Slide Attacks on the Sfinks Stream Cipher

Ali Alhamdan, Harry Bartlett, Ed Dawson, Leonie Simpson and Kenneth Koon-Ho Wong
Institute for Future Environments
Science and Engineering Faculty
Queensland University of Technology
Brisbane, Australia,
Email: a.alhamdan@student.qut.edu.au, {h.bartlett, e.dawson, lr.simpson, kk.wong}@qut.edu.au

***Abstract*—Sfinks is a shift register based stream cipher designed for hardware implementation and submitted to the eSTREAM project. In this paper, we analyse the initialisation process of Sfinks. We demonstrate a slid property of the loaded state of the Sfinks cipher, where multiple key-IV pairs may produce phase shifted keystream sequences. The state update functions of both the initialisation process and keystream generation and also the pattern of the padding affect generation of the slid pairs.**

***Index Terms*—Initialisation process, Sfinks, Slide attack, Slid pairs, Slid resynchronisation, Stream cipher.**

I. Introduction

The Sfinks [1] stream cipher was submitted to the eSTREAM project [2], the ECRYPT call for stream cipher proposals, in April 2005. It is a bit-based stream cipher that takes an 80-bit secret key and 80-bit IV as inputs into a 256-bit initial register. Sfinks is categorized as PROFILE 2A; suitable for hardware applications and with an associated authentication method. This research considers specifically the interaction between the initialisation process and the keystream generation process of the Sfinks stream cipher that can produce slid pairs generated by different key-IV pairs.

Most recent stream cipher proposals require an initialisation process as an essential part of the stream ciphers' specification. During the initialisation process, a secret key, k, and an initial value (IV), v, are loaded into an internal state and then processed to be diffused across the internal state. A well designed initialisation process for a keystream generator should ensure that each key-IV pair (k, v) generates a distinct and unpredictable keystream sequence. This is possible because the size of the internal state should be at least twice the key size to avoid time-memory trade-off attacks [3].

For stream ciphers, it is sometimes possible to find different key-IV pairs that produce phase shifted keystreams [4]–[8]. The state update function for the initialisation process defines a cycle of transitions of the internal state. Therefore, each (k, v) pair represents a point on such a cycle. If the length of the cycle(s) is smaller than the total number of possible key-IV pairs multiplied by the maximum length of the keystream, then there are overlaps between some keystream generated by different key-IV pairs. If it is possible to find a second loaded state generated by a pair (k', v') as a slid pair of the key-IV pair (k, v) after a number of iterations, α, then, the (k', v') pair is in the same cycle as the (k, v) pair. The probability of obtaining shifted keystream sequences from such a pair depends on the distance between (k', v') and (k, v) and the degree of similarity of the state update functions used during the initialisation process and the keystream generation process.

This paper is organized as follows. Section II describes resynchronisation and slide attacks on stream ciphers. Section III presents a full description of the Sfinks stream cipher. A theoretical analysis of slid pairs in the Sfinks cipher is presented in Section IV. In Section V, an experimental simulation supports the theoretical analysis. A practical procedure to attack Sfinks using the slid property is presented in Section VI. Section VII investigates the effect of a one bit modification of the padding patterns of Sfinks on slid pairs. Section VIII concludes with a discussion of the causes of slid pairs and methods for prevention or mitigation.

II. Resynchronisation and Slide Attacks

The slide attack is a generic attack which was firstly applied to block ciphers by Biryukov and Wagner in 1999 [8] and 2000 [9], and has also been applied to stream ciphers that are based on block ciphers, such as LEX [10] and WAKE-ROFB [8]. More recently slide attacks have been applied to other stream ciphers, such as Grain [4]–[6] and Trivium [7].

When a key-IV pair (k', v') produces a loaded state that can also be obtained from another key-IV pair (k, v) after a number of iterations α of the initialisation state update function, we refer to these two states as a slid pair. The keystream generated by the pair (k', v') may then be a phase-shifted version of the keystream generated by (k, v), shifted by α bits as shown in Figure 1. This occurs when the following properties hold for the initialisation process and key stream generation:

a) Iterations of the initialisation process are similar to each other.
b) Iterations of the keystream generation process are similar to each other.
c) State update functions for both of the initialisation and keystream generator processes have a degree of similarity.

Most stream ciphers meet the first two conditions above, but different ciphers vary greatly in the extent to which the third condition applies. Slid pairs can be expressed as $\{(k, v), (k', v'), \alpha\}$, where (k, v) is a key-IV pair that produces

978-1-4673-2393-2/12/ $31.00

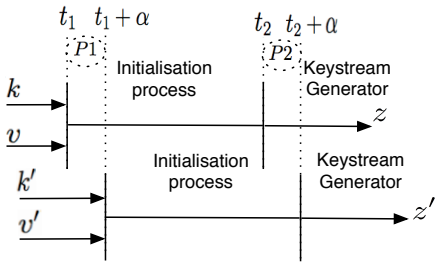

Fig. 1: slid pairs on stream cipher

another loaded state with key-IV pair (k', v') after α iterations of the initialisation state update function.

Previous analysis [4]–[7] focussed on the concern that slid pairs lead to phase shifted keystream, which may in turn reveal information about some or all of secret key bits. We consider this issue in further detail later. There is also a secondary concern with the existence of slid pairs, as follows. If the initialisation phase of a stream cipher involves N iterations of the state update function where the loaded state for pair (k, v) leads to a slid pair (k', v') after α iterations, then the initial state prior to keystream generation can be obtained from the loaded state for (k', v') after only $(N-\alpha)$ iterations. Thus the existence of slid pairs may imply a decrease in the effective number of iterations during the initialisation process. We now return to the issue of slid pairs leading to shifted keystream sequences.

Slide attacks have recently been applied to stream ciphers which are not based on block ciphers such as the Grain family of stream ciphers [11]–[13]. The analysis of Grain v1.0 [12] in [6] shows that for any key-IV pair (k, v) there exists a related (k', v') pair which generates a 1-bit shifted keystream with a probability of 2^{-2}. In general, if a key-IV pair, (k, v), produces another loaded state with (k', v') pair after α iterations, then the keystream generated by (k', v') is shifted by α bit(s) from the original keystream with probability $2^{-2\alpha}$ [4]. Zhang and Wang also demonstrated some sliding weak key-IV combinations that apply to various members of the Grain family of ciphers [5].

Slid pairs have also been reported for the Trivium stream cipher [14]. This cipher was analyzed by Priemuth-Schmid and Biryukov [7] in 2008. They found more than 2^{39} slid pairs out of 2^{80} keys.

The probability of obtaining a slid pair that results in a correspondingly shifted keystream depends on two factors: the probability of getting a legitimate loaded state with (k', v') pair after α iterations of the initialisation process $(P1)$ and the probability $(P2)$ that the first α iterations of the keystream generation for the (k, v) are same as the last α iterations of the initialisation for the (k', v'). The total probability for obtaining a slid pair that result in phase-shifted keystream is the product of these two probabilities, as shown in Figure 1.

III. Sfinks Stream Cipher

A. Description of Sfinks

The Sfinks stream cipher [1] has two main components: a shift register, S, and a nonlinear one-to-one inversion function INV, as shown in Figure 2. During the initialisation, it also uses a pipeline (memory) to delay the output of the INV function before combining this output with the shift register content. Additionally, another memory is used during keystream generation to delay the output bit of the shift register by 7 steps before combining it with a specific bit of the INV function to generate a new keystream bit.

Let s_t^i denote the contents of register stage i at time t, where $i = 0, 1, \ldots 255$ and $t \geq 0$. Sfinks uses an 80-bit secret key $k = k_0, \ldots, k_{79}$ and 80-bit initial value $v = v_0, \ldots, v_{79}$. The linear feedback function is described as following.

$$\begin{aligned} s_{t+1}^{255} = \; & s_t^{212} \oplus s_t^{194} \oplus s_t^{192} \oplus s_t^{187} \oplus s_t^{163} \oplus s_t^{151} \\ & \oplus s_t^{125} \oplus s_t^{115} \oplus s_t^{115} \oplus s_t^{107} \oplus s_t^{85} \oplus s_t^{66} \\ & \oplus s_t^{64} \oplus s_t^{52} \oplus s_t^{48} \oplus s_t^{14} \oplus s_t^{0} \end{aligned} \tag{1}$$

The nonlinear function INV can be considered as a 16×16 bit S-box. The inversion function is used during both initialisation and keystream generation, but in different ways in each case. Let x_t and $y_{6,t}$ denote the 16-bit input and output of INV respectively at time t, where $x_t = (x_t^{16}, \ldots, x_t^1)$ and $y_{6,t} = (y_{6,t}^{15}, \ldots, y_{6,t}^0)$. Let x_t^i denote the i-th bit of the input of the S-box, $y_{j,t}$ denote the j-th word of 16 bits ($j = 0, 1, \ldots, 6$) in the pipeline (memory) and $y_{j,t}^i$ denote the i-th bit of $y_{j,t}$, all at time t. INV is an invertible function, $\mathbb{F}_2^{16} \rightarrow \mathbb{F}_2^{16}$ that calculates the inverse of the 16-bit input, modulo the primitive polynomial $X^{16} + X^5 + X^3 + X^2 + 1$. The 16 input bits are taken from 16 register stages at time t as follows.

$$\begin{aligned} (x_t^{16}, \ldots, x_t^1) = & (s_t^{255}, s_t^{244}, s_t^{227}, s_t^{193}, s_t^{161}, s_t^{134}, s_t^{105}, \\ & s_t^{98}, s_t^{74}, s_t^{58}, s_t^{44}, s_t^{21}, s_t^{19}, s_t^{9}, s_t^{6}, s_t^{1}) \end{aligned} \tag{2}$$

The output word $y_{6,t}$ is delayed to become $y_{0,t}$. The delayed version of the 16-bit S-box output, $y_{0,t}$, is treated as 16 bit values as $y_{0,t} = (y_{0,t}^{15}, \ldots, y_{0,t}^0)$. During the initialisation, these bits are fed back to specified stages of the shift register. During keystream generation, only one bit of the output of the INV is used, and this bit contributes to the formation of the keystream bit.

