Impossible Differential Cryptanalysis of Safer++

Behnam Bahrak*, Taraneh Eghlidos†, Mohammad Reza Aref*
*Information System and Security Lab. (ISSL), Dept. of Electrical Engineering,
Sharif University of Technology, Tehran, Iran.
†Electronics Research Center,
Sharif University of Technology, Tehran, Iran.
bahrak@ee.sharif.edu, {teghlidos,aref}@sharif.edu

***Abstract*— In this paper, we describe an impossible differential property for 2.5 rounds of Safer++. It allows an impossible differential attack on 4 rounds of Safer++. The proposed attack requires 2^{23} chosen plaintexts and 2^{75} bytes of memory and performs 2^{84} 4-round Safer++ encryptions. The method developed to attack Safer++ can be applied to other block ciphers in Safer family.**[1]

I. INTRODUCTION

Safer++ is an iterated block cipher with variable key lengths of 128 and 256 bits and fixed block length of 128 bits [1]. The 128-bit key version of this cipher is a 7-round substitution-permutation network (SPN). Safer++ was submitted to the European pre-standardization project NESSIE [2] and was among the primitives selected for the second phase of this project.

SAFER was introduced by Massey in 1993 [3] and was analyzed in [4,5,6,7,8]. The weaknesses of this cipher result in several ciphers in this family: SAFER-K, SAFER-SK, SAFER+ and finally SAFER++. All these ciphers use the same S-boxes and share Pseudo-Hadamard-like mixing transforms.

Impossible differential attack [9] is a variant of differential cryptanalysis [10] which looks for differentials that hold with zero probability (or impossible differentials) to eliminate wrong keys and keep the right key. Safer++ was intensively analyzed in [11,12,13,14]. In this paper we study the security of 128-bit key version of this algorithm against the impossible differential attack. We use some weaknesses in Safer++ S-boxes and its diffusion property to apply an attack against 4-round variant of this cipher. The proposed attack requires 2^{23} chosen plaintexts and performs 2^{84} 4-round Safer++ encryptions.

The paper is organized as follows: In section 2 we briefly describe the Safer++ algorithm. Section 3 shows some properties of Safer++ round components. A new impossible differential property of Safer++ is introduced in section 4. In section 5, we propose an impossible differential attack on 4-round Safer++. Finally, section 6 concludes the paper.

II. A BRIEF DESCRIPTION OF SAFER++

This section includes a brief description of Safer++. You can refer to [1] for more details. Safer++ is an iterated block cipher in which every round consists of four layer: an upper key layer, a nonlinear layer, a lower key layer and a linear layer. The structure of one Safer++ round is illustrated in Figure 1. After the final round there is an output transformation which is similar to the upper key layer.

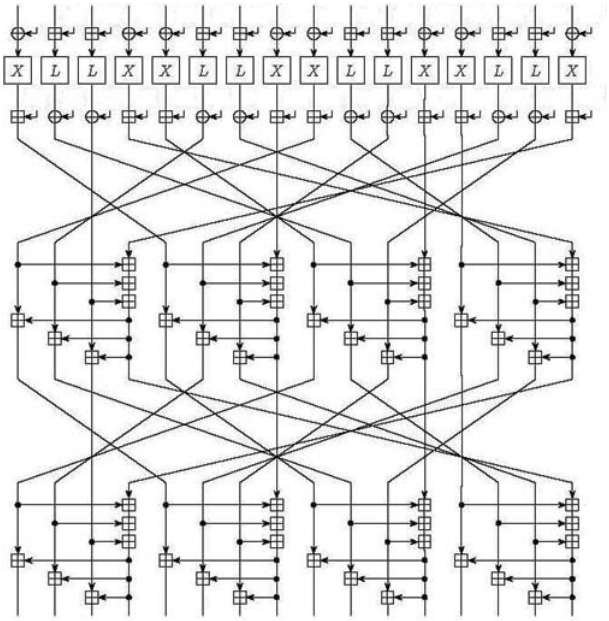

Fig. 1. One round encryption of Safer++

A. The Upper Key Layer

The upper key layer mixes a 16-byte subkey with the 16-byte block. Bytes 1, 4, 5, 8, 9, 12, 13 and 16 of the subkey are XORed to the corresponding bytes of the block and bytes 2, 3, 6, 7, 10, 11, 14 and 15 are combined using addition modulo 256.

