Differential Fault Analysis of Rabbit

Aleksandar Kircanski and Amr M. Youssef

Concordia Institute for Information Systems Engineering
Concordia University
Montreal, Quebec, H3G 1M8, Canada
{a_kircan,youssef}@ciise.concordia.ca

Abstract. Rabbit is a high speed scalable stream cipher with 128-bit key and a 64-bit initialization vector. It has passed all three stages of the ECRYPT stream cipher project and is a member of eSTREAM software portfolio. In this paper, we present a practical fault analysis attack on Rabbit. The fault model in which we analyze the cipher is the one in which the attacker is assumed to be able to fault a random bit of the internal state of the cipher but cannot control the exact location of injected faults. Our attack requires around $128 - 256$ faults, precomputed table of size $2^{41.6}$ bytes and recovers the complete internal state of Rabbit in about 2^{38} steps.

1 Introduction

The ECRYPT stream cipher project, also known as eSTREAM, is a project that aimed to identify new promising stream ciphers. The first call for stream cipher submissions was made in 2004 and it consisted of profile 1 and profile 2: software oriented ciphers and hardware oriented ciphers. The ciphers were put through a three-phase elimination process, finalizing in 2008, when four software oriented ciphers, including Rabbit, and three hardware oriented ciphers were selected as members of the eSTREAM portfolio.

After passing all three phases, Rabbit [4] (also see RFC 4503) has become a member of profile 1 eSTREAM portfolio. While originally designed with high software performance in mind, Rabbit turns out to be also very fast and compact in hardware. Fully optimized software implementations achieve an encryption speed of up to 3.7 clock cycles per byte (CPB) on a Pentium 3, and of 9.7 CPB on an ARM7. It uses a 128-bit secret key, 64-bit IV and generates 128 pseudo-random bits as keystream output at each iteration. The size of the secret internal state amounts to 513 bits, consisting of two sets of 8 32-bit words and one additional 1-bit value.

The security of Rabbit has been thoroughly investigated in a series of white papers published by the crypto lab at Cryptico A/S. These papers include analysis of the key setup function [9], analysis of IV-setup [13], mod n cryptanalysis [14], algebraic cryptanalysis [8] and periodic properties [11]. Also, a distinguishing attack requiring 2^{247} 128-bit samples was reported in [2]. The bias utilized in this attack was resulting from the bias in the Rabbit core function where it was shown that images of the Rabbit core function, g, have significantly less zeros than ones at each offset and this was used to show that there exists a bias in the least significant bit of certain keystream subblocks. This work was extended in [19], where the probability distribution of several keystream bits together was calculated by means of Fast Fourier Transform, using the techniques described in [22]. The complexity of the latter attack is 2^{158}. The authors also presented an attack in which the $2^{51.5}$ instantiations of the cipher are analyzed based on the first three keystream output blocks of each instantiation. The additional assumption is that certain differences expressed in terms of XOR among these $2^{51.5}$ internal states are known. This attack recovers all $2^{51.5}$ keys and requires 2^{32} precomputation steps, 2^{32} memory, and $2^{97.5}$ steps. According to the authors, the attack is given under an unusual cryptanalytic assumption. This attack was considered the first known key recovery attack on Rabbit.

In this paper we present a fault analysis attack on Rabbit. The fault model adopted is the one in which an attacker is assumed to be able to cause a bit-flip at a random location in the internal state of the cipher. However, the exact position of the flipped bit is unknown to the attacker. The attacker is also assumed to be able to reinitialize the cipher sufficient amount of times, iterate and obtain keystream words. The proposed attack requires around $128 - 256$ faults, an offline precomputed table of size $2^{41.6}$ bytes and recovers the complete internal state of Rabbit in about 2^{38} steps.

