Improved Linear Cryptanalysis of Reduced-Round MIBS

Aslı Bay[1], Jialin Huang[1,2], and Serge Vaudenay[1]

1. EPFL, Switzerland
2. Shanghai Jiao Tong University, China
email{asli.bay, serge.vaudenay}@epfl.ch, jlhuang.cn@gmail.com

Abstract. MIBS is a 32-round lightweight block cipher with 64-bit block size and two different key sizes, namely 64-bit and 80-bit keys. Bay et al. provided the first impossible differential, differential and linear cryptanalyses of MIBS. Their best attack was a linear attack on the 18-round MIBS-80. In this paper, we significantly improve their attack by discovering more approximations and mounting Hermelin et al.'s multidimensional linear cryptanalysis. We also use Nguyen et al.'s technique to have less time complexity. We attack on 19 rounds of MIBS-80 with a time complexity of $2^{74.23}$ 19-round MIBS-80 encryptions by using $2^{57.87}$ plaintext-ciphertext pairs. To the best of our knowledge, the result proposed in this paper is the best cryptanalytic result for MIBS, so far.

Keywords: multidimensional linear cryptanalysis, lightweight block ciphers, MIBS, RFID tags, sensor networks

1 Introduction

MIBS [ISSK09] is a lightweight block cipher suitable for constraint environments, such as RFID tags and sensor networks. MIBS was proposed by Izadi et al. in 2009; it has a simple Feistel structure and an SPN round function. The first and detailed cryptanalysis of reduced-round MIBS was realized by Bay et al. [BNV10] and they gave linear, differential and impossible differential cryptanalyses of MIBS. The best attack among them was the linear attack on the 18-round MIBS-80 with the time complexity of $2^{78.62}$ 18-round MIBS encryptions.

Linear cryptanalysis was proposed by Matsui and firstly applied to FEAL cipher [MY93] and subsequently to DES [Mat94b]. It is a known-plaintext attack and the adversary assumes that the plaintexts are independent and linearly distributed over the message space $\{0,1\}^n$. Essentially, the attack exploits linear (or affine) relations of plaintext, ciphertext and the key bits. Afterwards, Matsui [Mat94a] discovered that using two linear approximations together helps to reduce the data complexity of linear cryptanalysis. Simultaneously, Kaliski and Robshaw [JR94] introduced multiple linear cryptanalysis to reduce data complexities of Matsui's algorithms by using several approximations, but each linear approximation involves the same key bits. Then, Biryukov et al. [BCQ04] further improved this technique by using several linear approximations involving different key bits. However, both in Kaliski-Robshaw's and Biryukov et al.'s techniques, the statistical independence of each linear approximations is assumed. It is shown by Murphy in [Mur06] that this assumption may not hold in general. Biryukov et al. [BCQ04] is also added an enhancement heuristically to its method by using more approximations which are linearly and statistically dependent.

Afterwards, Baignères et al. [BJV04] proposed a statistical linear distinguisher, such that statistical independence of linear approximations is not needed anymore. In their technique, the attack is modeled as a hypothesis testing problem based on the log-likelihood ratio (LLR). They showed that the efficiency of a multidimensional distinguisher is measured by the distance of its distribution to the uniform distribution. This distance is then called the capacity (see Definition 1) which is directly related to the number of samples N needed for the attack.

Hermelin et al. further analyzed this technique for extending Matsui's Algorithm 1 and Matsui's Algorithm 2 to multiple dimensions by using some statistical techniques [HCN08,CHN09,HCN09,HN10,Her10,HN11,HN12]. They studied on the goodness-of-fit problem solved by the χ^2-statistic, the LLR-statistic method and the convolution method. They showed how to use correlations of one-dimensional linear approximations to determine multidimensional probability distributions (see Lemma 2), and to compute the capacity (see Lemma 3). They verified the new techniques on the AES candidate Serpent.

Table 1. Key recovery attacks on reduced-round MIBS

#Rounds	Data	Time	Memory	Cipher	Reference	Attack type
12	2^{59} CP	$2^{58.8}$	2^{62}	MIBS-80	[BNV10]	ID
13	2^{61} CP	2^{40}	2^{24}	MIBS-64	[BNV10]	DC
13	2^{61} CP	2^{56}	2^{24}	MIBS-80	[BNV10]	DC
14	2^{40} CP	$2^{37.2}$	2^{40}	MIBS-64	[BNV10]	DC
14	2^{40} CP	2^{40}	2^{40}	MIBS-80	[BNV10]	DC
17	2^{58} KP	2^{69}	2^{58}	MIBS-80	[BNV10]	LC
18[1]	$2^{63.47}$ KP	$2^{78.62}$	$2^{63.47}$	MIBS-80	[BNV10]	LC
19	$2^{57.87}$ KP	$2^{78.22}$	2^{76}	MIBS-80	Section 4	MLC
19	$2^{57.87}$ CP	$2^{74.23}$	2^{72}	MIBS-80	Section 5	MLC

Time complexity is number of reduced-round encryptions; DC: Differential Cryptanalysis; ID: Impossible Differential Attack; CP: Chosen Plaintext; KP: Known Plaintext; MLC: Multidimensional Linear Cryptanalysis

In this paper, we present a multidimensional linear attack on the 19-round MIBS-80 which outperforms the previous linear attack in terms of the number of rounds, time and data complexities. See Table 1 for the comparison of our results with the existing ones. We exploit Bay et al.'s attack by finding/using more linear approximations to reduce the data complexity. Moreover, we use Nguyen et al.'s

[1] We put the corrected complexities compared to [BNV10] by fixing a flaw in the attack.

approach to decrease time complexity of the attack, which enables us to attack on one more round.

The rest of the paper is organized as follows. We give a brief description of MIBS in Section 2. Some mathematical background and the detailed description of the linear attack in multiple dimensions are given in Section 3. Section 4 gives our multidimensional linear attack on the 19-round MIBS-80. Section 5 proposes a chosen-message version of the attack given in Section 4 on the 19-round MIBS-80. Finally, Section 6 concludes the paper.

2 Description of MIBS

MIBS [ISSK09] is a block cipher using the conventional Feistel structure (see Fig. 1). MIBS has a 64-bit block size supporting 64-bit and 80-bit keys and iterates 32 rounds for both key sizes. The round function F of MIBS is an SPN composed of an XOR layer with a round key, a layer of 4×4-bit S-Boxes (S layer), and a linear transformation layer (P layer), in this order. The components of the encryption process involved in F are explained as follows. Note that all internal operations in MIBS are nibble-wise, that is, on 4-bit words.

