Cryptanalysis of TWIS Block Cipher

Onur Koçak and Neşe Öztop

Institute of Applied Mathematics, Middle East Technical University, Turkey
{onur.kocak,noztop}@metu.edu.tr

Abstract. TWIS is a 128-bit lightweight block cipher that is proposed by Ojha et al. In this work, we analyze the security of the cipher against differential, impossible differential and linear attacks. For the differential case, we mount a full-round attack on TWIS and recover 12 bits of the 32-bit final subkey with 2^{21} complexity. For the other cases, we present distinguishers which can be extended to key recovery attacks. Also, we showed that the security of the cipher is only 62 bits instead of claimed 128 bits. Moreover, we introduce some observations that compromise the security of the cipher.
Keywords: TWIS, Lightweight Block Cipher, Differential Cryptanalysis, Impossible Differential Distinguisher, Linear Distinguisher.

1 Introduction

The pace of ubiquitous devices in daily life has been increased drastically in the last few years. As the usage increases, the privacy of the stored data and the security of the communication between these devices become questionable. The requirement for protection of data and communication makes the use of cryptographic algorithms inevitable. However, the standardized algorithms like AES[1] and SHA[2] or commonly used algorithms like Triple DES[3] and MD5[4] are not suitable for constrained devices. Therefore, recently, new lightweight algorithms which need low power consumption and hardware area, like Present[5], KATAN/KTANTAN[6], DESL[7], Grain[8] and TWIS[9] are designed for such constrained environments.

TWIS is a 128-bit block cipher designed to be used in ubiquitous devices. The cipher, which is inspired from CLEFIA[10], is a 2-branch generalized Feistel Network of 10 rounds. There is no key recovery attack on this cipher up to the authors knowledge. The only analysis is done by Su et al.[11] in which n-round iterative differential distinguishers are presented. However, as the probability of the iterative distinguishers are 1, they cannot be extended to a differential attack to get information about the key.

In this paper, we analyze the security of TWIS block cipher against differential, impossible differential and linear cryptanalysis. We mount a differential attack on full-round TWIS and recover 12 bits of the 32-bit final subkey with a complexity of 2^{21}. This is the first experimental result on TWIS. Also, we present a 9.5-round impossible distinguisher which can be extended to a key recovery attack, and a straightforward linear distinguisher. Furthermore, by making observations on the key schedule, we show that the cipher offers at most 62-bit security instead of claimed 128-bit. Besides, we mention the potential weaknesses due to the use of subkeys during the encryption and the choice of whitening subkeys. The paper is organized as follows. Section 2 gives a description of the round function and the key schedule of TWIS block cipher. In Section 3, a 10-round differential attack is presented. Linear and impossible differential distinguishers are proposed in Section 4 and Section 5, respectively. Some observations on the algorithm are given in Section 6. Finally Section 7 concludes the paper.

2 Description of TWIS Block Cipher

TWIS is a lightweight block cipher with 128-bit plaintext and key sizes each. Designers of the cipher are inspired from CLEFIA to design a lighter algorithm without compromising the

security. The algorithm is a 2-branch generalized Feistel Network, running on 10 rounds. At each round, two 32-bit subkeys are used. The key is mixed with the plaintext inside the G-function. Details of the G-function is given in Section 2.1.

Round subkeys are generated via key scheduling algorithm. Key scheduling part can be viewed as an $NFSR$ which updates the content using an S-box and a round constant. Details of the key scheduling algorithm is given in Section 2.2.

Notation The following notations are used throughout this paper:

- a $\oplus$ b: bitwise XOR of a and b
- a $\wedge$ b: bitwise AND of a and b
- $<<< i$: left rotation by i bits
- $>>> i$: right rotation by i bits
- ΔI: XOR difference between two inputs

Let $P = (P_0, P_1, P_2, P_3)$ and $C = (C_0, C_1, C_2, C_3)$ be the 128-bit plaintext and ciphertext respectively where P_i and C_i, $0 \leq i \leq 3$, are 32-bit words. Also, let RK_j be the 32-bit j^{th} subkey for $j = 0, .., 10$. Then, the encryption process can be summarized as in Algorithm 1. Likewise, Figure 2 shows the encryption schematically.

