Breaking the DECT Standard Cipher with Lower Time Cost

Lin Ding, Zhengting Li, Ziyu Guan, Xinhai Wang and Zheng Wu

***Abstract*—The DECT Standard Cipher (DSC) is a proprietary stream cipher used for encryption in the Digital Enhanced Cordless Telecommunications (DECT), which is a standard for short range cordless communication and widely deployed worldwide both in residential and enterprise environments. New weaknesses of the DSC stream cipher which are not discovered in previous works are explored and analyzed in this paper. Based on these weaknesses, new practical key recovery attacks and distinguishing attack on DSC with lower time cost are proposed. The first cryptanalytic result show that DSC can be broken in about 13.12 seconds in the known IV setting, when an offline phase that takes about 58.33 minutes is completed. After then, a distinguishing attack on DSC in the related key chosen IV setting is given, which has a time complexity of only 2 encryptions and a success probability of almost 1. Finally, based on the slide property, a key recovery attack on DSC with practical complexities is proposed. The experimental result shows that DSC can be broken on a common PC within about 44.97 seconds in the multiple related key setting. The attacks on DSC proposed in this paper clearly show that a well-designed initialization is absolutely necessary to design a secure stream cipher.**

***Index Terms*—Cryptanalysis, DECT Standard Cipher, Time-Memory-Data Trade-Off attack, Differential collision attack, Slide attack.**

I. Introduction

DIGITAL Enhanced Cordless Telecommunications (DECT), created by the European Telecommunications Standards Institute (ETSI) in the late 1980s, is a digital wireless technology for cordless telephony that is used for both domestic and business purposes [1]. It is designed for short-range which acts as an access method to major networks. Due to the flexible nature of DECT, it is one of the most commonly used systems for cordless phones besides Global System for Wireless Communications (GSM), Universal Mobile Telecommunications Service (UMTS) and Code Division Multiple Access (CDMA) around the globe. The number of DECT devices sold reaches 820 million with a proliferation of 100 million new devices per year. In the DECT standard, the DECT Standard Authentication Algorithm (DSAA) [2] is used to provide mutual authentication of devices, and the DECT Standard Cipher (DSC) [3] is used to provide encryption of the payload. However both features are optional and need not be implemented on a device. In 2009, two attacks on DECT [2,4] were presented and showed that some devices do not use encryption and authentication at all and can easily be eavesdropped on. After then, more works on the security of DECT voice communications were presented in [5]–[8]. It should be noted that all these attacks are not cryptanalytic attacks on the DSC stream cipher, since all of them are irrelevant to the detailed design of DSC.

The DECT Standard Cipher (DSC) is an asynchronous stream cipher used for encrypting payload of DECT transmissions such as cordless telephone calls. The stream cipher was kept secret, until it was publicly disclosed by Nohl, Tews and Weinmann [3] at FSE 2010. It was reverse-engineered from a DECT device using a combination of firmware probing and hardware reverse-engineering. As disclosed in [3], DSC is similar to the A5/1 stream cipher in GSM, and takes a 64-bit secret key and a 35-bit initialization vector (IV) to generate a keystream of variable length. The DECT standard supports frames of different lengths and formats. For common voice calls, each key and IV pair is used to generate a keystream of 720 bits which is split into two keystream segments of 360 bits each. The first 360 keystream bits are used to encrypt traffic from the base station to the phone, and the remaining 360 keystream bits are used to encrypt the frames sent by the phone. In each case, the first 40 keystream bits can be used to encrypt the C-Channel data (that contains control data), and the remaining 320 keystream bits are used to encrypt the actual voice data (B-field). If a frame contains no C-channel data, the first 40 bits are discarded. For more details, we refer the readers to the ETSI DECT standard [1].

Related works. Since the DSC stream cipher is a standard cipher used to provide confidentiality for cordless telephony in the digital wireless technology DECT, it has attracted a large amount of attention in the recent years due to its importance. Up to now, several attacks on DSC [3,9-12] have been proposed. In [3], beyond disclosing DSC, Nohl, Tews and Weinmann proposed the first cryptanalytic attack on DSC, often called NTW attack. The basic idea of NTW attack is to guess some internal state bits to remove the irregular clock control and then obtain a sufficiently large number of linear equations over the key and IV bits. To obtain 30 linear equations and reach a probability of success of 0.5, their attack requires at least 2^{15} available C-Channels generated by the same key together with 2^{15} different IVs. The NTW attack

Manuscript received October 12, 2023, accepted January 14, 2024. This work was supported by the National Natural Science Foundation of China under Grant 61602514, 61802437, 61902428, 62202493. (*Corresponding authors*: *Lin Ding*, *Zhengting Li*.)

Lin Ding is with the PLA SSF Information Engineering University, Zhengzhou 450001, China (e-mail: dinglin_cipher@163.com).

Zhengting Li is with the PLA SSF Information Engineering University, Zhengzhou 450001, China (e-mail: lizhengting0225@163.com).

Ziyu Guan is with the PLA SSF Information Engineering University, Zhengzhou 450001, China (e-mail: a1027495051@163.com).

Xinhai Wang is with the PLA SSF Information Engineering University, Zhengzhou 450001, China (e-mail: wxh1066559569@126.com).

Zheng Wu is with the PLA SSF Information Engineering University, Zhengzhou 450001, China (e-mail: g7001162@163.com).

has to solve a system of 30 linear equations following an exhaustive search over 34 key bits to recover the 64-bit secret key of DSC, which leads to a time complexity of about $2^{40.10}$ encryptions. Finally, the authors pointed out that an important reason that the DSC stream cipher is vulnerable against their attack is an insufficiently small number of initialization rounds before producing the first keystream bit.

After then, Weiner et al. [9] presented an optimized NTW attack and also an optimized FPGA implementation of their optimized NTW attack. This attack is able to more than double the success probability of the original NTW attack, depending of the number of available keystreams. In [10,11], Coisel and Sanchez presented an improved cryptanalysis of NTW attack, often called CS attack. The CS attack is able to quickly recover the secret key with a success rate of about 0.55 by analyzing 2^{13} available C-Channels generated by the same key together with 2^{13} different IVs and performing an exhaustive search over 2^{31} keys. As estimated in [12], the CS attack has a total time complexity of $2^{37.10}$. Based on an information collection method with unknown memory, Liu and Jin [12] introduced two improved attacks of CS attack. The one has the same time complexity with CS attack, while the success probability is improved from 0.55 to 0.89. The other has the same success probability with CS attack, while the time complexity is reduced by a factor of $2^{2.2}$. It should be noted that all these attacks [9]–[12] are follow-up works of NTW attack. If the number of initialization rounds of DSC is sufficiently increased, all existing attacks on DSC will not work anymore.

Our contributions. In this paper, new weaknesses of the DSC stream cipher which are not discovered in previous works are explored and analyzed. The comparisons of our cryptanalytic results with the previous attacks are listed in Table I. The contributions of this paper can be summarized as follows.

- Firstly, a dedicated Time-Memory-Data Trade-Off (TMDTO) attack on DSC is proposed with an online time complexity of $2^{32.32}$ encryptions, which improves the best cryptanalytic result of DSC by a factor of $2^{2.58}$, and an offline time complexity of $2^{40.38}$ encryptions. By using the optimized FPGA implementation of DSC in [9], the online and offline phase of our attack can be done in about 13.12 seconds and 58.33 minutes. Note that since the offline phase requires to be performed only once, our attack that has a low online time complexity is still very effective, particularly when the attacker wants to recover many secret keys in the online phase. The attack requires a memory space of 28.64 TiB, which is obviously feasible on a current PC. In addition, the attack requires $2^{40.67}$ keystream bits, which can be generated by $2^{31.18}$ known IVs and each known IV generate a keystream no more than 720 bits. Thus, the data complexity of our attack is entirely possible, which makes the attack feasible in practice. The attack has a success probability of 0.632. It should be noted that both of the previous attacks and the dedicated TMDTO attack on DSC are meaningful in reality. Specifically, when the standard stream cipher DSC is used to encrypt a short message that consists of a small amount of frames, the previous attacks are valid. In contrary, our dedicated TMDTO attack is clearly more effective when a long message needs to be encrypted with DSC.
- Secondly, a differential collision attack on DSC is proposed, based on the observation that the total size of the Key and IV of DSC is bigger than the total size of all four LFSRs. By solving the system of linear equations, a large number of differential collisions of DSC are found. As results, a distinguishing attack on DSC in the related key chosen IV setting is given, which has a time complexity of only 2 encryptions and requires 2 chosen IVs. The success probability is almost 1.
- Finally, the slide property of DSC is discovered and analyzed, and the result shows that two different Key-IV pairs can generate 1-bit shifted keystreams with probability $2^{-2.35}$ when some condition is satisfied. Clearly, this is a high probability compared with an ideal stream cipher that generates random keystreams. Based on the slide property, a key recovery attack on DSC with practical complexities in the multiple related key setting is proposed. We validate the cryptanalytic result by simulating the whole attack process on a common PC. The experimental result shows that the DSC stream cipher can be broken on a common PC within about 44.97 seconds in the multiple related key setting.