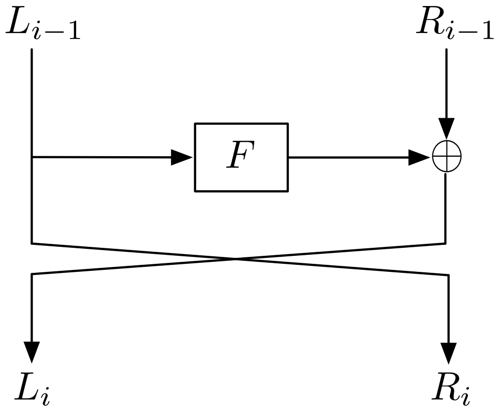

Fig. 1. The ith round of MIBS

Key addition: In each round i, $1 \leq i \leq 32$, the 32-bit input state s_i to the F function is XORed with the round key K_i, that is $s'_i = s_i \oplus K_i$, where "$\oplus$" denotes XOR.

S-Box layer S : After key addition, the state s'_i is split into eight nibbles and identical 4×4 S-Boxes (see Table 2) are applied in parallel.

Table 2. The S-Box of MIBS in hexadecimal notation

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	4	f	3	8	d	a	c	0	b	5	7	e	2	6	1	9

Linear transformation layer P : The input $(y_1, y_2, y_3, y_4, y_5, y_6, y_7, y_8)$ is transformed into its output $(y'_1,\ y'_2,\ y'_3,\ y'_4,\ y'_5,\ y'_6,\ y'_7,\ y'_8)$ by

$$\begin{aligned}
y'_1 &= y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_7 \oplus y_8;\\
y'_2 &= y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_6 \oplus y_7;\\
y'_3 &= y_1 \oplus y_2 \oplus y_3 \oplus y_5 \oplus y_6 \oplus y_8;\\
y'_4 &= y_2 \oplus y_3 \oplus y_4 \oplus y_7 \oplus y_8;\\
y'_5 &= y_1 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_8;\\
y'_6 &= y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_6;\\
y'_7 &= y_1 \oplus y_2 \oplus y_3 \oplus y_6 \oplus y_7;\\
y'_8 &= y_1 \oplus y_3 \oplus y_4 \oplus y_6 \oplus y_7 \oplus y_8,
\end{aligned}$$

where y_i's and y'_i's are 4-bit words.

Key Schedule: For our attack purposes, we mention only the 80-bit version of the key schedule generating 32-bit round keys K_i, for $1 \leq i \leq 32$. Let $state^i$ denote the ith round key state and let $state^0$ denote the 80-bit secret key. Considering bit numbering in right-to-left, 80-bit key schedule is formalized in Algorithm 1. In these algorithms, "$\ggg$" means bitwise rotation to right, "$\|$" means string concatenation, and "$\sim$" indicates a sequence of bit positions. In addition, S denotes the S-Box which is the same as the S-Box (see Table 2) in the round function. In the rest of this thesis, we denote by MIBS-80 (resp. MIBS-64) the 80-bit key (resp. 64-bit key) version of MIBS. Note that the input to the ith round is denoted by (L_{i-1}, R_{i-1}), with $(L_i, R_i) = (R_{i-1} \oplus F(K_i, L_{i-1}), L_{i-1}) \in \{0,1\}^{32}$ denoting the round output. Let (L_0, R_0) and (L_{32}, R_{32}) denote a plaintext block and a ciphertext block, respectively.

Algorithm 1 The 80-bit key schedule of MIBS.

```
for $i = 1$ to 32 do
    $state^i = state^{i-1} \ggg 19$
    $state^i = S[state^i[80 \sim 77]] \| S[state^i[76 \sim 73]] \| state^i[72 \sim 1]$
    $state^i = state^i[80 \sim 20] \| state^i[19 \sim 15] \oplus \text{RoundCounter} \| state^i[14 \sim 1]$
    $K_i = state^i[80 \sim 49]$
end for
```

3 Preliminary

3.1 Mathematical Background

We denote by $\mathbb{F}_2$ the field with two elements and $\mathbb{F}_2^m$ denotes the m-dimensional vector space over $\mathbb{F}_2$. Let X be a discrete random variable in $\mathbb{F}_2^m$ and $p = (p_0, p_1, \ldots, p_{2^m-1})$ be the probability distribution of X such that $p_\eta = \Pr(X = \eta)$, where $\eta \in \mathbb{F}_2^m$. The function $f = (f_1, \ldots, f_m) : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is called a vectorial Boolean function, where $f_i : \mathbb{F}_2^n \to \mathbb{F}_2$ is a Boolean function.

Definition 1. *Let $p = (p_0, p_1, \ldots, p_\eta)$ and $q = (q_0, q_1, \ldots, q_\eta)$ be two discrete probability distributions with sample space $\mathcal{S}$. Then, the* capacity *between p and q is*

$$C(p, q) = \sum_{\eta \in \mathcal{S}} \frac{(p_\eta - q_\eta)^2}{q_\eta}.$$

In the case where q is the uniform distribution θ, the capacity is denoted by $C(p)$. Let Y be a Bernoulli(p_0)-distributed random variable which takes values in $\{0, 1\}$ such that $\Pr(Y = 0) = p_0$. Then, the *correlation* of Y with zero is defined as

$$c(Y) = 2\Pr(Y = 0) - 1 = 2p_0 - 1. \tag{1}$$

The *bias* of Y, denoted as ε is equal to $c(Y)/2$. Let X be an m-bit random variable with probability distribution p and $a \in \mathbb{F}_2^m$. Then, we have

$$c(a \cdot X) = \sum_{\eta \in \mathbb{F}_2^m} (-1)^{a \cdot \eta} p_\eta. \tag{2}$$

The following lemma proves that the probability distribution p of m-bit random variable X, taking values from $\mathbb{F}_2^m$, is computed by the correlations of $a \cdot X$, where $a \in \mathbb{F}_2^m$.

Lemma 2. *([HCN08]) Let X be an m-bit random taking values from $\mathbb{F}_2^m$variable with probability distribution p, then*

$$p_\eta = 2^{-m} \sum_{a \in \mathbb{F}_2^m} (-1)^{a \cdot \eta} c(a \cdot X), \qquad \forall \eta \in \mathbb{F}_2^m.$$

Lemma 3. *([HCN08]) Let X be an m-bit random variable taking values from $\mathbb{F}_2^m$ and p be its probability distribution. Then, the capacity of p is*

$$C(p) = \sum_{a \in \mathbb{F}_2^m - \{0\}} c(a \cdot X)^2.$$

3.2 Matsui's Algorithm 2 in Multidimensional Linear Cryptanalysis

Let $f : \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a vectorial Boolean function and binary vectors $w_i \in \mathbb{F}_2^n$ and $u_i \in \mathbb{F}_2^n$ be the binary masks such that the pairs (u_i, w_i) are linearly independent. Each one-dimensional approximation of f is defined as a function g_i such that

$$g_i(X) = w_i \cdot f(X) \oplus u_i \cdot X, \quad \forall X \in \mathbb{F}_2^n,$$

where all g_i's are the base approximations for $i = 1, \ldots, m$. Let c_i be the correlation of g_i, $i = 1, \ldots, m$, and $g = (g_1, g_2, \ldots, g_m)$ be an m-dimensional vectorial Boolean function. Let $p = (p_0, p_1, \ldots, p_{2^m-1})$ be the probability distribution of g. This p can be computed from all possible one-dimensional correlations by Lemma 2.