Algorithm 1 The Encryption Process of TWIS

$(T_0, T_1, T_2, T_3) = (P_0 \oplus RK_0, P_1, P_2, P_3 \oplus RK_1)$
for $i = 1$ **to** 10 **do**
  $(X_0, X_1) = G(RK_{i-1}, T_0, T_1)$
  $T_2 = X_0 \oplus T_2$
  $T_3 = X_1 \oplus T_3$
  $T_1 = T_1 <<< 8$
  $T_3 = T_3 >>> 1$
  $(T_0, T_1, T_2, T_3) = (T_2, T_3, T_0, T_1)$
  $(X_0, X_1) = G(RK_i, T_0, T_3)$
  $T_1 = X_0 \oplus T_1$
  $T_2 = X_1 \oplus T_2$
  $T_2 = T_2 >>> 1$
  $T_3 = T_3 <<< 8$
end for
$(C_0, C_1, C_2, C_3) = (T_0 \oplus RK_2, T_1, T_2, T_3 \oplus RK_3)$

2.1 *G*-Function

G-function is the round function of TWIS block cipher. It provides confusion and diffusion between the branches. G-function takes three inputs of 32 bits; 32-bit subkey and 32-bit words from each two of the four branches, and outputs two 32-bit words. The G-function can be written as in Algorithm 2.

Algorithm 2 G-Function

$G(RK, X_0, X_1) = (Y_0, Y_1)$
  $Y_1 = X_1 \oplus F(RK, X_0)$
  $Y_0 = X_1$

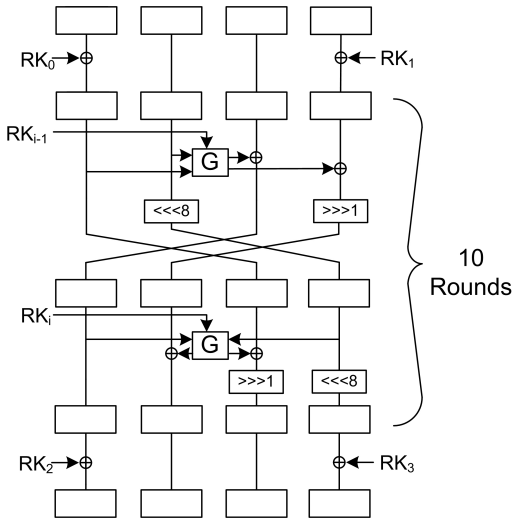

Fig. 1. Encryption Process

***F*-Function** F-function is the core of the G-function. Key mixing and confusion occurs within this function. F-function takes two 32-bit inputs, one of which is the subkey. It XORs the first parameter, the content of the corresponding branch, with the subkey and divides the resulting 32-bit word into four 8-bit words. Then, F-function applies a 6×8 S-box to each of the 8-bit words and swaps them as in the actual rounds. Finally, it concatenates four 8-bit words to form a 32-bit word. The F-function is formulated in Algorithm 3.

Algorithm 3 F-Function

$F(RK, Q)$
$Q = Q \oplus RK$
$Q = (Q_0, Q_1, Q_2, Q_3)$
$Q_0 = S(Q_0 \wedge 0x3f)$
$Q_1 = S(Q_1 \wedge 0x3f)$
$Q_2 = S(Q_2 \wedge 0x3f)$
$Q_3 = S(Q_3 \wedge 0x3f)$
$Q = (Q_2, Q_3, Q_0, Q_1)$

S-Box The S-box used in the F-function is a 6×8 S-box which is given in Table 1. The first two bits of the 6-bit input determine the row, the remaining 4 bits determine the column of the table and the corresponding value is given as the output. For example $S(0x24) = 0xf7$.

Although the output space is larger than the input space, there are some inputs that are mapped to the same output, like $S(30) = S(15)$. This enables a non-zero difference to be mapped to zero difference which is a weakness that can be exploited to mount differential type of attacks.

2.2 Key Schedule

The key schedule part of TWIS generates subkeys which are used in the F-functions. It produces 11 subkeys for the 10-round cipher. RK_0 and RK_1 are used as the initial whitening keys, while RK_2 and RK_3 are used as the final whitening keys. Notice that, RK_1, RK_2 and RK_3 are used three times, RK_{10} is used once and the rest of the subkeys are used twice. The key scheduling

Table 1. S-Box of TWIS

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	90	49	d1	c6	2f	33	74	fb	95	6d	82	ea	0e	b0	a8	1c
1	28	d0	4b	92	5c	ee	85	b1	c4	0a	76	3d	63	f9	17	af
2	bf	bf	19	65	f7	7a	32	20	16	ce	e4	83	9d	5b	4c	d8
3	ee	99	2e	f8	d4	9b	0f	13	29	89	67	cd	71	dd	b6	f4

algorithm uses the same S-box as in the F-function. In addition, it uses a diffusion matrix M to generate the subkeys from the master key which is given as

$$M = \begin{pmatrix} 0x01\ 0x02\ 0x04\ 0x06 \\ 0x02\ 0x01\ 0x06\ 0x04 \\ 0x04\ 0x06\ 0x01\ 0x02 \\ 0x06\ 0x04\ 0x02\ 0x01 \end{pmatrix}.$$

The key scheduling algorithm can be formulated as in Algorithm 4 and is shown in Figure 2.

