Journal of Scientia Iranica; Transactions D manuscript No.
(will be inserted by the editor)

Cryptanalysis of full-round SFN Block Cipher

a Lightweight Block Cipher, Targeting IoT Systems

Sadegh Sadeghi⋆[1], Majid Mahmoudzadeh Niknam[2], Nasour Bagheri⋆⋆[3], and Mohammad Reza Aref[1]

[1] *Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran.*
[2] *Department of Mathematics, Faculty of Mathematical Sciences and Computer, Kharazmi University, Tehran, Iran.*
[3] *Electrical Engineering Department, Shahid Rajaee Teacher Training University, Tehran, Iran, Postal code: 16788-15811.*

the date of receipt and acceptance should be inserted later

Abstract SFN is a lightweight block cipher designed to be compact in hardware and efficient in software for constrained environments such as the Internet of Things (IoT) edge devices. Compared to the conventional block ciphers that are either Feistel network-based or Substitution-Permutation (SP), it has a different structure and uses both the SP network structure and Feistel network structure to encrypt. The SFN supports key lengths of 96 bits and its block length is 64 bits and includes 32 rounds. In this paper, we propose a deterministic related-key distinguisher for 31 rounds of the SFN. We are able to use the proposed related-key distinguisher to attack the SFN in the known-plaintext scenario with the time complexity of $2^{60.58}$ encryptions. The data/memory complexity of those attacks are negligible. In addition, we will extend it to a practical chosen-plaintext-ciphertext key recovery attack on full SFN with the complexity of 2^{20}. We also experimentally verified this attack. Also, in the single key mode, we present a meet-in-the-middle attack against the full rounds for which the time complexity is 2^{80} the SFN calculations and the memory complexity is $2^{20.32}$ bytes. The data complexity of this attack is only two known plaintexts and their corresponding ciphertext.

Keywords Lightweight block cipher · SFN · Related-key differential cryptanalysis · Meet in the middle attack.

*present address: Department of Mathematics, Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan 45137-66731, Iran and Research Center for Basic Sciences and Modern Technologies (RBST), Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan 45137-66731, Iran. E-mail: s.sadeghi@iasbs.ac.ir
· **corresponding author, Tel/fax:+98-21-2297006, E-mail: na.bagheri@gmail.com

1 Introduction

A lightweight cryptography (LWC) design could be a cryptographic algorithm or protocol befitting for implementation in constrained environments including RFID tags, contactless smart cards, sensors, and so on.

A lightweight block cipher could be proper for such environments. It is worthy to note that block ciphers play a significant role in the security of communications as cryptography algorithms. Hence, the security analysis of block ciphers is of particular importance. To this end, in this paper, we apply related key and meet in the middle (MITM) attacks to analyze the lightweight block cipher SFN [1].

The notion of related key attack functions is based on the idea that the attacker has a prior awareness that (or chooses) there exists a relation between a number of keys and thus she can access the encryption functions under such related keys.The earliest attacks of this kind were developed independently by Biham [2] and Knudsen [3], and the concept of a related key attack was delineated by [2].

The meet in the middle attack is one of the types of known-plaintext attacks [4]. The attacker is able to know some plaintext and their ciphertexts. Using MITM attacks it may be possible to break ciphers, which have two or more secret keys for multiple encryptions using the same algorithm. For example, the 3DES cipher works in this way. The MITM attack is first proposed to attack DES [5]. There are numerous studies that pertain to MITM attacks on block ciphers, including [6–10].

SFN was proposed by Li *et.al* [1]. It is a 64-bit block cipher in which the round function uses both the

SP network structure and Feistel network structure to encrypt.

1.1 Our contribution.

In this paper, we present the first third party analysis of SFN block cipher, to the best of our knowledge, and our contributions are as follows:

- We introduce a deterministic related key differential distinguisher against 31 rounds of SFN.
- We also employ the proposed distinguisher to apply a full round related key differential attack on SFN to recover the main key (96 bits) in known-plaintext mode with time complexity $2^{60.58}$ and negligible data and memory complexity.
- We employ the proposed distinguisher to apply a full round related key differential attack on SFN to recover the main key (96 bits) with time complexity 2^{20}, data complexity $2^{17.92}$ and negligible memory complexityin chosen-plaintext-ciphertext mode. We also experimentally verify this attack.
- In single key mode, we introduce a meet in the middle attack on SFN to recover all the 96 bits of the main key with time complexity 2^{80} and memory complexity $2^{20.32}$ and negligible data complexity.

1.2 Outline.

This article is organized as follows. In Section 2 we present some notations and also a brief description of SFN block cipher. The description of the related key attack in the known-plaintext scenario is given in Section 3. We present the related key attack in the chosen-plaintext-ciphertext scenario in Section 4. Meet in the middle attack of the cipher is described in Section 5. Finally, the conclusion is presented in Section 6.

2 Preliminaries

In this section, we give some notations and a brief description of SFN block cipher which will be used in the following parts.

2.1 Notations

- $||$: is the concatenation of two binary strings.
- $X = (X_0 \cdots X_{15})$: represents a 64 bits string. X_0 is the lowest value of its nibbles and X_{15} is the highest value one.
- ΔX : represents a non-zero difference of X.
- P^i : represents the input of the $(i+1)^{\text{th}}$ round encryption $(i = 0, \cdots 31)$.
- RK : represents the front(low-value) 64 bits of the main keys.
- CK : represents the back(high-value) 32 bits of the main keys for control signal keys.
- $K = RK||CK$: represents the 96-bit main key.
- $S^i = (S^i_0 \cdots S^i_{15})$: represents the input of the i^{th} round encryption of SFN $(i = 1, \cdots, 32)$. S^i_0 is the lowest value nibble of S^i. It is also possible that S^i is represented by a 4×4 matrix:

$$S^i = \begin{bmatrix} S^i_0 & S^i_1 & S^i_2 & S^i_3 \\ S^i_4 & S^i_5 & S^i_6 & S^i_7 \\ S^i_8 & S^i_9 & S^i_{10} & S^i_{11} \\ S^i_{12} & S^i_{13} & S^i_{14} & S^i_{15} \end{bmatrix}.$$

- $RK^i = (RK^i_0 \cdots RK^i_{15})$: represents the $(i+1)^{\text{th}}$ round keys $(i = 0, \cdots, 31)$, RK^i_0 is the lowest value nibble of RK^i and $RK^{i\sim j}$ represents the i^{th} to j^{th} round keys.
- ΔRK^i : represents the difference of the $(i+1)^{\text{th}}$ round keys $(i = 0, \cdots, 31)$.
- CK_i : represents the i^{th} bit $(i = 0, \cdots, 31)$ of CK and $CK_{i\sim j}$ represents the i^{th} to j^{th} bit of CK.
- ΔCK : represents the difference of the control signal keys and ΔCK_i represents the difference of the i^{th} bit $(i = 0, \cdots, 31)$ of CK.
- RK^{in}_F, RK^{out}_F : represent the input and the output states of the Feistel *KeyExpansion* structure of the 32^{nd} round, respectively.
- RK^{in}_S, RK^{out}_S : represent the input and the output states of the SP *KeyExpansion* structure of the 32^{nd} round, respectively.
- P_F, P_S : represent the input states of the Feistel structure and SP structure of encryption in 32^{nd} round, respectively.
- 0^n : represents a sequence of n bits as 0, where n is a natural.
- $Enc(P, I)$: the encryption of P, I is a 32-bit string and $RK||I$ is as the main key.
- $Dec(C, I)$: the decryption of C, I is a 32-bit string and $RK||I$ is as the main key.
- $Enc(S^{(r+1)}, RK^{r\sim 31)}, CK_{r\sim 31}, r), (r = 0, \cdots, 31)$: the partial encryption of $S^{(r+1)}$, the encryption would start from round $(r+1)^{th}$ with round key RK^r and round control signal bit CK_r.
- $Dec(C, RK^{(r\sim 31)}, CK_{(r\sim 31)}, r), (r = 0, \cdots, 31)$: the partial decryption of C, the decryption would end after getting S^{r+1}.
- GF^4_2 : The finite field with 16 elements. In this field sum is XOR.

- $\alpha(t)$: represents the 32-bit string $0\cdots010\cdots0$, the only 1 is in the position of t, where $t = 0,\cdots,31$, e.g. $\alpha(31) = 0^{31}1,\ \alpha(5) = 0^5 10^{26}$.

2.2 Brief description of SFN

SFN, as a unique structure, consists of an SP network and a Feistel network [1]. Its block and the key lengths are 64 bits and 96 bits, respectively. The 96-bit main key is divided into the 64-bit round key as $RK^0 \in \{0,1\}^{64}$ and 32-bit control signal key as $CK \in \{0,1\}^{32}$. The RK^0 conducts *AddRoundKey* and *KeyExpansion*, and the $CK = CK_0 \parallel CK_1 \parallel \cdots \parallel CK_{30} \parallel CK_{31}$ is considered to be the control signal, and each bit of the control signal carries out one and only one round operation. In the case of a detailed signal key, when the bit of the control signal is 0, SFN chooses SP network structure to perform encryption or decryption, while the Feistel network structure conducts *KeyExpansion*. However, if the bit of the signal is 1, SFN selects Feistel network structure to carry out encryption or decryption and the SP network structure pursues *KeyExpansion* [1] (see Fig. 1).

The details of the SFN round function is given in Fig. 1. The 4-bit S-boxes S_1, and S_2 of SFN are defined as $S_1 = \{\texttt{C,A,D,3,E,B,F,7,8,9,1,5,0, 2,4,6}\}$ and $S_2 = \{\texttt{B,F,3,2,A,C,9,1,6,7,8,0, E,5,D,4}\}$, respectively. Both of the *MixRows* and *MixColumns layer* apply a matrix $M_{4\times4}$ which its 16 elements are in GF_2^4 (its characteristic polynomial is $x^4 + x + 1 = 0$):

$$M = \begin{bmatrix} 1 & 2 & 6 & 4 \\ 2 & 1 & 4 & 6 \\ 6 & 4 & 1 & 2 \\ 4 & 6 & 2 & 1 \end{bmatrix}.$$

In SFN the input of every rounds is represented as a 4×4 matrix say $S^i, i \in \{0,1,\cdots,31\}$, so the *MixColumns* and *MixRows layer* of i^{th} round can be represented by MS^i and S^iM respectively. For more details of SFN structure we refer the readers to [1].

3 Related Key with known-plaintext Attack

In this section, we will discuss the security of the SFN against the related key differential cryptanalysis in the known-plaintext scenario.