For an n-bit block cipher of r rounds, R^r denotes the last round of the cipher with its inverse R^{-r}, and $K_r \in \mathbb{F}_2^\ell$ denotes the last round key. K_r is also called the outer key. Let x and y' be the plaintext and ciphertext, respectively. The $(r-1)$-round m-dimensional linear approximation of the block cipher is written by

$$Ux \oplus Wy \oplus VK_{1,..,r-1},$$

where, $y = R^{-r}(y', K_r)$, $K_{1,..,r-1}$ is inner key bits, $U = (u_1, u_2, \ldots, u_m)^T$, $V = (v_1, v_2, \ldots, v_m)^T$ and $W = (w_1, w_2, \ldots, w_m)^T$ are the matrices of the masks for the texts and the inner key bits, respectively. The matrix V splits the inner key bits into 2^m equivalent classes such that $z = VK_{1,..,r-1} \in \mathbb{F}_2^m$. Furthermore, if $Ux \oplus Wy$ is distributed with p, then $Ux \oplus Wy \oplus VK_{1,..,r-1}$ is distributed with p^z, where all p^z's, $z \in \mathbb{F}_2^m$, are the permutations of each other. That is,

$$p^z_{\eta \oplus \alpha} = p^{z \oplus \alpha}_\eta, \quad \forall z, \eta, \alpha \in \mathbb{F}_2^m.$$

This implies that $C(p) = C(p^z)$, for all $z \in \mathbb{F}_2^m$. Now, we denote by $\widetilde{K_r}$ and $\widetilde{z}$ the right rth round key and the right inner key class, respectively. The aim of Matsui's Algorithm 2 for multidimensional linear attacks is to find $\widetilde{K_r}$ as well as $\widetilde{z}$. Note that we only attack on the last round key, but recovering more round keys is doable. The attack is mainly composed of four phases, namely, the *distillation phase*, the *analysis phase*, the *ranking phase* and the *search phase*.

In the distillation phase, we collect N plaintext-ciphertext pairs (x_1, y'_1), $\ldots$,(x_N, y'_N), where $x_1, \ldots, x_N$ are taken independently from the uniform distribution. By Algorithm 2, we compute the empirical probability distributions $q[K_r, \cdot]$ for eah key candidate K_r which are

$$q[K_r, \eta] = N^{-1}\#\{t : Ux_t \oplus WR^{-r}(y'_t, K_r) = \eta\}, \quad \forall \eta \in \mathbb{F}_2^m.$$

In the analysis phase, we choose the convolution method [HN10,Her10], as it is efficient in terms of both time and data complexities. Let p and q be the probability distributions of two m-bit random variables, X and Y, respectively. The ith component of the convolution of p and q is defined as

$$(q * p)_i = \sum_{\eta \in \mathbb{F}_2^m} q_\eta p_{i \oplus \eta}.$$

Algorithm 2 Distillation phase

```
procedure Compute $q((x_1, y'_1), \ldots, (x_N, y'_N),\ g_1, g_2, \ldots, g_m)$
    for $t = 1 \to N$ do
        for $K_r = 0 \to 2^\ell - 1$ do
            partially decrypt $y'_t$ and obtain $y_t = R^{-r}(y'_t, K_r)$
            for $i = 1 \to m$ do
                compute $\eta_i = u_i \cdot x_t \oplus w \cdot y_t$
            end for
            increment the counter $q[K_r, \eta]$ corresponding to $\eta = (\eta_1, \ldots, \eta_m)$
        end for
    end for
    update $q[K_r, \eta]$ as $q[K_r, \eta]/N$
end procedure
```

Algorithm 3 Analysis phase

```
procedure Compute Convolution($q$,$p$)
    for $K_r = 0 \to 2^\ell - 1$ do
        compute $q[K_r, \cdot] * p$ using Fast Fourier Transform (FFT)
        store $G(K_r) = \max_{z \in \mathbb{F}_2^m}(q[K_r, \cdot] * p)_z$ and $z'$ which is the index of the maximal component of $(q[K_r, \cdot] * p)$.
    end for
end procedure
```

Using the convolution method, the mark of each key candidate K_r is defined by

$$G(K_r) = \max_{z \in \mathbb{F}_2^m}(q[K_r, \cdot] * p)_z.$$

For each possible K_r, we find the maximal component of the convolution $(q[K_r, \cdot] * p)$, and record this maximal component as $G(K_r)$ together with its corresponding index z'. Because, the right key $\widetilde{K_r}$ is supposed to have the highest mark $G(K_r)$ and the right inner key $\widetilde{z}$ is recovered corresponding to $G(\widetilde{K_r})$. The detailed process of the analysis phase is mentioned in Algorithm 3.

In the ranking phase, we rank the key candidates by $G(K_r)$, and the keys are sorted in a decreasing order according to their ranking values. Under a predetermined advantage a, the right key candidate $\widetilde{K_r}$ should be within the position of $2^{\ell - a}$, where ℓ is the number of targeted key bits.

Then, the search phase, where the remaining key bits are searched and the correctness of the ranking result is verified, can be done by a number of trial encryptions according to the sorted candidate list.

3.3 Complexities of Multidimensional Linear Attacks (by the Convolution Method)

The advantage of the convolution method can be computed from the following Theorem.

Theorem 4. *[Her10] To distinguish the uniform distribution from the distributions of p^z which are close to the uniform one ($z \in \mathbb{F}_2^m$), the advantage of the convolution method for finding the last round key K_r is given by*

$$a = (\sqrt{NC(p)} - \Phi^{-1}(P_S))^2/2 - m,$$

where $P_S(> 0.5)$ is the success probability, N is the amount of data required for the attack, Φ is the cumulative distribution function of the normal distribution, $C(p)$ is the capacity of p and m is the dimension of the linear approximation.