Algorithm 4 The Key Scheduling Algorithm

$K = (K_1, K_2, \ldots, K_{16})$
for $i = 1$ **to** 11 **do**
  $K = K <<< 3$
  $K_i = S(K_i \wedge 0x3f)$
  $K_{15} = S(K_{15} \wedge 0x3f)$
  $K_{16} = K_{16} \oplus i$
  $RK_{i-1}^t = M \cdot (K_{13}K_{14}K_{15}K_{16})^t$
end for

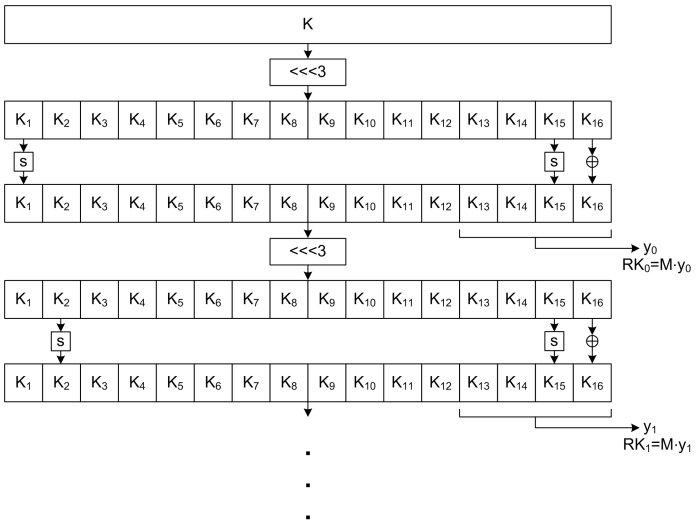

Fig. 2. Key Scheduling Algorithm

3 Differential Attack on TWIS

Differential cryptanalysis was introduced by Biham and Shamir [12] in 1990 and is one of the most effective techniques in block cipher cryptanalysis. It analyzes how the difference between two input values propagates after encryption of these inputs for some number of rounds. For TWIS block cipher, no differential analysis is given and it is left as a future work[9]. In[11], security of TWIS against differential cryptanalysis is evaluated by Su et al. and differential distinguishers for 10-round TWIS cipher are presented. In this section, we propose a key recovery attack on 10-round TWIS excluding the final key whitening. Our attack is based on a 9.5-round differential distinguisher which is explained in the following section.

3.1 9.5-round Differential Characteristic

In order to construct a differential characteristic with high probability, we choose the differences utilizing the following properties:

Property 1. The first 2 bits of 8-bit input which enters the S-box have no effect on the output because of the bitwise AND operation with $0x3f$.

Property 2. The input differences $0x01$ and $0x25$ cause zero output differences with probability 2^{-5}.

Property 1 enables us to have 1-round differentials with probability 1. Also, using Property 2, the number of active S-boxes can be decreased.

The inputs of the F-function are the 1^{st} and the 3^{rd} 32-bit words of the data which are interchanging in the swap operation. There is no rotation operation applied on the 3^{rd} word and the rotation on the 1^{st} word is a 1-bit right rotation. Therefore, if we have 80000000_x as input difference in the 3^{rd} word, this difference will produce zero differences after the F-function with probability 1 during the next four rounds by the first property. We extend such a 4-round characteristic by adding 3 rounds to the beginning and 2.5 rounds to the end of it. The best characteristic that we found for TWIS has probability 2^{-18} and is given in Table 2. For simplicity, we use the alternative round function depicted in Figure 3. In Table 2, the values ΔI_i refer to the input differences of the corresponding round. The output differences are not given additionally as they are the input differences of the next round.