3.1 First key recovery Attack

Consider the two secret related key inputs to be $K^0 = (RK^0 \parallel CK)$ and $\overline{K^0} = (RK^0 \parallel \overline{CK})$, where $\Delta CK = CK \oplus \overline{CK} = \alpha(31)$ and hence $\Delta K^0 = K^0 \oplus \overline{K^0} = 0^{64} \parallel \alpha(31) = 0^{95} \parallel 1$. Hence given C and $\overline{C}$, respectively produced by (P^0, K^0) and $(P^0, \overline{K^0})$, the output differentials after 31-round encryption are $\Delta P^{31} = 0^{64}$ and $\Delta K^{31} = (\Delta RK^{31} \parallel \Delta CK_0 \cdots \Delta CK_{31}) = (0^{64} \parallel \alpha(31) = 0^{95} \parallel 1)$ with a probability of 1, which is a distinguisher for 31^{st} rounds of the SFN. Since $\Delta CK_{31} = 1$, refer to Fig. 2, the adversary would not be able to determine the difference of ciphertexts (differential output). However, she gets a distinguisher for 31^{st} rounds of the SFN, and she is able to do key recovery on the 32^{nd} round of the cipher. The procedure of the key recovery of this round is given in Algorithm 1.

To determine the attack complexity, is dominated by 2^{64} guesses for RK_F^{32}, the last round key, and related partial decryption which costs 3 rounds of SFN. We should also exhaustively search the 32 bits of CK that are not involved in the first attack to find the correct key. Therefore, the total time complexity of the first attack is $(2^{64} \times 3)\frac{1}{32} + 2^{32} \simeq 2^{60.58}$ 32-round SFN encryptions.

Algorithm 1: The first key recovery attack on the SFN

- $\bullet RK^{31} \leftarrow$ *algorithm* 1
- **for** *each choice of* RK_F^{out} *(2^{64} choices)* **do**
  - 1) Decrypt one round Feistel Key Expansion to calculate the value of RK_F^{in}. Next, drive RK_S^{in}, encrypt one round SP KeyExpansion and calculate RK_S^{out};
  - 2) One round decryption of (C, RK_F^{out}) and $(\overline{C}, RK_S^{out})$ are assigned to P_S and P_F;
  - 3) **if** $P_S \oplus P_F = 0^{64}$ **then**
    - $RK^{31} \leftarrow RK_F^{out}$
  - **else**
    - (a).One round decryption of $(\overline{C}, RK_F^{out})$ and (C, RK_S^{out}) are assigned to P_S and P_F;
    - (b). **if** $P_S \oplus P_F = 0^{64}$ **then**
      - $RK^{31} \leftarrow RK_F^{out}$

4 Related-key with chosen-plaintext-ciphertext Attack

In this section we present a new attack to recover the main key in chosen-plaintext-ciphertext scenario with time complexity 2^{20} and data complexity $2^{17.92}$. Let us assume the adversary can choose an arbitrary ciphertext C and request from an oracle, say *O-RK*, the corresponding plaintext with the key $RK||CK$ or $RK||\overline{CK}$, where $CK \oplus \overline{CK}$ is a fixed 32 bits difference which its hamming weight is one. We denote the answers of *O-RK* with the key $RK||CK$ by $Dec(C, CK)$ and with the key $RK||\overline{CK}$ by $Dec(C, \overline{CK})$. Also, she or he can choose an arbitrary plaintext P and request from the oracle $O\text{-}RK$ the corresponding ciphertext with the key $RK||CK$ or $RK||\overline{CK}$ that we denote them by $Enc(P, CK)$ and $Enc(P, \overline{CK})$, respectively. In the attack, we will find the bit of CK_{31} first and then we recover the 64 bits of RK^{31} and after that, we look for CK_{30} and RK^{30}, then we find CK_{29} and RK^{29} and so on. Finally we get CK_0 and RK^0. Remember $K = RK \parallel CK = RK^0 \parallel CK_0 \parallel \cdots \parallel CK_{31}$, so the key K has been recovered.

Suppose P is an arbitrary plaintext and the adversary is given $Enc(P, CK)$ and $Enc(P, \overline{CK})$ where $CK \oplus \overline{CK} = 0x00000001$. In our method, we denote $Enc(P, CK)$ by C_0 and C_1, when the bit of CK_{31} is 0 or 1, respectively. It is obvious that there are so many relations between C_0 and C_1, and for using of them, we look at C_0 as a new plaintext, RK^{31} as a new main key, C_1 as a new corresponding ciphertext and at whole of them as a new scheme which we call it $\varGamma_{32}$, following Fig. 3. We found a multi-outputs differential characteristic for this new scheme (see Fig. 4). By using this differential characteristic and some other similar characteristics, we found CK_{31} first, RK_{31}, other control signal bits, and round keys then. For more details refer to subsection 4.2.

4.1 The structure of $\varGamma_{32}$

As we explained before, we denoted the new scheme by $\varGamma_{32}$ that C_0 is its plaintext, RK^{31} is its main key and C_1 is its ciphertext. Two 64 bits subkeys K_0 and K_1 are made from its main key RK^{31} (Fig. 3). K_0 and K_1 are made with Feistel and SP network structure, respectively. They are the first and last round key at $\varGamma_{32}$. The new scheme uses RK^{31} for second round key after K_0, and half of it (i.e. the 32 lowest value bits) before K_1 as third round key. We found a differential trail with the probability 2^{-22} for $\varGamma_{32}$. There was an interesting situation at this trail: the 8 lowest nibbles of the output of the trail were zero. It led us to choose a multi-output differential characteristic instead of a single-output for our purpose : we allowed the 8 highest nibbles of the output to be every differences. The value of the probability of this multi-output difference characteristic was 2^{-6} which was very greater than 2^{-22} (Fig. 4). On the other hand, because of being zero the values of the 8 lowest nibble of the output differences of the characteristic, we decided to consider the structure of *MixColumns* and *MixRows* layer to find an algebraic reason for it. After considering these two linear parts of $\varGamma_{32}$, we found an interesting fact: there existed a lot of similar multi-output characteristics for $\varGamma_{32}$, which the probabilities of them were at most 2^{-6} and at least 2^{-9}. We explain these differential characteristics in more detail in the following.

The differential characteristics for the $\varGamma_{32}$

As we explained before, we found a multi-output trail of $\varGamma_{32}$ which its probability was 2^{-6} (see Fig. 4). As you can see in it, the differences before *MixRows* and after *MixColumns* are $S^2 = (000030009000B000)$ and $S^4 = (FD49EF2D00000000)$, respectively. We define a 4×4 matrix M and denote the i^{th} differential state with a 4×4 matrix S^i, so in the trail of Fig. 4:

$$S^2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 \\ 9 & 0 & 0 & 0 \\ B & 0 & 0 & 0 \end{bmatrix}, S^4 = \begin{bmatrix} F & D & 4 & 9 \\ E & F & 2 & D \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, M = \begin{bmatrix} 1 & 2 & 6 & 4 \\ 2 & 1 & 4 & 6 \\ 6 & 4 & 1 & 2 \\ 4 & 6 & 2 & 1 \end{bmatrix}$$

It is easily seen the 8 high-value nibbles of S^4 are zero. The *MixRows* and *MixColumns* layer at i^{th} round are a multiplication of the above matrix M to the state matrix, from the right and left side respectively, so:

$$S^3 = S^2 M \Rightarrow S^4 = MS^3 \Rightarrow S^4 = MS^2 M.$$

After this observation we suppose instead of $3, 9, B$ in the first column of S^2 we put x, y, z and find them such that all of the elements of the third and fourth row of S^4 to be zero, i.e, :

$$\begin{bmatrix} * & * & * & * \\ * & * & * & * \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = M \cdot \begin{bmatrix} 0 & 0 & 0 & 0 \\ \mathbf{x} & 0 & 0 & 0 \\ \mathbf{y} & 0 & 0 & 0 \\ \mathbf{z} & 0 & 0 & 0 \end{bmatrix} \cdot M = M \cdot \begin{bmatrix} 0 & 0 & 0 & 0 \\ \mathbf{x} & \mathbf{2x} & \mathbf{6x} & \mathbf{4x} \\ \mathbf{y} & \mathbf{2y} & \mathbf{6y} & \mathbf{4y} \\ \mathbf{z} & \mathbf{2z} & \mathbf{6z} & \mathbf{4z} \end{bmatrix} \quad (1)$$

$$= \begin{bmatrix} \mathbf{2x{+}6y{+}4z} & \mathbf{2(2x{+}6y{+}4z)} & \mathbf{6(2x{+}6y{+}4z)} & \mathbf{4(2x{+}6y{+}4z)} \\ \mathbf{x{+}4y{+}6z} & \mathbf{2(x{+}4y{+}6z)} & \mathbf{6(x{+}4y{+}6z)} & \mathbf{4(x{+}4y{+}6z)} \\ \mathbf{4x{+}y{+}2z} & \mathbf{2(4x{+}y{+}2z)} & \mathbf{6(4x{+}y{+}2z)} & \mathbf{4(4x{+}y{+}2z)} \\ \mathbf{6x{+}2y{+}z} & \mathbf{2(6x{+}2y{+}z)} & \mathbf{6(6x{+}2y{+}z)} & \mathbf{4(6x{+}2y{+}z)} \end{bmatrix}$$

where the sign "$*$" can be every difference. These equations are equivalent with two below equations in GF_2^4:

$$\mathbf{4x{+}y{+}2z{=}0, 6x{+}2y{+}z{=}0} \Leftrightarrow \mathbf{y{=}7x, z{=}8x, x} \in GF_2^4. \quad (2)$$