It is important to note that unlike all previous cryptanalytic attacks on DSC in [3,9-12], increasing the number of initialization rounds of DSC can not strengthen the resistance of DSC against our attacks proposed in this paper. That is because all attacks proposed in this paper are irrelevant to the number of initialization rounds of DSC. Thus, the weaknesses of DSC discovered in this paper are different from the ones pointed out by previous works.

Feasibility and Impact of Our Attacks. In this paper, new practical key recovery attacks and distinguishing attack on DSC with lower time cost are proposed. In the following, we briefly discuss the real-world attack feasibility and the attack implications.

- To execute the attacks on DSC, the attacker needs to collect enough keystreams. In other words, the attacker needs to record the raw DECT data being sent over the wireless interface. As stated in [9], this can be easily achieved for the attacker by using a DECT PC-Card with a modified firmware or a generic software radio. Since the IV is generally used as an increasing counter for each frame, it is entirely feasible for the attacker to collect enough keystreams generated by the same key and different IVs.
- Nowadays, the DECT standard has reinforced its position as one of the main wireless communication protocols in Smart Home ecosystems, despite the massive adoption of mobile telephony [11]. The DSC stream cipher is designed to provide an adequate level of security against eavesdropping of DECT communications. Currently, the privacy of the personal voice communications of hundreds of millions of citizens depends on the security of

TABLE I
NEW ATTACKS ON DSC AND COMPARISONS WITH PREVIOUS ATTACKS

Attacks	Setting	Online time complexity	Data complexity	Memory complexity	Offline time complexity	IVs	Related keys	Success probability	Ref.
Key recovery attack	Single key	$2^{40.10}$ encryptions	$2^{15} \times 40 \approx 2^{20.32}$ bits	-	-	2^{15}	-	0.50	[3]
Key recovery attack	Single key	$2^{40.10}$ encryptions	$2^{15} \times 40 \approx 2^{20.32}$ bits	-	-	2^{15}	-	0.90	[9]
Key recovery attack	Single key	$2^{37.10}$ encryptions	$2^{13} \times 40 \approx 2^{18.32}$ bits	-	-	2^{13}	-	0.55	[10,11]
Key recovery attack	Single key	$2^{37.10}$ encryptions	$2^{13} \times 40 \approx 2^{18.32}$ bits	-	-	2^{13}	-	0.89	[12]
Key recovery attack	Single key	$2^{34.90}$ encryptions	$2^{13} \times 40 \approx 2^{18.32}$ bits	-	-	2^{13}	-	0.55	[12]
Key recovery attack	Single key	$2^{32.32}$ encryptions	$2^{31.18} \times 720 \approx 2^{40.67}$ bits	28.64 TiB	$2^{40.38}$ encryptions	$2^{31.18}$	-	0.632	**Ours**
Distinguishing attack	Related key	2 encryptions	$2 \times 40 = 80$ bits	-	-	2	1	1	**Ours**
Key recovery attack	Related key	$2^{22.01}$ encryptions	$2^{20.81}$ bits	-	-	$2^{11.81}$	111	1	**Ours**

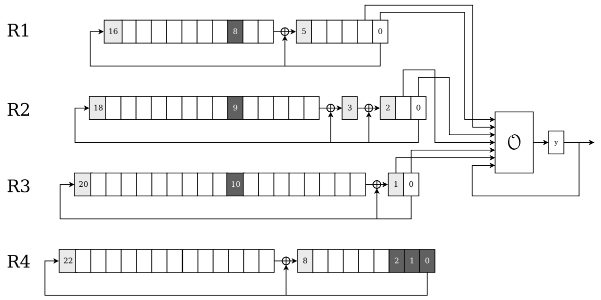

Fig. 1. An overview of the keystream generator of DSC

this encryption algorithm. Once the secret key of DSC is recovered, the attacker can eavesdrop the actual DECT communications arbitrarily, which is probably a serious threat to the DECT communication users.

This paper is organized as follows. A brief description of the DSC stream cipher is given in section II. In section III, a dedicated Time-Memory-Data Trade-Off attack on DSC is introduced. In section IV, the differential collision attack on DSC is given. The slide attacks on DSC are presented in section V. Concluding remarks are given in Section VI.

II. A Brief Description of DSC

In this section, we introduce briefly the DECT Standard Cipher (DSC). The full description is detailed in [3]. An overview of the keystream generator of DSC is depicted in Fig. 1.

The internal state of DSC consists of 4 Galois LFSRs $R1, R2, R3$ and $R4$ of lengths 17, 19, 21 and 23 respectively, as well as a single memory bit y for the output combiner. The states of these four LFSRs at time t are denoted as $\left(a_0^{(t)}, \cdots, a_{16}^{(t)}\right)$, $\left(b_0^{(t)}, \cdots, b_{18}^{(t)}\right)$, $\left(c_0^{(t)}, \cdots, c_{20}^{(t)}\right)$ and $\left(d_0^{(t)}, \cdots, d_{22}^{(t)}\right)$, respectively. The update functions of these four LFSRs are given as follows.

LFSR $R1$:

$$a_{16}^{(t+1)} = a_0^{(t)},\ a_5^{(t+1)} = a_6^{(t)} \oplus a_0^{(t)}$$
$$a_i^{(t+1)} = a_{i+1}^{(t)} \text{ for } i \in \{0, \cdots, 4, 6, \cdots, 15\}$$

LFSR $R2$:

$$b_{18}^{(t+1)} = b_0^{(t)},\ b_3^{(t+1)} = b_4^{(t)} \oplus b_0^{(t)},\ b_2^{(t+1)} = b_3^{(t)} \oplus b_0^{(t)}$$
$$b_i^{(t+1)} = b_{i+1}^{(t)} \text{ for } i \in \{0, 1, 4, \cdots, 17\}$$

LFSR $R3$:

$$c_{20}^{(t+1)} = c_0^{(t)},\ c_1^{(t+1)} = c_2^{(t)} \oplus c_0^{(t)}$$
$$c_i^{(t+1)} = c_{i+1}^{(t)} \text{ for } i \in \{0, 2, \cdots, 19\}$$

LFSR $R4$:

$$d_{22}^{(t+1)} = d_0^{(t)},\ d_8^{(t+1)} = d_9^{(t)} \oplus d_0^{(t)}$$
$$d_i^{(t+1)} = d_{i+1}^{(t)} \text{ for } i \in \{0, \cdots, 7, 9, \cdots, 21\}$$

The $R4$ is a regularly clocked LFSR, while the other three, i.e., $R1, R2, R3$, are irregularly clocked LFSRs. For each clock, $R4$ is clocked three times whereas each of $R1, R2, R3$ is clocked either two or three times. More specifically, the number of times e_i that the LFSR $Ri\,(i = 1, 2, 3)$ is clocked is calculated as follows.

$$e_1 = 2 + \left(d_0^{(t)} \oplus b_9^{(t)} \oplus c_{10}^{(t)}\right)$$
$$e_2 = 2 + \left(d_1^{(t)} \oplus a_8^{(t)} \oplus c_{10}^{(t)}\right)$$
$$e_3 = 2 + \left(d_2^{(t)} \oplus a_8^{(t)} \oplus b_9^{(t)}\right)$$

The output combiner function f is a cubic Boolean function on seven variables, which consist of the rightmost two bits of the LFSRs $R1, R2, R3$ as well as the memory bit y. The specification of f is given in algebraic normal form as

$$\begin{aligned} & f\left(a_0^{(t)}, a_1^{(t)}, b_0^{(t)}, b_1^{(t)}, c_0^{(t)}, c_1^{(t)}, y\right) \\ = {} & a_1^{(t)} a_0^{(t)} y \oplus b_1^{(t)} a_0^{(t)} y \oplus c_0^{(t)} a_0^{(t)} y \oplus b_0^{(t)} a_1^{(t)} a_0^{(t)} \\ & \oplus b_1^{(t)} b_0^{(t)} a_0^{(t)} \oplus c_0^{(t)} b_0^{(t)} a_0^{(t)} \oplus a_1^{(t)} y \oplus c_0^{(t)} y \oplus c_1^{(t)} y \\ & \oplus a_1^{(t)} a_0^{(t)} \oplus b_0^{(t)} a_1^{(t)} \oplus c_1^{(t)} a_0^{(t)} \oplus b_1^{(t)} \oplus c_1^{(t)} \end{aligned}$$

The output of the output combiner function f gives a keystream bit and is loaded into the memory bit for the next clock.