Table 2. 9.5-round Differential Characteristic

Rounds	ΔI_0	ΔI_1	ΔI_2	ΔI_3	**# Active S-boxes**	**I/O Diff. for S-box**	**Probability**
1	02000000_x	00000000_x	00000000_x	$0000A600_x$	1	$0x02 \rightarrow 0xA6$	2^{-4}
2	00000000_x	00000000_x	01000000_x	00000000_x	1	$0x01 \rightarrow 0x00$	2^{-5}
3	01000000_x	00000000_x	00000000_x	00000000_x	1	$0x01 \rightarrow 0x00$	1*
4	00000000_x	00000000_x	00800000_x	00000000_x	0	-	1
5	00800000_x	00000000_x	00000000_x	00000000_x	0	-	1
6	00000000_x	00000000_x	00400000_x	00000000_x	0	-	1
7	00400000_x	00000000_x	00000000_x	00000000_x	0	-	1
8	00000000_x	00000000_x	00200000_x	00000000_x	1	$0x20 \rightarrow 0x83$	2^{-4}
9	00200000_x	00000000_x	80000041_x	00000000_x	2	$0x20 \rightarrow 0x83$ $0x01 \rightarrow 0x00$	2^{-5}*
9.5	80000041_x	80000041_x	00100000_x	00000000_x	1	$0x01 \rightarrow 0x00$	1*
	80000041_x	00004180_x	80100041_x	$C0000020_x$	-	-	

Notice that, in Table 2, the probability values of some rounds are marked with an asterisk(*) and these values are also relatively higher when considering the number of active S-boxes. The reason for high probability is that the cipher uses the same subkey for two consecutive G-functions and this makes them identical. To clarify, let x and $\bar{x}$ be two input values to G and y, $\bar{y}$ be the two corresponding output values. Then, if x and $\bar{x}$ are input to the next G-function which uses the same subkey, the outputs will again be y and $\bar{y}$. Hence, if an input pair with input difference Δx produces outputs with difference Δy with some probability p in G, then the same output difference Δy is produced with probability 1 when the input difference is Δx for the next G-function that uses the same subkey. Therefore, the probability of a differential characteristic that involves such G-functions is p instead of p^2. If each G-function were using different subkeys, the probability of the characteristic would be 2^{-32}.

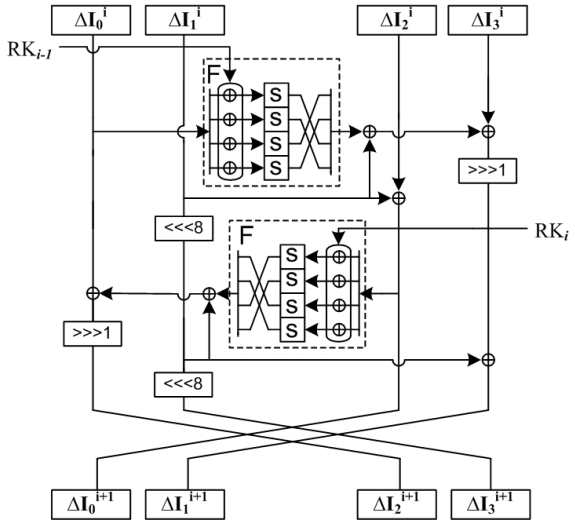

Fig. 3. Alternative Round Function

3.2 10-round Differential Attack

We perform a key-recovery attack on 10-round TWIS, excluding the final key whitening, by using the 9.5-round differential characteristic given in Section 3.1 and recover 12 bits of the last round subkey RK_{10}. Adding a half round to the end of the given 9.5-round differential characteristic and simply tracing the differences, we obtain the difference between ciphertext pairs as (80100041_x,$C00041A0_x$, $????????_x$, 00418000_x).

The attack proceeds as follows:

1. Take $N = c.2^{18}$ plaintext pairs $P^i = (P_0^i, P_1^i, P_2^i, P_3^i)$, $P^{i^*} = (P_0^{i^*}, P_1^{i^*}, P_2^{i^*}, P_3^{i^*})$ such that $P^i \oplus P^{i^*} = (02000000_x, 00000000_x, 00000000_x, 0000A600_x)$ and obtain their corresponding ciphertexts $C^i = (C_0^i, C_1^i, C_2^i, C_3^i)$, $C^{i^*} = (C_0^{i^*}, C_1^{i^*}, C_2^{i^*}, C_3^{i^*})$ by encrypting these plaintexts for 10 rounds of TWIS.
2. Check the first 64-bit and the last 32-bit ciphertext difference whether $C_0^i \oplus C_0^{i^*} = 80100041_x$, $C_1^i \oplus C_1^{i^*} = C00041A0_x$ and $C_3^i \oplus C_3^{i^*} = 00418000_x$ and keep the text pairs satisfying these equations.
3. As the input differences of the S-boxes in the 10^{th} round are $0x3f \cdot 80 = 0x0$, $0x3f \cdot 10 = 0x10$, $0x3f \cdot 00 = 0x0$ and $0x3f \cdot 41 = 0x01$, one can attack the 2^{nd} and 4^{th} 8-bit words of RK_{10}.

