complexities.

Acknowledgement

The authors thank Nathan Keller for his insights and comments, Atul Luykx and Elmar Tischhauser for reading and commenting on early versions of this paper,

and Idan Dorfman for his mental support. The authors also thank the anonymous referees for their comments and suggestions. The generous support of the Israeli Ministry of Science and Technology of the first author is acknowledged.

References

1. Anonymous Authors: Source Code of the Attack. Available Upon Request (2013)
2. Biham, E., Dunkelman, O., Keller, N.: Related-Key Boomerang and Rectangle Attacks. In Cramer, R., ed.: EUROCRYPT. Volume 3494 of Lecture Notes in Computer Science., Springer (2005) 507–525
3. Biham, E., Shamir, A.: Differential Cryptanalysis of DES-like Cryptosystems. J. Cryptology **4**(1) (1991) 3–72
4. Daemen, J.: Cipher and Hash Function Design. Strategies based on linear and differential cryptanalysis. PhD thesis, Katholieke Universiteit Leuven (1995) René Govaerts and Joos Vandewalle (promotors).
5. Daemen, J., Govaerts, R., Vandewalle, J.: Block ciphers based on modular arithmetic. In Wolfowicz, W., ed.: Proceedings of the 3rd Symposium on State and Progress of Research in Cryptography, Rome,IT, Fondazione Ugo Bordoni (1993) 80–89
6. Hong, S., Kim, J., Lee, S., Preneel, B.: Related-Key Rectangle Attacks on Reduced Versions of SHACAL-1 and AES-192. In Gilbert, H., Handschuh, H., eds.: FSE. Volume 3557 of Lecture Notes in Computer Science., Springer (2005) 368–383
7. Jia, K., Chen, J., Wang, M., Wang, X.: Practical Attack on the Full MMB Block Cipher. In Miri, A., Vaudenay, S., eds.: Selected Areas in Cryptography. Volume 7118 of Lecture Notes in Computer Science., Springer (2011) 185–199
8. Kelsey, J., Schneier, B., Wagner, D.: Related-key cryptanalysis of 3-WAY, Biham-DES, CAST, DES-X, NewDES, RC2, and TEA. In Han, Y., Okamoto, T., Qing, S., eds.: ICICS. Volume 1334 of Lecture Notes in Computer Science., Springer (1997) 233–246
9. Kim, J., Kim, G., Hong, S., Lee, S., Hong, D.: The Related-Key Rectangle Attack - Application to SHACAL-1. In Wang, H., Pieprzyk, J., Varadharajan, V., eds.: ACISP. Volume 3108 of Lecture Notes in Computer Science., Springer (2004) 123–136
10. Lai, X., Massey, J.L.: A Proposal for a New Block Encryption Standard. In Damgård, I., ed.: EUROCRYPT. Volume 473 of Lecture Notes in Computer Science., Springer (1990) 389–404
11. Lai, X., Massey, J.L.: Markov Ciphers and Differentail Cryptanalysis. In Davies, D.W., ed.: EUROCRYPT. Volume 547 of Lecture Notes in Computer Science., Springer (1991) 17–38
12. Wagner, D.: The Boomerang Attack. In Knudsen, L.R., ed.: FSE. Volume 1636 of Lecture Notes in Computer Science., Springer (1999) 156–170
13. Wang, M., Nakahara, J., Sun, Y.: Cryptanalysis of the Full MMB Block Cipher. In Jr., M.J.J., Rijmen, V., Safavi-Naini, R., eds.: Selected Areas in Cryptography. Volume 5867 of Lecture Notes in Computer Science., Springer (2009) 231–248

A Probabilities for the Transitions $X \xrightarrow{\otimes G_0} \bar{\delta}$ and $Y \xrightarrow{\otimes G_3} \bar{\delta}$

In this Appendix we present a list of transitions from some input differences to δ with respect to modular multiplication by G_0 and G_3, and their probabilities:

Input Difference (X)	Probability	$-log_2(p)$
$\texttt{7FBFFB64}_x$	$32768 \cdot 2^{-32}$	17
$\texttt{45440164}_x$	$31872 \cdot 2^{-32}$	17.03
$\texttt{7F3FFB64}_x$	$31744 \cdot 2^{-32}$	17.04
$\texttt{C5440164}_x$	$28032 \cdot 2^{-32}$	17.22
$\texttt{4000C164}_x$	$26912 \cdot 2^{-32}$	17.28
$\texttt{C000C164}_x$	$26336 \cdot 2^{-32}$	17.31
$\texttt{90440164}_x$	$26112 \cdot 2^{-32}$	17.32
$\texttt{88240164}_x$	$26112 \cdot 2^{-32}$	17.32
$\texttt{08240164}_x$	$26112 \cdot 2^{-32}$	17.32
$\texttt{80240164}_x$	$26112 \cdot 2^{-32}$	17.32
$\texttt{00240164}_x$	$26112 \cdot 2^{-32}$	17.32
$\texttt{10440164}_x$	$26112 \cdot 2^{-32}$	17.32
$\texttt{90C40164}_x$	$25344 \cdot 2^{-32}$	17.37
$\texttt{10C40164}_x$	$25344 \cdot 2^{-32}$	17.37
$\texttt{C0014404}_x$	$25024 \cdot 2^{-32}$	17.38
$\texttt{C0014164}_x$	$24992 \cdot 2^{-32}$	17.39
$\texttt{C00A0164}_x$	$24960 \cdot 2^{-32}$	17.39
$\texttt{80012404}_x$	$24576 \cdot 2^{-32}$	17.41
$\texttt{80011C04}_x$	$24576 \cdot 2^{-32}$	17.41
$\texttt{00012404}_x$	$24576 \cdot 2^{-32}$	17.41
$\texttt{00011C04}_x$	$24576 \cdot 2^{-32}$	17.41
$\texttt{400A0164}_x$	$24192 \cdot 2^{-32}$	17.43
$\texttt{40014164}_x$	$24160 \cdot 2^{-32}$	17.43
$\texttt{40014404}_x$	$24128 \cdot 2^{-32}$	17.44
$\texttt{D77FFB64}_x$	$23328 \cdot 2^{-32}$	17.49

25 Most Probable Transitions for $X \xrightarrow{\otimes G_0} \bar{\delta}$

Input Difference (Y)	Probability	$-log_2(p)$
$\texttt{7FFD7FF1}_x$	$17920 \cdot 2^{-32}$	17.87
$\texttt{FFF7FED1}_x$	$16640 \cdot 2^{-32}$	17.97
$\texttt{7FFD7FF9}_x$	$16384 \cdot 2^{-32}$	18
$\texttt{409004D1}_x$	$14848 \cdot 2^{-32}$	18.14
$\texttt{C09004D1}_x$	$14336 \cdot 2^{-32}$	18.19
$\texttt{7FFBF9D1}_x$	$14336 \cdot 2^{-32}$	18.19
$\texttt{5FDED9D1}_x$	$12480 \cdot 2^{-32}$	18.39
$\texttt{400801C9}_x$	$12304 \cdot 2^{-32}$	18.41
$\texttt{7FFBFDD9}_x$	$12288 \cdot 2^{-32}$	18.41
$\texttt{C00801C9}_x$	$12272 \cdot 2^{-32}$	18.41
$\texttt{5FFD79D1}_x$	$12032 \cdot 2^{-32}$	18.44
$\texttt{D41004D1}_x$	$11400 \cdot 2^{-32}$	18.52
$\texttt{775F7FF9}_x$	$11280 \cdot 2^{-32}$	18.53
$\texttt{775F7FF1}_x$	$11280 \cdot 2^{-32}$	18.53
$\texttt{541004D1}_x$	$10632 \cdot 2^{-32}$	18.62
$\texttt{77FDFE51}_x$	$10016 \cdot 2^{-32}$	18.70
$\texttt{FFF7D9D1}_x$	$9984 \cdot 2^{-32}$	18.71
$\texttt{7FDFD851}_x$	$9984 \cdot 2^{-32}$	18.71
$\texttt{C30011D1}_x$	$9760 \cdot 2^{-32}$	18.74
$\texttt{508011D1}_x$	$9632 \cdot 2^{-32}$	18.76
$\texttt{430011D1}_x$	$9504 \cdot 2^{-32}$	18.78
$\texttt{805041D1}_x$	$9472 \cdot 2^{-32}$	18.79
$\texttt{005041D1}_x$	$9472 \cdot 2^{-32}$	18.79
$\texttt{C41111D1}_x$	$9456 \cdot 2^{-32}$	18.79
$\texttt{908011D1}_x$	$9440 \cdot 2^{-32}$	18.79

25 Most Probable Transitions for $Y \xrightarrow{\otimes G_3} \bar{\delta}$

Table 4. Most Probable Transitions for Multiplications in MMB