Initialization. The DSC stream cipher takes a 64-bit key and a 35-bit IV as input. At the beginning of initialization, all four LFSRs and the memory bit y are filled with zeros. Then each of four LFSRs is clocked 128 times. Here, the most significant bit of each LFSR is generated by the bit that is shifted out, XORed with one input bit. The 128 input bits are introduced in the sequence $k_0, \cdots, k_{63}, iv_0, \cdots, iv_{34}, 0, \cdots, 0$. Note that the irregular clock control is not used, and each LFSR is clocked once during this key and IV loading process. When all input bits have been loaded, 40 pre-cipher rounds are performed. In these pre-cipher rounds, the irregular clock control is used, while the output is discarded. If one or more

LFSRs become the all-zero state after executing 11 rounds, the most significant bits of the corresponding LFSRs are forcibly set to 1 before starting the next round.

III. A Dedicated Time-Memory-Data Trade-Off Attack on DSC

Recall the brief description of DSC above. After the key and IV loading process, the DSC stream cipher has to perform 40 pre-cipher rounds to complete the initialization. In these pre-cipher rounds, the irregular clock control is used, while the output is discarded. This means that each of 40 pre-cipher rounds in the initialization is the same with one round of the keystream generation apart from that the output is discarded in the initialization. Based on this important observation, a dedicated Time-Memory-Data Trade-Off (TMDTO) attack on DSC will be proposed in this section. Introduced independently by Babbage [13] and Golić [14], TMDTO attack offers a generic technique to reverse one-way functions, where one can trade off time, memory and data costs and which are especially effective against stream ciphers with small internal state size.

We start with constructing a Boolean function that the attacker tries to invert. The function f takes the 81-bit internal state which consists of the state of four LFSRs and the memory bit as input, and performs 121 rounds of the keystream generation, where a total of 121 keystream bits are generated. The function f intercepts the latter 81 bits from the 121 keystream bits as output. Like a typical TMDTO attack, our attack on DSC consists of two phases, i.e., the offline phase and the online phase. In the offline phase, the attacker pre-computes a table using the function f he is trying to invert. In the online phase, the attacker captures enough keystreams generated by the fixed key and different IVs, and checks if one keystream fragment happens to be in the table constructed in the offline phase. In the attack on DSC, we suppose that the attacker is given a set of r keystream fragments, i.e., $Z_1, \cdots, Z_r$, each of which is consecutive and consists of 81 bits, and he is asked to find a pre-image x of any one of these r keystream fragments such that $f(x) = Z_l$ with $1 \leq l \leq r$, where the pre-image x denotes an 81-bit internal state of DSC.

In the offline phase, the attacker constructs a two-column table, denoted as Q. The detailed process of the offline phase can be described as an algorithm, called **Algorithm 1**, as follows.

Algorithm 1 The offline phase

```
1. Randomly pick $m$ different pre-images, i.e., $x_1, \cdots, x_m$.
2. For $i$ from 1 to $m$, do the following:
    • For each pre-image $x_i$, compute $y_i = f(x_i)$;
3. Store the pairs $(y_i, x_i)$ in the two-column table $Q$ indexed
   by the value of $y_i$.
```

In **Algorithm 1**, the step 2 has to apply the function f to each of m different pre-images, and thus it takes m evaluations of the function f in time. Note that one evaluation of the function f indicates the time cost of generating 121 keystream bits. In step 3 of **Algorithm 1**, the pairs (y_i, x_i) are stored in the two-column table Q indexed by the value of y_i, and thus this step takes about $m\log_2 m$ simple comparisons in time. When m is not large, it is easy to see that the time cost of $\log_2 m$ simple comparisons is far less than the time cost of one evaluation of the function f. Therefore, the offline phase takes about $P = m$ evaluations of the function f in time, and requires a memory space of $M = (81 + 81)\, m \approx 2^{7.34} m$ bits.

In the online phase, the attacker wishes to find one pre-image x such that $f(x) = Z_l$ with $1 \leq l \leq r$, using the two-column table Q constructed in the offline phase. The detailed process of the online phase can be described as an algorithm, called **Algorithm 2**, as follows.

Algorithm 2 The online phase

```
1. For $l$ from 1 to $r$, do the following:
    • Check whether $Z_l$ is in the first column of the two-column
      table $Q$. If yes, read the corresponding $x$ in the second
      column of the two-column table $Q$ and go to step 2;
      otherwise, go back to try the next keystream fragment.
2. Output the found pre-image $x$.
```

In step 1 of **Algorithm 2**, it has to make a check for each of r keystream fragments. Thus, the online phase takes $T = r$ table lookup operations in time.

Since there are r keystream fragments which are available to the attacker and the offline phase covers $P = m$ pre-images of the function f, the probability that there is at least one keystream fragment passing the check of **Algorithm 2**, denoted as p, can be calculated as

$$p = 1 - \left(1 - \tfrac{m}{N}\right)^r \approx 1 - e^{-\frac{mr}{N}}$$

Where $N = 2^{81}$ denotes the total number of the pre-images of the function f. Thus, $mr \geq N$ should be satisfied such that at least one pre-image is found with a significant success probability in the online phase.

Now, we have obtained a dedicated TMDTO attack on the DSC stream cipher. The attack has an offline time complexity of $P = m$ evaluations of the function f, and an online time complexity of $T = r$ table lookup operations. It requires a memory space of $M = 2^{7.34} m$ bits and r keystream fragments, each of which is consecutive and consists of 81 bits. The attack succeeds with a probability of about $p = 1 - e^{-\frac{mr}{N}}$. Since each key and IV pair can only generate a keystream with a limited length of 720 bits, and thus $720 - 81 + 1 = 640 \approx 2^{9.32}$ keystream fragments can be available to the attacker for each known IV. It means that the r keystream fragments can be generated by $2^{-9.32} r$ known IVs. The IV size of DSC is 35, and then it has $2^{-9.32} r \leq 2^{35}$. Thus, the restriction $r \leq 2^{44.32}$ must be satisfied in the attack. The data complexity of the attack can be calculated as $2^{-9.32} r \cdot 720 \approx 2^{0.17} r$ bits.

It is easy to see that the time complexity units of the offline phase and online phase are different and should be unified to facilitate comparison. Let δ_1 denote the ratio of the time cost of one evaluation of the function f to the time cost of one

TABLE II
THE RESULTS OF TMDTO ATTACKS ON THE DSC STREAM CIPHER

m	T	M	D	P	p
$2^{38.5}$	$2^{34.25}$ encryptions	7.16 TiB	$2^{42.67}$ bits	$2^{38.38}$ encryptions	0.632
$2^{40.5}$	$2^{32.32}$ encryptions	28.64 TiB	$2^{40.67}$ bits	$2^{40.38}$ encryptions	0.632
$2^{42.5}$	$2^{30.39}$ encryptions	114.56 TiB	$2^{38.67}$ bits	$2^{42.38}$ encryptions	0.632

encryption of DSC, and then the attack has an offline time complexity of $P = m\delta_1$ encryptions. Since the two-column table Q constructed in the offline phase consists of m entries, thus one table lookup operation in the online phase consists of about $\log_2 m$ simple bit operations. Let δ_2 denote the ratio of the time cost of one simple bit operation to the time cost of one encryption of DSC, and then the attack has an online time complexity of $T = r\delta_2 \log_2 m$ encryptions. We have made an experiment on a PC with 2.5 GHz Intel Pentium 4 processor. In this experiment, we first execute 2^{24} evaluations of the function f, and it takes about 161.842 seconds. We also execute 2^{24} simple bit operations, and it takes about 0.015 second. In the meantime, we execute 2^{24} encryptions of DSC, and it takes about 175.973 seconds. Thus, we have obtained the experimental value of the ratios δ_1 and δ_2 as

$$\delta_1 = 161.842/175.973 \approx 2^{-0.12}$$
$$\delta_2 = 0.015/175.973 \approx 2^{-13.52}$$