From Theorem 4, the data complexity N of the convolution method is approximated to

$$N = \frac{a+m}{C(p)},$$

where $C(p)$ is the capacity of p, a denotes the advantage, and m is the dimension of the linear approximation.

The time complexity of the analysis phase is $m2^{m+\ell}$ operations. In the ranking phase, sorting can be done within $\ell 2^{\ell}$. In addition, the time complexity of the search phase depends on the advantage, i.e. 2^{k-a}, where k is the master key size. Algorithm 2 computes the empirical probability distributions for all keys K_r with $\mathcal{O}(mN2^{\ell})$ time complexity, in a conventional way. However, this complexity can be reduced to $\mathcal{O}(N + \lambda 2^{m+\ell})$ in [NWW11], where $\lambda > 0$ differs by the attack cases. That is, if only the last round(s) keys are attacked, then $\lambda = 4m + 3\ell$, and if both first and last rounds keys are attacked, then $\lambda = 3m + 3\ell$. This is not somewhat different from the extension of the Collard's method for reducing the time complexity of the Matsui's Algorithm 2 [CSQ07].

However, in [NWW11], they consider linear approximations having the same output masks and they compute the empirical probability distribution for both cases. In our attack, linear approximations have different input and output masks, we slightly modify their strategy to use and present as follows. Note that in our case the active S-boxes in the first and last rounds are the same for all linear approximations even with different masks.

Lemma 5. *Let $g_a = f_1^a(x, y') \oplus f_2^a(x, K_1) \oplus f_3^a(y', K_r)$ be all linear approximations involved in an attack, and f_1^a, f_2^a and f_3^a be boolean functions, for all $a \in \mathbb{F}_2^m$. Assume w.l.o.g. that g_i's, for $i = 1, \ldots, m$, constitute m base linear approximations which linearly span the rest of approximations. Let K_1 and K_r denote the respective key bits in the first and last rounds. Let x and y' be the plaintext and ciphertext, respectively. Then, in the distillation phase, the probability distribution $q[k, \eta]$ of $g = (g_1, g_2, \ldots, g_m)$, where $\eta \in \mathbb{F}_2^m$ and K contains ℓ bits from both K_1 and K_r in total, is computed with $\mathcal{O}(mN + (2m+3\ell)2^{\ell+m} + 2^{\ell})$ time complexity and $\mathcal{O}(2^{m+\ell})$ memory complexity.*

Proof. For the details of this proof please refer to Appendix A. The proof can also be implied by the attack procedure explained in Section 4.

4 Multidimensional Linear Cryptanalysis of Reduced-Round MIBS-80

In this section, we apply a 12-dimensional linear attack on the 19 rounds of MIBS-80 by using the convolution method [Her10].

4.1 Previous 16-round Linear Approximations

Bay et al. [BNV10] found a set of six 16-round linear approximations with 31 active S-boxes. Namely, these linear approximations are due to the six possible instantiations from the linear approximation table (LAT) (see Appendix B) of MIBS, $(w, z) \in \{(2_x, 6_x), (6_x, 2_x), (4_x, e_x), (e_x, 4_x), (8_x, d_x), (d_x, 8_x)\}$, where the symmetry $w \overset{\text{S-Box}}{\rightarrow} z$ and $z \overset{\text{S-Box}}{\rightarrow} w$ (both with the same bias 2^{-2}) is exploited. Each of these 16-round linear approximations has a bias $\varepsilon = 2^{-31}$. Since $c = 2|\varepsilon|$, their correlations are $c = 2^{-30}$. The set of 16-round linear approximations can be found in Appendix C. Note that the input mask to the ith round is denoted by $(\mathit{\Gamma} L_{i-1}, \mathit{\Gamma} R_{i-1})$. The $(i+1)$th round input mask is the ith round output mask. Values subscripted by "x" are in hexadecimal base.

4.2 Our set of 16-round Linear Approximations

We exploit the 16-round linear approximations mentioned in Section 4.1 with 31 active S-boxes. We find 594 more by using different combinations of nonzero masks in the last round of them. Strictly speaking, we cut the first 15 rounds of these linear approximations. The total bias of each is 2^{-29}. Since the last round input mask to the function F is free to choose, there are ten possible nonzero input masks with nonzero biases for each w, that is, there are two nonzero input masks with a bias of 2^{-2} and eight input masks with a bias of 2^{-3}. The values of w's and their corresponding $\bar{y}$'s and $\bar{z}$'s (here, w, $\bar{y}$ and $\bar{z}$ form the masks together) are given in Table 3 in a group manner. Hence, we take all possible combinations of all values for $(\bar{y}, \bar{z})$ and obtain 600 linear approximations[2] depicted in Table 4. Note that the last pair of bit masks in Table 4 stands for the output masks after the swapping of half blocks in a round. These approximations are indeed generated by twelve base linear approximations. For example, one set of twelve base approximations is $(w, \bar{y}, \bar{z}) \in B = \{(2_x, 6_x, 6_x), (2_x, 6_x, b_x), (2_x, b_x, 6_x), (4_x, 9_x, e_x), (4_x, 9_x, 9_x), (4_x, e_x, 9_x), (8_x, b_x, d_x), (8_x, b_x, b_x), (8_x, d_x, b_x), (d_x, 8_x, 8_x), (d_x, 8_x, c_x), (d_x, c_x, 8_x)\}$. In detail, 24 of 600 linear approximations have biases 2^{-31}, 192 of them have biases 2^{-32} and 384 of them have biases 2^{-33}. Since, $c = 2|\varepsilon|$, they have respective correlations $c_1 = 2^{-30}$, $c_2 = 2^{-31}$ and $c_3 = 2^{-32}$. The capacity of the 12-dimensional system is lower bounded by

$$24 \times (2^{-30})^2 + 192 \times (2^{-31})^2 + 384 \times (2^{-32})^2 = 2^{-53.415}.$$

Note that we ignore the rest of $2^{12} - 601$ approximations as they have negligible correlations.

[2] These approximations contain the previous ones mentioned in Section 4.1.