B. The Nonlinear Layer

The nonlinear layer uses two bytewise functions: the exponential function (X) and the logarithmic function (L). X and L are defined as:

$$X(a) = (45^a \ mod\ 257)\ mod\ 256$$
$$L(a) = \log_{45}^{(a)} \ mod\ 257 \quad and\ L(0) = 128$$

[1]This work was partially supported by Iran Telecommunications Research Center and the cryptography chair of Iranian NSF.

It can be easily checked that X and L are mutually inverse. In the nonlinear layer, bytes 1, 4, 5, 8, 9, 12, 13 and 16 are applied to the function X, and other bytes are sent through the function L.

C. The Lower Key Layer

The lower key layer mixes a 16-byte subkey with the 16-byte block. Bytes 2, 3, 6, 7, 10, 11, 14 and 15 of the subkey are XORed to the corresponding bytes of the block and bytes 1, 4, 5, 8, 9, 12, 13 and 16 are combined using addition modulo 256.

D. The Linear Layer

The linear transformation uses a 4-point Pseudo Hadamard Transform (4-PHT) and a permutation. The linear layer first change the order of input bytes and then applies the 4-PHT to groups of four bytes. The output of the linear layer is obtained after iterating this operation twice. The linear layer and its inverse can be represented by matrices A and A^{-1} in $GF(2^8)$. These matrices are shown in Figure 2.

$$A = \begin{pmatrix}
1 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 4 & 2 & 2 & 2 & 1 & 1 & 2 & 1 \\
2 & 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 1 & 2 & 4 & 2 & 2 \\
2 & 2 & 4 & 2 & 2 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 2 & 1 & 2 & 1 & 1 & 1 \\
4 & 2 & 2 & 2 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 \\
1 & 1 & 2 & 1 & 2 & 1 & 1 & 1 & 2 & 4 & 2 & 2 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 2 & 2 & 4 & 2 & 2 & 1 & 1 & 1 \\
1 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 1 & 2 & 1 \\
1 & 1 & 2 & 1 & 4 & 2 & 2 & 2 & 1 & 2 & 1 & 1 & 1 & 1 & 1 & 1 \\
1 & 1 & 1 & 1 & 2 & 4 & 2 & 2 & 1 & 1 & 2 & 1 & 2 & 1 & 1 & 1 \\
1 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 2 & 2 & 4 & 2 \\
2 & 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 \\
1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 2 & 1 & 4 & 2 & 2 & 2 \\
2 & 4 & 2 & 2 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 1 & 2 & 1 \\
2 & 1 & 1 & 1 & 2 & 2 & 4 & 2 & 1 & 1 & 1 & 1 & 1 & 2 & 1 & 1 \\
1 & 1 & 2 & 1 & 2 & 1 & 1 & 1 & 1 & 2 & 1 & 1 & 1 & 1 & 1 & 1
\end{pmatrix}$$

$$A^{-1} = \begin{pmatrix}
0 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & -1 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & -4 & 0 & 0 & 1 & -1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & -4 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & -1 \\
0 & 0 & -1 & 16 & -1 & 0 & -4 & 1 & 0 & -4 & 0 & 1 & -4 & -1 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & -1 \\
1 & 0 & 0 & -1 & 0 & 0 & 0 & -4 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & -4 & 0 & 0 & 1 & -1 & 0 & 1 & 1 & 0 \\
-4 & 0 & 0 & 1 & 0 & 0 & 0 & 16 & -1 & -1 & -4 & 1 & 0 & -4 & -1 & 1 \\
1 & 1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & -4 & 0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & -4 & 0 & 0 & 1 & -1 \\
0 & 1 & 0 & -1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & -4 & 0 & 0 & 1 & 0 \\
-1 & -4 & 0 & 1 & -4 & -1 & -1 & 1 & 0 & 0 & 0 & 16 & 0 & 0 & -4 & 1 \\
0 & 0 & 1 & -1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & -4 \\
0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & -4 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & -1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & -4 \\
0 & -1 & -4 & 1 & 0 & -4 & 0 & 1 & -4 & 0 & -1 & 1 & -1 & 0 & 0 & 16
\end{pmatrix}$$

Fig. 2. Matrices of the linear transformation

E. The Key Schedule

The key schedule expands the 16-byte master key into the required number of subkeys. Let $K = [k_{1,1}, k_{1,2}, ..., k_{1,16}]$ denotes the master key. The key schedule generates fifteen 16-byte subkeys (two subkeys for each of rounds and an additional subkey for the output transformation). These subkeys are generated as follows:

- The first subkey is equal to the master key: $K_1 = K$
- Let $k_{1,17} = k_{1,1} \oplus k_{1,2} \oplus ... \oplus k_{1,16}$
- For $i \geq 2$
  $k_{i,j} = k_{i-1,j} \lll 3$
  $K_{i,j} = k_{i,(i+j-2 \ mod \ 17)+1} + B_{i,j} \ mod \ 256$
  $K_i = [K_{i,1}, K_{i,2}, ..., K_{i,16}]$

In the above expressions, $B_{i,j}$ is a known constant and $\lll 3$ denotes cyclic shift to the left by 3 bits.
It is obvious that $K_{i,j}$ can be computed as a function of $k_{1,(i+j-2 \ mod \ 17)+1}$.