2 Fault Analysis

Cryptanalytic attacks can be broadly classified into two classes. In the first class of attacks, the attacker tries to exploit any weakness in the underlying mathematical structure of the cipher. This type includes, for example, differential cryptanalysis, linear cryptanalysis and algebraic attacks. The second class of attacks are implementation dependent attacks, which include side channel attacks, such as timing analysis [18] and power analysis [17], and fault analysis attacks. In fault analysis attacks [5], the attacker applies some kind of physical influence on the internal state of the cryptosystem, such as ionizing radiation which flips random bits in devices' memory. By examining the results of cryptographic operations under such faults, it is often possible to deduce information about the secret key or the secret internal state of the cipher.

Fault attacks were first introduced by Boneh *et al.* [5] in 1996 where they described attacks that targeted the RSA public key cryptosystem by exploiting a faulty Chinese remainder theorem computation to factor the modulus n. Subsequently, fault analysis attacks were extended to symmetric systems such as DES [3] and later to AES [15] and other primitives. Fault analysis attacks became a more realistic serious threat after cheap and low-tech methods of applying faults were presented (e.g., [1,23]).

Hoch and Shamir [16] showed that fault analysis attacks present a powerful tool against stream ciphers as well. Stream ciphers based on LFSRs, LILI-128 and SOBER-t32 as well as RC4 were shown to be insecure in a fault analysis model in which the attacker does not have the ability to choose the exact location of the induced fault. In the case of RC4, the key recovery attack requires 2^{16} faults and 2^{26} keystream words. In [6], RC4 was assessed using a different fault model in which the attacker may specify the location at which the fault is induced but can not specify the value of injected faults. The attack requires 2^{16} induced faults. Another more advanced fault analysis attack on RC4 which requires 2^{10} faults was also introduced in the same paper.

Hojsík and Rudolf [20] presented an attack on another eSTREAM cipher, Trivium [7]. The attack recovers the secret internal state using 42 fault injections. The fault model used is the one in which the attacker is able to flip a random bit in the internal state of Trivium without being able to exactly control its location. This work was subsequently improved in [21], providing an attack that recovers Trivium inner state with only 3.2 fault injections on average. The authors used different cipher representation and were able to reduce high-degree equations to linear ones, concluding that a change in the way by which the cipher is represented may result in a better attack.

In this paper, we use the same model as the one used in fault analysis of Trivium [20,21]. The attacker is assumed to be able to flip a random bit in the internal state of the cipher without being able to exactly control its location. In other words, the exact location of induced fault is assumed to be unknown to the attacker.

The rest of the paper is organized as follows. The Rabbit specifications that are relevant to our attack are briefly reviewed in the next section. The main idea behind our attack is presented in section 4.1. The procedure used to determine the location of induced faults is described in section 4.2 and the complete attack is described in section 4.3. Finally, the attack success probability and its associated complexity are analyzed in section 5.

3 Specification of Rabbit Stream Cipher

Internal state of Rabbit consists of 513 bits. It includes: eight 32-bit values: $x_{0,t}$, $\cdots x_{7,t}$, eight 32-bit counters, $c_{0,t}, \dots c_{7,t}$, and one additional bit $\phi_{7,t}$, used in the counter update. When the cipher steps from time t to time $t+1$, the counter is updated independently of x values, by adding known a_i values, corrected with carries ϕ as follows:

$$\begin{aligned} c_{0,t+1} &= c_{0,t} + a_0 + \phi_{7,t} \\ c_{j,t+1} &= c_{j,t} + a_j + \phi_{j-1,t+1}, 1 \le j \le 7 \end{aligned}$$

where

$$\phi_{j,t+1} = \begin{cases} 1 - 1_{\mathbb{Z}/2^{32}\mathbb{Z}}(c_{0,t} + a_0 + \phi_{7,t}) & \text{if } \; j = 0 \\ 1 - 1_{\mathbb{Z}/2^{32}\mathbb{Z}}(c_{j,t} + a_j + \phi_{j-1,t+1}) & \text{if } \; j > 0 \end{cases}$$

and $a_0 = a_3 = a_6 = 4D34D34D$, $a_1 = a_4 = a_7 = D34D34D3$, $a_2 = a_5 = 34D34D34D$. Function $\mathbf{1}_{\mathbb{Z}/2^{32}\mathbb{Z}}$ is defined by

$$\mathbf{1}_{\mathbb{Z}/2^{32}\mathbb{Z}}(x) = \begin{cases} 0 \text{ if } \; x \ge 2^{32} \\ 1 \; \text{ if } \; x < 2^{32} \end{cases}$$