So there exist **15 nonzero** solution for triple **(x,y,z)** when **x** varies from 1 to F. As it is seen the three nonzero input differences **x**, **y** and **z** at S^2 in the characteristic of Fig. 4 are in the fourth column of the Table 1. In fact, there are a lot of other similar characteristics that their differences S^2 in them are like (0000**x**000**y**000**z**000) and the nonzero differences **x**, **y** and **z** can be chosen from every column of the Table 1. On the other hand in a characteristic of this kind, if $S^0 =$ (**i**000**j**0000000**k**000) then the differential nibbles **i**, **j** and **k** are corresponded to **y**, **z** and **x** (Fig. 5), i.e. they must be such that they

can become $\mathbf{y}$, $\mathbf{z}$ and $\mathbf{x}$ after passing the S_1-*box*, by positive probabilities. So if in a characteristic of this kind S^2 is fixed then S^0 can varies.
If $S^2 = (0000\mathbf{3}000\mathbf{9}000\mathbf{B}000)$ and $S^0 = (\mathbf{i}000\mathbf{j}0000000\mathbf{k}000)$, then by considering the DDT of S_1-box, the differential nibbles $\mathbf{i}, \mathbf{j}$ and $\mathbf{k}$ can vary while $\mathbf{i} \in \{2,3,5,8,9,B,C\}, \mathbf{j} \in \{4,7,8,9,B,D,F\}$ and $\mathbf{k} \in \{4,6,7,8,9,D,F\}$. The reason is : e.g. $\mathbf{i}$ must be a difference that has a positive probability for going to 9 after passing S_1-*box*. So by using ninth column of the DDT of S_1-*box* (Table (2)) $\mathbf{i}$ has to be in the set $\{2,3,5,8,9,B,C\}$.
The almost same situation exists for $\mathbf{j}$ or $\mathbf{k}$. With a similar way if in a characteristic of this kind the values of $\mathbf{i}$, $\mathbf{j}$ or $\mathbf{k}$ at S^0 are fixed then the values of $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ at S^2 can vary. Suppose $S^0 = (\mathbf{3}000\mathbf{7}0000000\mathbf{4}000)$ and $S^2 = (0000\mathbf{x}000\mathbf{y}000\mathbf{z}000)$ then the nonzero difference $\mathbf{x}$, $\mathbf{y}$ or $\mathbf{z}$ must be one of differences which the differences $4, 3$ or 7 can go to them after passing the S_1-*box* with positive probability respectively. So from the DDT of S_1-*box* $\mathbf{x} \in \{1,2,3,4,5,8,B\}$, $\mathbf{y} \in \{4,6,7,8,9,D,F\}$ and $\mathbf{z} \in \{3,5,6,B,D,E\}$. But by considering these three sets and the Table 1, we conclude there exist only three cases for $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$, i.e. $(\mathbf{x},\mathbf{y},\mathbf{z}) \in \{(3,9,B),(4,F,6),(5,8,E)\}$. The other values for $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ can not occur, e.g. $\mathbf{x}$ does not be 1: if $\mathbf{x} = 1$ then by Table 1, the differences $\mathbf{y}$ and $\mathbf{z}$ must be 7 and 8 respectively(the value of the second and third row of the first column of Table 1), but in the set of values for $\mathbf{z}$, there is not the value 8, so it is not possible that $\mathbf{x} = 1$.
It may be possible that there exists only one solution for S^2 when the S^0 is fixed, e.g. if $S^0 = (\mathbf{2}000\mathbf{B}0000000\mathbf{6}000)$ then there is only one case for $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$, i.e. $(\mathbf{x},\mathbf{y},\mathbf{z}) = (3,9,B)$, therefore S^2 must be $(0000\mathbf{3}000\mathbf{9}000\mathbf{B}000)$. In this case the probability of the characteristic is equal to the multiplication of three probabilities $\Pr_{S_1}(2 \to 9) \times \Pr_{S_1}(B \to B) \times \Pr_{S_1}(6 \to 3) = \frac{4}{16} \times \frac{4}{16} \times \frac{4}{16} = 2^{-6}$. By these method we collected some characteristics of this kind and their probabilities, they can be seen in Table 3.
On the other hand, if in the equality 1 we change all the elements of the column 1 and elements of columns 2, 3 or 4 of the middle matrix with each other, then two equations 2 stay the same without any changes. We explain the reasons for the case of column 2, the two other columns are the same. For column 2 the relations 1 is changed as follows:

$$\begin{bmatrix} * & * & * & * \\ * & * & * & * \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = M \cdot \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \mathbf{x} & 0 & 0 \\ 0 & \mathbf{y} & 0 & 0 \\ 0 & \mathbf{z} & 0 & 0 \end{bmatrix} \cdot M = M \cdot \begin{bmatrix} 0 & 0 & 0 & 0 \\ \mathbf{2x} & \mathbf{x} & \mathbf{4x} & \mathbf{6x} \\ \mathbf{2y} & \mathbf{y} & \mathbf{4y} & \mathbf{6y} \\ \mathbf{2z} & \mathbf{z} & \mathbf{4z} & \mathbf{6z} \end{bmatrix} \quad (3)$$

$$= \begin{bmatrix} \mathbf{2(2x+6y+4z)} & \mathbf{2x+6y+4z} & \mathbf{4(2x+6y+4z)} & \mathbf{6(2x+6y+4z)} \\ \mathbf{2(x+4y+6z)} & \mathbf{x+4y+6z} & \mathbf{4(x+4y+6z)} & \mathbf{6(x+4y+6z)} \\ \mathbf{2(4x+y+2z)} & \mathbf{4x+y+2z} & \mathbf{4(4x+y+2z)} & \mathbf{6(4x+y+2z)} \\ \mathbf{2(6x+2y+z)} & \mathbf{6x+2y+z} & \mathbf{4(6x+2y+z)} & \mathbf{6(6x+2y+z)} \end{bmatrix}$$

As it can be seen in equality 3, these equality show the two previous equations 2 are still valid. With these properties it is straightforward that one can conclude in a multi-outputs trail of this kind, it is possible to rotate the input difference nibbles of S^0 by $1, 2$ or 3 nibbles from the low-value nibbles to the high-value ones (see Table 4). At this table, the numbers in the fourth column show the number of nibbles for the rotation at the input of characteristics. It should be noted we considered the most left bit as the LSB.

Remark 1 We built a Mixed Integer Linear Programming (MILP) [11, 12] model for Γ_{32} and by using it, we looked for a characteristic from C_1 to C_0 such that its input differential to be $S^0 = (\mathbf{2}000\mathbf{B}0000000\mathbf{6}000)$ and at its output differential the 8 low-value nibbles to be zero. After that we found that there is not any such characteristic from C_1 to C_0 whit positive probability. By using this property we find the control signal bits of CK, for more details refer to section 4.2.

4.2 Second key recovery procedure

In this section, we show that by decryption different ciphertext and encrypting the results again under related keys, the 96 bits of the main secret key can be extracted with the time complexity of 2^{20} and data complexity $2^{17.92}$. In the attack the adversary has access to an oracle, say O-RL, and for an arbitrary plaintext P or for an arbitrary ciphertext C she or he can receive from the oracle:

1. the ciphertext $Enc(P, Ck)$.
2. the ciphertext $Enc(P, \overline{Ck})$, while the hamming weight of $Ck \oplus \overline{Ck}$ is 1.
3. the plaintext $Dec(C, Ck)$.
4. the plaintext $Dec(C, \overline{Ck})$, while the hamming weight of $Ck \oplus \overline{Ck}$ is 1.

To the best of our knowledge this kind of attack, in related key mode, which applies chosen-plaintext and ciphertext simultaneously is introduced for the first time , so we call it "chosen-plaintext-ciphertext" related key attack. algorithm 2 is for recovering the main key of the SFN and it has some steps as follows:

- recovering the control signal bit CK_{31} by algorithm 3.
- recovering the round key K_0 by running four times algorithm 4 and one time algorithm 5.

Algorithm 2: Recovering the main key of the SFN

- The 96 bits of main key $\leftarrow$ *Algorithm* 2

1) $CK_{31} \leftarrow Algorithm\ \ 3$
2) **for** $j \in \{0,1,2,3\}$ **do**
   $((K_0)_j, (K_0)_{4+j}, (K_0)_{12+j}) \leftarrow$ *algorithm* 4 (j, CK_{31})
3) $((K_0)_8, (K_0)_9, (K_0)_{10}, (K_0)_{11}) \leftarrow Algorithm\ \ 5$ $((K_0)_l,\ l \in (GF_2^4 - \{8,9,10,11\})$
4) $RK^{31} \leftarrow K_0$ ▷ By using the K_0 and with the Feistel structure in backward direction at Γ_{32} get the RK^{31}.
5) **for** $l \in \{30, \cdots, 0\}$ **do**
   a) $CK_l \leftarrow Algorithm6\ (l, CK_i, RK^i (l+1 \le i \le 31))$;
   b) $RK^l \leftarrow RK^{l+1}$; ▷ By using the RK^{l+1} and with the Feistel structure or the SP structure in backward direction get the RK^l when $CK_l = 0$ or 1.
6) Return $RK^0 \parallel CK_0 \parallel \cdots \parallel CK_{31}$ as the secret key of the SFN cipher.

Algorithm 3: Recovering the control signal bit CK_{31}

- $CK_{31} \leftarrow$ *algorithm* 3

1) $n = 900$, $S^0 = (2000B00000006000)$, $CK_{31} = 1$;
2) $\overline{CK} = CK \oplus \alpha(31)$;
3) **for** $i \in \{1, 2, \cdots, n\}$ **do**
   a) $C(i) \xleftarrow{\$} \{0,1\}^{64}$;
   b) $C'(i) = C(i) \oplus S^0$;
   c) $\overline{C(i)} = Enc\big(Dec(C(i), CK), \overline{CK}\big)$;
   d) $\overline{C'(i)} = Enc\big(Dec(C'(i), CK), \overline{CK}\big)$;
   e) $\Delta(i) = \overline{C(i)} \oplus \overline{C'(i)}$;
   f) **if** *8 low-value nibbles of* $\Delta(i)$ *are zero* **then**
      put $CK_{31} = 0$,
      $i = n$;

return CK_{31}.

- recovering the control signal bit CK_l and after that the round key RK^l, by running algorithm 7, 31 times for $l \in \{30, \cdots, 0\}$ respectively.

Then it returns the $K = RK^0 \parallel CK_0 \parallel \cdots \parallel CK_{31}$ as the main key of the SFN cipher. An overview of the second key recovery attack on SFN as schematically is shown in Fig. 6.