III. Properties of the Components

A. Diffusion in the linear layer

The linear layer guarantees that a single active byte (a byte with nonzero difference) at the input of the layer will cause at least ten active bytes at the output [6], but it has been proved that the branch number of the linear transformation is 5 [13], i.e. we can fine an input with two active bytes which results in an output with only three active bytes. For example if the input difference is $\Delta In = (0, 0, 128, 128, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$, the output difference will be $\Delta Out = (0, 0, 0, 0, 0, 128, 0, 0, 128, 0, 0, 0, 0, 0, 0, 128)$.

B. Properties of the S-boxes

Safer++ S-boxes have some interesting mathematical properties. It is not hard to see that the following expressions hold for X and L functions [5]:

$$X(a) + X(a + 128) = 1 \qquad (3.2.1)$$
$$L(a) - L(1 - a) = 128 \qquad (3.2.2)$$

In above expressions '+' and '-' represents addition and subtraction in modulo 256 respectively.

C. Key Mixture Property

It can be easily shown that if '+' defines the addition in $GF(2^n)$ and $\oplus$ defines the XOR operation, then we will have :

$$\forall x \in GF(2^n) : x \oplus 2^{n-1} = x + 2^{n-1}$$

In the case of Safer++, we have: $\forall x \in GF(2^8) : x \oplus 128 = x + 128$. Using the above equation, we can see that if the difference of two numbers in $GF(2^8)$ is 128, after XORing these numbers with a fixed k, the difference of them will remain 128.

IV. Impossible Differential Property of Safer++

A. Notations

In this paper we use the following notations: X_i^I denotes the input block of the i-th round, while X_i^U, X_i^S, X_i^L and X_i^O denote the intermediate value after applying the upper key layer, the nonlinear layer, the lower key layer and the linear layer in round i, respectively. Obviously $X_{i-1}^O = X_i^I$ holds for $i \geq 2$. We denote the i-th subkey by K_i, i.e. in round j we use K_{2j-1} and K_{2j} as upper and lower subkeys. Δ represents the difference in $GF(2^8)$. 'od' and 'ev' denote odd numbers

and even numbers, respectively and '?' represents an unknown value. We also denote the j-th byte of X_i by $X_{i,j}$ where $j \in \{1, 2, ..., 16\}$.

B. 2.5-Round Impossible Differential Property

In [14] a 1.75-round impossible differential property was presented. In this section we propose a new 2.5-round impossible differential property which our attack is based on.

Theorem 4.1: If X_1^S has the difference $\Delta X_1^S = (0, 0, 128, 128, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$, then X_3^O can not have the difference $\Delta X_3^O = (?, -(b + 4c + d - e), ?, ?, 128, b, ?, ?, ?, ?, ?, c, ?, ?, d, e)$ where b, c, d and e are known values.

Proof: Let $\Delta X_1^S = (0, 0, 128, 128, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$. Because of the key mixture property (see 3.3) we have $\Delta X_1^L = \Delta X_1^S$, which leads to $\Delta X_1^O = A \times \Delta X_1^L = (0, 0, 0, 0, 0, 128, 0, 0, 128, 0, 0, 0, 0, 0, \ 0, 128) = \Delta X_2^I$ (the matrix A represents the linear transformation). Again using key mixture property we have $\Delta X_2^U = \Delta X_2^I$. According to (3.2.1), we know that if inputs of the function X have the difference 128, the sum of the outputs in modulo 256 will be 1 and because 256 is even, these outputs should have an odd difference, otherwise their sum can not be odd in modulo 256. Hence after applying S-boxes in second round we have: $\Delta X_2^S = (0, 0, 0, 0, 0, ?, 0, 0, od, 0, 0, 0, 0, 0, 0, od)$ and because key mixture has no influence on being even or odd: $\Delta X_2^L = (0, 0, 0, 0, 0, ?, 0, 0, od, 0, 0, 0, 0, 0, 0, od)$. According to entries of matrix A, we conclude $\Delta X_2^O = A \times \Delta X_2^L = (?, ?, ?, ev, ?, ?, od, ?, ev, ev, ?, ?, od, ?, ev, ?)$.