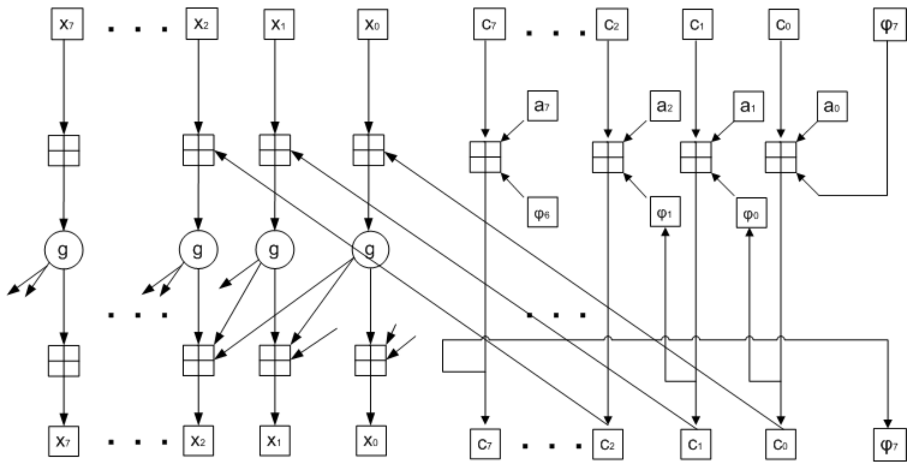

Fig. 1. Simplified view of the state update function of Rabbit, rotations omitted

The x values are updated by

$$\begin{aligned}
x_{0,t+1} &= g_{0,t} + (g_{7,t} \lll 16) + (g_{6,t} \lll 16)\\
x_{1,t+1} &= g_{1,t} + (g_{0,t} \lll 8) + g_{7,t}\\
x_{2,t+1} &= g_{2,t} + (g_{1,t} \lll 16) + (g_{0,t} \lll 16)\\
x_{3,t+1} &= g_{3,t} + (g_{2,t} \lll 8) + g_{1,t}\\
x_{4,t+1} &= g_{4,t} + (g_{3,t} \lll 16) + (g_{2,t} \lll 16)\\
x_{5,t+1} &= g_{5,t} + (g_{4,t} \lll 8) + g_{3,t}\\
x_{6,t+1} &= g_{6,t} + (g_{5,t} \lll 16) + (g_{4,t} \lll 16)\\
x_{7,t+1} &= g_{7,t} + (g_{6,t} \lll 8) + g_{5,t}
\end{aligned} \tag{1}$$

where

$$g_{j,t} = (x_{j,t} + c_{j,t+1})^2 \oplus [(x_{j,t} + c_{j,t+1})^2 \gg 32] \tag{2}$$