Table 3. Possible values for w, $\bar{y}$ and $\bar{z}$.

w	$\bar{y}$, $\bar{z}$	bias
2_x	$6_x, b_x$	2^{-2}
	$2_x, 3_x, 4_x, 5_x, 8_x, 9_x, e_x, f_x$	2^{-3}
4_x	$9_x, e_x$	2^{-2}
	$2_x, 3_x, 4_x, 5_x, a_x, b_x, c_x, d_x$	2^{-3}
6_x	$1_x, 2_x$	2^{-2}
	$8_x, 9_x, a_x, b_x, c_x, d_x, e_x, f_x$	2^{-3}
8_x	b_x, d_x	2^{-2}
	$1_x, 3_x, 5_x, 7_x, 8_x, a_x, c_x, e_x$	2^{-3}
d_x	$8_x, c_x$	2^{-2}
	$2_x, 3_x, 6_x, 7_x, a_x, b_x, e_x, f_x$	2^{-3}
e_x	$4_x, c_x$	2^{-2}
	$1_x, 3_x, 5_x, 7_x, 9_x, b_x, d_x, f_x$	2^{-3}

4.3 A 12-dimensional Linear Attack on the 19 rounds of MIBS-80

We use all possible $2^{12} - 1$ (dependent and independent) linear approximations generated from the base approximations mentioned in Section 4.2. We attack on 19 rounds of MIBS-80 by using the convolution method [Her10] together with the (modified) Nguyen et al.'s approach [NWW11] (see Section 3.3). We recover some key bits from the last round key as well as the first and the second rounds' keys, by considering them as a combined ℓ-bit key. Note that the 600 linear approximations that we found in Section 4.2 are useful to compute the (lower bound of) capacity of the system which is used for finding the required number of the data for the attack.

We perform a key-recovery attack on 19 rounds of MIBS-80 by placing the 16-round linear approximations of 12-dimension between rounds 3 and 18. Because the capacity of the 12-dimensional linear approximation is $2^{-53.415}$, according to Theorem 4, the data requirement is $N = 2^{57.874}$ plaintext-ciphertext pairs by taking 10 bits of advantage, i.e. $a = 20$. According to Theorem 4 our attack's success probability is $P_S = 0.9$. We recover some part of the key bits from the first, second and the last rounds. We guess K_1 (except $K_{1,3}$), $K_{2,6}$, K_{19} (except $K_{19,3}$), which make 60 round key bits in total. Notice that the third S-Box of the first round is not active due to the fact that its output is not needed to compute the input to the first and the sixth S-Boxes in the second round. All four phases of the attack are explained as follows.

We call $T = P \oplus S$, composed of the S and the P layers of MIBS. Let $\alpha = 0w0w000w$, $\beta = 00\bar{y}0000\bar{z}$, where w, $\bar{y}$ and $\bar{z}$ take their values from the set B. Notice that any set of base approximations is acceptable. According to the set B, the combined coefficients (represented as 12-dimensional vectors) of the $2^{12} - 12 - 1$ approximations are also known.

Let $K' = (K_1, K_2, K_{19})$ contain 96 bits, $v = (L_0, R_0, R_{19})$ and $K' \oplus v = (K_1 \oplus L_0, K_2 \oplus R_0, K_{19} \oplus R_{19})$. Let us define $b_1 : \{0,1\}^{2^{96}} \to \{0,1\}^{2^{32}}$ such that

Table 4. A set of 16-round linear approximations, where "$*$" can be 2^{-3}, 2^{-4} or 2^{-5}, and w and z take the values as we mentioned in Section 4.1.

Round i	ΓL_{i-1}	ΓR_{i-1}	Number of active S-Boxes	Bias
1	$0w0w000w$	00000000	0	2^{-1}
2	00000000	$0w0w000w$	2	2^{-3}
3	$0w0w000w$	$00z0000z$	3	2^{-4}
4	$00z0000z$	$w0ww000w$	2	2^{-3}
5	$w0ww000w$	$z0zz000z$	2	2^{-3}
6	$z0zz000z$	$00w0000w$	3	2^{-4}
7	$00w0000w$	$0z0z000z$	2	2^{-3}
8	$0z0z000z$	00000000	0	2^{-1}
9	00000000	$0z0z000z$	2	2^{-3}
10	$0z0z000z$	$00w0000w$	3	2^{-4}
11	$00w0000w$	$z0zz000z$	2	2^{-3}
12	$z0zz000z$	$w0ww000w$	2	2^{-3}
13	$w0ww000w$	$00z0000z$	3	2^{-4}
14	$00z0000z$	$0w0w000w$	2	2^{-3}
15	$0w0w000w$	00000000	0	2^{-1}
16	00000000	$0w0w000w$	2	$*$
17	$0w0w000w$	$00\bar{y}0000\bar{z}$	-	-

$b_1(K' \oplus v) = K_1 \oplus L_0$. Similarly, define $b_2(K' \oplus v) = K_2 \oplus R_0$, and $b_3(K' \oplus v) = K_{19} \oplus R_{19}$. Then, the base and combined approximations are

$$h^a(P, C) \oplus g^a(K', (P, C)_\ell),$$

where

$$h^a(P, C) = \alpha_a \cdot L_0 \oplus \alpha_a \cdot R_{19} \oplus \beta_a \cdot L_{19},$$

and

$$\begin{aligned} g^a(K', (P, C)_\ell) = g^a(K' \oplus v) = \\ g'^a(b_1(K' \oplus v), b_2(K' \oplus v), b_3(K' \oplus v)) = \\ g'^a(K_1 \oplus L_0, K_2 \oplus R_0, K_{19} \oplus R_{19}) = \\ \alpha_a \cdot T(T(L_0 \oplus K_1) \oplus R_0 \oplus K_2) \oplus \beta_a \cdot T(R_{19} \oplus K_{19}) \end{aligned}$$

Here, $(P, C)_\ell$ denotes the ℓ text bits interacting with the ℓ-bit attacked key bits. According to α_a and β_a, we know only 64 bits of $K' \oplus v$ involved in the computation of g^a. Thus, hereafter when we mention $g^a(K' \oplus v)$, we mean that the function g^a acts directly on these 64 bits. That is, $K' \oplus v$ actually denotes the 64 active bits. Here, K' includes the 64 active key bits in K_1, K_2 and K_{19} and v is the 64 bits of texts.

The base approximations are denoted as $h^a(P, C) \oplus g^a(K', (P, C)_\ell)$, where $a = 1, \ldots, 12$. After examining the key schedule of MIBS-80, it can be verified that K_1 and K_2 share four attacked bits in common, that is $K_2[3 \sim 0] = K_1[22 \sim 19]$, where "$\sim$" indicates a sequence of bit positions. Therefore, while K' denotes the 64 key bits interacting with the text bits, K denotes the 60 key bits considering the key schedule.