In the reverse direction, suppose $\Delta X_3^O = (?, -(b + 4c + d - e), ?, ?, 128, b, ?, \ ?, ?, ?, ?, c, ?, ?, d, e)$. So we have $\Delta X_3^L = A^{-1} \times \Delta X_3^O = (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, 128, ?, ?, ?, ?, ?, ?)$, which leads to $\Delta X_3^S = (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, 128, ?, ?, \ \ ?, ?, ?, ?)$ (key mixture property). Because of (3.2.2) and by a discussion similar to the previous part, we know that if outputs of the function L has the difference 128, the inputs should have an odd difference, so $\Delta X_3^U = (?, ?, ?, ?, ?, ?, ?, ?, ?, ?, od, ?, ?, ?, ?, ?, ?)$ and as a result $\Delta X_3^I = (?, ?, ?, ?, ?, ?, ?, \ ?, ?, ?, od, ?, ?, ?, ?, ?, ?)$ (key mixture has no influence on being even or odd). $\Delta X_{3,10}^I$ is odd and $\Delta X_{2,10}^O$ is even, but that is impossible because $X_3^I = X_2^O$. ■

V. Impossible Differential Attack on 4-Round Safer++

Using the above impossible differential property, we can attack a 4-round variant of Safer++. Figure 3 illustrates the attack.

A. The attack procedure

In order to decrease data complexity, we use *'structures'*. Here by structure we mean a set of 2^{16} plaintexts

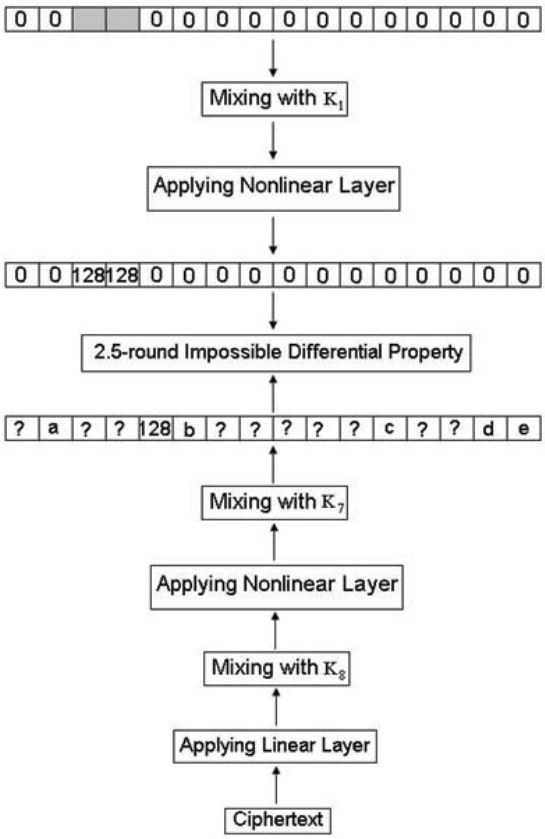

Fig. 3. Impossible differential attack on 4-round Safer++

which have fixed values in all but two bytes 3 and 4. Such a structure proposes $2^{16} \times (2^{16} - 1) \times \frac{1}{2} \approx 2^{31}$ pairs of plaintexts. The procedure of this attack is as follows:

Step 1. Take 2^n structures (i.e. $2^n \times 2^{16} = 2^{n+16}$ plaintexts, so $2^n \times 2^{31} = 2^{n+31}$ plaintext pairs). Ask for the encryptions of structures by 4-round Safer++. Apply the inverse linear transformation on all 2^{n+16} ciphertexts, i.e. compute $X_4^L = A^{-1} \times C$ for all of them.

Step 2. Guess the 8-bit value for $K_{8,5}$ (fifth byte of the lower subkey in round 4) and compute $X_{4,5}^U = L(X_{4,5}^L - K_{8,5} \ mod \ 256)$ for all pairs. Choose pairs whose difference $\Delta X_{4,5}^U$ are 128. The probability of such a difference is 2^{-8} and consequently we expect to have $2^{n+31} \times 2^{-8} = 2^{n+23}$ pairs under this condition.