Table II lists the results of TMDTO attacks on the DSC stream cipher varying according to different values of m and r. It is easy to see that there exists a trade-off between the complexities of our attack on DSC. As shown in Table II, $m = r = 2^{40.5}$ is a reasonable choice, which leads to a dedicated TMDTO attack on DSC with an online time complexity of $2^{32.32}$ encryptions and an offline time complexity of $2^{40.38}$ encryptions. As shown in the experiment above, executing 2^{24} encryptions of DSC takes about 175.973 seconds on a PC. If the attack is implemented on a PC, the online phase will take about 15.62 hours, while the offline phase will take about 173.70 days, which is too large in time cost. In fact, the time cost can be significantly reduced if a fast implementation is used. According to the optimized FPGA implementation of [9], about 408.8 million keys per second can be exhausted. Thus, we can implement the online phase and offline phase of our attack in about $2^{32.32} \big/ 408.8 \times 10^6 \approx 13.12$ seconds and $2^{40.38} \big/ 408.8 \times 10^6 \times 60 \approx 58.33$ minutes, when the optimized FPGA implementation is used in our attack. In addition, the attack requires a memory space of about 28.64 TiB which is obviously feasible on a current PC and a data complexity of $2^{40.67}$ bits. It should be noted that the required $2^{40.67}$ bits can be generated by $2^{-9.32} r = 2^{-9.32} \times 2^{40.5} = 2^{31.18}$ known IVs, as about $2^{9.32}$ keystream fragments can be available to the attacker for each known IV. Thus, the data complexity of our attack is entirely possible, which makes the attack feasible in practice. The attack has a success probability of 0.632.

IV. DIFFERENTIAL COLLISION ATTACK ON DSC

At the beginning of initialization, all four LFSRs are filled with zeros, and then the 64-bit key and 35-bit IV are loaded into the four LFSRs by clocking each of them 128 times. Note that the irregular clock control is not used, and each LFSR is clocked once during this key and IV loading process. For convenience of description, we call the state of four LFSRs after this process *initial state* and denote as $S^{(128)} = \left(a_0^{(128)}, \cdots, a_{16}^{(128)}, b_0^{(128)}, \cdots, b_{18}^{(128)}, c_0^{(128)}, \cdots, c_{20}^{(128)}, d_0^{(128)}, \cdots, d_{22}^{(128)}\right)$. It is easy to see that the total size of all four LFSRs is 80, which is smaller than the total size of the Key and IV, i.e., $64 + 35 = 99$. Thus, there must be collisions in the initial state of DSC. More specifically, there must be different Key-IV pairs which generate the same initial state and then the same keystream.

Let $\Delta K = (\Delta k_0, \cdots, \Delta k_{63})$ and $\Delta IV = (\Delta iv_0, \cdots, \Delta iv_{34})$ denote the key difference and IV difference, respectively. Denote by $\Delta S^{(t)} = \left(\Delta a_0^{(t)}, \cdots, \Delta a_{16}^{(t)}, \Delta b_0^{(t)}, \cdots, \Delta b_{18}^{(t)}, \Delta c_0^{(t)}, \cdots, \Delta c_{20}^{(t)}, \Delta d_0^{(t)}, \cdots, \Delta d_{22}^{(t)}\right)$ the state difference of four LFSRs at time t. Clearly, $\Delta S^{(0)} = (0, \cdots, 0)$ always holds, since all four LFSRs are filled with zeros at the beginning of initialization. After then, the 64-bit key and 35-bit IV are loaded into the four LFSRs by clocking each of them 128 times. Since the irregular clock control is not used in this process, $\Delta S^{(128)}$ can be certainly linearly expressed by ΔK and ΔIV. Now, an algorithm is given as follows, to iteratively compute the linear expression of $\Delta S^{(128)}$ over ΔK and ΔIV.

Algorithm 3 Computing the linear expression of $\Delta S^{(128)}$ over ΔK and ΔIV

```
1. Set $\left(\Delta\lambda^{(0)}, \cdots, \Delta\lambda^{(127)}\right) \leftarrow (\Delta k_0, \cdots, \Delta k_{63}, \Delta iv_0, \cdots, \Delta iv_{34}, 0, \cdots, 0)$;
2. Set $\Delta S^{(0)} \leftarrow (0, \cdots, 0)$;
3. For $t$ from 0 to 127, do the following:
    • $\Delta a_{16}^{(t+1)} \leftarrow \Delta a_0^{(t)} \oplus \Delta\lambda^{(t)}$;
    • $\Delta a_5^{(t+1)} \leftarrow \Delta a_6^{(t)} \oplus \Delta a_0^{(t)}$;
    • $\Delta a_i^{(t+1)} \leftarrow \Delta a_{i+1}^{(t)}$ for $i \in \{0, \cdots, 4, 6, \cdots, 15\}$;
    • $\Delta b_{18}^{(t+1)} \leftarrow \Delta b_0^{(t)} \oplus \Delta\lambda^{(t)}$;
    • $\Delta b_3^{(t+1)} \leftarrow \Delta b_4^{(t)} \oplus \Delta b_0^{(t)}$;
    • $\Delta b_2^{(t+1)} \leftarrow \Delta b_3^{(t)} \oplus \Delta b_0^{(t)}$;
    • $\Delta b_i^{(t+1)} \leftarrow \Delta b_{i+1}^{(t)}$ for $i \in \{0, 1, 4, \cdots, 17\}$;
    • $\Delta c_{20}^{(t+1)} \leftarrow \Delta c_0^{(t)} \oplus \Delta\lambda^{(t)}$;
    • $\Delta c_1^{(t+1)} \leftarrow \Delta c_2^{(t)} \oplus \Delta c_0^{(t)}$;
    • $\Delta c_i^{(t+1)} \leftarrow \Delta c_{i+1}^{(t)}$ for $i \in \{0, 2, \cdots, 19\}$;
    • $\Delta d_{22}^{(t+1)} \leftarrow \Delta d_0^{(t)} \oplus \Delta\lambda^{(t)}$;
    • $\Delta d_8^{(t+1)} \leftarrow \Delta d_9^{(t)} \oplus \Delta d_0^{(t)}$;
    • $\Delta d_i^{(t+1)} \leftarrow \Delta d_{i+1}^{(t)}$ for $i \in \{0, \cdots, 7, 9, \cdots, 21\}$.
```

By **Algorithm 3**, we can easily obtain the linear expression of $\Delta S^{(128)}$ over ΔK and ΔIV. More specifically, each of

TABLE III
A DIFFERENTIAL COLLISION PAIR OF DSC

Key difference $\Delta K = (\Delta k_0, \cdots, \Delta k_{63})$: 0,1,1,0,1,1,0,0,0,1,0,1,1,1,0,1,0,1,0,1,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0,1,0,0,0,0,1,1,1,0,0,1,1,0,1,1,1,0,0,0,1,1,0,1,1,1,0,0,1,1
IV difference $\Delta IV = (\Delta iv_0, \cdots, \Delta iv_{34})$: 0,1,0,1,0,1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
K_1: 1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
IV_1: 1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
K_2: 1,1,1,0,1,1,0,0,0,1,0,1,1,1,0,1,0,1,0,1,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0,1,0,0,0,0,1,1,1,0,0,1,1,0,1,1,1,0,0,0,1,1,0,1,1,1,0,0,1,1
IV_2: 1,1,0,1,0,1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
Keystream: 1,0,0,1,0,1,0,1,1,0,1,1,1,1,1,1,1,0,1,0,0,1,0,1,0,1,1,1,1,0,1,0,1,1,1,0,0,0,0,0$\cdots$
K_1: 1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
IV_1: 1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
K_2: 1,0,1,0,1,1,0,0,0,1,0,1,1,1,0,1,0,1,0,1,0,0,1,1,1,1,1,0,0,0,1,1,1,0,0,1,0,0,0,0,1,1,1,0,0,1,1,0,1,1,1,0,0,0,1,1,0,1,1,1,0,0,1,1
IV_2: 1,0,0,1,0,1,1,1,1,1,0,1,1,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1
Keystream: 1,1,0,1,0,1,0,0,0,1,0,1,1,0,0,1,1,0,1,1,0,1,0,1,0,1,1,0,0,0,1,1,1,0,1,1,0,0,1,0$\cdots$

80 state bit differences in $\Delta S^{(128)}$ can be expressed by one linear polynomial over ΔK and ΔIV. It is easy to see that the condition $\Delta S^{(128)} = 0$ should be satisfied to find collisions in the initial state of DSC. Once a nonzero key and IV difference (i.e., $(\Delta K, \Delta IV)$) which makes $\Delta S^{(128)} = 0$ to be satisfied is found, any two key-IV pairs satisfying the key and IV difference $(\Delta K, \Delta IV)$ will generate the same initial state and then the same keystream. For convenience of description, we introduce a definition as follows.