Distillation Phase:

1. Collect $N = 2^{57.874}$ plaintext-ciphertext pairs (P_t, C_t), for $t = 1, \ldots, N$.
2. The table containing all empirical correlations $r[a, K]$ for each 60-bit key K can be computed as follows.
   (a) Construct $T^{2^{12} \times 2^{64}}$ in a way that for all $a \in \mathbb{F}_2^{12}$, for all $(P_t, C_t) = ({L_0}^t \| R_0^t, L_{19}^t \| R_{19}^t)$, $t = 1, \ldots, N$, if $h^a(P_t, C_t) = 0$, increment the counter $T[a, (P_t, C_t)_l]$, otherwise decrement it. Afterwards, update $T[a, (P_t, C_t)_l] = T[a, (P_t, C_t)_l]/N$. The time complexity of this is $N \times 2^{12}$ computations of h^a. The memory complexity for T is $2^{m+l} = 2^{76}$ block units, and hereafter each block unit denotes a size of $\log_2 N$ bits[3]. However, this table can be generated by constructing another table $E^{2^{12} \times 2^{64}}$ in the next step to reduce its time complexity.
   (b) Build another table $E^{2^{12} \times 2^{64}}$ by

$$E[z_1, z_2] = \#\{(P_t, C_t), t = 1, ..., N | (h^1(P_t, C_t), ..., h^{12}(P_t, C_t)) = z_1, \\ \text{and } (P_t, C_t)_l = z_2\}, \text{where } z_1 \in \mathbb{F}_2^{12}, z_2 \in \mathbb{F}_2^{64}.$$

   The time complexity of this step is $12 \times N \approx 2^{61.459}$ computations of h^a, i.e. two XOR operations. As one-round encryption is equivalent to more than seven XOR operations including the cost of S-Boxes, the above complexity can be regarded as $12 \times N \times 2/7 = 2^{59.652}$ one-round encryptions. The required memory complexity for E is $2^{m+l} = 2^{76}$ block units.
   (c) Now, build the table T from tables E and H. Let $H^{2^{12} \times 2^{12}}$ be a matrix such that $H[i, j] = (-1)^{i \cdot j}$, $\forall i, j \in \mathbb{F}_2^{12}$. For each fixed v, (that is, for each column vector in T), we can compute

$$T[a, v] = \sum_{z_1=0}^{2^{12}-1} (-1)^{a \cdot z_1} E[z_1, v]$$

   by the relation

$$T[\cdot, v] = HE[\cdot, v],$$

   with the time complexity 12×2^{12} multiplications. As there are 2^{64} columns in T, the total time complexity is $12 \times 2^{64+12} \approx 2^{79.585}$ multiplications, which is equal to $3/7 \times 2^{79.585} \approx 2^{78.363}$ one-round encryptions[4]. The memory complexity for H is 2^{22} bytes[5].

[3] Each entry in T is at most N, thus at most $\log_2 N$ bits are needed.

[4] We assume that three XORs correspond to one multiplication.

[5] Each entry in H is either 1 or -1.

(d) We can now construct the matrix $r^{2^{12} \times 2^{64}}$ having entries $r[a, K']$ from table T as

$$r[a, K'] = \sum_{v=0}^{2^{64}-1} (-1)^{g^a(K' \oplus v)} T[a, v].$$

Again, the memory complexity for r is about $2^{12+64} = 2^{76}$ block units. Then, for each $a \in \mathbb{F}_2^{12}$, compute S^a from g^a. Here, S^a has the size of $2^{64} \times 2^{64}$ and $S^a[i,j] = S^a[K', v] = (-1)^{g^a(K' \oplus v)}$. The time complexity of constructing each S^a is 3×2^{64} one-round MIBS encryptions. We will compute

$$r[a, \cdot] = S^a T[a, \cdot].$$

Note that each S^a is a circulant matrix. Thus, vector $r[a, \cdot]$ is calculated with $3 \times 64 \times 2^{64}$ multiplications. The total time complexity of computing r is $2^{12} \times 3 \times 64 \times 2^{64} = 2^{83.584}$ multiplications, that is, $3/7 \times 2^{83.584} \approx 2^{82.362}$ one-round encryptions. Due to the fact that S^a is a circulant matrix, only the first row is needed to be stored. Hence, the memory complexity is 2^{57} bytes.

(e) According to the common key bits in K' brought by the key schedule, K' actually has only 60 bits required to be guessed. Thus, we select the possible 2^{60} keys in r and eliminate the wrong keys based on the key schedule. We update $r[a, K']$ to $r[a, K]$ which only contains the columns for possible keys.

3. Finally, find $q[K, \cdot]$ for only key bits $K \in \mathbb{F}_2^{60}$ from the empirical correlations $r[a, K]$, by using Lemma 2. Hence, we have

$$q[K, \cdot] = 2^{-12} H r[\cdot, K],$$

where $H^{2^{12} \times 2^{12}}$ is exploited again, i.e. $H[i,j] = (-1)^{i \cdot j}$. The time complexity of computing the row vector of $q[K, \cdot]$ is 12×2^{12} multiplications. The total time complexity of this step is $2^{60} \times 12 \times 2^{12} \approx 2^{75.585}$ multiplications, i.e. $3/7 \times 2^{75.585} \approx 2^{74.363}$ one-round encryptions. Also, $2^{12+60} = 2^{72}$ block units are needed for storing q.

The total time complexity is computed by summing up all above steps. $1/19 \times (2^{59.652} + 2^{78.363} + 2^{82.362} + 2^{74.363}) \approx 2^{78.207}$ 19-round MIBS encryptions. The total memory complexity is 2^{76} block units.

Analysis Phase: We use the convolution method [Her10] explained in Section 3.2. As mentioned before, the necessary number of data for the attack is $2^{57.874}$ plaintext-ciphertext pairs. Since we obtain the empirical probability distribution from the observed data, and we can compute the theoretical probability distributions from 600 linear approximations by Lemma 2, we can directly apply the convolution method described in Algorithm 3. Then, we obtain the right key and the right inner key class by sorting their marks. The time needed for this phase is about $m2^{m+\ell} = 2^{75.585}$ multiplications, equivalently $3/7 \times 1/19 \times 2^{75.585} \approx 2^{70.115}$ 19-round encryptions, where $m = 12$ and $\ell = 60$.

Ranking and Search Phases: As we get a 10-bit advantage, and $2^{60-10} = 2^{50}$ candidate keys kept from the analysis phase are ranked according to their maximal marks with $2^{65.907}$ time complexity. Thus, we make $2^{50+20} = 2^{70}$ trial encryptions to find the correct 80-bit secret key.

To sum up, the total time and memory complexities of the attack are the sum of complexities of all phases (the complexity of the ranking phase can be ignored), namely $2^{78.207} + 2^{70.115} + 2^{70} + 2^{65.907} = 2^{78.217}$ 19-round MIBS encryptions and 2^{76} block units, respectively.