Step 3. Guess the 40-bit value at bytes 2, 6, 12, 15 and 16 for the K_8. Decrypt partially these bytes in the last round, i.e. compute $X_4^U[2, 6, 15] = X(X_4^L[2, 6, 15] \oplus K_8[2, 6, 15])$ and $X_4^U[12, 16] = L(X_4^L[12, 16] - K_8[12, 16] \ mod \ 256)$.

Step 4. Guess the 8-bit value for $K_{7,12}$. According to the Safer++ key schedule, we can compute $K_{7,16}$ from the guessed value for $K_{8,15}$ in step 3, because both of them are functions of $k_{1,5}$. For all remaining pairs compute $X_3^O[12, 16] = X_4^U[12, 16] \oplus K_7[12, 16]$ and find ΔX_3^O in bytes 12 and 16. Note that in bytes 2, 6 and 15 the subkey is added in modulo 256, so we

have $\Delta X_3^O = \Delta X_4^L$ in these bytes. On the other hand $\Delta X_{3,5}^O = \Delta X_{4,5}^L = 128$ because of the key mixture property. So we have ΔX_3^O in bytes (2,5,6,12,15,16) for all pairs. Choose pairs whose difference ΔX_3^O have the following property:
$(\Delta X_{3,2}^O + \Delta X_{3,6}^O - 4\Delta X_{3,12}^O + \Delta X_{3,15}^O - \Delta X_{3,16}^O)\ mod\ 256 = 0$
The above expression holds with the probability of 2^{-8}, so we expect to have $2^{n+23} \times 2^{-8} = 2^{n+15}$ pairs under this condition.

Step 5. In this step, we eliminate wrong 16-bit values at bytes 3 and 4 for the K_1 by showing that the impossible differential property holds, if these keys were used. The probability of a wrong 16-bit value at bytes 3 and 4 for K_1 is $(1-2^{-16})$, so after analyzing all 2^{n+15} remaining pairs, we expect only $N = 2^{16} \times (1-2^{-16})^{2^{n+15}}$ wrong values of the two bytes of K_1 remain. For $n = 7$, the expected number is about $2^{16} \times e^{-64} \approx 2^{-74}$ and we can expect that only the right subkey remains. Unless the initial guess of the 56-bit value of the last round key K_8 and K_7 are correct, it is expected that we can eliminate the whole 16-bit value of K_1 in this step. Since the wrong values for K_8, K_7, K_1 occur with the small probability of $(2^8)^7 \times 2^{-74} = 2^{-18}$, if a 16-bit value remains for K_1, we can assume that the guessed 56-bit value for K_8 and K_7 are correct.

B. The attack complexity

According to step 1 and because we need to have $n = 7$, the data complexity of the attack is 2^{23} chosen plaintexts. The time complexity of the attack is consisted of five parts:
Step 1 requires $2^{n+16} \times \frac{1}{4} = 2^{n+14}$ one round encryptions, because for each of 2^{n+16} ciphertexts, we should apply one of four round operations.

Step 2 requires $2 \times 2^8 \times 2^{n+31} \times \frac{1}{16} \times \frac{2}{4} = 2^{n+35}$ one round encryptions, because for each of 2^8 guessed keys (for byte $K_{8,5}$), we should apply two operations on one of 16 bytes.

Step 3 requires $2 \times 2^8 \times 2^{40} \times 2^{n+23} \times \frac{5}{16} \times \frac{2}{4} = 2^{n+69}$ one round encryptions, because for each of 2^8 guessed keys in step 2, we should guess 2^{40} values in this step and for all of these keys we should apply two operations on five bytes of the block.

Step 4 requires $2 \times 2^{48} \times 2^8 \times 2^{n+23} \times \frac{2}{16} \times \frac{1}{4} = 2^{n+75}$ one round encryptions, because for each of the 2^{48} guessed keys in previous steps, we should guess 2^8 keys in this step and apply one operation on two bytes of the block.

Step 5 requires $2 \times 2^{56} \times 2^{16} \times \left\{1 + (1-2^{-16}) + ... + (1-2^{-16})^{2^{n+15}}\right\} \times \frac{2}{16} \times \frac{1}{4} \approx 2^{86}$ one round encryptions, because for each of the 2^{56} guessed keys in previous steps, we should analyze 2^{16} keys in this step. On the other hand, in each iteration of this step we should analyze $N \times (1-2^{-16})$ where N is the number of remaining keys from previous iteration. Analyzing a key in this step consists of applying two operations on two bytes of each plaintext pairs.
Consequently for $n = 7$, the overall complexity of the attack on 4-round Safer++ is about $\frac{2^{21}+2^{42}+2^{76}+2^{82}+2^{86}}{4} \approx 2^{84}$ encryptions. We can find 9 bytes of the master key by this attack. Other bytes of the master key can be found by a simple exhaustive search, so the whole key can be recovered in time complexity of $2^{84} + (2^8)^7 \approx 2^{84}$ encryptions. Meanwhile, $2^{72} \times 9 \approx 2^{75}$ bytes of memory are needed to store the list of deleted key values for the attack.