Definition 1. *For the DSC stream cipher, a nonzero key and IV difference $(\Delta K, \Delta IV)$ is called a* ***differential collision*** *of DSC, if $\Delta S^{(128)} = 0$ satisfies.*

To make $\Delta S^{(128)} = 0$ to be satisfied, a system of 80 linear equations over ΔK and ΔIV can be obtained, where a total of $64 + 35 = 99$ variables, i.e., $\Delta k_0, \cdots, \Delta k_{63}, \Delta iv_0, \cdots, \Delta iv_{34}$, are involved in this system. We have calculated the rank of the coefficient matrix of the system obtained by **Algorithm 3**, and find that it is exactly equal to 80. It indicates that for any nonzero key difference, there exist 2^{19} IV differences such that each IV difference together with the given nonzero key difference is a differential collision of DSC. A differential collision of DSC found by solving the system is given in the first row of Table III. Two collisions are also presented as examples. In each of these two collisions, two different key-IV pairs which generate the same keystream are listed in the table.

Based on the differential collision $(\Delta K, \Delta IV)$ in Table III, a differential distinguishing attack on DSC can be proposed. Specifically, when the differential collision in Table III is satisfied, two different Key-IV pairs of DSC certainly generate the same keystream, while an ideal stream cipher should generate completely random keystreams when different Key-IV pairs are inputted. Thus, a distinguishing attack can be easily constructed using the differential collision property of DSC. It should be noted that the distinguishing attack on DSC is proposed in the related key chosen IV setting. Since the IV is public and can be freely chosen by the attacker in cryptanalysis of stream ciphers, thus the chosen IV setting can be easily satisfied. As pointed out by Biham and Dunkelman in [15], related-key attack is also a standard attack scenario in cryptanalysis of symmetric ciphers. It has important practical significance and can be implied to many stream ciphers. For instance, related-key weaknesses of the RC4 stream cipher led to a practical attack on the WEP protocol [16]. It is generally known that in the related key setting [17]–[19], the encryption can be performed using two different keys that have a linear or nonlinear relationship known to the attacker, while the values of these keys are unknown. The detailed process of the differential distinguishing attack on DSC can be described as an algorithm, called **Algorithm 4**, as follows.

Algorithm 4 A differential distinguishing attack on DSC

1. For the secret key K, do the following:
   - Randomly choose an IV IV, generate a sequence by the (K, IV) pair;
   - For the related key K' satisfying $K' = K \oplus \Delta K$, choose an IV IV' satisfying $IV' = IV \oplus \Delta IV$, generate a sequence by the (K', IV') pair;
   - Check whether the two sequences generated by (K, IV) and (K', IV') are exactly the same, if yes, it concludes that the two sequences are generated by DSC; otherwise, it concludes that the two sequences are random.
2. Output "DSC" or "Random".

In **Algorithm 4** above, only two encryptions of DSC needs to be executed: one for the (K, IV) pair and one for the related (K', IV') pair. Thus, the distinguishing attack on DSC has a time complexity of only two encryptions, requiring two chosen IVs and one related key. The attack has a success probability of almost 1, when the length of keystreams generated by the two different Key-IV pairs is large enough, e.g., 40.

V. Slide Attacks on DSC

In this section, we explore the existence of slide property in the DSC stream cipher, and propose related key attacks on the cipher. The slide property is a general cryptanalytic tool to exploit potential weaknesses of symmetric primitives and has been applied to many well-known stream ciphers, e.g., Trivium [20], Salsa20 [20], Grain-like [21]–[24], Decim v2 [25], WG-like [26]–[28], SNOW 3G [29], GEA-1 [30] and GEA-2 [30].

A. Slide Property of DSC

Denote by $K_0 = (k_0, \cdots, k_{63})$ and $IV_0 = (iv_0, \cdots, iv_{34})$ the 64-bit key and 35-bit IV of DSC, respectively, and $S_0^{(t)}$ the state of four LFSRs generated by (K_0, IV_0) at time t. Let $(K_i, IV_i)\,(i = 1, \cdots, 7)$ be seven related key-IV pairs of (K_0, IV_0). The relation of $(K_i, IV_i)\,(i = 1, \cdots, 7)$ and (K_0, IV_0) is defined as follows.

$$\begin{aligned}
&K_0 = (k_0, k_1, k_2, \cdots, k_{63}), IV_0 = (iv_0, \cdots, iv_{34})\\
&K_1 = (k_0 \oplus 1, k_1, k_2, k_3, \cdots, k_{63}), IV_1 = IV_0\\
&K_2 = (k_0, k_1 \oplus 1, k_2, k_3, \cdots, k_{63}), IV_2 = IV_0\\
&K_3 = (k_0 \oplus 1, k_1 \oplus 1, k_2, k_3, \cdots, k_{63}), IV_3 = IV_0\\
&K_4 = (k_0, k_1, k_2 \oplus 1, k_3, \cdots, k_{63}), IV_4 = IV_0\\
&K_5 = (k_0 \oplus 1, k_1, k_2 \oplus 1, k_3, \cdots, k_{63}), IV_5 = IV_0\\
&K_6 = (k_0, k_1 \oplus 1, k_2 \oplus 1, k_3, \cdots, k_{63}), IV_6 = IV_0\\
&K_7 = (k_3, \cdots, k_{63}, k_0, k_1, k_2), IV_7 = (iv_3, \cdots, iv_{34}, 0, 0, 0)
\end{aligned}$$

Let $S_i^{(t)}$ be the state of four LFSRs generated by (K_i, IV_i) at time t, $i = 1, \cdots, 7$. Since all four LFSRs and the memory bit are filled with zeros at the beginning of initialization, $S_i^{(0)} = (0, \cdots, 0)$ always holds for $0 \le i \le 7$. Then each of these four LFSRs is regularly clocked 128 times. In this process, the memory bit y is fixed to be zero and the most significant bit of each LFSR is generated by the bit that is shifted out, XORed with one input bit. For (K_0, IV_0), if the first three input bits are all equal to zero, i.e., $k_0 = 0, k_1 = 0$ and $k_2 = 0$, then $S_0^{(3)} = 0$ definitely holds. Now, an observation can be obtained as follows.

Observation 1. For $0 \le i \le 6$, the states $S_i^{(3)}$ and $S_7^{(0)}$ satisfy the following relation according to the values of k_0, k_1 and k_2.

- $S_0^{(3)} = S_7^{(0)} = 0$ satisfies if $k_0 = 0, k_1 = 0$ and $k_2 = 0$ simultaneously hold
- $S_1^{(3)} = S_7^{(0)} = 0$ satisfies if $k_0 = 1, k_1 = 0$ and $k_2 = 0$ simultaneously hold
- $S_2^{(3)} = S_7^{(0)} = 0$ satisfies if $k_0 = 0, k_1 = 1$ and $k_2 = 0$ simultaneously hold
- $S_3^{(3)} = S_7^{(0)} = 0$ satisfies if $k_0 = 1, k_1 = 1$ and $k_2 = 0$ simultaneously hold
- $S_4^{(3)} = S_7^{(0)} = 0$ satisfies if $k_0 = 0, k_1 = 0$ and $k_2 = 1$ simultaneously hold
- $S_5^{(3)} = S_7^{(0)} = 0$ satisfies if $k_0 = 1, k_1 = 0$ and $k_2 = 1$ simultaneously hold
- $S_6^{(3)} = S_7^{(0)} = 0$ satisfies if $k_0 = 0, k_1 = 1$ and $k_2 = 1$ simultaneously hold

Supposing $S_i^{(3)} = S_7^{(0)} = 0$ satisfies for some $i \in \{0, \cdots, 6\}$, it is easy to see that $S_i^{(3+j)} = S_7^{(j)}$ directly holds for $1 \le j \le 61$. Since the input bits to generate $S_i^{(65)}$and $S_7^{(62)}$ are iv_0 and k_0, respectively, thus $S_i^{(65)} = S_7^{(62)}$ satisfies if $k_0 = iv_0$ holds. Similarly, $S_i^{(66)} = S_7^{(63)}$ satisfies if $k_0 = iv_0$ and $k_1 = iv_1$ simultaneously hold, and then $S_i^{(67)} = S_7^{(64)}$ satisfies if $k_0 = iv_0, k_1 = iv_1$ and $k_2 = iv_2$ simultaneously hold. Thus, a new observation can be obtained as follows.