During initialisation, the output of the S-box is stored in a pipeline (memory), $y_{j,t}$ for $0 \leq j \leq 6$, to manage the delay of 7 steps. The total required memory to store seven 16-bit output bits is 112 memory bits. Consequently, the total effective state during the initialisation process is the sum of shift register size and the number of memory bits: 368 bits. During keystream generation, a memory of only 7 bits, m_i for $0 \leq i \leq 6$, is used to perform a delay of 7 steps to stage s^0 that is used to generate a new keystream bit, z_t. The memory stages, $y_{j,t}^0$ are also used to delay $y_{6,t}^0$ for this purpose, but the remaining 15 bits of S-box from $y_{j,t}^1$ to $y_{j,t}^{15}$ for $0 \leq j \leq 6$ are not used

during the keystream generation. Therefore, the total effective state size during keystream generation is 270.

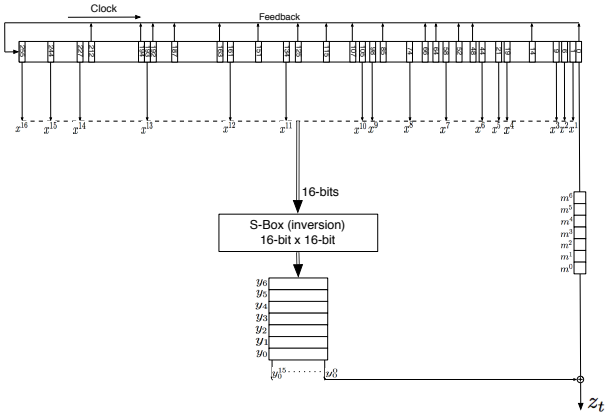

Fig. 2: Keystream generator of Sfinks stream cipher

1) Initialisation Process: The initialisation process takes as input the 80-bit key and 80-bit IV and performs 128 iterations to produce the 368-bit initial state. Once this initial state is obtained, keystream generation can begin. The initialisation process is performed in two phases, which we refer to as *loading* and *diffusion*.

Note: the reference implementation of Sfinks uses $t = -128$ to denote the start of the diffusion process but, for simplicity, we will use $t = 0$ to denote this time point.

a) Loading Phase: Firstly, all of the register stages are set to zero. Then the 80-bit key and 80-bit IV are transferred to specified register positions $s_0^{96+i} = k_i$, for $0 \leq i \leq 79$, and $s_0^{176+i} = v_i$, for $0 \leq i \leq 79$. The register stage $s_0^{95} = 1$ and the remaining $s_0^i = 0$, for $0 \leq i \leq 94$. The output of the S-box is set to all-zero for the first seven 16-bit outputs, $y_{j,0}^i = 0$ for $0 \leq j \leq 6$ and $0 \leq i \leq 15$ (the initial value of the pipeline). In [1] this process is described as necessary to clear the pipeline stages in the hardware implementation and to provide the initial values of the output of the S-box to allow for the delay of 7 steps.

When both the secret key and IV have been transferred and the rest of the state bits (the rest of shift register and the memory cells) are fixed to the designated values, the Sfinks stream cipher is in its *loaded state*. Following this, the diffusion phase begins.

b) Diffusion Phase: The diffusion phase consists of 128 iterations of the initialisation state-update function. Each iteration can be considered as a function which maps the state space to itself. After the diffusion phase is completed, the keystream generator is said to be in its *initial state*.

The state update function for the initialisation process uses Equation 1 and the output of the nonlinear S-box function. The S-box output feeds back into 16 specified stages of the shift register with a time delay of 7 steps as detailed below. Figure 3 gives a general overview of state update function during the diffusion phase of the Sfinks stream cipher.

$$s_t^i = s_{t-1}^{i+1} \oplus y_{0,t-1}^{i \bmod 16} \tag{3}$$

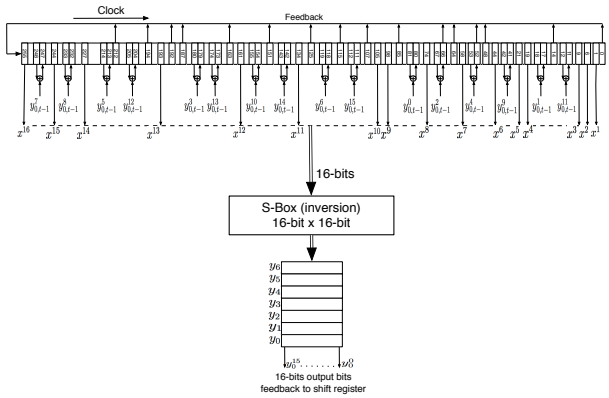

Fig. 3: Initialisation processes of Sfinks stream cipher

for $i = \{11,\ 17,\ 41,\ 52,\ 66,\ 80,\ 111,\ 118,\ 142,\ 154,\ 173,\ 179,\ 204, 213,\ 232,\ 247\}$. All other bits are shifted normally, i.e. $s_t^i = s_{t-1}^{i+1}$ for all other i's . At each iteration, the shift register is clocked and then the INV function is called to calculate the inverse of the 16-bit input to the S-box. This is stored as the S-box output in the memory where the first input is the first output. The 16-bit output of the memory is XORed with the contents of the 16 specified stages of the shift register to form the contents of another 16 stages of the shift register. The S-box function is the only nonlinear component in the initialisation process. A complete description of the state update function is:

$$s_t^i = \begin{cases} s_{t-1}^{i+1} & \text{for } i = \{0, 1, \ldots, 254\} \text{ except } \{11, 17, 41, 52, 66, 80, \\ & 111, 118, 142, 154, 173, 179, 204, 213, 232, 247\} \\ s_{t-1}^{i+1} \oplus y_{0,t-1}^{i \bmod 16} & \text{for } i = \{11, 17, 41, 52, 66, 80, 111, 118, 142, 154, \\ & 173, 179, 204, 213, 232, 247\} \\ \bigoplus_j s_{t-1}^j & \text{for } i = 255 \\ & \text{for } j = \{0, 14, 48, 52, 64, 66, 85, 107, 115, 125, 151, \\ & 163, 187, 192, 194, 212\} \end{cases}$$

$$y_{j,t} = \begin{cases} y_{j+1,t-1} & \text{for } j = \{0, 1, 2, 3, 4, 5\} \\ INV(x^{16}, \ldots, x^1) & \text{for } j = 6 \\ & (x^{16}, \ldots, x^1) = (s_t^{255}, s_t^{244}, s_t^{227}, s_t^{193}, s_t^{161}, s_t^{134}, \\ & s_t^{105}, s_t^{98}, s_t^{74}, s_t^{58}, s_t^{44}, s_t^{21}, s_t^{19}, s_t^9, s_t^6, s_t^1) \end{cases}$$

2) Keystream generation: At $t = 128$, the Sfinks stream cipher has completed the initialisation processes and is ready for keystream generation. A memory component of 7 stages is used to apply a delay of 7 steps to the rightmost bit in the shift register s_t^0. During keystream generation, the output of the INV function is not fed back to shift register, S, and so the register feedback is linear. The rightmost bit in the shift register s_t^0 is shifted to a memory stage m_t^6 at time t. The least significant bit of the 16-bit output value of the S-box, $y_{0,t}^0$ is XORed with the value of stage m_t^0 to produce each keystream bit z_t. The output equation is, $z_t = m_{t-1}^0 \oplus y_{0,t-1}^0$. where

$$m_t^i = \begin{cases} m_{t-1}^{i+1} & \text{for } i = \{0, 1, 2, 3, 4, 5\} \\ s_{t-1}^0 & \text{for } i = 6 \end{cases}$$

IV. SLID PAIRS USING SFINKS

The main purpose of this analysis is to identify conditions under which a slid key-IV pair of the Sfinks cipher is obtained from a specific loaded key-IV after a number of iterations α. Comparing the properties listed in Section II with the Sfinks cipher, we note that properties (a) and (b) apply, but the state update functions during the initialisation and keystream generation processes are somewhat different. The format of the loaded state is specified. For a given particular key-IV pair, the task is to identify when the next loaded state may occur during the initialisation process. Moreover, it is important to determine relationship between the original loaded key-IV pair (k, v) and the next slid key-IV pair (k', v'), which is derived from the original key-IV after a given number of iterations α.

Analysis of the Sfinks stream cipher initialisation process is complicated by the delay of 7 steps in feeding the S-box output back into the shift register, S. However, we observe that the correspondence between the content of stage $s^{95} = 1$ and the next output feedback stage from the S-box s^{80} may lead to slid pairs after a number of iterations α. In the remainder of this research, we refer to stages of S which provide inputs to the S-box as input stages and stages of S which receive outputs from the S-box as output stages respectively.

Some conditions need to be met for loaded state of Sfinks to produce a slid pair after α iterations. Slid pairs can occur after α iterations if the content of the stage s_α^{95} is 1 and the stages from s_α^{94} to s_α^0 are all zeros (to follow the Sfinks' loading format). The input bits to the S-box from $(x_t^{10}$ to $x_t^1)$ should be zeros except some cases as shown below. The output of the S-box $(y_{0,t}^{11}, y_{0,t}^9, y_{0,t}^4, y_{0,t}^2, y_{0,t}^1, y_{0,t}^0)$ should also be zeros except $y_{0,15}^0$ at $t = 15$ and other special cases as shown later.

The content of stage $s_\alpha^{95-\alpha}$ can become 0 by flipping the content of the stage s_0^{95} to 0 during the α iterations. The content of the stage s_α^{95} should be 1. These can be achieved by two steps. Firstly, the content of the stage s_0^{95} can be flipped after 15 iterations, due to the next output stage of the S-box s_{15}^{80}. That requires that the S-box output $y_{0,15}^0 = 1$, so that $s_{15}^{80} \oplus y_{6,15}^0 = 0$. Secondly, to ensure the content of stage s_α^{95} is 1 after α iterations, the content of the stage $s_0^{95+\alpha}$ should be 1. The content of stages $s_0^{95+\alpha-1}$ to s_0^{96} should be zeros and the output stages that lie between s^{95} and s^0 should be maintained to meet the required contents, zeros, during the α iterations. Based on these requirements, it is impossible to get slid pairs before $t = 15$.