5 The Chosen-Plaintext Version of the Attack

The time and memory complexities of the attack on the reduced-round MIBS-80 detailed in Section 4 can be reduced by fixing some plaintext bits corresponding to the active S-Boxes like in [KM01]. The main reason is that we do not need to guess the key bits corresponding to the fixed data, as any output parity of the S-Box will always be fixed, that is, it is always 0 or 1. Due to the fact that we need $2^{57.874}$ number of plaintext-ciphertext pairs for the attack, we have the freedom of fixing 4-bits of plaintexts corresponding to the one S-box. Let us consider 4 fixed plaintext bits for the first active S-Box of the first round. Hence, the number of guessed key bits becomes $\ell = 56$ bits instead of $\ell = 60$ bits. By updating the previous attack according to this method with using the same number of data, we obtain $2^{74.228}$ 19-round encryptions of time and 2^{72} block units of memory.

6 Conclusion

This paper proposes a multidimensional linear cryptanalysis on the reduced-round MIBS-80. The attack is faster than the previous linear attack, also requires less data complexity thanks to exploiting many linear approximations. As far as we know, the result proposed in this paper is the best cryptanalytic result for MIBS-80, so far.

References

[BCQ04] Alex Biryukov, Christophe De Cannière, and Michaël Quisquater. On Multiple Linear Approximations. In *CRYPTO'04*, volume 3152 of *Lecture Notes in Computer Science*, pages 1–22. Springer, 2004.

[BJV04] Thomas Baignères, Pascal Junod, and Serge Vaudenay. How Far Can We Go Beyond Linear Cryptanalysis? In Pil Joong Lee, editor, *ASIACRYPT'04*, volume 3329 of *Lecture Notes in Computer Science*, pages 432–450. Springer, 2004.

[BNV10] Aslı Bay, Jorge Nakahara, and Serge Vaudenay. Cryptanalysis of Reduced-Round MIBS Block Cipher. In Swee-Huay Heng, Rebecca N. Wright, and Bok-Min Goi, editors, *CANS'10*, volume 6467 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2010.

[CHN09] Joo Yeon Cho, Miia Hermelin, and Kaisa Nyberg. A New Technique for Multidimensional Linear Cryptanalysis with Applications on Reduced Round Serpent. In Pil Joong Lee and Jung Hee Cheon, editors, *ICISC'08*, volume 5461 of *Lecture Notes in Computer Science*, pages 383–398. Springer, 2009.

[CSQ07] Baudoin Collard, François-Xavier Standaert, and Jean-Jacques Quisquater. Improving the Time Complexity of Matsui's Linear Cryptanalysis. In Kil-Hyun Nam and Gwangsoo Rhee, editors, *ICISC'07*, volume 4817 of *Lecture Notes in Computer Science*, pages 77–88. Springer, 2007.

[HCN08] Miia Hermelin, Joo Yeon Cho, and Kaisa Nyberg. Multidimensional Linear Cryptanalysis of Reduced Round Serpent. In Yi Mu, Willy Susilo, and Jennifer Seberry, editors, *ACISP'08*, volume 5107 of *Lecture Notes in Computer Science*, pages 203–215. Springer, 2008.

[HCN09] Miia Hermelin, Joo Yeon Cho, and Kaisa Nyberg. Multidimensional Extension of Matsui's Algorithm 2. In Orr Dunkelman, editor, *FSE'09*, volume 5665 of *Lecture Notes in Computer Science*, pages 209–227. Springer, 2009.

[Her10] Miia Hermelin. Multidimensional linear cryptanalysis. 2010.

[HN10] Miia Hermelin and Kaisa Nyberg. Dependent Linear Approximations: The Algorithm of Biryukov and Others Revisited. In Josef Pieprzyk, editor, *CT-RSA'10*, volume 5985 of *Lecture Notes in Computer Science*, pages 318–333. Springer, 2010.

[HN11] Miia Hermelin and Kaisa Nyberg. Linear Cryptanalysis Using Multiple Linear Approximations. In *IACR Cryptology ePrint Archive*, volume 2011, 2011.

[HN12] Miia Hermelin and Kaisa Nyberg. Multidimensional linear distinguishing attacks and Boolean functions. In *Cryptography and Communications*, volume 4, pages 47–64, 2012.

[ISSK09] Maryam Izadi, Babak Sadeghiyan, Seyed Saeed Sadeghian, and Hossein Arabnezhad Khanooki. MIBS: A New Lightweight Block Cipher. In *CANS'09*, volume 5888 of *Lecture Notes in Computer Science*. Springer, 2009.

[JR94] Burton S. Kaliski Jr. and Matthew J. B. Robshaw. Linear Cryptanalysis Using Multiple Approximations. In Yvo Desmedt, editor, *CRYPTO '94*, volume 839 of *Lecture Notes in Computer Science*, pages 26–39. Springer, 1994.

[KM01] Lars R. Knudsen and John Erik Mathiassen. A Chosen-Plaintext Linear Attack on DES. In Bruce Schneier, editor, *FSE'00*, volume 1978 of *Lecture Notes in Computer Science*, pages 262–272. Springer, 2001.

[Mat94a] Mitsuru Matsui. The First Experimental Cryptanalysis of the Data Encryption Standard. In Yvo Desmedt, editor, *CRYPTO'94*, volume 839 of *Lecture Notes in Computer Science*, pages 1–11. Springer, 1994.

[Mat94b] Mitsuru Matsui. Linear Cryptoanalysis Method for DES Cipher. In Tor Helleseth, editor, *EUROCRYPT'93*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer, 1994.

[Mur06] S. Murphy. The Independence of Linear Approximations in Symmetric Cryptanalysis. *IEEE Transactions on Information Theory*, 52(12):5510–5518, 2006.

[MY93] Mitsuru Matsui and Atsuhiro Yamagishi. A New Method for Known Plaintext Attack of FEAL Cipher. In Rainer A. Rueppel, editor, *EUROCRYPT'92*, volume 658 of *Lecture Notes in Computer Science*, pages 81–91. Springer, 1993.

[NWW11] Phuong Ha Nguyen, Hongjun Wu, and Huaxiong Wang. Improving the Algorithm 2 in Multidimensional Linear Cryptanalysis. In Udaya Parampalli and Philip Hawkes, editors, *ACISP'11*, volume 6812 of *Lecture Notes in Computer Science*, pages 61–74. Springer, 2011.

A Proof of Lemma 5

For the sake of understanding, we use the same notations as that of Nguyen's paper [NWW11].