Extracting the CK_{31} (algorithm 3)

Suppose S^0 is the input difference of characteristic No.1 in Table 3. For $i = 1, \cdots, n$ where n is a natural number greater than 64, choose a random ciphertext $C(i)$ and put $C'(i) = C(i) \oplus S^0$ and quarry from the oracle *O-RK* to produce $Dec(C(i), CK)$ and $Dec(C'(i), CK)$, and then we quarry for $\overline{C(i)} = Enc\big(Dec(C(i), CK), CK \oplus \alpha(31)\big)$ and $\overline{C'(i)} = Enc\big(Dec(C', CK), CK \oplus \alpha(31)\big)$ and define $\Delta(i) = \overline{C(i)} \oplus \overline{C'(i)}$. The probability of characteristic No. 1 is 2^{-6}, so we expected that the values of the 8 low-value nibbles of $\Delta(i)$ to be zero (suitable case) in at least $n/64 \ge 1$ times out of these n cases. If for one i the $\Delta(i)$ satisfies the condition, then conclude $CK_{31} = 0$, the reason for this conclusion is the remark 1, and otherwise conclude $CK_{31} = 1$. For $n = 900$ the probability of the case in which $CK_{31} = 0$ and there is not any suitable case for i, so the algorithm returns an incorrect value for CK_{31}, is equal to $(1-2^{-6})^{900} \simeq 7 \times 10^{-7}$. It is obvious by choosing larger n, we can make the previous probability smaller and smaller. Therefore we expect the algorithm to return the correct value for CK_{31} when we choose a value sufficiently large for n. We examined it for ten million random cases when we had chosen $n = 900$: the algorithm returned the correct value in all cases, so we choose this value for n in the algorithm.

Extracting three nibbles of the RK^{31} (algorithms 4 and 6)

We want to explain the algorithm 4 and a little about the algorithm 6 which is called by it, here. Suppose $K_0 = ((K_0)_0 \cdots (K_0)_{15})$ and $j \in \{0,1,2,3\}$. Inputs of the algorithm 4 are a value j and the signal bit CK_{31} and its output is nibbles $(K_0)_j, (K_0)_{j+4}, (K_0)_{j+12}$. In "3" we define $a_0 = j, a_1 = j+4, a_2 = j+12$: the indexes of nibbles of K_0 which we want to recover them, and $b_0 = 8, b_1 = 12, b_2 = 4$: which the nibble with index $a_i - j$ in the differential S^0 after swapping and passing from S_1*-box* goes to the nibble with index b_i in the differential S^2 (Fig. 5). Also we introduce for $r \in \{0,1,2\}$, the sets J_r which we want to choose the values of $(K_0)_{a_r}$ from their elements. At the first in "1" we put $J_r = GF_2^4$ and step by step they are updated (in "4.(b)*(2)") and become smaller until all of them have only one element. Then if the conditions are satisfied in "(c)", we choose and return the unique element of J_r as the nibble $(K_0)_{a_r}$ in "(c)*". We introduce and initialize three 16 elements vectors CTR_0, CTR_1, CTR_2 in "1". The algorithm 6 updates three vectors CTR_0, CTR_1, CTR_2 and the algorithm 4 calls it several times (in "4.(a)"). For $k \in \{0, 1, \cdots, 15\}$ the element $CTR_r[k]$ counts how many times the value k satisfies the conditions for $(K_0)_{a_r}$: in "3.(g)" at the algorithm 6 when k satisfies the conditions, the value of $CTR_r[k]$ is added one unit.
Also we define three sets MS_r such that the set MS_r has all k which the value of $CTR_r[k]$ is the maximum value between all 16 elements of vector CTR_r (in "4.(b)*(1)"). These three sets are initialized to ϕ at the first (in "1."), and they are applied for updating the sets J_r: for $r \in \{0,1,2\}$ we update the sets J_r by intersecting them by MS_r (in "4.(b)*(2)").

For $m = 1$ after updating the sets J_r, if each one of them have one element, the algorithm return their unique element as the nibbles $(K_0)_{a_r}$ in "4.(c)*" , if not it tries $m = 2$. After that if some of the sets J_r have more than one elements, the program adds one unit to *Rep* and repeats the previous steps again. If for all values of m and *Rep* the algorithm can not find the three nibbles of K_0, then it print: "The three nibbles of K_0 can not be found with these values of parameters: the number of m or the value of *Rep*, or both of them should be increased" and then return the value of flag. In our experiments with $m \in \{1, 2\}$ the maximum value for *Rep* was 12 and by notice the maximum value of *Rep* in the algorithm, i.e. 30, this case will not occur in real experiments.

Algorithm 4: Recovering three nibbles of K_0 in Γ_{32}

- $(K_0)_j, (K_0)_{(4+j)}, (K_0)_{(12+j)} \leftarrow$ *algorithm* 4 (j, CK_{31})

1) $J_0 = J_1 = J_2 = GF_2^4$,
   $CTR_0[16] = CTR_1[16] = CTR_2[16] = \{0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0\}$,
   $MS_0 = MS_1 = MS_2 = \phi$,
   $flag = 0, \quad n = 30$;
2) **if** $CK_{31} = 0$ **then**
   | $CK' = CK$;
   else
   | $CK' = CK \oplus \alpha(31)$;
3) $a_0 = j, a_1 = 4 + j, a_2 = 12 + j$,
   $b_0 = 8, b_1 = 12, b_2 = 4$;
4) **for** $Rep \in \{1, 2, \cdots, n\}$ **do**
   for $m \in \{1, 2\}$ **do**
   a) $(CTR_r, r \in \{0, 1, 2\}) \leftarrow$ *Algorithm* 6 $(m, j, CK', CTR_r, J_r, r \in \{0, 1, 2\})$;
   b) **for** $r \in \{0, 1, 2\}$ **do**
   * **if** $|J_r| > 1$ **then**
   1) put MS_r equal to the set of all k which the value of $CTR_r[k]$ is equal to the maximum value of all values of 16 elements of CTR_r;
   2) $J_r = J_r \cap MS_r$;
   c) **if** $|J0| = |J_1| = |J_2| = 1$ **then**
   flag = 1, $\quad m = 2, \quad n = 30$;
   * return the value of flag and for $r \in \{0, 1, 2\}$ the unique element of J_r as the nibble $(K_0)_{a_r}$.
5) **if** *flag = 0* **then**
   print: "The three nibbles of K_0 can not be found with these values of parameters: the value of m, *Rep*, or both of them have to be increased", and then return the value of the flag.

Extracting the bits of CK_l (algorithms 7)

In algorithm 7 we denote a ciphertext which encrypted l rounds at the SFN by $C_{(l)}$, so the plaintext is equal to $C_{(0)}$ and the complete ciphertext which encrypted 32 rounds is equal to $C_{(32)}$, that usually we denote it by C. Suppose l is a fixed value in the set $\{0, 1, \cdots, 30\}$ and given the values of CK_j, RK^j for every $j = l+1, \cdots, 31$. For recovering CK_l the same as algorithm 3, suppose S^0 is the input differential of characteristic No.1 in Table 3: S^0 is initialized at "1.". In "2." define $\overline{CK} = CK \oplus \alpha(l)$ and in "3.(a)" for $i = 1, \cdots, n$, where $n = 900$, choose a random 64 bits string $C_{(l+1)}(i)$ and at "3.(b)" put

Algorithm 5: Recovering four nibbles of RK^{31}, exhaustive search

- $(K_0)_l (l \in \{8, 9, 10, 11\}) \leftarrow$ *algorithm* $5(CK_{31}, (K_0)_l (l \in (GF_2^4 - \{8, 9, 10, 11\}))$

1) $C_0 \xleftarrow{\$} \{0, 1\}^{64}$;
2) **if** $CK_{31} = 0$ **then**
   | $CK' = CK$;
   else
   | $CK' = CK \oplus \alpha(31)$;
3) $\overline{CK'} = CK' \oplus \alpha(31)$;
4) $C_1 = Enc\big(Dec(C_0, CK'), \overline{CK'}\big)$;
5) **for** $i \in \{0, \cdots, 2^{16} - 1\}$ **do**
   a) $K'_0 = ((K_0)_0 \parallel \cdots \parallel (K_0)_7 \parallel i \parallel (K_0)_{12} \parallel \cdots \parallel (K_0)_{15})$;
   b) Find $\overline{RK^{31}}$; ▷ By the Feistel structure in backward direction at Γ_{32} cipher and using the value of K'_0 find $\overline{RK^{31}}$;
   c) Find C'_1; ▷ At Γ_{32} cipher, for plaintext C_0 and main key $\overline{RK^{31}}$ find the corresponding ciphertext: C'_1;
   d) **if** $C'_1 = C_1$ **then**
   Return the value of i as $(K_0)_8 \parallel \cdots \parallel (K_0)_{11}$.

Algorithm 6: Updating three sets CTR_r for $r \in \{0, 1, 2\}$

- $CTR_r (r \in \{0, 1, 2\}) \leftarrow$ *algorithm* 6 $(m, j, CK', CTR_r, J_r (r \in \{0, 1, 2\}))$

1) $n = 900$,
   $\overline{CK'} = CK' \oplus \alpha(31)$;
2) $a_0 = j \ , \ a_1 = 4 + j \ , \ a_2 = 12 + j$,
   $b_0 = 8 \ , \ b_1 = 12 \ , \ b_2 = 4$;
3) **for** $i \in \{1, \cdots, n\}$ **do**
   a) Initialize S^0, S^2; ▷ put them equal to the inputs differential of characteristic number m with j nibbles rotation of table 4.
   b) $C_0(i) \xleftarrow{\$} \{0, 1\}^{64}$;
   c) $C'_0(i) = C_0(i) \oplus S^0$;
   d) $C_1(i) = Enc\big(Dec(C_0(i), CK'), \overline{CK'}\big)$;
   e) $C'_1(i) = Enc\big(Dec(C'_0(i), CK'), \overline{CK'}\big)$;
   f) $\Delta(i) = C_1(i) \oplus C'_1(i)$;
   g) **if** *8 low-value nibbles of $\Delta(i)$ are zero* **then**
   for $r \in \{0, 1, 2\}$ **do**
   if : $|J_r| > 1$ **then**
   for $k \in \{0, 1, \cdots, 15\}$ **do**
   if $S_1((C_0(i))_{a_r} \oplus k) \oplus S_1((C'_0(i))_{a_r} \oplus k) = (S^2)_{b_r}$ **then**
   | * add $CTR_r[k]$ one unit;

return CTR_0, CTR_1 and CTR_2.