VI. Conclusion

We have proposed a new impossible differential attack against Safer++ reduced to 4 rounds. As shown in Table 1, the performance of this attack is much better than the previous impossible differential attack [14] and it is because of the efficiency of the new impossible differential property. In comparison with other attacks, we can see that this attack requires the least number of chosen plaintexts for breaking the same number of rounds and its time complexity is better than most of them.

TABLE I
Comparison of our results with other chosen plaintext attacks on Safer++

Rounds	Data	Workload	Type	Reference
2.75	2^{64}	2^{60}	Imp. Diff.	[14]
4	2^{23}	2^{84}	Imp. Diff.	This paper
3	2^{81}	2^{101}	Linear	[12]
4	2^{48}	2^{70}	Multiset	[11]
4.5	2^{48}	2^{94}	Multiset	[11]
4	2^{64}	2^{112}	Integral	[13]

References

[1] J.L.Massey, G. H.Khachatrian, and M. K.Kuregian, "Nomination of SAFER++ as candidate algorithm for the New European Schemes for Signatures, Integrity and Encryption (NESSIE)" Primitive submitted to NESSIE by Cylink Corp., Sept. 2000.

[2] NESSIE Project New European Schemes for Signatures, Integrity and Encryption. http://cryptonessie.org.

[3] J. L. Massey, "SAFER K-64: A byte-oriented block-ciphering algorithm" in Fast Software Encryption, FSE93 (R. J. Anderson, ed.), vol. 809 of Lecture Notes in Computer Science, pp. 117, Springer-Verlag, 1994.

[4] J. Kelsey, B. Schneier, and D. Wagner, "Key-schedule cryptanalysis of 3-WAY, IDEA, G-DES, RC4, SAFER, and Triple-DES" in Advances in Cryptology CRYPTO96 (N. Koblitz, ed.), vol. 1109 of Lecture Notes in Computer Science, pp. 237251, Springer-Verlag, 1996.

[5] L. R. Knudsen,"A detailed analysis of SAFER K", Journal of Cryptology, vol. 13, no. 4, pp. 417436, 2000.

[6] J. L.Massey, "On the optimality of SAFER+ diffusion" in Proceedings of the Second AES Candidate Conference, National Institute of Standards and Technology, Mar. 1999.

[7] S. Murphy, "An analysis of SAFER", Journal of Cryptology, vol. 11, no. 4, pp. 235251, 1998.

[8] J. Nakahara Jr, B. Preneel, and J. Vandewalle, ”Linear cryptanalysis of reduced- round versions of the SAFER block cipher family” in Fast Software Encryption, FSE 2000 (B. Schneier, ed.), vol. 1978 of Lecture Notes in Computer Science, pp. 244261, Springer-Verlag, 2001.

[9] E. Biham, A. Biryukov, and A. Shamir, ”Cryptanalysis of Skipjack Reduced to 31 Rounds”, Advances in Cryptology, proc. of EUROCRYPT 99, Lecture Notes in Comput. Sci., vol 1592, Springer-Verlag, 1999, pp. 12-23.

[10] E. Biham and A. Shamir, ”Differential cryptanalysis of DES-like cryptosystems”, Journal of Cryptology., 1991,4(1), pp. 3-72.

[11] A. Biryukov, C. Canniere and G. Dellkrantz, ”Cryptanalysis of Safer++”, Advances in Cryptology, proc. of CRYPTO’03, Lecture Notes in Comput. Sci., vol 2729, Springer-Verlag, 2003, pp. 195-211.

[12] J. Nakahara, B. Preneel, and J. Vandewalle, ”Linear cryptanalysis of reduced-round safer++”, In Proceedings of the second NESSIE Workshop, 2001.

[13] G. Piret and J. Quisquater, ”Integral Cryptanalysis on reduced-round Safer++”, 2003.

[14] J. Nakahara Jr, ”Cryptanalysis and Design of Block Ciphers”, PhD thesis, Katholieke Universiteit Leuven, June 2003.