For $\alpha \geq 15$ iterations, it may be possible that slid pairs occur. If $\alpha = 15$, then the content of stage s_0^{110} should be 1 as it will be shifted to s_{15}^{95} after 15 iterations. Note that this value of 1 will pass the input stages s^{105} at $t = 5$ and s^{98} at $t = 12$. The input stage s^{105} should be considered where the output stages $(y_{6,t}^{11}, y_{6,t}^9, y_{6,t}^4, y_{6,t}^2, y_{6,t}^1, y_{6,t}^0)$ are zeros. The content of the input stage s^{98} will be 1 at $t = 12$ and its output will feed back at $t = 19$. This is after getting the required slid pair but will affect the content of the pipeline.

We applied exhaustive search over the inputs and outputs of the S-box to find appropriate inputs and outputs to meet the required conditions. For $\alpha = 15$, there is no input value to the S-box with $(x_t^9, \ldots, x_t^1) = (0, \ldots, 0)$ and $x_t^{10} = 1$ that gives $(y_{6,t}^{11}, y_{6,t}^9, y_{6,t}^4, y_{6,t}^2, y_{6,t}^1, y_{6,t}^0) = (0, \ldots, 0)$. Therefore more iterations are required to obtain a slid pair. Additionally, the only input to the S-box that satisfies the condition $(x_t^{10}, \ldots, x_t^1) = (0, \ldots, 0)$ is the 16 zeros $(0, \ldots, 0)$ inputs that gives 16 output of zeros $(0, \ldots, 0)$ at any time t. So, this type of input can not flip the content of s^{80}. Likewise, it is not possible to satisfy the conditions for a slid pair to occur at $t = 16$.

However, for $\alpha \geq 17$ iterations, slid pairs are possible using specific conditions between the outputs of the S-box. In this case we look for two output stages with the shortest distance between them, namely s^{11} and s^{17}. These conditions are applied to the input and output of the S-box during the α iterations. According to the Sfinks' format pattern, the input of the S-box $(x_t^{10}, \ldots, x_t^1)$ must be zeros $(0, \ldots, 0)$ during the affected iterations (from $0 \leq t \leq (\alpha - 7)$ except some iterations). For example, the content of the input stage $s_{\alpha-10}^{105}$ will be 1, which is the input bit of S-box $x_{\alpha-10}^{10} = 1$. These conditions of the input of the S-box apply up to $t = \alpha - 7$. The output of the S-box during the α iterations should maintain the Sfinks' pattern at $t = \alpha$. Therefore, the output bit of the S-box $(y_{0,t}^{11}, y_{0,t}^9, y_{0,t}^4, y_{0,t}^2, y_{0,t}^1, y_{0,t}^0)$ should be zeros. There are some exceptions for these conditions. For example, at time $t = 15$ the $y_{0,15}^0$ should have 1 to flip the value of s_{15}^{80} to 0. As well, it may be required to flip some bits and flip them again to 0 during the α iterations. This process is according to the available input-output pairs of the S-box that follow the required conditions as shown later.

The new-bit of the shift register s_t^{255} is an input of the S-box, x_t^{16}. Therefore, it introduces another condition for the linear feedback function of the shift register. If the required new-bit of the shift register is 0 and the real new-bit is 1, there is a freedom to flip any tap-bit of the linear feedback function to get the required bit without changing any other condition.

The cases $\alpha = 17$, 18 and 19 are investigated in Section V. A general method of generating slid pairs can be applied until $\alpha = 23$ (by fixing 56 key bits and 45 IV bits). After that, the situation becomes more complicated because the interaction between the linear feedback function and the S-box output stages is more complex. Another method may be required to identify slid pairs for higher values of α.

A. Limitations

Recall that the initial loaded state of Sfinks includes a pipeline that contains all zeros. However, it is not possible to obtain a slid pair with all zeros in the pipeline, for the following reason: the content of stage s^{98} will be 1 at $t = \alpha - 3$. Therefore in the last 7 of the α iterations, the content of the input stages of the S-box would not be all zeros (since $x_{\alpha-3}^{98} = 1$). This implies that the output of the S-box at this time, $t = \alpha - 3$, also can not be all zeros.

V. Experimental work

A. Experimental procedures

This section describes the procedures used to obtain the first occurrence of slid pairs, and the simulation of the Sfinks cipher to obtain slid pairs at time $\alpha = 17$, 18 and 19. This simulation requires an exhaustive search over the inputs and outputs of the S-box.

1) slid pair at $\alpha = 17$:

- Apply a specific input to the S-box at $t = 1$ to get $y^1_{6,1} = 1$. So, the input to the S-box satisfies $(x^{10}_1, \ldots, x^1_1) = (0, \ldots, 0)$ and the output must have $(y^{11}_{6,1}, y^9_{6,1}, y^4_{6,1}, y^2_{6,1}, y^0_{6,1}) = (0, \ldots, 0)$ and $y^1_{6,1} = 1$. This will flip s^{17}_8 to 1.
- For $2 \leq t \leq 6$, the input to the S-box must satisfy $(x^{10}_t, \ldots, x^1_t) = (0, \ldots, 0)$ and give $(y^{11}_{6,t}, y^9_{6,t}, y^4_{6,t}, y^2_{6,t}, y^1_{6,t}, y^0_{6,t}) = (0, \ldots, 0)$.
- The value of $s^{17}_8 = 1$ will be shifted to s^{11}_{14} after 6 iterations.
- At time $t = 7$, a specific input to the S-box is required so that its output will flip the value of s^{11}_{14} to 0 at $t = 14$ with the input to the S-box $(x^9_7, \ldots, x^1_7) = (0, \ldots, 0)$ and $x^{10}_7 = 1$ and the output $(y^9_{6,7}, y^4_{6,7}, y^2_{6,7}, y^1_{6,7}, y^0_{6,7}) = (0, \ldots, 0)$ and $y^{11}_{6,7} = 1$. This output will result in flipping the s^{11}_{14} to 0.
- At $t = 8$, the input to the S-box should be specified to be $(x^1_8, \ldots, x^{10}_8) = (0, \ldots, 0)$. This input must result in the output $(y^{11}_{6,8}, y^9_{6,8}, y^4_{6,8}, y^2_{6,8}, y^1_{6,8}) = (0, \ldots, 0)$ and $y^0_{6,8} = 1$. At $t = 15$, it will flip the value of s^{80}_{15} to 0.
- For $9 \leq t \leq 10$, apply the following input to the S-box to insure $(x^{10}_t, \ldots, x^1_t) = (0, \ldots, 0)$ to get $(y^{11}_{6,t}, y^9_{6,t}, y^4_{6,t}, y^2_{6,t}, y^1_{6,t}, y^0_{6,t}) = (0, \ldots, 0)$.
- After $t = 10$, it is not important to restrict the input to the S-box.
- To maintain the new-bit, s^{255}_t for $t \geq 1$, requires 10 tap-bits of the linear feedback function. The tap-bits are chosen to be s^{125}_0 to s^{134}_0 that carry k^{29} to k^{38} respectively.

Table I shows an exhaustive search of the required input/output pairs of the S-box that satisfy the above procedure. The c_1, c_2, c_3, b_1 and b_2 are for later reference in the next section.

TABLE I: Input/output of the S-box to get the slid pairs at $\alpha = 17$

Input			Output	
time	Value		time	Value
1	0011110000000000	$\Longrightarrow$	8	0110000100000010
7	$c_1 = 0001101000000000$	$\Longrightarrow$	14	0000110001101000
	$c_2 = 1010111000000000$			1000110011000000
	$c_3 = 1100111000000000$			0011110100100000
8	$b_1 = 1000000000000000$	$\Longrightarrow$	15	1010000101101001
	$b_2 = 0111100000000000$			0011000010000001

2) slid pair at $\alpha = 18$:

- From $t = 1$, apply the following input to the S-box to insure the $(x^{10}_1, \ldots, x^1_1) = (0, \ldots, 0)$ to get $(y^{11}_{6,1}, y^9_{6,1}, y^4_{6,1}, y^2_{6,1}, y^1_{6,1}, y^0_{6,1}) = (0, \ldots, 0)$.
- Apply a specific input to the S-box at $t = 2$ to get $y^1_{6,2} = 1$ at $t = 9$, the input to the S-box is $(x^{10}_t, \ldots, x^1_t) = (0, \ldots, 0)$ and the output is $(y^{11}_{6,t}, y^9_{6,t}, y^4_{6,t}, y^2_{6,t}, y^0_{6,t}) = (0, \ldots, 0)$ and $y^1_{6,t} = 1$. This will flip s^{17}_9 to 1.
- The value of $s^{17}_9 = 1$ will be shifted to s^{11}_{15} after 6 iterations.
- For $3 \leq t \leq 7$, apply the following inputs to the S-box to insure $(x^{10}_t, \ldots, x^1_t) = (0, \ldots, 0)$ to get $(y^{11}_{6,t}, y^9_{6,t}, y^4_{6,t}, y^2_{6,t}, y^1_{6,t}, y^0_{6,t}) = (0, \ldots, 0)$.
- At time $t = 8$, a specific input to the S-box is required so that its output will flip the value of s^{11}_{15} and s^{80}_{15} to 0 at $t = 15$ with the input to the S-box $(x^9_8, \ldots, x^1_8) = (0, \ldots, 0)$ and $x^{10}_8 = 1$ and the output $(y^9_{6,8}, y^4_{6,8}, y^2_{6,8}, y^1_{6,8}) = (0, \ldots, 0)$ $y^0_{6,8} = 1$ and $y^{11}_{6,8} = 1$. This output will result in flipping s^{11}_{15} and s^{80}_{15} to 0.
- For $9 \leq t \leq 11$, apply the following input to the S-box to insure $(x^{10}_t, \ldots, x^1_t) = (0, \ldots, 0)$ to get $(y^{11}_{6,t}, y^9_{6,t}, y^4_{6,t}, y^2_{6,t}, y^1_{6,t}, y^0_{6,t}) = (0, \ldots, 0)$.
- After $t = 11$, it is not important to restrict the input to the S-box.
- To maintain the new-bit, s^{255}_t for $t \geq 1$, requires 11 tap-bits of the linear feedback function. The tap-bits are chosen to be s^{212}_0 to s^{222}_0 that carry v^{36} to v^{46} respectively.