Algorithm 7: Recovering the control signal bit CK_l for a l in the set $\{0, \cdots, 30\}$

- $CK_l \leftarrow Algorithm\ 7$ $(l, CK_i, RK^i(l+1 \leq i \leq 31))$

1) $n = 900$,
 $S^0 = (2000B00000006000)$,
 $CK_l = 1$;
2) $\overline{CK} = CK \oplus \alpha(l)$;
3) **for** $i \in \{1, 2, \cdots, n\}$ **do**
 a) $C_{(l+1)}(i) \xleftarrow{\$} \{0,1\}^{64}$;
 b) $C'_{(l+1)}(i) = C_{(l+1)}(i) \oplus S^0$;
 c) $C(i) = Enc(C_{(l+1)}(i), RK^{(l+1)\sim 31}, CK_{(l+1)\sim 31}, l+1)$,
 $C'(i) = Enc(C'_{(l+1)}(i), RK^{(l+1)\sim 31}, CK_{(l+1)\sim 31}, l+1)$;
 d) $\overline{C(i)} = Enc\big(Dec(C(i), CK), \overline{CK}\big)$;
 e) $\overline{C'(i)} = Enc\big(Dec(C'(i), CK), \overline{CK}\big)$;
 f) $\overline{C_{(l+1)}(i)} = Dec(\overline{C(i)}, RK^{(l+1)\sim 31}, CK_{(l+1)\sim 31}, l+1)$,
 $\overline{C'_{(l+1)}(i)} = Dec(\overline{C'(i)}, RK^{(l+1)\sim 31}, CK_{(l+1)\sim 31}, l+1)$;
 g) $\Delta_{(l+1)}(i) = \overline{C_{(l+1)}(i)} \oplus \overline{C'_{(l+1)}(i)}$;
 h) **if** *8 low-value nibbles of* $\Delta_{(l+1)}(i)$ *are zero* **then**
  $CK_{(l)} = 0 \ \ i = n$;
4) Return $CK_{(l)}$ as the signal bit CK_l.

$C'_{(l+1)}(i) = C_{(l+1)}(i) \oplus S^0$. Consider both of $C_{(l+1)}(i)$ and $C'_{(l+1)}(i)$ as ciphertexts which encrypted $l+1$ rounds with the signal string CK. In "3.(c)" by using the control signal bits CK_i and round keys RK^i for $i \geq l+1$, encrypt $C_{(l+1)}(i)$ and $C'_{(l+1)}(i)$ for $31-l$ rounds encryption more, to reach the complete ciphertexts $C_{(32)}(i) = C(i)$ and $C'_{(32)}(i) = C'(i)$. In "3.(d),3.(e)" request from the oracle *O-RK* to give the $Dec(C(i), CK), Dec(C'(i), CK)$ and $Enc\big(Dec(C(i), CK), \overline{CK}\big), Enc\big(Dec(C'(i), CK), \overline{CK}\big)$ which we denote them by $\overline{C(i)}, \overline{C'(i)}$ respectively. In "3.(f)" by using the control signal bits CK_i and round keys RK^i for $i \geq l+1$, decrypt $\overline{C(i)}, \overline{C'(i)}$ for $31-l$ rounds, to reach two $l+1$ encrypted ciphertexts $\overline{C_{(l+1)}(i)}$ and $\overline{C'_{(l+1)}(i)}$. In "3.(g)" define $\Delta_{(l+1)}(i)$ and in "3.(h)" if 8 low-value nibbles of $\Delta_{(l+1)}(i)$ are zero, then put $CK_{(l)} = 0$ and return it as the signal bit CK_l, and otherwise go to next i. If there is not any i which for it the condition at "3.(h)" satisfies, then return $CK_{(l)}$ which is initialized at "1." to 1, as the signal bit CK_l. For the probability of correctness of obtained CK_l refer to remark 1 and Extracting the CK_{31} at 4.2.

Extracting the main key (algorithm 2)

For recovering the main key of the SFN cipher, one can apply the algorithm 2. First in "1." by calling the algorithm 3 recover the signal bit CK_{31}. Then in "2." assume the number j, as the number for nibbles rotation at the input of the characteristics of Table 3, and by calling 4 times the algorithm 4, recover the nibbles with indexes $\{0, 1, 2, 3, 4, 5, 6, 7, 12, 13, 14, 15\}$ and after that in "3." by calling the algorithm 5, by exhaustive search, recover the nibbles with indexes $\{8, 9, 10, 11\}$ of the key K_0 at the new scheme Γ_{32}. By knowing the value of K_0 and with notice to the structure of the new scheme Γ_{32}, in "4." with Feistel structure in backward direction, get the value of round key RK^{31}. Then at "5." for $l \in \{30, \cdots, 0\}$, in 31 steps and at each step, by calling the algorithm 7 and recovering the signal bit CK_l, with Feistel or SP structure in backward direction (if CK_l is equal to 0 or 1 respectively), recover the round key RK^l. Finally in "6." return the value $RK^0 \parallel CK_0 \parallel \cdots \parallel CK_{31}$ as the secret key of the SFN block cipher.

Complexity of the second recovery attack

Suppose we consider the maximum time of encryption or decryption for a 64 bits plaintext or ciphertext, as a unit of time. Hear we want to compute an upper bound of the computational time complexity in the term of this unit for the second key recovery attack or related key with chosen-plaintext-ciphertext attack for recovering the 96 bits main key of the SFN block cipher. For this purpose we should compute the time for running the algorithm 2. First we compute the time complexity for the algorithms 3, 4, 5, 7 and then we compute the time complexity of the algorithm 2 which calls these algorithm inside itself. In each case we also compute the data complexity.

1. The time for running the algorithm 3 is dominated by the rows "3.(c)" and "3.(d)". At each row there are one decryption and one encryption, so the time complexity of this algorithm is upper bonded to $2 \times 2 \times n = 2 \times 2 \times 900 = 3600 \leq 2^{11.82}$. At this algorithm in "3.(a)" a ciphertext is chosen randomly and it is repeated almost 900 times, so its data complexity is upper bounded to $900 \times 1 = 2^{9.82}$ ciphertexts.
2. First we compute the time complexity of the algorithm 6 which is called inside of the algorithm 4. The run-time of the algorithm 6 is dominated by the rows "3.(d)" and "3.(e)", and at each row there are one encryption and one decryption. These two row are done 900 times, so its run-time as same as the algorithm 3, is upper bounded to $2^{11.82}$. The data complexity of the algorithm 6 is similar to the algorithm 3 and is upper bounded to $2^{9.82}$.
 The run-time at the algorithm 4 is dominated by the row "4.(a)". This row is done at most $2n = 2 \times 30 = 60$ times, therefore its run-time is upper bounded to $60 \times 2^{11.82} \leq 2^{17.73}$. The data complexity

of the algorithm 4 is related to calling the algorithm 6, and this algorithm is called at most $2 \times 30 = 60$, by considering the data complexity of the algorithm 6, we can conclude the data complexity of the algorithm 4 is upper bounded to $60 \times 2^{9.82} \leq 2^{15.73}$.

3. The run-time of the algorithm 5 is related to rows "4.", "5.(b)" and "5.(c)". At "4." there are one decryption and one encryption. The run-time for each row of two rows "5.(b)" and "5.(c)" is less than $\frac{1}{32}$ of the run-time for one round of the SFN cipher, so the total run-time for the algorithm 5 can be computed as follows: $2 + 2^{16} \times (2 \times \frac{1}{32}) \leq 2^{12.01}$. The data complexity of the algorithm 5 is one ciphertext which is chosen randomly at "1.".
4. The rows "3.(c)","3.(d)","3.(e)" and "3.(f)" are the main role at the run-time of the algorithm 7. The run-times of rows "3.(c)" or "3.(f)" are less than one encryption or decryption respectively which both are less than one unit. The run-time of rows "3.(d)" or "3.(e)" are 2 units. These 4 rows are done at most 900 times, so the time complexity of the algorithm 7 is upper bounded to $900 \times (1+2+2+1) \leq 900 \times 6 \leq 2^{12.40}$. The data complexity of the algorithm 7 is dominated by "3.(a)". This row is done 900 times at most, so the data complexity of this algorithm is equal to $900 \times 1 = 900 \leq 2^{9.82}$.
5. The run-time of the algorithm 2 is related to two sources:first, calling other algorithms at row "1." one time (algorithm 2), at row "2." 4 times (algorithm 4), at row "3." one time (algorithm 5), at row "5.(a)" 31 times (algorithm 6) which the run-time of all these algorithms have been computed before, and second the computing rounds key at "4." one time,"5.(b)" 31 times, which their run-time are less than one round decryption or $\frac{1}{32}$ unit of time. Therefore the total time complexity of the algorithm 2 is upper bounded to $2^{11.82} + 4 \times 2^{17.73} + 2^{12.01} + 31 \times 2^{12.40} + \frac{1}{32} + 31 \times \frac{1}{32} \leq 2^{20}$. The data complexity of the algorithm 2 by noticing the data complexities of other algorithm which it calls them is as follows: $2^{9.82} + 4 \times 2^{15.73} + 1 + 31 \times 2^{9.82} \leq 2^{17.92}$.

4.3 Experimental results:

By noticing the small time complexity of the second key recovery attack on the SFN block cipher, i.e. 2^{20}, a practical experiment was possible. So we decided to make a program to check it experimentally. The algorithms 3, 4, 5 and 7 are the main role at the algorithm 2. The algorithm 7 is almost similar to the algorithm 3 and the algorithm 5 is an exhaustive search, so we made a program by C++ language [1] for checking experimentally the algorithms 3 and 4: first program for recovering the signal bit CK_{31} and the second one for recovering the nibbles $\{0, 1, \cdots, 7, 12, 13, 14, 15\}$ of the round key K_0 at new scheme Γ_{32}. Our program can find both of CK_{31} and nibbles $(K_0)_k$ for $k \in \{0, 1, \cdots, 7, 12, 13, 14, 15\}$ separately and it is based on the algorithms 3, 6 and 4. It has been checked 10 billion times for recovering the signal bit CK_{31} and the nibbles $\{(K_0)_0, (K_0)_4, (K_0)_{12}\}$, $\{(K_0)_1, (K_0)_5, (K_0)_{13}\}$, $\{(K_0)_2, (K_0)_6, (K_0)_{14}\}$, and than $\{(K_0)_3, (K_0)_7, (K_0)_{15}\}$, and it always returned the correct values for them. The program ran on a laptop with below specifications in less than one second:
Intel(R) Core(TM) i7-6500U CPU, @ 2.50GHz 2.59 GHz, RAM 8.00 GB (7.87 GB usable), 64-bit operating system, x64-based processor.

The algorithm works as follows: first, the signal bit CK_{31} and the 64 bits round key RK^{31} are chosen randomly. By the value of RK^{31}, the keys K_0, and K_1 of Γ_{32} are made, and their value could be used only by oracle O-RK. The algorithm recovers the signal bit CK_{31} first, and after that the value of three nibbles $(K_0)_j, (K_0)_{j+4}, (K_0)_{j+12}$ where $j \in \{0, 1, 2, 3\}$ is fixed . In the algorithm the number "Rep", as in the algorithm 4, is used for repetition, also at each repetition when the algorithm wants to choose a random ciphertext, for more randomness, "Rep" is used as a coefficient of the number which is generated by "*rand*" function of `C++`. The other notations are the same as ones at the algorithms 3 and 4.