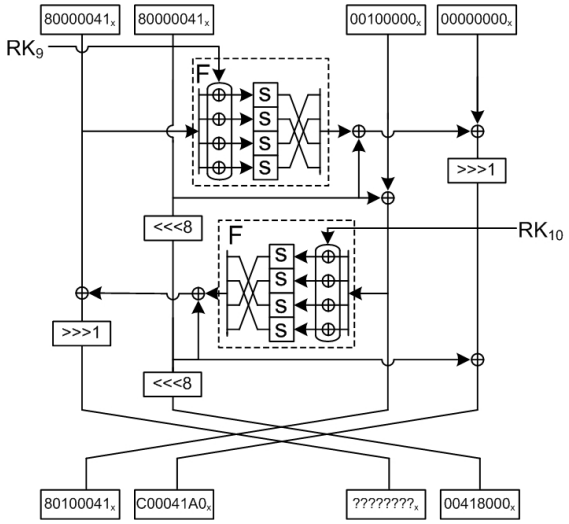

Fig. 4. Last Round of the Attack

However, since two bits of each word vanish after bitwise AND operation, we can retrieve 12 bits of the subkey. Therefore, keep a counter for each possible value of the 12 bits of the subkey RK_{10} corresponding to the second and the fourth bytes.

4. Inputs of the last F-function are (C_0^i, RK_{10}) and (C_0^{i*}, RK_{10}). XOR of output difference of this F-function and $((00418000_x) >>> 8)$ should be equal to the XOR of 80000041_x and $(\Delta C_2^i <<< 1)$. So, for each pair of plaintexts and their corresponding ciphertexts (C^i, C^{i*}), increment the counter for the corresponding value of the subkey RK_{10} when the following equations holds:

$$F(C_0^i, RK_{10}) \oplus F(C_0^{i*}, RK_{10}) \oplus 00004180_x = 80000041_x \oplus (\Delta C_2^i <<< 1).$$

5. Adopt the key with the highest counter as the right key.

The signal to noise ratio S/N of the attack is calculated as 2^5. This value can be calculated from

$$S/N = \frac{2^k \cdot p}{\alpha \cdot \beta}$$

where k is the key bits we try to derive, p is the probability, α is the average count of the subkeys per counted plaintext pair and β is the ratio of the counted pairs to all pairs. As we search for the 12 bits of the final subkey, $k = 12$, and the probability is $p = 2^{-18}$. The expected number of suggested subkeys is $\alpha = 2$. The checking condition for the output of the F-function is 12 bits. So, S/N ratio can be computed as

$$S/N = \frac{2^{-18} \cdot 2^{12}}{2 \cdot 2^{-12}} = 2^5.$$

According to [12], about $c = 4$ right pairs is enough to uniquely determine the 12 bits of RK_{10}. Therefore, the number of required plaintext pairs is $N = 4 \cdot 2^{18} = 2^{20}$ and this makes the data complexity of the attack 2^{21} chosen plaintexts. Step (1) requires 2^{21} 10-round encryptions. After Step (2), there remains $2^{20} \cdot 2^{-18} = 2^2$ right pairs. Step (3) requires 2^{12} counters. For Step (4), $2^2 \cdot 2 \cdot \frac{1}{2}$ 1-round computations are required which can be ignored. Hence, time complexity of this attack is 2^{21} 10-round encryptions and the memory complexity is 2^{12}. Moreover, as the

two attacked 6-bit words are independent from each other, one can keep two counters of 6 bits instead of a single counter of12 bits, which reduces the memory complexity to 2^7.

The implementation of the attack verifies the results given in this section. Using the reference implementation of TWIS and taking $c = 4$, it takes only 15 seconds on a laptop[1] to get the 12 bits of the final subkey. By optimizing the reference code, the attack time can be decreased.

4 Impossible Differential Distinguisher for TWIS

Impossible differential analysis[13, 14] is a variant of differential analysis. The fundamental difference between two analysis methods is that in differential cryptanalysis the attacker tries to exploit possible input-output difference pairs to get information about the correct key, while in impossible differential cryptanalysis the attacker tries to find events that never occur and use differentials with probability zero, called impossible differentials. In this section, we analyze the security of TWIS with respect to impossible differential cryptanalysis and present a distinguisher of 9.5 rounds.

While building the impossible differential characteristic, we were inspired from the differential characteristic given in Table 2. We combine two differential characteristics with probability one and obtain a contradiction by using the miss-in-the-middle approach[15]. The impossible differential characteristic is depicted in Figure 5, in which "0" denotes the 32-bit word consisting of all zeros.