Table II gives a result of an exhaustive search of the required input/output pairs of the S-box that satisfy the above procedure.

TABLE II: Input/output of the S-box to get the slid pairs at $\alpha = 18$

Input			Output	
time	Value		time	Value
2	0011110000000000	$\Longrightarrow$	9	0110000100000010
8	$d_1 = 1011011000000000$	$\Longrightarrow$	15	1000100000001001
	$d_2 = 0000111000000000$			1010100101001001

3) slid pair at $\alpha = 19$:

- For $1 \leq t \leq 2$, apply the zeros inputs to the S-box to insure $(x^{10}_t, \ldots, x^1_t) = (0, \ldots, 0)$ to get $(y^{11}_{6,t}, y^9_{6,t}, y^4_{6,t}, y^2_{6,t}, y^1_{6,t}, y^0_{6,t}) = (0, \ldots, 0)$.
- Apply specific input to the S-box at $t = 3$ to get $y^1_{6,3} = 1$. So , the input to the S-box is $(x^{10}_3, \ldots, x^1_3) = (0, \ldots, 0)$ and the output is $(y^{11}_{6,3}, y^9_{6,3}, y^4_{6,3}, y^2_{6,3}, y^0_{6,3}) = (0, \ldots, 0)$ and $y^1_{6,3} = 1$. This will flip s^{17}_{10} to 1 at time $t = 10$.
- For $4 \leq t \leq 7$, apply the zeros input to the S-box to insure $(x^{10}_t, \ldots, x^1_t) = (0, \ldots, 0)$ to get $(y^{11}_{6,t}, y^9_{6,t}, y^4_{6,t}, y^2_{6,t}, y^1_{6,t}, y^0_{6,t}) = (0, \ldots, 0)$.
- At $t = 8$, the input to the S-box should be specified to be $(x^1_8, \ldots, x^{10}_8) = (0, \ldots, 0)$. This input will result in the output $(y^{11}_{6,8}, y^9_{6,8}, y^4_{6,8}, y^2_{6,8}, y^1_{6,8}) = (0, \ldots, 0)$ and $y^0_{6,8} = 1$. At $t = 15$, it will flip the value of s^{80}_{15} to 0.
- The value of s^{17}_{10} = 1 will be shifted to s^{11}_{16} after 6 iterations.
- At time $t = 9$, a specific input to the S-box is required so that its output will flip the value of s^{11}_{16} to 0. The input to the S-box are $(x^9_9, \ldots, x^1_9) = (0, \ldots, 0)$ and $x^{10}_9 =$

1 and the output $(y^9_{6,9}, y^4_{6,9}, y^2_{6,9}, y^1_{6,9}, y^0_{6,9}) = (0, \ldots, 0)$ and $y^{11}_{6,9} = 1$. This output will result in flipping s^{11}_{16} to 0 at time $t = 16$.

- For $10 \leq t \leq 12$, apply the zeros input to the S-box to insure that $(x^{10}_t, \ldots, x^1_t) = (0, \ldots, 0)$ to get $(y^{11}_{6,t}, y^9_{6,t}, y^4_{6,t}, y^2_{6,t}, y^1_{6,t}, y^0_{6,t}) = (0, \ldots, 0)$.
- After $t = 12$, it is not important to restrict the input to the S-box.
- To maintain the new-bit, s^{255}_t for $t \geq 1$, requires 12 tap-bits of the linear feedback function. The tap-bits are chosen to be s^{212}_0 to s^{223}_0 that carry v^{36} to v^{47} respectively.

As shown above, Table III shows all the possible input/output values to the S-box using an exhaustive search of the required of the S-box that satisfy the above procedure.

TABLE III: Input/output of the S-box to get the slid pairs at $\alpha = 19$

Input			Output	
time	Value		time	Value
3	0011110000000000	$\Longrightarrow$	10	0110000100000010
8	$b_1 = 1000000000000000$	$\Longrightarrow$	15	1010000101101001
	$b_2 = 0111100000000000$			0011000010000001
9	$e_1 = 0001101000000000$	$\Longrightarrow$	16	0000110001101000
	$e_2 = 1010111000000000$			1000110011000000
	$e_3 = 1100111000000000$			0011110100100000

B. Findings and results

This section focuses on the result of the analysis of slid pairs according to the previous procedures that are applied to the Sfinks cipher. The findings of slid pairs are presented after different number of iterations at $t = 17$, 18 and 19. At each specific number of iterations, the slid pairs have specific conditions and relationship between these slid key-IV pairs. To remind the reader, $y^i_{j,t}$ denotes the i-th bit of j-th word of the memory at time t, for $i \in \{0, 1, \ldots, 15\}$.

1) slid pair at $\alpha = 17$: The slid pair is found for a key-IV with conditions. These conditions to get another key-IV pair at $\alpha = 17$ are to have specific values for some key-IV bits. There are 40 and 30 bits of key and IV respectively which should have specific values as shown below.

$$k^i = 0,\ i \in \{75, 74, 71, 70, 69, 68, 67, 48, 47, 46, 44, 43, 42, 41, 40, 19, 18, 17, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0\}$$
$$k^i = 1,\ i \in \{72, 66, 39, 16\}$$

$$(k^{73}, k^{45}) = \begin{cases} (0,0) & \text{for } x_8 = b_1 \text{ and } x_7 = c_1 \\ (0,1) & \text{for } x_8 = b_1 \text{ and } x_7 = c_2 \\ (0,1) & \text{for } x_8 = b_1 \text{ and } x_7 = c_3 \\ (1,0) & \text{for } x_8 = b_2 \text{ and } x_7 = c_1 \\ (1,1) & \text{for } x_8 = b_2 \text{ and } x_7 = c_2 \\ (1,1) & \text{for } x_8 = b_2 \text{ and } x_7 = c_3 \end{cases}$$

$$v^i = 0,\ i \in \{78, 77, 74, 73, 72, 71, 70, 69, 61, 60, 57, 56, 55, 54, 53, 27, 26, 23, 22, 21, 20, 19\}$$
$$v^i = 1,\ i \in \{52, 18\}$$

$$(v^{76}, v^{75}, v^{59}, v^{58}, v^{25}, v^{24}) = \begin{cases} (0,0,0,0,0,1) & \text{for } x_8 = b_1 \text{ and } x_7 = c_1 \\ (0,0,0,1,0,0) & \text{for } x_8 = b_1 \text{ and } x_7 = c_2 \\ (0,1,0,0,0,0) & \text{for } x_8 = b_1 \text{ and } x_7 = c_3 \\ (1,0,1,0,1,1) & \text{for } x_8 = b_2 \text{ and } x_7 = c_1 \\ (1,0,1,1,1,0) & \text{for } x_8 = b_2 \text{ and } x_7 = c_2 \\ (1,1,1,0,1,0) & \text{for } x_8 = b_2 \text{ and } x_7 = c_3 \end{cases}$$

There are 40 and 50 free key and IV bits respectively as shown below.

$$k^i,\ i \in \{79, 78, 77, 76, 65, 64, 63, 62, 61, 60, 59, 58, 57, 56, 55, 54, 53, 52, 51, 50, 49, 38, 37, 36, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21, 20\}$$

$$v^i,\ i \in \{79, 68, 67, 66, 65, 64, 63, 62, 51, 50, 49, 48, 47, 46, 45, 44, 43, 42, 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 30, 29, 28, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0\}$$

There are 6 possibilities of the input to the S-box to satisfy the required condition to get slid pairs at $t = 17$. The second key-IV, (k', v'), pair can be expressed in the form of the first key-IV, (k, v), pair as follow.
Note: Terms printed in boldface are known bits, the values of which depend on the input of the S-box as shown in Table IV.