The result related to recovering three nibbles $(K_0)_0$, $(K_0)_4$, $(K_0)_{12}$ for 4 random cases **one** to **four** are shown at Table 5. The first column shows the number of characteristic at Table 3. The second column to seventh one show the sets J_0, MS_0, J_1, MS_1 and J_2, MS_2 respectively. The penultimate column shows the number of random ciphertexts used for recovering the nibbles, and the numbers of pairs with specified input/output differentials between them. The blue color are the recovered nibbles of K_0 at last row, while they are red at the K_0 in first row and other places in every recovering. In each row, for $r \in \{0, 1, 2\}$ the set J_r is equal to intersection of the set MS_r at the same row with the set J_r at the previous row. When the number of elements in the set J_r becomes 1, then its element is $(K_0)_{a_r}$ which a_r is 0, 4 or 12.

At this experiment for recovering the three nibbles $(K_0)_0$, $(K_0)_4$, $(K_0)_{12}$, the number of random ciphertexts that were used: in case **one** was equal to $3(2\times900) = 5400$ (Rep=3), in case **two** to **four** was equal to $2\times900 = 1800$

[1] Related codes are available at
`https://github.com/MajidMNiknam/SFN-cipher/commit/8688ecaaed83e49633d942176c40c22154b879ac`

(Rep=1). Also for 10 billion repetitions for recovering these nibbles, the average and its maximum number of random ciphertexts that were used were equal to 2314.76 and 19800 in case **one**, 2173.90 and 12600 in case **two**, 2174.50 and 14400 in case **three**, 4045.47 and 21600 in case **four**, respectively. So the experimental data complexity of this algorithm on average was $2^{11.98}$ and its maximum was $2^{14.4}$, while the theoretical data complexity for this algorithm has been computed before $2^{15.73}$.

5 Meet in the middle attack

In MITM attack, the cipher is divided into two parts and the main idea is that the subkeys of key bits in both parts of the cipher can be guessed independently. In 2010, Bogdanov et al., introduced a new variant of MITM attack (3-subset MITM attack) on block ciphers [6]. Instead of considering two subsets of key bits, they considered three subsets as A_1 that shows the key bits used only in the first part, A_2 that shows the key bits used only in the second part, and A_0 that shows the key bits used in two parts of the cipher.

Following the SFN's description, given the 96-bit main key $K = RK\|CK$, the fraction $RK \in \{0,1\}^{64}$ is used to generate the round keys, and $CK \in \{0,1\}^{32}$ is used as the control signal to determine whether in each round key-expansion/round-function the Feistel structure is used or the SP one. Notice that each bit of the control signal is used in one and only one round of the SFN, hence, this block cipher will be an appropriate candidate for the 3-subset MITM attack. Therefore, inspired by [6], suppose $A_1 = CK_{0\sim15}, A_2 = CK_{16\sim31}$, and $A_0 = RK$, are the three subsets of key bits used in SFN structures. The procedure of the key recovery of the SFN in the 3-subset meet in the middle attack is given in algorithm 8. Now, if the adversary guesses are correct then the internal values should match, i.e., $P^{16} = P'^{16}$. These happen for the correct guess of keys with the probability of 1 while for the wrong guess of keys the matching probability would be $2^{-|P^{16}|}$. Therefore, with a probability of about $2^{-|P^{16}|}$ this match would result in a false positive, but overall the number of key candidates is reduced to about $2^{|K|} \times 2^{-|P^{16}|} = 2^{96-64} = 2^{32}$ after applying algorithm 8. Thus, the number of key candidates is small enough that it has no effect on attack complexity. However, by considering another known plaintext/ciphertext (P', C') the number of key candidates can be reduced to about $2^{32} \times 2^{-64} = 2^{-32}$ and thus the target key will be obtained.

Following the previous discussion, considering the cost of the decryption round the same as the cost of the encryption round, the time complexity of the provided attack would be equal to

$$\underbrace{2^{|A_0|}(\frac{1}{2}(2^{|A_1|} + 2^{|A_2|}))}_{\text{algorithm 8}} + \underbrace{((2^{|K|-|P^{16}|}) + (2^{|K|-|P^{16}|} \times 2^{|C'|}) + \cdots)}_{\text{key testing}}$$
$$= 2^{64}(\frac{1}{2}(2^{16} + 2^{16})) + 2^{32} + 2^{-32} \simeq 2^{80},$$

calls to the SFN. Therefore, in total, we need only two known plaintext/ciphertext pairs (one for applying algorithm 8 and one for the key testing step). The memory complexity of the attack is dominated by matching step in algorithm 8 which is at most $2^{16} \times (80+80)$ bits, $2^{20.32}$ bytes.

Algorithm 8: The 3-subset MITM attack of the SFN

```
•K ← algorithm 8(A_0, A_1, A_2)
for a known plaintext/ciphertext pair do
    for each 2^64 choice of key bits in A_0 do
        for each 2^16 choice of key bits in A_1 do
            Encrypt 16 rounds SFN to calculate the
              value of P^16;
        for each 2^16 choice of A_2 do
            Decrypt 16 rounds SFN to calculate the
              value of P'^16;
        Execute matching between the values of P^16
          and P'^16 on 64 bits;
        if P^16 = P'^16 then
            Return the related key as correct round
              key.
        else
            Abort the related key.
```

6 Conclusion

This paper investigates the security level on the SFN against the related key attack. The encryption of the SFN involves an SP network structure and a Feistel network structure. The SFN fixes a 64-bit block with a 96-bit key. We have proposed an attack, in the known-plaintext scenario, taking advantage of the related key distinguisher. With this attack, we have shown that FSN provides at most $2^{60.58}$ encryptions security. We also proposed a chosen-plaintext-ciphertext related key attack on the SFN with the complexity of 2^{20}. In addition, in the single key mode, we presented a meet in the middle attack for which the time complexity was 2^{80} and the memory complexity was $2^{20.32}$ bytes. The attack complexity should be compared with the complexity of exhaustive key search which is 2^{96}.

Acknowledgments

Nasour Bagheri was supported by Shahid Rajaee Teacher Training University under grant number 4968.

References

1. Li, L., Liu, B., and Zhou, Y. et al, "SFN: A new lightweight block cipher," *Microprocessors and Microsystems*, vol. 60, pp. 138–150, 2018.
2. Biham, E. "New types of cryptanalytic attacks using related keys," *Journal of Cryptology*, vol. 7, no. 4, pp. 229–246, 1994.
3. Knudsen, L. R. "Cryptanalysis of LOKI 91," in *International Workshop on the Theory and Application of Cryptographic Techniques*, pp. 196–208, Springer, 1992.
4. Diffie, W., and Hellman, M. "Exhaustive cryptanalysis of the nbs data encryption standard," *IEEE Computer Society Press*, vol. 10, no. 6, pp. 74–84, 1977.
5. Diffie, W. and Hellman, M. E. "Special feature exhaustive cryptanalysis of the nbs data encryption standard," *Computer*, vol. 10, no. 6, pp. 74–84, 1977.
6. Bogdanov, A., and Rechberger, C., "A 3-subset meet-in-the-middle attack: cryptanalysis of the lightweight block cipher ktantan," in *International Workshop on Selected Areas in Cryptography*, pp. 229–240, Springer, 2010.
7. Ahmadi, S., and Aref, M. R. "Generalized meet in the middle cryptanalysis of block ciphers with an automated search algorithm," *IEEE Access*, vol. 8, pp. 2284–2301, 2019.
8. Dong, X., Wei, Y., and Gao, W. et al, "New meet-in-the-middle attacks on fox block cipher," *The Computer Journal*, 2022.
9. Ahmadi, S, Ahmadian, Z., and Mohajeri, J., et al, "Low-data complexity biclique cryptanalysis of block ciphers with application to piccolo and hight," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 10, pp. 1641–1652, 2014.
10. Liu, F., Sarkar, S., and Wang, G., et al, "Algebraic meet-in-the-middle attack on lowmc." Cryptology ePrint Archive, Paper 2022/019, 2022. `https://eprint.iacr.org/2022/019`.
11. Mouha, N., Wang, Q. and Gu, D., et al, "Differential and linear cryptanalysis using mixed-integer linear programming," in *International Conference on Information Security and Cryptology*, pp. 57–76, Springer, 2011.
12. Sun, S., Hu, L., and Wang, M., et al, "Towards finding the best characteristics of some bit-oriented block ciphers and automatic enumeration of (related-key) differential and linear characteristics with predefined properties," *Cryptology ePrint Archive, Report*, vol. 747, p. 2014, 2014.

Biographies

Sadegh Sadeghi received his Ph.D. in mathematical cryptography from Kharazmi University in 2019. His Ph.D. dissertation focused on automated cryptanalysis of lightweight symmetric. He was a postdoctoral researcher in the Electrical Engineering Department at the Sharif University of Technology, Tehran. He is currently an associate professor at the department of mathematics, Institute for Advanced Studies in Basic Sciences (IASBS) and Research Center for Basic Sciences and Modern Technologies (RBST), Institute for Advanced Studies in Basic Sciences (IASBS), Zanja, Iran. His main research interests are cryptanalysis and the security of protocols

Majid Mahmoudzadeh Niknam is a free researcher in cryptography and a mathematics teacher. He received his BSc degree in applied mathematics from the Sharif University of Technology, MSc degree in pure math from Tarbiat Modares University, and a Ph.D. degree from Kharazmi University in applied mathematics (cryptography). His research interests are linear attack, differential attack, and key recovery in lightweight block ciphers and Authenticated Encryption with Associated Data (AEAD).

N. Bagheri received the M.S. and Ph.D. degrees in electrical engineering from the Iran University of Science and Technology (IUST), Tehran, Iran, in 2002 and 2010, respectively. He is currently a Full Professor in the Electrical Engineering Department, at Shahid Rajaee Teacher Training University, Tehran, and the head of CPS2 laboratory there. He is also a part-time Researcher with the Institute for Research in Fundamental Sciences. He is the author of more than 100 articles on information security and cryptology. His research interests include cryptology, more precisely, designing and analysis of symmetric schemes, such as lightweight ciphers, e.g., block ciphers, hash functions, authenticated encryption schemes, cryptographic protocols for constrained environments, such as RFID tags and IoT edge devices, and hardware security, e.g., the security of symmetric schemes against side-channel attacks, such as fault injection and power analysis.