Observation 2. Supposing $S_i^{(3)} = S_7^{(0)} = 0$ satisfies for some $i \in \{0, \cdots, 6\}$, then $S_i^{(67)} = S_7^{(64)}$ satisfies if $k_0 = iv_0, k_1 = iv_1$ and $k_2 = iv_2$ simultaneously hold.

Again, supposing $S_i^{(67)} = S_7^{(64)}$ satisfies for some $i \in \{0, \cdots, 6\}$, it is easy to see that $S_i^{(67+j)} = S_7^{(64+j)}$ directly holds for $1 \le j \le 61$. However, the state $S_i^{(129)}$ is generated by $S_i^{(128)}$ using one pre-cipher round, while $S_7^{(128)}$ is generated by $S_7^{(125)}$ using three regular clocks. Denote by y' the value of the memory bit generated by $S_i^{(128)}$ after one pre-cipher round. If $S_i^{(129)} = S_7^{(128)}$ satisfies for some $i \in \{0, \cdots, 6\}$ and $y' = 0$ holds, the 81-bit internal state generated by (K_i, IV_i) will be the same with that generated by (K_7, IV_7). It means that (K_i, IV_i) will generate 1-bit shifted keystream with respect to (K_7, IV_7). For convenience of description, we introduce a definition as follows.

Definition 2. *For the DSC stream cipher, the (K, IV) and related (K', IV') are called a **slide pair** of DSC, if (K', IV') generate 1-bit shifted keystream with respect to (K, IV).*

Let p be the probability that (K_i, IV_i) and (K_7, IV_7) are a slide pair of DSC under the condition that $S_i^{(67)} = S_7^{(64)}$ satisfies for some $i \in \{0, \cdots, 6\}$. To calculate the value of probability p, we have made an experiment. In this experiment, we randomly choose 2^8 keys with $(k_0, k_1, k_2) = (0, 0, 0)$. For each key, we choose 2^8 different IVs, where the values of iv_0, iv_1 and iv_2 are fixed to be zeros and the remaining 32 IV bits (i.e., $iv_3, \cdots, iv_{34}$) are randomly chosen. The result shows that a total of 12821 slide pairs are found in the 2^8 random keys and 2^8 different IVs. Thus, the probability p can be calculated as $p = 12821 \big/ \big(2^8 \times 2^8\big) \approx 2^{-2.35}$. To further strengthen the accuracy of the experimental result, we increase the numbers of keys and IVs to be 2^{10}, and then the result shows that a total of 206222 slide pairs are found in the 2^{10} random keys and 2^{10} different IVs. Now, the probability p can be recalculated as $p = 206222 \big/ \big(2^{10} \times 2^{10}\big) \approx 2^{-2.35}$, which is the same with the former experimental result. Furthermore, when the numbers of keys and IVs are further increased to be 2^{12}, a total of 3286619 slide pairs are found in the 2^{12} random keys and 2^{12} different IVs, and thus the probability p can be recalculated as $p = 3286619 \big/ \big(2^{12} \times 2^{12}\big) \approx 2^{-2.35}$, which confirms the accuracy of the former experimental result again. Thus, when $S_i^{(67)} = S_7^{(64)}$ satisfies for some $i \in \{0, \cdots, 6\}$, (K_i, IV_i) and (K_7, IV_7) are a slide pair of DSC with probability $2^{-2.35}$. Table IV gives two slide pairs of DSC found in this experiment as examples. As shown in this table, (K', IV') generates 1-bit shifted keystream with respect to (K, IV).

In the meanwhile, we have made another experiment. In this experiment, we randomly choose 2^8 keys with $(k_0, k_1, k_2) \neq (0, 0, 0)$. For each key, we choose 2^8 different IVs, where $(k_0, k_1, k_2) = (iv_0, iv_1, iv_2)$ are satisfied and the remaining 32 IV bits (i.e., $iv_3, \cdots, iv_{34}$) are randomly chosen. The result shows that none of slide pairs is found in the 2^8 random keys and 2^8 different IVs. To further verify the experimental result, we increase the numbers of keys and IVs to be 2^{10}. Still, none of slide pairs is found in the 2^{10} random keys and 2^{10} different IVs. The experimental result remains unchanged when the numbers of keys and IVs are further increased to be 2^{12}.

The two experiments above clearly show that if the condition $S_i^{(67)} = S_7^{(64)}$ satisfies for some $i \in \{0, \cdots, 6\}$, a large number of slide pairs can be easily found. In contrary, when the condition $S_i^{(67)} = S_7^{(64)}$ does not satisfy for any $i \in \{0, \cdots, 6\}$, the probability that there exists at least one slide pair is extremely small. Thus, we can judge whether the condition $S_i^{(67)} = S_7^{(64)}$ satisfies for some $i \in \{0, \cdots, 6\}$, according to the number of found slide pairs. Considering the results of the two experiments above, it is easy to see that the judgement can succeed with an extremely high probability. Let $K_0 = (k_0, \cdots, k_{63})$ be the 64-bit secret key of DSC. Now, we

TABLE IV
Two slide pairs of DSC found in the experiment

K: 0,0,0,1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,1,0,1,1,0,0,1,0,0,0,0,0,1,1,0,1,0,1,0,0,1,1,0,1,0,1,0,1,0,0,0,0,0,1,1,0,1,1,0,0,0
IV: 0,0,0,0,1,1,0,1,1,1,1,1,1,0,0,1,0,1,1,1,1,0,0,0,1,1,1,0,0,0,0,1,0,0,1
Keysteam: 1,1,1,0,1,0,1,1,0,0,0,0,0,1,1,1,1,0,0,0,0,1,0,1,0,1,1,1,0,0,0,0,0,0,1,1,1,1,0,1,0,0,0,1,1,0,1,1,1,0,1,1,0,1,0,1,0,0,1,0,0,0,1,0$\cdots$

K': 1,1,1,1,1,0,0,0,0,0,1,1,1,1,1,1,1,1,1,0,1,0,1,1,0,0,1,0,0,0,0,0,1,1,0,1,0,1,0,0,1,1,0,1,0,1,0,1,0,0,0,0,0,1,1,0,1,1,0,0,0,0,0,0
IV': 0,1,1,0,1,1,1,1,1,1,0,0,1,0,1,1,1,1,0,0,0,1,1,1,0,0,0,0,1,0,0,1,0,0,0
Keystream: 1,1,0,1,0,1,1,0,0,0,0,0,1,1,1,1,0,0,0,0,1,0,1,0,1,1,1,0,0,0,0,0,0,1,1,1,1,0,1,0,0,0,1,1,0,1,1,1,0,1,1,0,1,0,1,0,0,1,0,0,0,1,0,1$\cdots$

K: 0,0,0,1,0,0,1,1,1,0,1,1,0,0,1,1,1,1,1,1,0,0,1,1,1,1,1,0,1,1,1,0,1,1,0,0,1,1,0,0,1,0,0,1,1,1,0,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,1
IV: 0,0,0,0,0,0,0,1,1,1,1,1,0,0,0,0,1,1,0,1,1,0,1,0,1,0,0,1,1,1,1,1,1,0,0
Keystream: 0,1,0,1,0,1,0,1,1,0,0,0,0,0,0,1,1,1,1,1,0,1,1,0,1,0,1,1,1,0,1,0,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,0,1,0,1,0,1,0,1,0,1,1,1,0,0,0,0,1$\cdots$

K': 1,0,0,1,1,1,0,1,1,0,0,1,1,1,1,1,1,0,0,1,1,1,1,1,0,1,1,1,0,1,1,0,0,1,1,0,0,1,0,0,1,1,1,0,0,0,1,1,0,1,0,0,1,1,0,0,0,0,0,0,1,0,0,0
IV': 0,0,0,0,1,1,1,1,1,0,0,0,0,1,1,0,1,1,0,1,0,1,0,0,1,1,1,1,1,1,0,0,0,0,0
Keystream: 1,0,1,0,1,0,1,1,0,0,0,0,0,0,1,1,1,1,1,0,1,1,0,1,0,1,1,1,0,1,0,1,0,1,1,1,1,0,1,1,1,1,1,1,1,1,0,1,0,1,0,1,0,1,0,1,1,1,0,0,0,0,1,0$\cdots$

aim at recovering the three key bits k_0, k_1, k_2 of DSC by using an algorithm, called **Algorithm 5**, as follows. This algorithm utilizes seven related keys $K_1, \cdots, K_7$, as defined above.