$$\begin{aligned}
v'^{79} &= v^{52} \oplus \mathbf{y^5_{6,8}} \oplus v^{34} \oplus v^{32} \oplus v^{27} \oplus v^3 \oplus k^{71} \oplus k^{45} \oplus k^{35} \oplus k^{27} \oplus k^5 \\
v'^{78} &= v^{51} \oplus \mathbf{y^5_{6,7}} \oplus v^{33} \oplus v^{31} \oplus v^{26} \oplus v^2 \oplus k^{70} \oplus k^{44} \oplus k^{34} \oplus k^{26} \oplus k^4 \\
v'^{77} &= v^{50} \oplus v^{32} \oplus v^{30} \oplus v^{25} \oplus v^1 \oplus k^{69} \oplus k^{43} \oplus k^{33} \oplus k^{25} \oplus k^3 \\
v'^{76} &= v^{49} \oplus v^{31} \oplus v^{29} \oplus v^{24} \oplus v^0 \oplus k^{68} \oplus k^{42} \oplus k^{32} \oplus k^{24} \oplus k^2 \\
v'^{75} &= v^{48} \oplus v^{30} \oplus v^{28} \oplus v^{23} \oplus k^{79} \oplus k^{67} \oplus k^{41} \oplus k^{31} \oplus k^{23} \oplus k^1 \\
v'^{74} &= v^{47} \oplus v^{29} \oplus v^{27} \oplus v^{22} \oplus k^{78} \oplus k^{66} \oplus k^{40} \oplus k^{30} \oplus k^{22} \oplus k^0 \oplus 1 \\
v'^{73} &= v^{46} \oplus v^{28} \oplus v^{26} \oplus v^{21} \oplus k^{77} \oplus k^{65} \oplus k^{39} \oplus k^{29} \oplus k^{21} \oplus 1 \\
v'^{72} &= v^{45} \oplus v^{27} \oplus v^{25} \oplus v^{20} \oplus k^{76} \oplus k^{64} \oplus k^{38} \oplus k^{28} \oplus k^{20} = 0 \\
v'^{71} &= v^{44} \oplus v^{26} \oplus v^{24} \oplus v^{19} \oplus k^{75} \oplus k^{63} \oplus k^{37} \oplus k^{27} \oplus k^{19} \oplus 1 = 0 \\
v'^{70} &= v^{43} \oplus v^{25} \oplus v^{23} \oplus v^{18} \oplus k^{74} \oplus k^{62} \oplus k^{36} \oplus k^{26} \oplus k^{18} = \mathbf{a_1} \\
v'^{69} &= v^{42} \oplus v^{24} \oplus v^{22} \oplus v^{17} \oplus k^{73} \oplus k^{61} \oplus k^{35} \oplus k^{25} \oplus k^{17} = \mathbf{a_2} \\
v'^{68} &= v^{41} \oplus v^{23} \oplus v^{21} \oplus v^{16} \oplus k^{72} \oplus k^{60} \oplus k^{34} \oplus k^{24} \oplus 1 = 0 \\
v'^{67} &= v^{40} \oplus v^{22} \oplus v^{20} \oplus v^{15} \oplus k^{71} \oplus k^{59} \oplus k^{33} \oplus k^{23} \oplus k^{15} = 0 \\
v'^{66} &= v^{39} \oplus v^{21} \oplus v^{19} \oplus v^{14} \oplus k^{70} \oplus k^{58} \oplus k^{32} \oplus k^{22} \oplus k^{14} = 0 \\
v'^{65} &= v^{38} \oplus v^{20} \oplus v^{18} \oplus v^{13} \oplus k^{69} \oplus k^{57} \oplus k^{31} \oplus k^{21} \oplus k^{13} = 0 \\
v'^{64} &= v^{37} \oplus v^{19} \oplus v^{17} \oplus v^{12} \oplus k^{68} \oplus k^{56} \oplus k^{30} \oplus k^{20} \oplus k^{12} = 0 \\
v'^{63} &= v^{36} \oplus v^{18} \oplus v^{16} \oplus v^{11} \oplus k^{67} \oplus k^{55} \oplus k^{29} \oplus k^{19} \oplus k^{11} = 0
\end{aligned}$$

$$\begin{aligned}
v'^{62} &= v^{79} & v'^{57} &= v^{74} = 0 & v'^{52} &= v^{69} = 0 \\
v'^{61} &= v^{78} = 0 & v'^{56} &= v^{73} = 0 & v'^{51} &= v^{68} \\
v'^{60} &= v^{77} = 0 & v'^{55} &= v^{72} = 0 & v'^{50} &= v^{67} \\
v'^{59} &= \mathbf{v^{76}} & v'^{54} &= v^{71} \oplus \mathbf{y^8_{6,8}} & v'^{49} &= v^{66} \\
v'^{58} &= \mathbf{v^{75}} & v'^{53} &= v^{70} \oplus \mathbf{y^8_{6,7}} & v'^{48} &= v^{65}
\end{aligned}$$

$v'^{47} = v^{64} \oplus 1$
$v'^{46} = v^{63}$
$v'^{45} = v^{62}$
$v'^{44} = v^{61} = 0$
$v'^{43} = v^{60} = 0$
$v'^{42} = \mathbf{v^{59}}$
$v'^{41} = \mathbf{v^{58}}$
$v'^{40} = v^{57} = 0$
$v'^{39} = v^{56} = 0$
$v'^{38} = v^{55} = 0$
$v'^{37} = v^{54} = 0$
$v'^{36} = v^{53} = 0$
$v'^{35} = 1 \oplus \mathbf{y^5_{6,8}}$
$v'^{34} = v^{51} \oplus \mathbf{y^5_{6,7}}$
$v'^{33} = v^{50}$
$v'^{32} = v^{49}$
$v'^{31} = v^{48}$
$v'^{30} = v^{47}$
$v'^{29} = v^{46}$
$v'^{28} = v^{45}$
$v'^{27} = v^{44}$
$v'^{26} = v^{43} \oplus \mathbf{y^{12}_{6,8}}$
$v'^{25} = v^{42} \oplus \mathbf{y^{12}_{6,7}}$
$v'^{24} = v^{41}$
$v'^{23} = v^{40}$
$v'^{22} = v^{39}$
$v'^{21} = v^{38}$
$v'^{20} = v^{37}$
$v'^{19} = v^{36}$
$v'^{18} = v^{35}$
$v'^{17} = v^{34}$
$v'^{16} = v^{33}$
$v'^{15} = v^{32}$
$v'^{14} = v^{31}$
$v'^{13} = v^{30}$
$v'^{12} = v^{29}$
$v'^{11} = v^{28}$
$v'^{10} = v^{27} = 0$
$v'^{9} = v^{26} = 0$
$v'^{8} = \mathbf{v^{25}}$
$v'^{7} = \mathbf{v^{24}}$
$v'^{6} = v^{23} = 0$
$v'^{5} = v^{22} = 0$

$v'^{4} = v^{21} = 0$
$v'^{3} = v^{20} = 0$
$v'^{2} = v^{19} = 0$
$v'^{1} = 1 \oplus \mathbf{y^3_{6,8}}$
$v'^{0} = v^{17} \oplus \mathbf{y^3_{6,7}}$
$k'^{79} = v^{16}$
$k'^{78} = v^{15}$
$k'^{77} = v^{14}$
$k'^{76} = v^{13}$
$k'^{75} = v^{12} \oplus 1$
$k'^{74} = v^{11} \oplus \mathbf{y^{13}_{6,7}}$
$k'^{73} = v^{10}$
$k'^{72} = v^{9}$
$k'^{71} = v^{8}$
$k'^{70} = v^{7}$
$k'^{69} = v^{6}$
$k'^{68} = v^{5} \oplus 1$
$k'^{67} = v^{4}$
$k'^{66} = v^{3}$
$k'^{65} = v^{2}$
$k'^{64} = v^{1}$
$k'^{63} = v^{0}$
$k'^{62} = k^{79}$
$k'^{61} = k^{78}$
$k'^{60} = k^{77}$
$k'^{59} = k^{76}$
$k'^{58} = k^{75} = 0$
$k'^{57} = k^{74} = 0$
$k'^{56} = \mathbf{k^{73}}$
$k'^{55} = k^{72} \oplus 1 = 0$
$k'^{54} = k^{71} = 0$
$k'^{53} = k^{70} = 0$
$k'^{52} = k^{69} = 0$
$k'^{51} = k^{68} = 0$
$k'^{50} = k^{67} = 0$
$k'^{49} = k^{66} = 1$
$k'^{48} = k^{65}$
$k'^{47} = k^{64}$
$k'^{46} = k^{63}$
$k'^{45} = k^{62}$
$k'^{44} = k^{61}$
$k'^{43} = k^{60}$
$k'^{42} = k^{59}$

$k'^{41} = k^{58}$
$k'^{40} = k^{57}$
$k'^{39} = k^{56}$
$k'^{38} = k^{55}$
$k'^{37} = k^{54} \oplus 1$
$k'^{36} = k^{53}$
$k'^{35} = k^{52}$
$k'^{34} = k^{51}$
$k'^{33} = k^{50}$
$k'^{32} = k^{49}$
$k'^{31} = k^{48} = 0$
$k'^{30} = k^{47} = 0$
$k'^{29} = k^{46} = 0$
$k'^{28} = \mathbf{k^{45}}$
$k'^{27} = k^{44} = 0$
$k'^{26} = k^{43} = 0$
$k'^{25} = k^{42} = 0$
$k'^{24} = k^{41} = 0$
$k'^{23} = k^{40} = 0$
$k'^{22} = k^{39} = 1$
$k'^{21} = k^{38}$
$k'^{20} = k^{37} \oplus \mathbf{y^6_{6,8}}$
$k'^{19} = k^{36} \oplus \mathbf{y^6_{6,7}}$
$k'^{18} = k^{35}$
$k'^{17} = k^{34}$
$k'^{16} = k^{33}$
$k'^{15} = k^{32}$
$k'^{14} = k^{31}$
$k'^{13} = k^{30} \oplus \mathbf{y^{15}_{6,8}}$
$k'^{12} = k^{29} \oplus \mathbf{y^{15}_{6,7}}$
$k'^{11} = k^{28}$
$k'^{10} = k^{27}$
$k'^{9} = k^{26}$
$k'^{8} = k^{25}$
$k'^{7} = k^{24}$
$k'^{6} = k^{23}$
$k'^{5} = k^{22}$
$k'^{4} = k^{21}$
$k'^{3} = k^{20}$
$k'^{2} = k^{19} = 0$
$k'^{1} = k^{18} = 0$
$k'^{0} = k^{17} = 0$

TABLE IV: 6 alternative inputs and outputs to the S-box at $\alpha = 17$

Conditions		Input										
x_8	x_7	$\mathbf{a_1}$	$\mathbf{v^{76}}$	$\mathbf{v^{59}}$	$\mathbf{v^{25}}$	$\mathbf{k^{73}}$	$\mathbf{a_2}$	$\mathbf{v^{75}}$	$\mathbf{v^{58}}$	$\mathbf{v^{24}}$	$\mathbf{k^{45}}$	
b_1	c_1	1	0	0	0	0	0	0	0	1	0	
b_1	c_2	1	0	0	0	0	1	0	1	0	1	
b_1	c_3	1	0	0	0	0	1	1	0	0	1	...
b_2	c_1	0	1	1	1	1	0	0	0	1	0	
b_2	c_2	0	1	1	1	1	1	0	1	0	1	
b_2	c_3	0	1	1	1	1	1	1	0	0	1	

Output													
	$\mathbf{y^5_{6,8}}$	$\mathbf{y^8_{6,8}}$	$\mathbf{y^{12}_{6,8}}$	$\mathbf{y^3_{6,8}}$	$\mathbf{y^6_{6,8}}$	$\mathbf{y^{15}_{6,8}}$	$\mathbf{y^5_{6,7}}$	$\mathbf{y^8_{6,7}}$	$\mathbf{y^{12}_{6,7}}$	$\mathbf{y^3_{6,7}}$	$\mathbf{y^{13}_{6,7}}$	$\mathbf{y^6_{6,7}}$	$\mathbf{y^{15}_{6,7}}$
	1	1	0	1	1	1	1	0	0	1	0	1	0
	1	1	0	1	1	1	0	0	0	0	0	1	1
...	1	1	0	1	1	1	1	1	1	0	1	0	0
	0	0	1	0	0	0	1	0	0	1	0	1	0
	0	0	1	0	0	0	0	0	0	0	0	1	1
	0	0	1	0	0	0	1	1	1	0	1	0	0

Note that 32 bits of v' and 23 bits of k' are fixed by the above relations.