M.R. Aref received his BSc degree in 1975 from the University of Tehran, Iran, and MSc and Ph.D. degrees in 1976 and 1980, respectively, from Stanford University, Stanford, CA, USA, all in Electrical Engineering. He returned to Iran in 1980 and since then, he has been actively engaged in academic activities. He was a former Faculty member of Isfahan University of Technology from 1982 to 1995. He has been working as a Professor of Electrical Engineering at Sharif University of Technology, Tehran since 1995 and has published more than 230 technical papers in communication and information theory and cryptography in international journals and conference proceedings. His current research interests include areas of communication theory, information theory, and cryptography.

Figure and Table Captions

List of Figures

List of Tables

Table 1: The fifteen solution for $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$. Every column is a solution. The blue column is the values for the characteristic of Fig.4.

x	1	2	**3**	4	5	6	7	8	9	A	B	C	D	E	F
y	7	E	**9**	F	8	1	6	D	A	3	4	2	5	C	B
z	8	3	**B**	6	E	5	D	C	4	F	7	A	2	9	1

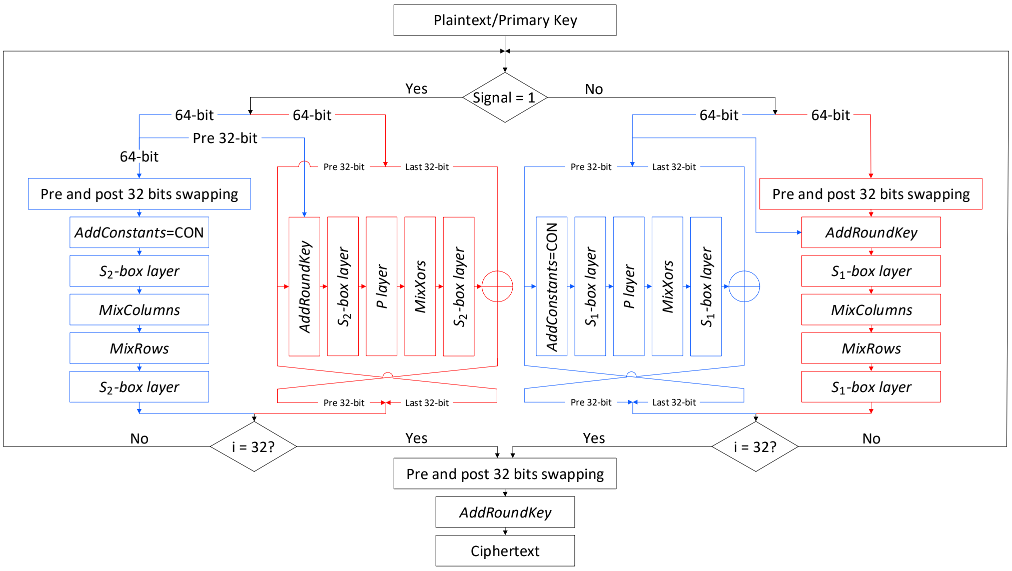

Fig. 1: Encryption procedure of SFN cipher [1].

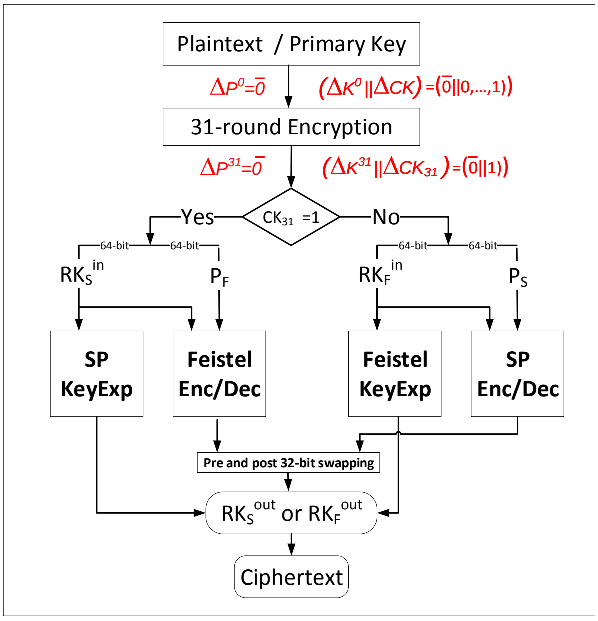

Fig. 2: A Distinguisher on full SFN, where $\overline{0}$ means 0^{64} .

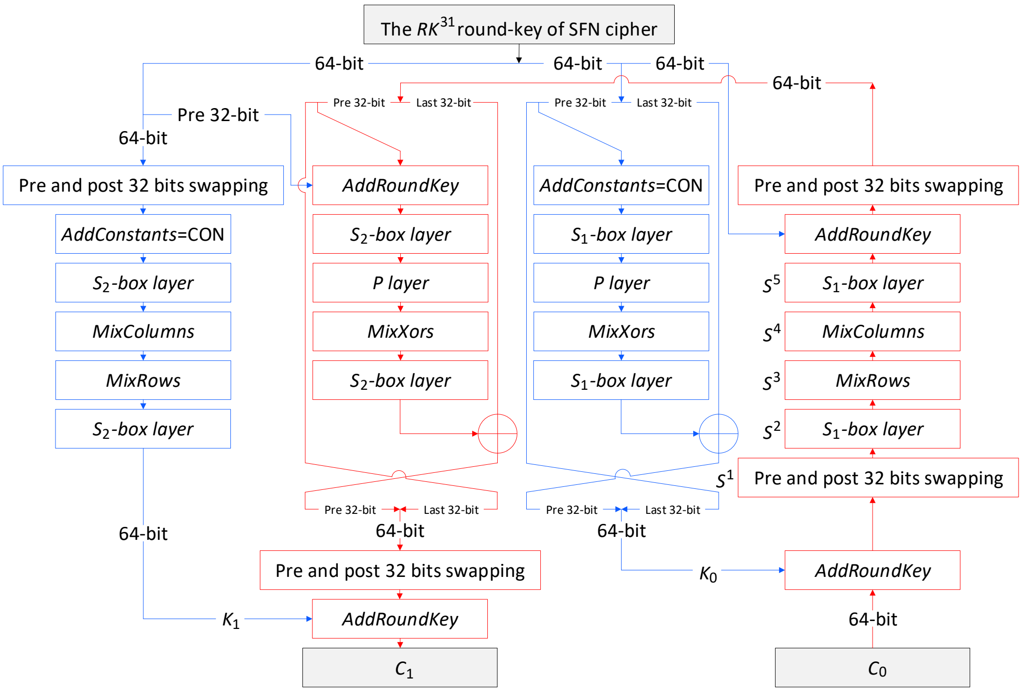

Fig. 3: The new scheme Γ_{32} from C_0 to C_1 for the SFN cipher, where the C_0 and C_1 are the ciphertext when CK_{31} is 0 or 1, respectively. At Γ_{32} two 64 bits subkeys K_0 and K_1 are made from the round key RK^{31} with Feistel and SP network, respectively.

Table 2: Differential Distribution Table (DDT) of the S_1-*box*.

x/y	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	2	4	0	2	2	2	0	2	0	0	0	0	0	2	0
2	0	4	0	0	4	0	0	0	0	4	0	0	4	0	0	0
3	0	0	0	0	2	0	4	2	2	2	0	0	0	2	0	2
4	0	2	4	2	2	2	0	0	2	0	0	2	0	0	0	0
5	0	2	0	0	2	0	0	4	0	2	4	0	2	0	0	0
6	0	2	0	4	0	0	0	2	2	0	0	0	2	2	0	2
7	0	0	0	2	0	4	2	0	0	0	0	2	0	4	2	0
8	0	2	0	2	2	0	2	0	0	2	0	2	2	0	2	0
9	0	0	4	2	0	2	0	0	2	2	0	2	2	0	0	0
A	0	0	0	0	0	4	0	0	0	0	4	0	0	4	0	4
B	0	0	0	0	2	0	0	2	2	2	0	4	0	2	0	2
C	0	0	4	0	0	2	2	0	2	2	0	0	2	0	2	0
D	0	0	0	2	0	0	2	4	0	0	4	2	0	0	2	0
E	0	2	0	0	0	0	0	2	2	0	0	0	2	2	4	2
F	0	0	0	2	0	0	2	0	0	0	4	2	0	0	2	4

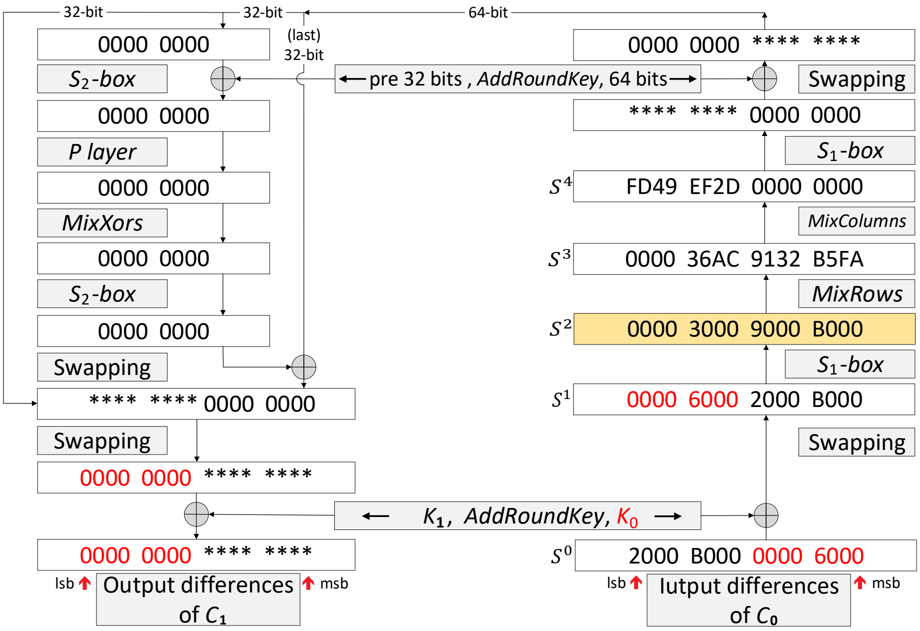

Fig. 4: A multi-output characteristic (because the stars can vary) for the new scheme Γ_{32} from C_0 to C_1. The values of differences of input nibbles 0, 4 and 12 can vary simultaneously. In fact the *nibble*0, *nibble*4 and *nibble*12 of input differences must choose from a proper set. It is while its 8 low-value output nibbles differences do not vary and stay zero. Also, the positions of 16 input nibbles can rotate to the right (from LSB to MSB) by 1, 2 or 3 nibbles, e.g. there exist a similar characteristics with input difference $0x0060000000B00020$ and the same previous output.