In the left part of Figure 5, the input difference (0,0,Δy,0), Δy=00800000_x, is proceeded for 4.5 rounds in the forward direction and the difference (Δt,0,0,0), Δt=00200000_x, is obtained. On the other part, starting from the last round of the characteristic, the output difference (Δt,0,0,0) is traced backwards for 5 rounds and (0,0,Δx,0) difference where Δx=01000000_x, is acquired. However, we cannot have (Δt,0,0,0) = (0,0,Δx,0) since both Δt and Δx are non-zero differences. Therefore, (0,0,Δy,0) $\nrightarrow$ (Δt,0,0,0) after 9.5 rounds.

This characteristic can be extended to an impossible differential attack by adding half round to the beginning of the characteristic. By guessing the initial subkeys, wrong values can be eliminated and one will be left with the actual value of the subkeys.

5 Linear Distinguisher for TWIS

Linear cryptanalysis[16], is a generic method which exploits the linear relations among plaintext, ciphertext and key bits. Using the linear approximations of non-linear round function elements, one can gather information on the key bits. One can also provide distinguishers for an algorithm by using its linear properties. In this section, we present linear distinguishers for full TWIS with probability 1. These distinguishers, similar to the impossible differential distinguisher, can be extended to a key recovery attack by adding some number of rounds to the beginning or the end of the characteristic.

In order to obtain the linear characteristics, we use Property 1 in Section 3.1. Since the bits in the first two positions of the S-box input are omitted, if we choose masks that trace those bits, after the S-box operation the mask vanishes. There are 4 S-boxes in the F-function, so there are 8 bit positions that are omitted in an F-function. Therefore, one can find 2^8 trivial masks that vanishes after the first half of the first round. These masks can be used to distinguish the cipher from a random mapping. Moreover, as the masks vanish in the first round, the distinguishers apply arbitrary number of rounds of TWIS. Also, by choosing proper bits, one can increase the number of masks which vanish in the following rounds. In Figure 5, one of the discussed

[1] 2.2 Ghz Intel Core2Duo Processor, 2 GB Ram, Ubuntu 10.10 64 bit Operating System.