2) slid pair at $\alpha = 18$: For slid pair at $\alpha = 18$, the conditions to get another key-IV pair are to have specific values for some key-IV bits at $t = 0$. There are 43 and 33 bits of key and IV respectively which must have fixed values as shown below.

$$k^i = 0,\ i \in \{76, 75, 74, 72, 71, 70, 69, 68, 66, 49, 48, 47, 45, 44, 43, 42, 41, 39, 20, 19, 18, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, , 0\}$$

$$k^i = 1,\ i \in \{67, 46, 40, 17\}$$

$$k^{73} = \begin{cases} (0) & \text{for } x_8 = d_1 \\ (1) & \text{for } x_8 = d_2 \end{cases}$$

$$v^i = 0,\ i \in \{79, 78, 77, 76, 75, 74, 73, 72, 71, 70, 69, 62, 61, 60 58, 57, 56, 55, 54, 52, 28, 27, 26, 24, 23, 22, 21, 20, 18\}$$

$$v^i = 1,\ i \in \{53, 19\}$$

$$(v^{59}, v^{25}) = \begin{cases} (1,1) & \text{for } x_8 = d_1 \\ (0,0) & \text{for } x_8 = d_2 \end{cases}$$

There are 37 and 47 free key and IV bits respectively as shown.

$$k^i,\ i \in \{79, 78, 77, 65, 64, 63, 62, 61, 60, 59, 58, 57, 56, 55, 54, 53, 52, 51, 50, 38, 37, 36, 35, 34, 33, 32, 31, 30, 29, 28, 27, 26, 25, 24, 23, 22, 21\}$$

$$v^i,\ i \in \{68, 67, 66, 65, 64, 63, 51, 50, 49, 48, 47, 46, 45, 44, 43, 42, 41, 40, 39, 38, 37, 36, 35, 34, 33, 32, 31, 30, 29, 17, 16, 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0\}$$

There are two possibilities of the input to the S-box to satisfy the required conditions to get a slid pair at $\alpha = 18$. The second key-IV pair, (k', v'), can be expressed in term of the first key-IV , (k, v), as follow.
Note: Terms printed in boldface are known bits, the values of which depend on the input of the S-box as shown in Table V.

$v'^{79} = v^{53} \oplus v^{35} \oplus v^{33} \oplus v^{28} \oplus v^4 \oplus k^{72} \oplus k^{46} \oplus k^{36} \oplus k^{28} \oplus k^6$
$v'^{78} = v^{52} \oplus v^{34} \oplus v^{32} \oplus v^{27} \oplus v^3 \oplus k^{71} \oplus k^{45} \oplus k^{35} \oplus k^{27} \oplus k^5$
$v'^{77} = v^{51} \oplus v^{33} \oplus v^{31} \oplus v^{26} \oplus v^2 \oplus k^{70} \oplus k^{44} \oplus k^{34} \oplus k^{26} \oplus k^4$
$v'^{76} = v^{50} \oplus v^{32} \oplus v^{30} \oplus \mathbf{v^{25}} \oplus v^1 \oplus k^{69} \oplus k^{43} \oplus k^{33} \oplus k^{25} \oplus k^3$
$v'^{75} = v^{49} \oplus v^{31} \oplus v^{29} \oplus v^{24} \oplus v^0 \oplus k^{68} \oplus k^{42} \oplus k^{32} \oplus k^{24} \oplus k^2$
$v'^{74} = v^{48} \oplus v^{30} \oplus v^{28} \oplus v^{23} \oplus k^{79} \oplus k^{67} \oplus k^{41} \oplus k^{31} \oplus k^{23} \oplus k^1 \oplus 1$
$v'^{73} = v^{47} \oplus v^{29} \oplus v^{27} \oplus v^{22} \oplus k^{78} \oplus k^{66} \oplus k^{40} \oplus k^{30} \oplus k^{22} \oplus k^0$
$v'^{72} = v^{46} \oplus v^{28} \oplus v^{26} \oplus v^{21} \oplus k^{77} \oplus k^{65} \oplus k^{39} \oplus k^{29} \oplus k^{21} \oplus 1 = 0$
$v'^{71} = v^{45} \oplus v^{27} \oplus \mathbf{v^{25}} \oplus v^{20} \oplus k^{76} \oplus k^{64} \oplus k^{38} \oplus k^{28} \oplus k^{20} = 0$
$v'^{70} = v^{44} \oplus v^{26} \oplus v^{24} \oplus v^{19} \oplus k^{75} \oplus k^{63} \oplus k^{37} \oplus k^{27} \oplus k^{19} = 0$
$v'^{69} = v^{43} \oplus \mathbf{v^{25}} \oplus v^{23} \oplus v^{18} \oplus k^{74} \oplus k^{62} \oplus k^{36} \oplus k^{26} \oplus k^{18} = \mathbf{a_3}$
$v'^{68} = v^{42} \oplus v^{24} \oplus v^{22} \oplus v^{17} \oplus \mathbf{k^{73}} \oplus k^{61} \oplus k^{35} \oplus k^{25} \oplus k^{17} \oplus 1 = 0$
$v'^{67} = v^{41} \oplus v^{23} \oplus v^{21} \oplus v^{16} \oplus k^{72} \oplus k^{60} \oplus k^{34} \oplus k^{24} \oplus k^{16} = 0$
$v'^{66} = v^{40} \oplus v^{22} \oplus v^{20} \oplus v^{15} \oplus k^{71} \oplus k^{59} \oplus k^{33} \oplus k^{23} \oplus k^{15} = 0$
$v'^{65} = v^{39} \oplus v^{21} \oplus v^{19} \oplus v^{14} \oplus k^{70} \oplus k^{58} \oplus k^{32} \oplus k^{22} \oplus k^{14} = 0$
$v'^{64} = v^{38} \oplus v^{20} \oplus v^{18} \oplus v^{13} \oplus k^{69} \oplus k^{57} \oplus k^{31} \oplus k^{21} \oplus k^{13} = 0$
$v'^{63} = v^{37} \oplus v^{19} \oplus v^{17} \oplus v^{12} \oplus k^{68} \oplus k^{56} \oplus k^{30} \oplus k^{20} \oplus k^{12} = 0$
$v'^{62} = v^{36} \oplus v^{18} \oplus v^{16} \oplus v^{11} \oplus k^{67} \oplus k^{55} \oplus k^{29} \oplus k^{19} \oplus k^{11} = 0$