Algorithm 5 **Recovering the three key bits k_0, k_1, k_2 of DSC**

```
1. Choose $R$ different IVs $IV_0^i = (iv_0^i, \cdots, iv_{34}^i)$, $i = 1, \cdots, R$, where the values of $iv_0^i, iv_1^i$ and $iv_2^i$ are fixed to be zeros and the remaining 32 IV bits (i.e., $iv_3^i, \cdots, iv_{34}^i$) are randomly chosen.
2. Set $n_j \leftarrow 0$ for $j = 0, \cdots, 6$.
3. For $i$ from 1 to $R$, do the following:
    • Set $IV_l^i \leftarrow IV_0^i$ for $l = 1, \cdots, 6$;
    • Set $IV_7^i = (iv_3^i, \cdots, iv_{34}^i, 0, 0, 0)$;
    • For $j$ from 0 to 6, do the following:
        – Check whether $(K_7, IV_7^i)$ and $(K_j, IV_j^i)$ are a slide pair, if yes, $n_j \leftarrow n_j + 1$; otherwise, go back to try the next pair.
4. For $j$ from 0 to 6, check whether the value of $n_j$ is close to $2^{-2.35}R$, if yes, return the value of $j$; otherwise, if $n_j = 0$ always holds for $j = 0, \cdots, 6$, return 7.
```

In **Algorithm 5**, the parameter n_j is a counter, and it indicates the number of slide pairs formed by (K_7, IV_7^i) and (K_j, IV_j^i) with $1 \leq i \leq R$. If the algorithm outputs the value of j with $0 \leq j \leq 6$, it means that the first three keys bits of K_j are all equal to zero, then the attacker can easily recover the three key bits k_0, k_1, k_2. For example, if the algorithm outputs 3, it has $k_0 \oplus 1 = 0, k_1 \oplus 1 = 0, k_2 = 0$, and thus $k_0 = 1, k_1 = 1, k_2 = 0$. If the algorithm outputs 7, it means that $k_0 = k_1 = k_2 = 1$ holds. Thus, the three key bits k_0, k_1, k_2 of DSC can be recovered by **Algorithm 5**.

In this algorithm, for each of R different IVs, it has to check whether (K_7, IV_7^i) and (K_j, IV_j^i) are a slide pair for $0 \leq j \leq 6$. Thus, **Algorithm 5** has a time complexity of $R \cdot 8 = 2^3 R$ encryptions, requiring 7 related keys and R chosen IVs. It should be noted that the size of keystream required for each (K_7, IV_7^i) and (K_j, IV_j^i) is quite small and far less than 720 bits, since we only utilize the keystream to make a check whether (K_7, IV_7^i) and (K_j, IV_j^i) are a slide pair. To ensure that the 64-bit secret key of DSC can be uniquely recovered, each (K_7, IV_7^i) and (K_j, IV_j^i) in this algorithm generates a keystream of size 64 which is certainly large enough to make this check. Therefore, **Algorithm 5** requires $R \cdot 8 \cdot 64 = 2^9 R$ keystream bits.

By considering the experimental results above, it is easy to see that $R = 2^8$ is obviously enough such that **Algorithm 5** succeeds in recovering the three key bits k_0, k_1, k_2 with probability almost 1. To verify this, we make an experiment. In this experiment, a total of 1000 keys are randomly chosen, and for each random key we execute **Algorithm 5** with $R = 2^8$. The result show that **Algorithm 5** succeeds for each of these 1000 keys. Thus, it has a success probability of almost 1. Up to now, a key recovery attack on DSC has been proposed based on **Algorithm 5**. The attack has a time complexity of about $2^3 R + 2^{64-3} = 2^3 \cdot 2^8 + 2^{61} \approx 2^{61}$ encryptions, since the remaining 61 key bits should be exhausted after the three key bits k_0, k_1, k_2 are recovered by **Algorithm 5**. It requires 7 related keys, 2^8 chosen IVs and $2^9 \cdot 2^8 = 2^{17}$ keystream bits. The attack can recover all 64 key bits of DSC with a success probability of almost 1.

It should be noted that there certainly exist other slide pairs for longer shifts, but much more related keys are required to find the useful slide pair, which will weaken the practicality of proposed key recovery attack on DSC. Therefore, we no longer consider the slide properties with longer shifts.

B. A Practical Related Key Attack on DSC

Clearly, the time complexity of the key recovery attack on DSC above is better than exhaustive key search, but it is still too time-consuming to be practical on a common PC. In fact, the time complexity can be reduced significantly using more related keys. In this subsection, we give a practical key recovery attack on DSC in the multiple related key setting.

The practical key recovery attack on DSC utilizes $8h$ keys, i.e., $K_0, \cdots, K_7, \cdots, K_{8h-8}, \cdots, K_{8h-7}$, and consists of h steps. In the i-th step with $0 \leq i < h$, the attack utilizes 8 keys $K_{8i}, \cdots, K_{8i+7}$. The relation of K_{8i} and K_{8i+8} is defined as $K_{8i+8} = K_{8i} \lll 3$, where $\lll 3$ is the left cyclic shift operator by 3 bits. The relation of $K_{8i+j}\,(j = 1, \cdots, 7)$ and K_{8i} is defined as the same in the first subsection of section V. In the i-th step, the attacker executes **Algorithm 5** once using 8 keys $K_{8i}, \cdots, K_{8i+7}$, and then recovers three key bits $k_{3i}, k_{3i+1}, k_{3i+2}$ of the 64-bit key of DSC. Since the attack consists of h steps, thus a total of $3h$ key bits can be recovered, which leads to a time complexity of $2^{11}h$ encryptions. After this process, the remaining $64-3h$ key bits can be recovered by making an exhaustive search. Thus, the

TABLE V
THE CRYPTANALYTIC RESULTS OF DSC IN THE MULTIPLE RELATED KEY SETTING

h	Time complexity	Data complexity	Required chosen IVs	Required related keys
13	$2^{11} \times 13 + 2^{64-3\times 13} \approx 2^{25}$	$2^{17} \times 13 \approx 2^{20.70}$	$2^{8} \times 13 \approx 2^{11.70}$	103
14	$2^{11} \times 14 + 2^{64-3\times 14} \approx 2^{22.01}$	$2^{17} \times 14 \approx 2^{20.81}$	$2^{8} \times 14 \approx 2^{11.81}$	111
15	$2^{11} \times 15 + 2^{64-3\times 15} \approx 2^{19.08}$	$2^{17} \times 15 \approx 2^{20.91}$	$2^{8} \times 15 \approx 2^{11.91}$	119
16	$2^{15} + 2^{16} \approx 2^{16.58}$	$2^{17} \times 16 = 2^{21}$	$2^{8} \times 16 = 2^{12}$	127

total time complexity of the attack on DSC can be calculated as $2^{11}h + 2^{64-3h}$ encryptions. This attack requires $8h - 1$ related keys, 2^8h chosen IVs and $2^{17}h$ keystream bits. Table V lists the cryptanalytic results of DSC in the multiple related key setting varying according to different values of h.

As shown in Table V, it is easy to see that there exists a trade-off between the complexities and the number of required related keys. To make the attack practical on a common PC with as less required related keys as possible, $h = 14$ is a reasonable choice, which leads to a key recovery attack on DSC with a time complexity of about $2^{22.01}$ encryptions. We have validated the result by simulating the whole attack process. The results show that the attacker can recover 42 key bits by executing **Algorithm 5** fourteen times within 0.54 second on average, and the remaining 22 bits can be recovered by making an exhaustive search, which can be done within 44.43 seconds on average. The simulation was implemented on a PC with 2.5 GHz Intel Pentium 4 processor. The experimental result shows that the DSC stream cipher can be broken within about 44.97 seconds in the multiple related key setting.

VI. CONCLUSIONS

As a standard stream cipher used to provide confidentiality for cordless telephony in the digital wireless technology DECT, the DSC stream cipher has attracted a large amount of attention in the recent years due to its importance. In this paper, we have discovered some new weaknesses of DSC which are not found in previous works. Based on these weaknesses, new practical key recovery attacks and distinguishing attack on DSC with lower time cost are proposed. The cryptanalytic results show that the DSC stream cipher can be broken in the known IV setting or related key setting. It is worth mentioning that all attacks proposed in this paper are irrelevant to the number of initialization rounds of DSC, and thus increasing the number of initialization rounds can not strengthen the resistance of DSC against our attacks. This is different from the previous cryptanalytic attacks on DSC. Our attacks on DSC clearly show that a well-designed initialization is absolutely necessary to design a secure stream cipher.

In 2011, an improved version of DSC called DSC2 was published and standardized by ETSI in [31]. It is built based on the block cipher AES and completely different from the stream cipher DSC. To meet current security requirements, ETSI followed the long-term goal to remove the support of DSC from DECT devices. However, algorithm change in the massive devices usually requires the consent of several parties and takes a long time. Thus, the new attacks on DSC are still realistic and meaningful, as DSC2 is not completely deployed in all DECT devices.