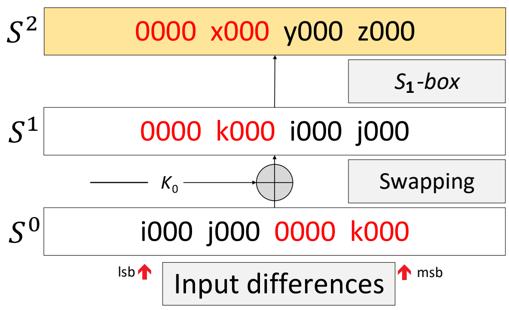

Fig. 5: The first three state of a similar multi-outputs differential characteristic to the trail of Fig. 4 for Γ_{32}.

Table 3: Some multi-output characteristics for Γ_{32}, where PS and PT respectively denote the probability of a single trail and the sum of the probabilities of all trails as a total probability.

No.	S^0	S^2	$\#S^2$	PS	PT
1	(**2**000**B**00000000**6**000)	(0000**3**000**9**000**B**000)	1	2^{-6}	2^{-6}
2	(**3**000**4**00000000**7**000)	(0000**3**000**9**000**B**000)	1	2^{-9}	2^{-9}
3	(**3**000**4**00000000**4**000)	(0000**3**000**9**000**B**000) (0000**1**000**7**000**8**000)	2	2^{-9} 2^{-9}	2^{-8}
4	(**2**000**B**00000000**4**000)	(0000**3**000**9**000**B**000) (0000**4**000**7**000**B**000)	2	2^{-9} 2^{-9}	2^{-8}
5	(**3**000**7**00000000**4**000)	(0000**3**000**9**000**B**000) (0000**4**000**F**000**6**000) (0000**5**000**8**000**E**000)	3	2^{-9} 2^{-9} 2^{-9}	$2^{-7.41}$

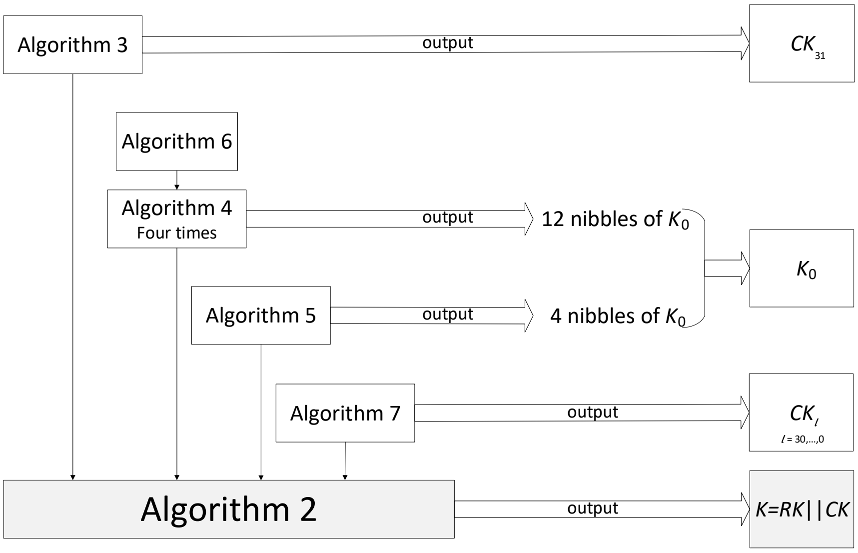

Fig. 6: Framework of the second key recovery attack of SFN cipher.

Table 4: It is possible to rotate the input differences S^0 at the multi-output characteristics for Γ_{32} by $1, 2$ or 3 nibbles from the low-value nibbles to the high-value ones. At first column, numbers 10 to 13, $\cdots$, 50 to 53 show the characteristic number 1, $\cdots$, 5 of Table 3 and their rotations with $1, 2$ or 3 nibbles at their inputs respectively.

No.	S^0	S^2	# nibbles	PS	PT
10	(**2**000**B**0000000**6**000)	(0000**3**000**9**000**B**000)	0	2^{-6}	2^{-6}
11	(0**2**000**B**0000000**6**00)	(00000**3**000**9**000**B**00)	1	2^{-6}	2^{-6}
12	(00**2**000**B**0000000**6**0)	(000000**3**000**9**000**B**0)	2	2^{-6}	2^{-6}
13	(000**2**000**B**0000000**6**)	(0000000**3**000**9**000**B**)	3	2^{-6}	2^{-6}
20	(**3**000**4**0000000**7**000)	(0000**3**000**9**000**B**000)	0	2^{-9}	2^{-9}
21	(0**3**000**4**0000000**7**00)	(00000**3**000**9**000**B**00)	1	2^{-9}	2^{-9}
22	(00**3**000**4**0000000**7**0)	(000000**3**000**9**000**B**0)	2	2^{-9}	2^{-9}
23	(000**3**000**4**0000000**7**)	(0000000**3**000**9**000**B**)	3	2^{-9}	2^{-9}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
50	(**3**000**7**0000000**4**000)	(0000**3**000**9**000**B**000) (0000**4**000**F**000**6**000) (0000**5**000**8**000**E**000)	0	2^{-9} 2^{-9} 2^{-9}	$2^{-7.4150}$
51	(0**3**000**7**0000000**4**00)	(00000**3**000**9**000**B**00) (00000**4**000**F**000**6**00) (00000**5**000**8**000**E**00)	1	2^{-9} 2^{-9} 2^{-9}	$2^{-7.4150}$
52	(00**3**000**7**0000000**4**0)	(000000**3**000**9**000**B**0) (000000**4**000**F**000**6**0) (000000**5**000**8**000**E**0)	2	2^{-9} 2^{-9} 2^{-9}	$2^{-7.4150}$
53	(000**3**000**7**0000000**4**)	(0000000**3**000**9**000**B**) (0000000**4**000**F**000**6**) (0000000**5**000**8**000**E**)	3	2^{-9} 2^{-9} 2^{-9}	$2^{-7.4150}$

Table 5: The experimental results for recovering CK_{31}, and three nibbles with indexes $0, 4$ and 12 of K_0 at Γ_{32} for 4 random keys: **one** to **four**. The blue color are the recovered nibbles of K_0 at last row in every case, while they are red at the K_0 in first row and other places.

One	$CK_{31} = 1$	$K_0 = 0x18be\ 6784\ a484\ ef55$			$RK^{31} = 0x0029\ 4823\ 18be\ 6784$			
No. of ch.	$J_0 = GF_2^4$	MS_0	$J_1 = GF_2^4$	MS_1	$J_2 = GF_2^4$	MS_2	# pair	Rep
1	$\{5, 7, c, e\}$	$\{5, 7, c, e\}$	$\{4, 6, d, f\}$	$\{4, 6, d, f\}$	$\{8, a, c, e\}$	$\{8, a, c, e\}$	900, (18)	1
2	$\{5, 7, c, e\}$	GF_2^4	$\{4, 6, d, f\}$	GF_2^4	$\{8, a, c, e\}$	GF_2^4	900, (0)	
1	$\{5, 7, c, e\}$	$\{5, 7, c, e\}$	$\{4, 6, d, f\}$	$\{4, 6, d, f\}$	$\{8, a, c, e\}$	$\{8, a, c, e\}$	900, (16)	2
2	$\{5, 7, c, e\}$	GF_2^4	$\{4, 6, d, f\}$	GF_2^4	$\{8, a, c, e\}$	GF_2^4	900, (0)	
1	$\{5, 7, c, e\}$	$\{5, 7, c, e\}$	$\{4, 6, d, f\}$	$\{4, 6, d, f\}$	$\{8, a, c, e\}$	$\{8, a, c, e\}$	900, (13)	3
2	$\{5\}$	$\{5, 6\}$	$\{4\}$	$\{0, 4\}$	$\{e\}$	$\{9, e\}$	900, (5)	
Two	$CK_{31} = 0$	$K_0 = 0x42da\ 718d\ dcc0\ 40b9$			$RK^{31} = 0x02a4\ 3024\ 42da\ 718d$			
No.	$J_0 = GF_2^4$	MS_0	$J_1 = GF_2^4$	MS_1	$J_2 = GF_2^4$	MS_2	# pair	Rep
1	$\{0, 2, 9, b\}$	$\{0, 2, 9, b\}$	$\{0, 2, 9, b\}$	$\{0, 2, 9, b\}$	$\{8, a, c, e\}$	$\{8, a, c, e\}$	900, (15)	1
2	$\{9\}$	$\{9, a\}$	$\{0\}$	$\{0, 4\}$	$\{a\}$	$\{a, d\}$	900, (2)	
Three	$CK_{31} = 1$	$K_0 = 0x284c\ 4948\ c30c\ 6a77$			$RK^{31} = 0x43c7\ 76fa\ 284c\ 4948$			
No.	$J_0 = GF_2^4$	MS_0	$J_1 = GF_2^4$	MS_1	$J_2 = GF_2^4$	MS_2	# pair	Rep
1	$\{5, 7, c, e\}$	$\{5, 7, c, e\}$	$\{5, 7, c, e\}$	$\{5, 7, c, e\}$	$\{8, a, c, e\}$	$\{8, a, c, e\}$	900, (14)	1
2	$\{7\}$	$\{4, 7\}$	$\{c\}$	$\{8, c\}$	$\{c\}$	$\{b, c\}$	900, (2)	
Four	$CK_{31} = 0$	$K_0 = 0x67d4\ 0097\ 6ea0\ d1d5$			$RK^{31} = 0x1db3\ 707f\ 67d4\ 0097$			
No.	$J_0 = GF_2^4$	MS_0	$J_1 = GF_2^4$	MS_1	$J_2 = GF_2^4$	MS_2	# pair	Rep
1	$\{5, 7, c, e\}$	$\{5, 7, c, e\}$	$\{0, 2, 9, b\}$	$\{0, 2, 9, b\}$	$\{0, 2, 4, 6\}$	$\{0, 2, 4, 6\}$	900, (11)	1
2	$\{5\}$	$\{5, 6\}$	$\{0\}$	$\{0, 4\}$	$\{4\}$	$\{3, 4\}$	900, (2)	