$v'^{61} = v^{79} = 0$
$v'^{60} = v^{78} = 0$
$v'^{59} = v^{77} = 0$
$v'^{58} = v^{76} = 0$
$v'^{57} = v^{75} = 0$
$v'^{56} = v^{74} = 0$
$v'^{55} = v^{73} = 0$
$v'^{54} = v^{72} = 0$
$v'^{53} = v^{71} \oplus \mathbf{y^8_{6,8}}$
$v'^{52} = v^{70} = 0$
$v'^{51} = v^{69} = 0$
$v'^{50} = v^{68}$
$v'^{49} = v^{67}$
$v'^{48} = v^{66}$
$v'^{47} = v^{65} \oplus 1$
$v'^{46} = v^{64}$
$v'^{45} = v^{63}$
$v'^{44} = v^{62} = 0$
$v'^{43} = v^{61} = 0$
$v'^{42} = v^{60} = 0$
$v'^{41} = \mathbf{v^{59}}$
$v'^{40} = v^{58} = 0$
$v'^{39} = v^{57} = 0$
$v'^{38} = v^{56} = 0$
$v'^{37} = v^{55} = 0$
$v'^{36} = v^{54} = 0$
$v'^{35} = v^{53} = 1$
$v'^{34} = v^{52} = 0$
$v'^{33} = v^{51}$
$v'^{32} = v^{50}$
$v'^{31} = v^{49}$
$v'^{30} = v^{48}$
$v'^{29} = v^{47}$
$v'^{28} = v^{46}$
$v'^{27} = v^{45}$
$v'^{26} = v^{44}$
$v'^{25} = v^{43}$
$v'^{24} = v^{42}$
$v'^{23} = v^{41}$
$v'^{22} = v^{40}$
$v'^{21} = v^{39}$
$v'^{20} = v^{38}$
$v'^{19} = v^{37}$
$v'^{18} = v^{36}$
$v'^{17} = v^{35}$
$v'^{16} = v^{34}$
$v'^{15} = v^{33}$
$v'^{14} = v^{32}$
$v'^{13} = v^{31}$
$v'^{12} = v^{30}$
$v'^{11} = v^{29}$
$v'^{10} = v^{28} = 0$
$v'^{9} = v^{27} = 0$
$v'^{8} = v^{26} = 0$
$v'^{7} = \mathbf{v^{25}}$
$v'^{6} = v^{24} = 0$
$v'^{5} = v^{23} = 0$
$v'^{4} = v^{22} = 0$
$v'^{3} = v^{21} = 0$
$v'^{2} = v^{20} = 0$
$v'^{1} = v^{19} = 1$
$v'^{0} = v^{18} \oplus 1 = 1$
$k'^{79} = v^{17}$
$k'^{78} = v^{16}$
$k'^{77} = v^{15}$
$k'^{76} = v^{14}$
$k'^{75} = v^{13}$
$k'^{74} = v^{12} \oplus \mathbf{y^{13}_{6,8}}$
$k'^{73} = v^{11}$
$k'^{72} = v^{10}$
$k'^{71} = v^{9}$
$k'^{70} = v^{8}$
$k'^{69} = v^{7}$
$k'^{68} = v^{6} \oplus 1$
$k'^{67} = v^{5}$
$k'^{66} = v^{4}$
$k'^{65} = v^{3}$
$k'^{64} = v^{2}$
$k'^{63} = v^{1}$
$k'^{62} = v^{0}$
$k'^{61} = k^{79}$
$k'^{60} = k^{78}$
$k'^{59} = k^{77}$
$k'^{58} = k^{76} = 0$
$k'^{57} = k^{75} = 0$
$k'^{56} = k^{74} = 0$
$k'^{55} = \mathbf{k^{73}}$
$k'^{54} = k^{72} = 0$
$k'^{53} = k^{71} = 0$
$k'^{52} = k^{70} = 0$
$k'^{51} = k^{69} = 0$
$k'^{50} = k^{68} = 0$
$k'^{49} = k^{67} = 1$
$k'^{48} = k^{66} = 0$
$k'^{47} = k^{65}$
$k'^{46} = k^{64}$
$k'^{45} = k^{63}$
$k'^{44} = k^{62}$
$k'^{43} = k^{61}$
$k'^{42} = k^{60}$
$k'^{41} = k^{59}$
$k'^{40} = k^{58}$
$k'^{39} = k^{57}$
$k'^{38} = k^{56}$
$k'^{37} = k^{55} \oplus 1$
$k'^{36} = k^{54}$
$k'^{35} = k^{53}$
$k'^{34} = k^{52}$
$k'^{33} = k^{51}$
$k'^{32} = k^{50}$
$k'^{31} = k^{49} = 0$
$k'^{30} = k^{48} = 0$
$k'^{29} = k^{47} = 0$
$k'^{28} = k^{46} = 1$
$k'^{27} = k^{45} = 0$
$k'^{26} = k^{44} = 0$
$k'^{25} = k^{43} = 0$
$k'^{24} = k^{42} = 0$
$k'^{23} = k^{41} = 0$
$k'^{22} = k^{40} = 1$
$k'^{21} = k^{39} = 0$
$k'^{20} = k^{38}$
$k'^{19} = k^{37} \oplus \mathbf{y^6_{6,8}}$
$k'^{18} = k^{36}$
$k'^{17} = k^{35}$
$k'^{16} = k^{34}$
$k'^{15} = k^{33}$
$k'^{14} = k^{32}$
$k'^{13} = k^{31}$
$k'^{12} = k^{30} \oplus 1$
$k'^{11} = k^{29}$
$k'^{10} = k^{28}$
$k'^{9} = k^{27}$
$k'^{8} = k^{26}$
$k'^{7} = k^{25}$
$k'^{6} = k^{24}$
$k'^{5} = k^{23}$
$k'^{4} = k^{22}$
$k'^{3} = k^{21}$
$k'^{2} = k^{20} = 0$
$k'^{1} = k^{19} = 0$
$k'^{0} = k^{18} = 0$

Note that 43 bits of v' and 25 bits of k' are fixed by the above relations.

TABLE V: 2 alternative inputs and outputs to the S-box at $\alpha = 18$

Condition	Input				Output		
x_8	$\mathbf{a_3}$	$\mathbf{v^{59}}$	$\mathbf{v^{25}}$	$\mathbf{k^{73}}$	$\mathbf{y^{13}_8}$	$\mathbf{y^8_8}$	$\mathbf{y^6_8}$
b_1	1	1	1	0	0	0	0
b_2	0	0	0	1	1	1	1

3) slid pair at $\alpha = 19$: For the slid pair at $\alpha = 19$, there are 46 and 35 bits of key and IV respectively which must take fixed value as follow.

$$k^i = 0, i \in \{77, 76, 75, 74, 72, 71, 70, 69, 67, 66, 50, 49, 48, 46, 45, 44, 43, 42, 40, 39, 21, 20, 19, 17, 16, \ldots, 0\}$$
$$k^i = 1, i \in \{68, 41, 18, 17\}$$

$$(k^{73}, k^{47}) = \begin{cases} (0,0) & \text{for } x_8 = b_1 \text{ and } x_9 = e_1 \\ (0,1) & \text{for } x_8 = b_1 \text{ and } x_9 = e_2 \\ (0,1) & \text{for } x_8 = b_1 \text{ and } x_9 = e_3 \\ (1,0) & \text{for } x_8 = b_2 \text{ and } x_9 = e_1 \\ (1,1) & \text{for } x_8 = b_2 \text{ and } x_9 = e_2 \\ (1,1) & \text{for } x_8 = b_2 \text{ and } x_9 = e_3 \end{cases}$$

$$v^i = 0, i \in \{79, 78, 76, 75, 74, 73, 72, 71, 70, 69, 63, 62, 61, 58, 57, 56, 55, 53, 52, 29, 28, 27, 24, 23, 22, 21, 19, 18\}$$
$$v^i = 1, i \in \{54, 20\}$$

$$(v^{77}, v^{76}, v^{60}, v^{59}, v^{26}, v^{25}) = \begin{cases} (0,0,0,0,1,0) & \text{for } x_8 = b_1 \text{ and } x_9 = e_1 \\ (0,0,1,0,0,0) & \text{for } x_8 = b_1 \text{ and } x_9 = e_2 \\ (1,0,0,0,0,0) & \text{for } x_8 = b_1 \text{ and } x_9 = e_3 \\ (0,1,0,1,1,1) & \text{for } x_8 = b_2 \text{ and } x_9 = e_1 \\ (0,1,1,1,0,1) & \text{for } x_8 = b_2 \text{ and } x_9 = e_2 \\ (1,1,0,1,0,1) & \text{for } x_8 = b_2 \text{ and } x_9 = e_3 \end{cases}$$

As for the cases $\alpha = 17$ and 18, expressions can again be determined for (k', v') in terms of (k, v). These are not reported in this paper.

All the slid pairs are obtained using computer simulation for all possible inputs and outputs of the S-box. As shown previously, slid pairs occur according to specific conditions at time $t \geq 17$. Slid pairs occur with a limitation, where the pipeline can not be reset to all zeros.

C. Shifted keystream

We now consider the additional constraints which must be satisfied in order for slid pair (k, v) and (k', v') to generate shifted keystream sequences. These constraints depend on the degree of the similarity between the state update functions during the initialisation and keystream generation processes.

To obtain a shifted keystream from the slid pairs at $\alpha = 17$, all the output of the S-box for the last 17 iterations of the initialisation process (from $t = 111$ to $t = 128$) should be all zeros to get the linear feedback during these iterations of the initialisation process. Therefore, the input of the S-box from $t = 104$ to $t = 121$ should be zeros. To ensure all the relevent input bits are zeros, there are 187 bits which should be fixed and another 17 bits to be fixed for the linear feedback function. The total fixed bits are 204 bits. The probability of satisfying 204 independent constraints is 2^{-204}. In this case, the free bits at the loaded state are 90 bits. Hence, there are 2^{90} possible slid pairs and the probability of slid pair gives an out of phase keystream is 2^{-204}. Therefore, the expected number of shifted keystream sequences from these slid pairs $= 2^{-204} \times 2^{90} = 2^{-114}$. So, it seems extremely unlikely that any shifted keystream will result from any of the slid pairs.

Similarly, for the slid pairs at $\alpha = 18$ and 19, the last 18 and 19 outputs of the S-box should be zeros respectively.

The total fixed bits in these cases are 211 and 218 bits. The probability of satisfying 211 and 218 independent constraints are 2^{-211} and 2^{-218} respectively. In this case, the free bits at the loaded state are 84 and 79 bits. Therefore expected number of keystream out of phase shifted by 18 and 19 bits from any of the relevant slid pairs are $2^{(-211+84)} = 2^{-127}$ and $2^{(-218+79)} = 2^{-139}$.

VI. Attack Procedure

Since Sfinks has an 80-bit key and a 256-bit shift register, it is not feasible to simply guess the whole secret key and then generate the keystream to check whether the guess is correct. However, if it is possible to identify the occurrence of a slid pair (extremely unlikely as discussed in Section V), this enables us to use the resulting relationship to reduce the number of key bits that need to be guessed, forming the basis for an attack on the cipher. The attack presented in this paper uses the properties of the slide attacks. We assume a known-plaintext attack scenario. The following algorithm outlines the attacking procedure.

This algorithm is described for 17-bit shifted versions of keystream. However, it can be extended to other shifts as well. The algorithm depends on two keystreams with unknown secret keys and known IV's such that the resulting keystreams are shifted versions of one another as mentioned in Section II. Note that we are interested here to find the secret key.

Recalling the specification of the loading phase of Sfinks in Section III, it is impossible to find two shifted keystreams generated by a secret key with different IV's. Therefore, in this case, we are focusing on two keystreams generated by diferent key-IV pairs, (k, v) and (k', v').

Attack Algorithm:

Inputs: Pairs of plaintext and ciphertext.

Step1: From the known-plaintext-ciphertext attack, XOR the plaintext with the corresponding ciphertext to get the keystream sequence

Step2: Divide each keystream sequence into separate keystreams. Each keystream corresponds to a different key-IV combination based on the rekying process.

Step3: Check the similarity of $keystream_i$ and $keystream_j \ll 17$ for every available key-IV pair, using $keystream_i \oplus keystream_j \ll 17$.

- If so, note the key-IV pairs (k, v) and (k', v').
- If not, the algorithm fails for this 17-bit shift.

Step4: If shifted keystream has been identified, guess and check each possible key, as follows

- Using the relevant of $t = 17$ equations in Section V, where the IV's v and v' are known.
- Due to the slid pairs, there are 40 and 23 bits of the k and k' are known and fixed,
- Guess the 40 free key bits of k in the relevant set of equations in Section V.
- Use these free key bits and the known IV's to calculate the remaining key bits of k' using the related equations in Section V.
- Use the generated secret key k' with the corresponding v' to generate its keystream using the Sfinks algorithm.
- Compare between the generated keystream and the known keystream.
  - If they are identical, the secret keys k and k' have been found.
  - If not, repeat the process for another guess of k and calculate the k'.