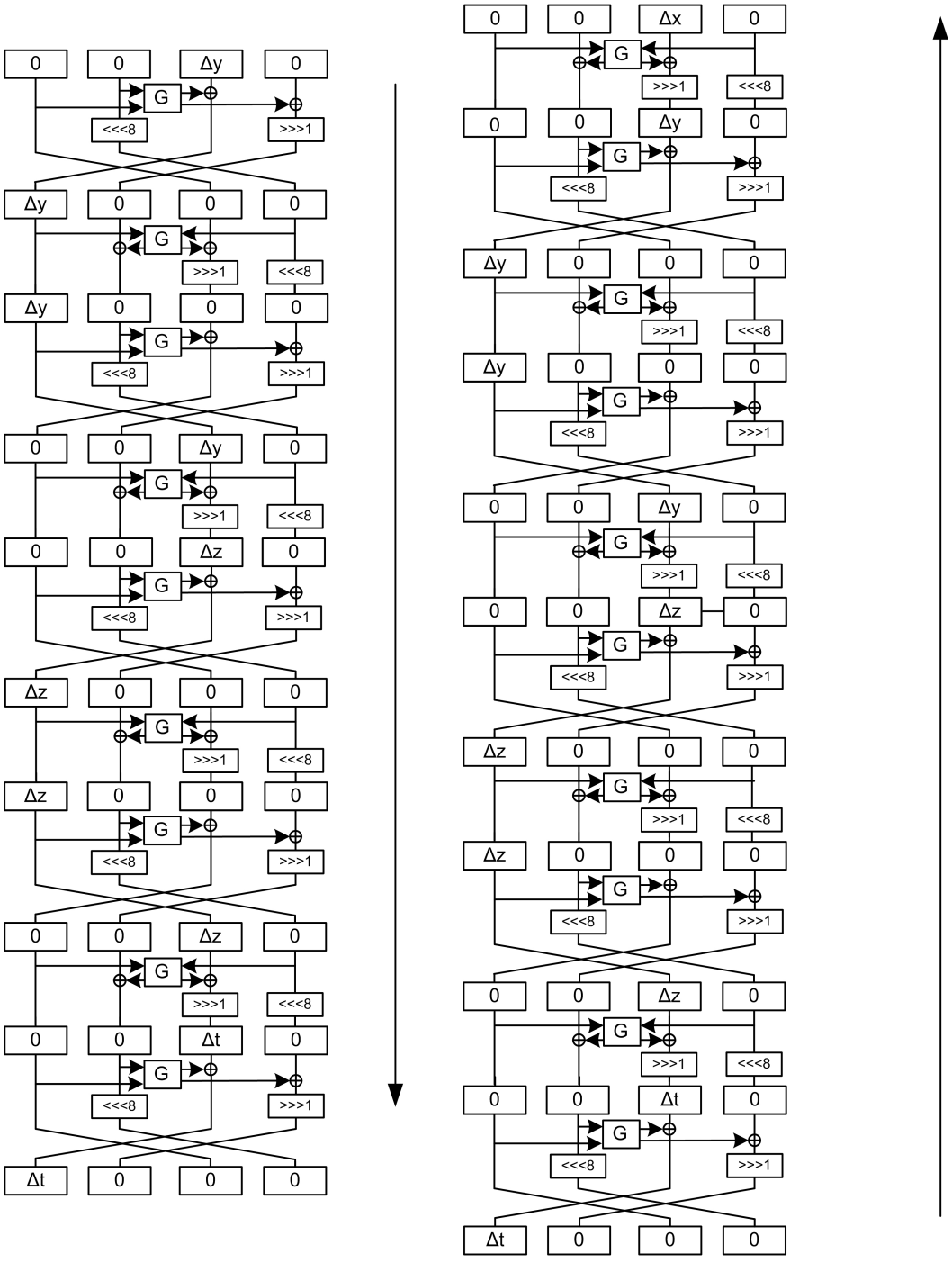

Fig. 5. Impossible Differential Characteristic where Δx=01000000$_x$, Δy=00800000$_x$, Δt=00200000$_x$, and Δz=00400000$_x$.

distinguishers is illustrated where [0] denotes the linear mask on the first bit of 32-bit word and [−] denotes no masking.

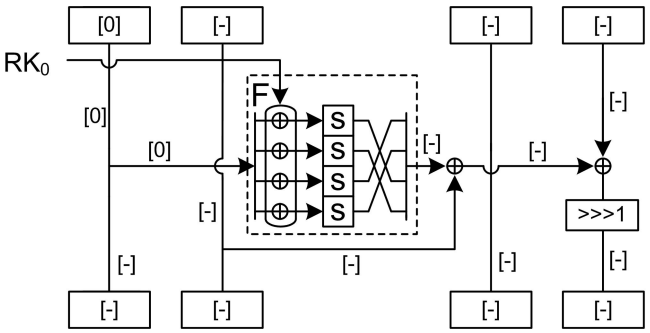

Fig. 6. Linear Distinguisher for TWIS

6 Key Related Observations

This section is devoted to the observations on TWIS block cipher. These observations, which are mainly on key scheduling algorithm, include very basic design flaws like actual key size and trivial related key distinguishers that compromise the security of the algorithm.

The most important flaw with the key schedule is that it does not use all bits of the master key. Instead, it uses only 62 bits of the 128-bit key. The first subkey is generated from the first 3 and the last 29 bits of the master key. Each remaining subkeys will be generated by left rotation of the modified key by 3. So, in order to generate 10 more subkeys, algorithm uses the first 33 bits of the master key. Therefore, key scheduling algorithm uses the first 33 and the last 29 bits of the key to derive the 11 subkeys. Hence the security of the cipher is 62-bit instead of claimed 128-bit.

Another flaw arises from the S-box used in the key schedule. The S-box is used in the same manner with data processing part, so, one can find many related key distinguishers for TWIS. In order to form a related key distinguisher, it is enough to use a difference between two keys, where the difference coincides to the bit positions that are not processed by the S-box. For example, take K and $\overline{K}$ such that $\overline{K_0} = K_0 \oplus 0x10$. After the initial 3-bit rotation, the difference $0x10$ becomes $0x80$. Then, applying S-box to both K and $\overline{K}$, one gets $S(K \wedge 0x3f) = S(\overline{K} \wedge 0x3f)$ and the difference between the keys will be cancelled after S-box operation. Since there is no other difference between the keys, all the subkeys will be exactly the same. This means, if one encrypts P with K and $\overline{K}$, he gets the same ciphertexts with probability one. The number of related key distinguishers can be increased by choosing the key differences that coincide the first two bit positions of 8-bit S-box input.

Also, in the data processing part, the data is XORed with the subkey and then S-box is applied to the XORed data. As S-box ignores the first two bits of the 8-bit input, 8 bits of the key is thrown away after this operation. So, the actual subkeys are 24 bits instead of 32 bits.

The key whitening, which is introduced to increase the security, is used in an inappropriate way. Notice that RK_0 is XORed to P_0 as the key whitening which also again XORed to P_0 in the first round inside the G-function. In this way RK_0 will be cancelled in G and it has no effect on the first G-function. So, if one knows the plaintext or specifically P_0, then he also knows the output of the G-function without knowing the key. Therefore, the cipher can be considered as 9,5 rounds.