Furthermore, to strengthen the DSC stream cipher to provide an adequate level of security, some countermeasures can be adopted. On the one hand, the number of pre-cipher rounds must be increased, e.g., from 40 to 80, to avoid the previous cryptanalytic attacks on DSC. On the other hand, the key size and internal state size of DSC need to be extended, and the key/IV loading should be modified to avoid our attacks on DSC. Since DSC2 is built based on the block cipher AES and completely different from the stream cipher DSC, it is an interesting open problem how to design a DSC-like stream cipher that can be immune to all known attacks on DSC. We leave it as a future work.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers and the associate editor for their valuable comments and suggestions.

REFERENCES

[1] ETSI. ETSI DECT Official Website [Online]. http://www.etsi.org/technologies-clusters/technologies/dect/.

[2] S. Lucks, A. Schuler, E. Tews, R.P. Weinmann, and M. Wenzel, “Attacks on the DECT authentication mechanisms,”, in *Proceedings of the Cryptographers’ Track at RSA Conference*, San Francisco, CA, USA, Apr. 2009, pp. 48-65.

[3] K. Nohl, E. Tews, and R.P. Weinmann, “Cryptanalysis of the DECT standard cipher,” in *Proceedings of the 17th International Workshop on Fast Software Encryption*, Seoul, Korea, Feb. 2010, pp. 1-18.

[4] H.G. Molter, K. Ogata, E. Tews, and R.P. Weinmann, “An Efficient FPGA Implementation for an DECT Brute-Force Attacking Scenario,” in *Proceedings of the Fifth International Conference on Wireless and Mobile Communications*, Cannes, France, Aug. 2009, pp. 82-86.

[5] P. McHardy, A. Schuler, and E. Tews, “Interactive decryption of DECT phone calls,” in *Proceedings of the Fourth ACM Conference on Wireless Network Security*, Hamburg, Germany, Jun. 2011, pp. 71-78.

[6] I. Sanchez, G. Baldini, D. Shaw, and R. Giuliani, “Experimental Passive Eavesdropping of Digital enhanced cordless telecommunication voice communications through lowcost software defined radios,” *Security and Communication Networks*, vol. 8, no. 3, pp. 403-417, Mar. 2014.

[7] I. Coisel, and I. Sanchez, “Practical interception of DECT encrypted voice communication in unified communications environments,” in *Proceedings of the 2014 IEEE Joint Intelligence and Security Informatics Conference*, The Hague, Netherlands, Sep. 2014, pp. 115-122.

[8] I. Coisel, I. Sanchez, and D. Shaw, “Physical attacks against the lack of perfect forward secrecy in DECT encrypted communications and possible countermeasures,” in *Proceedings of the 11th International Conference on Wireless and Mobile Communications*, Dubrovnik, Croatia, Aug. 2015, pp. 594-599.

[9] M. Weiner, E. Tews, B. Heinz, and J. Heyszl, “FPGA implementation of an improved attack against the DECT standard cipher,” in *Proceedings of the 13th International Conference on Information Security and Cryptology*, Seoul, Korea, Dec. 2010, pp. 177-188.

[10] I. Coisel, and I. Sanchez, “Improved cryptanalysis of the DECT standard cipher,” in *Proceedings of the 17th International Workshop on Cryptographic Hardware and Embedded Systems*, Saint-Malo, France, Sep. 2015, pp. 269-286.

[11] I. Coisel, and I. Sanchez, “Improved cryptanalysis of the DECT standard cipher,” *Journal of Cryptographic Engineering*, vol. 6, no. 2, pp. 155-169, Mar. 2016.

[12] H. Liu, and C. Jin, “An Improvement of the CS Attack to DSC Cipher,” *The Computer Journal*, vol. 62, no. 8, pp. 1158-1165, Aug. 2019.

[13] S. Babbage, “Improved exhaustive search attacks on stream ciphers,” in *Proceedings of European Convention on Security and Detection 1995*, Brighton, UK, May. 1995, pp. 161-166.

[14] J. Golić, “Cryptanalysis of alleged A5 stream cipher,” in *Proceedings of International Conference on the Theory and Application of Cryptographic Techniques 1997*, Konstanz, Germany, May. 1997, pp. 239-255.

[15] E. Biham and O. Dunkelman, “Differential cryptanalysis in stream ciphers,” Cryptology ePrint Archive, Report 2007/218, 2007 [Online]. http://eprint.iacr.org/.

[16] S. Fluhrer, I. Mantin, A. Shamir, “Weaknesses in the Key Scheduling Algorithm of RC4,” in *Proceedings of the 8th Annual International Workshop on Selected Areas in Cryptography*, Toronto, Ontario, Canada, Aug. 16-17, 2001, pp. 1-24.

[17] E. Biham, “New Types of Cryptanalytic Attacks Using Related Keys,” in *Proceedings of Workshop on the Theory and Application of of Cryptographic Techniques 1993*, Lofthus, Norway, May. 1993, pp. 398-409.

[18] S. Lucks, “Ciphers Secure against Related-Key Attacks,” in *Proceedings of the 11th International Workshop on Fast Software Encryption*, Delhi, India, Feb. 2004, pp. 359-370.

[19] M. Ciet, G. Piret, and J. Quisquater, “Related-key and slide attacks: Analysis, connections, and improvements,” in *Proceedings of 2002 IEEE International Symposium on Information Theory*, Lausanne, Switzerland, Jun. 2002, pp. 315-325.

[20] D. Priemuth-Schmid and A. Biryukov, “Slid Pairs in Salsa20 and Trivium,” in *Proceedings of the 9th International Conference on Cryptology in India*, Kharagpur, India, Dec. 2008, pp. 1-14.

[21] C. D. Cannière, Ö. Kücük, and B. Preneel, “Analysis of Grain’s initialization algorithm,” in *Proceedings of the First International Conference on Cryptology in Africa*, Casablanca, Morocco, Jun. 2008, pp. 276-289.

[22] L. Yuseop, J. Kitae, S. Jaechul, and H. Seokhie, “Related-key chosen IV attacks on Grain-v1 and Grain-128,” in *Proceedings of the 13th Australasian Conference on Information Security and Privacy*, Wollongong, Australia, Jul. 2008, pp. 321-335.

[23] L. Ding and J. Guan, “Related key chosen IV attack on Grain-128a stream cipher,” *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 5, pp. 803-809, May. 2013.

[24] S. Banik, S. Maitra, S. Sarkar, and S. T. Meltem, “A chosen IV related key attack on Grain-128a,” in *Proceedings of the 18th Australasian Conference on Information Security and Privacy*, Brisbane, QLD, Australia, Jul. 2013, pp. 13-26.

[25] L. Ding and J. Guan, “Related-key chosen IV attack on Decim v2 and Decim-128,” *Mathematical and Computer Modelling*, vol. 55, no. 1-2, pp. 123-133, Jan. 2012.

[26] L. Ding, C. Jin, J. Guan, S. Zhang, T. Cui, D. Han, and W. Zhao, “Cryptanalysis of WG family of stream ciphers,” *The Computer Journal*, vol. 58, no. 10, pp. 2677-2685, Oct. 2015.

[27] L. Ding, C. Jin, J. Guan, and Q. Wang, “Cryptanalysis of lightweight WG-8 stream cipher,” *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 4, pp. 645-652, Apr. 2014.

[28] L. Ding, D. Gu, L. Wang, C. Jin, and J. Guan, “A real-time related key attack on the WG-16 stream cipher for securing 4G-LTE networks,” *Journal of Information Security and Applications*, vol. 63, Dec. 2021, 103015.

[29] A. Kircanski and A. Youssef, “On the sliding property of SNOW 3G and SNOW 2.0,” *IET Information Security*, vol. 5, no. 4, pp. 199-206, Dec. 2011.

[30] L. Ding, Z. Wu, X. Wang, Z. Guan and J. LI, “New Attacks on the GPRS Encryption Algorithms GEA-1 and GEA-2,” *IEEE Transactions on Information Forensics and Security*, vol. 17, pp. 2878-2889, Aug. 2022.

[31] ETSI. Digital Enhanced Cordless Telecommunications (DECT); Common Interface (CI); Part 7: Security features. Final draft ETSI EN 300 175-7 V2.4.0 (2011-12). ETSI DECT Official Website [Online]. https://www.etsi.org/deliver/etsi_en/300100_300199/30017507/02.04.00_40/en_30017507v020400o.pdf.