Step5: Use the secret key k and k' with known IV's to decrypt the entire related ciphertexts.

Outputs: The secret keys k and k'.

In step 3, if two keystreams have been identified as shifted keystream by 17 bits, then the guessing process should cover the 40 free key bits instead of the 80 key bits. For each guess, it requires to substitute the guessed key bits into the relevant equations in Section V to find the secret key k'. At this time, it is not clear whether the guessed key is correct or not. The proposed key must be verified by generating a keystream that should be same as the known keystream.

If the generated keystream is similar to the known keystream then the secret key is correct, if not use another guess. Note that, the above algorithm is for 17-bit shifts. If it fails, then it is possible to try for 18-bit shifts or more.

Attacking Complexity: In step 3, the comparison depends on the available length of the plaintext and its corresponding ciphertext. If an attacker has enough plaintext-ciphertext pairs, it is possible to check the shifted keystreams. As mentioned in Section V, the probability of obtaining shifted keystreams from a given slid pair is 2^{-204}, and the total number of possible slid pairs is 2^{90}. Then the expected number of shifted keystreams from slid pairs is 2^{-114}. Thus this cipher appears to be secure for slid attacks, but the presence of slid pairs is still a concern.

VII. Analysis of Sfinks with slight modification

This section analyses a slightly modified version of Sfinks, where the modification is applied to the padding format of Sfinks during the loading phase. This analysis shows that the effect of a minor change in the padding increases the probability of occurrence of slid pairs and the probability of obtaining shifted keystreams from given slid pairs.

In this modification, we change the pattern of the padding to be zeros for all stages (s^{95} to s^{90}). In other words, we set the content of stage s^{95} to 0 instead of 1. The slid pairs may now occur after only one clock under two conditions. Firstly, the 16-bit input to the S-box must be zero at time $t = 1$. This requires us to fix the values of 8 stages (s_1^{255},s_1^{244}, s_1^{227}, s_1^{193}, s_1^{161}, s_1^{134}, s_1^{105}, s_1^{98}) to be zeros to ensure that the pipeline is still in the initial condition. The last seven of these stages can be expressed in terms of key and IV bits as (k^{66}, k^{39}, k^{10}, k^{3}, v^{69}, v^{52}, v^{18}). In addition, since $s^{255} = x^{16}$ must be zero, the linear feedback function imposes a condition that the XOR of the following bits must be zero: (k^{68}, k^{56}, k^{30}, k^{20}, k^{12}, v^{34}, v^{19}, v^{17}, v^{12}), so that $s_1^{255} = 0$. Secondly, the content of s_0^{96} must be similar to the padding, that is $s_0^{96} = s_0^{95} = 0$.

Thus, a slid pair with a zeroed pipeline can be obtained by fixing 9 bits out of the 160 key-IV bits. Therefore, the proportion of valid key-IV pairs which lead to these slid pairs is 2^{-9}. Hence there are $2^{160-9} = 2^{151}$ slid pairs for $\alpha = 1$.

A slid pair can lead to an out-of-phase keystream by fixing another 16 bits to ensure that the last iteration of the initialisation process at $t = 128$ is linear. Specifically, we require that all 16 input bits of the S-box at time $t = 121$ should be zeros. The probability of satisfying these 16 independent constraints is 2^{-16}. Therefore, the expected proportion of Key-IV pairs that lead to phase shifted keystream with 1 step is $2^{-9} \times 2^{-16} = 2^{-25}$ and the expected number of out of phase keystream sequences is $2^{151} \times 2^{-16} = 2^{135}$.

We see that a small change in the padding format during the loading phase of the initialisation has hugely increased the probability of obtaining both slid pairs and corresponding shifted keystream. As expressed in this section and Section V, there are 2^{151} and 2^{90} out of 2^{160} possible slid key-IV pairs for the modified version and current cipher respectively. The probabilities of a slid pair that gives a shifted keystream are 2^{-16} and 2^{-204} for these versions respectively.

VIII. Conclusion

The occurrence of slid pairs in a stream cipher depends on the similarity of state update functions and the similarity of each iteration in the initialisation process and keystream generation. For most stream ciphers, the iterations of the initialisation process are identical, and the iterations of the keystream generation process are identical too, although the state update function of the initialisation process may differ from the state update function of the keystream generation. Therefore, slid (k, v) pairs with shifted keystream sequences in stream ciphers are possible. For the Sfinks cipher, the state update functions of initialisation and keystream generation processes are quite different, where the initialisation update function is nonlinear and the keystream generation state update is a purely linear function.

For Sfinks, the padding pattern and the form of the state update function defer the first slid pair to 17 iterations. It is possible to find slid pairs by fixing 70 bits out of 160 key-IV bits. Although the proportion of valid key-IV pairs which lead to these slid pairs is small (2^{90}) out of (2^{160}), the existence of such pairs is undesirable property, which can be avoided by carefully designing the cipher.

It is obvious that the state update functions of the initialisation process and the keystream generator and the padding system affect the complexity of slid pairs. In the slightly modified version of Sfinks when the padding consists entirely of zeros, it is possible to get a phase shifted keystream based on a slid pair every 2^{25} initialisations. Note that for Sfinks without this modification, it is extremely unlikely that any of the slid pairs which occur would lead to phase shifted keystream.

It is a general belief among the designers of stream ciphers that if the number of iterations of a cipher is increased, the cipher may become very strong. We have demonstrated that the slid pairs can occur in Sfinks at time $t \geq 17$ for key-IV pairs, although these pairs are very unlikely to lead to phase shifted keystream. However, the fact that slid pairs can occur in the cipher is still a concern because it is equivalent to reducing the effective number of iterations during initialisation, and this may decrease the security of the initialisation process of the cipher.

The probability of success for slide attacks on stream ciphers like Sfinks can be reduced by increasing the size of the internal state, destroying the similarity of the state update functions of the initialisation process and keystream generation or designing the pattern of padding in a proper way to defer the slid pairs as long as possible. For example, if the content of the stage s^{40} in Sfinks is changed to a 1 in the padding format, this will defer the occurrence of slid pair to $\alpha \geq 24$ and the relevant conditions on the state contents will also be more complicated as mentioned previously. This will result in a lower probability of slid pairs occurring.

References

[1] A. Braeken, J. Lano, N. Mentens, B. Preneel, and I. Verbauwhede, "SFINKS: A synchronous stream cipher for restricted hardware environments," eSTREAM, ECRYPT Stream Cipher Project, Report 2005/026, 2005, www.ecrypt.eu.org/stream/ciphers/sfinks/sfinks.ps.

[2] eSTREAM: European Network of Excellence for Cryptology, "The ECRYPT Stream Cipher Project," http://www.ecrypt.eu.org/stream/.

[3] J. Hong and P. Sarkar, "Rediscovery of time memory tradeoffs," Cryptology ePrint Archive, Report 2005/090, 2005, http://eprint.iacr.org/.

[4] C. D. Cannière, Ö. Kücük, and B. Preneel, "Analysis of Grain's initialization algorithm," in *Progress in Cryptology AFRICACRYPT 2008*, ser. Lecture Notes in Computer Science, S. Vaudenay, Ed., vol. 5023. Springer Berlin / Heidelberg, 2008, pp. 276–289.

[5] H. Zhang and X. Wang, "Cryptanalysis of stream cipher grain family," Cryptology ePrint Archive, Report 2009/109, 2009, http://eprint.iacr.org/.

[6] Ö. Küçük, "Slide resynchronization attack on the initialization of grain 1.0," eSTREAM, ECRYPT Stream Cipher Project, Report 2006/044, 2006, http://www.ecrypt.eu.org/stream.

[7] D. Priemuth-Schmid and A. Biryukov, "Slid Pairs in Salsa20 and Trivium," in *Progress in Cryptology - INDOCRYPT 2008*, ser. Lecture Notes in Computer Science, D. Chowdhury, V. Rijmen, and A. Das, Eds., vol. 5365. Springer Berlin / Heidelberg, 2008, pp. 1–14.

[8] A. Biryukov and D. Wagner, "Slide attacks," in *Fast Software Encryption*, ser. Lecture Notes in Computer Science, L. Knudsen, Ed., vol. 1636. Springer Berlin / Heidelberg, 1999, pp. 245–259.

[9] A. Biryukov and D. Wagner, "Advanced slide attacks," in *Advances in Cryptology EUROCRYPT 2000*, ser. Lecture Notes in Computer Science, B. Preneel, Ed., vol. 1807. Springer Berlin / Heidelberg, 2000, pp. 589–606.

[10] H. Wu and B. Preneel, "Resynchronization attacks on WG and LEX," in *Fast Software Encryption*, ser. Lecture Notes in Computer Science, M. Robshaw, Ed., vol. 4047. Springer Berlin / Heidelberg, 2006, pp. 422–432.

[11] M. Hell, T. Johansson, and W. Meier, "Grain - A Stream Cipher for Constrained Environments," eSTREAM, ECRYPT Stream Cipher Project, Report 2005/010, 2005, http://www.ecrypt.eu.org/stream.

[12] M. Hell, T. Johansson, and W. Meier, "Grain: a stream cipher for constrained environments," *International Journal of Wireless and Mobile Computing*, vol. 2, no. 1, pp. 86–93, 2007.

[13] M. Hell, T. Johansson, and W. Meier, "A stream cipher proposal: Grain-128," in *2006 IEEE International Symposium on Information Theory*, July 2006, pp. 1614 –1618.

[14] C. D. Cannière and B. Preneel, "Trivium - A Stream Cipher Construction Inspired by Block Cipher Design Principles," in *Information Security*, ser. Lecture Notes in Computer Science, S. Katsikas, J. López, M. Backes, S. Gritzalis, and B. Preneel, Eds., vol. 4176. Springer Berlin / Heidelberg, 2006, pp. 171–186.