Furthermore, the choice of final whitening subkeys results in a weakness. If one can determine the whole 32-bits of RK_2 and RK_{10} by attacking the final round, he can also determine the subkeys in between trivially. As the S-box is not invertible (one has to guess the ignored two bits) and there are 3 unknown bits coming from left rotation, it is not possible to go backwards from RK_{10}. Also, one cannot go forwards from RK_2 because of the rotation. However, knowing both, one can determine the missing bits and recover the subkeys $RK_3, RK_4, \ldots, RK_9$ by going backwards from RK_{10}, forwards from RK_2, and checking the known bits. One can recover RK_1, by going backwards from RK_2, with 2^5 computations since there are 3 unknown bits from the rotation and 2 unknown bits from the inverse of the S-box. Similarly, RK_0 can be recovered using RK_1 with 2^5 complexity.

Besides, the diffusion of the key bits into the plaintext is not sufficient. This is a result of using an 8-bit word-wise permutation instead of a bitwise permutation and 8-bit S-box. This enables the attacker to mount an exhaustive search for a 32-bit subkey by dividing it into four 8-bit words without the knowledge of the remaining 24 bits. The complexity of such a search will be $4 \cdot 2^8 = 2^{10}$ instead of 2^{32}. However, in TWIS case, since the S-box ignores two input bits, one can recover the active subkey with $4 \cdot 2^6 = 2^8$ complexity.

7 Conclusion and Future Work

In this paper, we analyze the security of TWIS block cipher against differential, impossible differential and linear attacks. Our results show that 10-round TWIS, when we exclude the final key whitening, is not resistant against differential attack. We recover half of the active key bits with 2^{21} chosen plaintexts. Also, we present distinguishers using the impossible differential and linear analysis techniques. These distinguishers can be extended to key recovery attacks. Finally, we propose some important observations on the algorithm.

As future a work, we aim to apply the mentioned attacks on full TWIS and mount related-key attacks by using the weaknesses in the key schedule.

References

1. National Institute for Science and Technology (NIST). Advanced Encryption Standard (FIPS PUB 197). http://www.csrc.nist.gov/publications/fips/fips197/fips-197.pdf, 2001.
2. National Institute of Standards and Technology. Federal Information Processing Standard 180-2 Secure Hash Standard. http://csrc.nist.gov/publications/fips/, 2002.
3. William C. Barker, National Institute of Standards, and Technology (U.S.). *Recommendation for the Triple Data Encryption Algorithm (TDEA) block cipher [electronic resource] / William C. Barker*. U.S. Dept. of Commerce, Technology Administration, National Institute of Standards and Technology, Gaithersburg, MD :, 2004.
4. Ronald L. Rivest. The MD5 Message-Digest Algorithm. http://tools.ietf.org/rfc/rfc1321.txt, 1992.
5. Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In *CHES*, pages 450–466, 2007.
6. Christophe De Cannière, Orr Dunkelman, and Miroslav Knezevic. KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers. In *CHES*, pages 272–288, 2009.
7. Axel Poschmann, Gregor Leander, Kai Schramm, and Christof Paar. New Light-Weight Crypto Algorithms for RFID. In *ISCAS*, pages 1843–1846, 2007.
8. Martin Hell, Thomas Johansson, and Willi Meier. Grain: A Stream Cipher for Constrained Environments. *IJWMC*, 2(1):86–93, 2007.
9. Shrikant Ojha, Naveen Kumar, Kritika Jain, and Sangeeta Lal. TWIS - A Lightweight Block Cipher. In *ICISS*, pages 280–291, 2009.
10. Taizo Shirai, Kyoji Shibutani, Toru Akishita, Shiho Moriai, and Tetsu Iwata. The 128-Bit Blockcipher CLEFIA (Extended Abstract). In *FSE*, pages 181–195, 2007.
11. Bozhan Su, Wenling Wu, Lei Zhang, and Yanjun Li. Some Observations on TWIS Block Cipher. Cryptology ePrint Archive, Report 2010/066, 2010. http://eprint.iacr.org/.

12. Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In *CRYPTO*, pages 2–21, 1990.
13. Lars Knudsen. DEAL - A 128-bit Block Cipher. In *NIST AES Proposal*, 1998.
14. Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack Reduced to 31 Rounds Using Impossible Differentials. In *EUROCRYPT*, pages 12–23, 1999.
15. Eli Biham, Alex Biryukov, and Adi Shamir. Miss in the Middle Attacks on IDEA and Khufu. In *FSE*, pages 124–138, 1999.
16. Mitsuru Matsui. Linear Cryptoanalysis Method for DES Cipher. In *EUROCRYPT*, pages 386–397, 1993.