Proof. According to Lemma 2, we can compute the empirical probability distribution of m-dimensional linear approximations from their one-dimensional empirical correlations. We now define a matrix $r^{2^m \times 2^\ell}$ composed of the correlations of all g^a's, for $a \in \mathbb{F}_2^m$ and $K \in \mathbb{F}_2^\ell$. Notice that $r[a, K] = 1$, for all $K \in \mathbb{F}_2^\ell$ if $a = (0, \ldots, 0)$. Otherwise, we compute $r[a, K]$ by constructing a table $T^{2^m \times 2^\ell}$. This table T is built as follows: for all $a \in \mathbb{F}_2^m$, we increment the counter $T[a, (x_t, y_t')_\ell]$, where $(x_t, y_t')_\ell$ denotes the ℓ-bits of plaintexts and ciphertexts interacting with attacked key bits from K_1 and K_r, if $f_1^a(x_t, y_t') = 0$, otherwise decrement it. Then, update $T[a, K] = T[a, K]/N$.

However, the table T can be built in a more efficient way by providing another table $E^{2^m \times 2^\ell}$ which is

$$E[h_1, h_2] = \#\{(x_t, y_t'),\, t = 1, \ldots N | (f_1^1(x_t, y_t'), ..., f_1^m(x_t, y_t')) = h_1, (x_t, y_t')_\ell = h_2\},$$

for all $h_1 \in \mathbb{F}_2^m$, for all $h_2 \in \mathbb{F}_2^\ell$. The time complexity of building E is mN. The memory complexity required for E is 2^{m+l} block units, here the size of each block unit is $\log_2 N$, as the possible biggest value stored in E is N. Now, we build the table T from tables E and H, where $H^{2^m \times 2^m}$ is a Hadamard matrix such that $H[i, j] = (-1)^{i \cdot j}$ for $i, j \in \mathbb{F}_2^m$. Hence, $T[\cdot, v] = HE[\cdot, v]$, for each fixed v, we have

$$T[a, v] = \sum_{h_1=0}^{2^m-1} (-1)^{a \cdot h_1} E[h_1, v].$$

As H is a Hadamard matrix, for each column $v \in \mathbb{F}_2^\ell$, $T[\cdot, v]$ is obtained by $m2^m$ time complexity. Thus, for the whole T, the time complexity is $m2^{\ell+m}$. The memory complexity for H is 2^{2m-7} bytes.

We now write

$$r[a, K] = \sum_{v=0}^{2^\ell-1} (-1)^{f^a(K \oplus v)} T[a, K], \qquad a \in \mathbb{F}_2^m.$$

Here, $f^a = f_2^a \oplus f_3^a$. We note that only ℓ bits from x and y' are used in f^a. Now, we define another $2^\ell \times 2^\ell$ table S^a depending on a for all $i, j \in \mathbb{F}_2^\ell$ such that $S^a[i, j] = (-1)^{f^a(i \oplus j)}$. The time complexity of constructing all S^a is 2^ℓ because

S^a is a circulant matrix and the active S-Boxes in our case are the same. Hence, we get

$$r[a,\cdot] = S^a T[a,\cdot].$$

Due to the fact that S^a is a *circulant* matrix (see [CSQ07]), $r[a,\cdot]$ is computed in $3\ell 2^\ell$ operations with 2^ℓ bytes to store the first row of S^a. The total time complexity of computing r is $3\ell 2^{\ell+m}$ multiplications. Finally, by using Lemma 2, we get $q[K,\cdot] = 2^{-m} H r[\cdot, K]$ with $m2^m$ time complexity. The total time complexity of computing q is $m2^{\ell+m}$. The memory complexity for q is $2^{m+\ell}$ block units.

By summing up all above steps, the time complexity is $\mathcal{O}(mN + (2m + 3\ell)2^{\ell+m} + 2^\ell)$. Here, we use $\mathcal{O}$ notation because the exact value depends on the detailed time unit. According to the size of the used matrices, the memory complexity is $2^{m+\ell}$ block units.

B Linear Approximation Table (LAT) of the S-Box in MIBS

	0_x	1_x	2_x	3_x	4_x	5_x	6_x	7_x	8_x	9_x	a_X	b_x	c_x	d_x	e_x	f_x
0_x	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1_x	0	-2	0	2	0	-2	-4	-2	2	0	-2	0	2	0	2	-4
2_x	0	0	-2	-2	-2	2	-4	0	0	4	2	-2	-2	-2	0	0
3_x	0	2	2	0	2	0	0	2	-2	4	0	2	0	2	-2	-4
4_x	0	-2	-2	4	-2	0	0	2	0	-2	2	0	-2	0	-4	-2
5_x	0	0	-2	2	2	-2	0	0	2	2	0	4	-4	0	2	2
6_x	0	-2	4	2	0	-2	0	-2	0	2	4	-2	0	2	0	2
7_x	0	4	0	0	0	-4	0	0	-2	-2	2	-2	-2	-2	2	-2
8_x	0	2	2	4	0	2	-2	0	-2	0	0	2	2	-4	0	2
9_x	0	0	2	-2	-4	-4	-2	2	0	0	-2	2	0	0	-2	2
a_x	0	-2	0	-2	-2	0	2	-4	-2	0	2	4	0	-2	0	-2
b_x	0	0	4	0	-2	2	2	2	4	0	0	0	-2	-2	2	-2
c_x	0	0	0	0	2	-2	2	-2	2	2	-2	-2	0	-4	-4	0
d_x	0	2	0	-2	2	0	-2	0	4	-2	4	2	2	0	-2	0
e_x	0	4	-2	2	-4	0	2	-2	2	2	0	0	2	2	0	0
f_x	0	2	2	0	0	2	-2	-4	0	-2	-2	0	-4	2	-2	0

C Previous 16-round Linear Approximations

Round i	ΓL_{i-1}	ΓR_{i-1}	Number of active S-Boxes	Bias
1	$0w0w000w$	00000000	0	2^{-1}
2	00000000	$0w0w000w$	2	2^{-3}
3	$0w0w000w$	$00z0000z$	3	2^{-4}
4	$00z0000z$	$w0ww000w$	2	2^{-3}
5	$w0ww000w$	$z0zz000z$	2	2^{-3}
6	$z0zz000z$	$00w0000w$	3	2^{-4}
7	$00w0000w$	$0z0z000z$	2	2^{-3}
8	$0z0z000z$	00000000	0	2^{-1}
9	00000000	$0z0z000z$	2	2^{-3}
10	$0z0z000z$	$00w0000w$	3	2^{-4}
11	$00w0000w$	$z0zz000z$	2	2^{-3}
12	$z0zz000z$	$w0ww000w$	2	2^{-3}
13	$w0ww000w$	$00z0000z$	3	2^{-4}
14	$00z0000z$	$0w0w000w$	2	2^{-3}
15	$0w0w000w$	00000000	0	2^{-1}
16	00000000	$0w0w000w$	2	2^{-3}
17	$0w0w000w$	$00z0000z$	-	-