The 128-bit keystream output block $s_{t+1}^{[127..0]}$, is constructed as follows:

$$\begin{aligned}
s_{t+1}^{[15..0]} &= x_{0,t+1}^{[15..0]} \oplus x_{5,t+1}^{[31..16]}, & s_{t+1}^{[31..16]} &= x_{0,t+1}^{[31..16]} \oplus x_{3,t+1}^{[15..0]}\\
s_{t+1}^{[47..32]} &= x_{2,t+1}^{[15..0]} \oplus x_{7,t+1}^{[31..16]}, & s_{t+1}^{[63..48]} &= x_{2,t+1}^{[31..16]} \oplus x_{5,t+1}^{[15..0]}\\
s_{t+1}^{[79..64]} &= x_{4,t+1}^{[15..0]} \oplus x_{1,t+1}^{[31..16]}, & s_{t+1}^{[95..80]} &= x_{4,t+1}^{[31..16]} \oplus x_{7,t+1}^{[15..0]}\\
s_{t+1}^{[111..96]} &= x_{6,t+1}^{[15..0]} \oplus x_{3,t+1}^{[31..16]}, & s_{t+1}^{[127..112]} &= x_{6,t+1}^{[31..16]} \oplus x_{1,t+1}^{[15..0]}
\end{aligned} \tag{3}$$

Figure 1 shows a simplified view of the Rabbit state update function. The description of the key setup scheme of Rabbit is omitted since it does not play a role in the attack outlined in this paper.

4 Differential Fault Analysis Attack

Throughout the rest of this paper, faulty words will be denoted same as non-faulty ones, except that a "′" sign will be added. This way, faulty Rabbit internal state words at time t will be denoted by $x'_{i,t}$, $c'_{j,t}$, $\phi'_{7,t}$. The whole Rabbit internal state at time t, consisting of $[(x_{i,t})_{i=0\ldots7}, (c_{i,t})_{i=0\ldots7}, \phi_{7,t}]$, will be denoted by S_t. Accordingly, its faulty counterpart will be denoted by S'_t. We will also use "$+$" to denote addition mod 32, unless otherwise stated.

Faulty keystream output at step t will be denoted by s'_t. The i-th 16-bit segment of word s will be denoted by $s^{(i)}$. For example $s_t^{(1)}$ denotes $s^{[31..16]}$, i.e., bits 16 to 31 of word s_t.

According to our fault analysis model, the attacker has the power to flip a bit within the internal state of the cipher, that is $x_{i,t}, c_{i,t}$, $i = 0, \ldots 7$, $\phi_{7,t}$ but the attacker can not control or know the exact location of the induced fault (both at the bit and at the word level).

4.1 The Main Idea

Before stating the complete attack procedure, we provide a motivational example that illustrates the idea behind the attack. Let states of Rabbit at step t, S_t and S'_t, differ only in i-th bit of word $x_{0,t}$. Consequently, $x'_{0,t} + c'_{0,t+1} = x_{0,t} + c_{0,t+1} + \sigma 2^i$, for some unknown $\sigma \in \{-1, +1\}$ and $i \in \{0, \ldots 31\}$. Then, with high probability, $g'_{0,t} \neq g_{0,t}$ and $g'_{i,t} = g_{i,t}$ for $i = 1..7$. This implies that $x'_{i,t+1} \neq x_{i,t+1}$, for $i = 0, 1, 2$ and $x'_{i,t+1} = x_{i,t+1}$ for $i = 3..7$. In particular, since $x'^{[31..16]}_{5,t+1} = x^{[31..16]}_{5,t+1}$ and $x'^{[15..0]}_{3,t+1} = x^{[15..0]}_{3,t+1}$, then using the first line in Eq. (3), the following holds

$$\begin{aligned} s'^{[15..0]}_{t+1} &= x'^{[15..0]}_{0,t+1} \oplus x^{[31..16]}_{5,t+1} \quad & s^{[15..0]}_{t+1} &= x^{[15..0]}_{0,t+1} \oplus x^{[31..16]}_{5,t+1} \\ s'^{[31..16]}_{t+1} &= x'^{[31..16]}_{0,t+1} \oplus x^{[15..0]}_{3,t+1} \quad & s^{[31..16]}_{t+1} &= x^{[31..16]}_{0,t+1} \oplus x^{[15..0]}_{3,t+1} \end{aligned}$$

Thus guessing $x^{[31..16]}_{5,t+1}$ and $x^{[15..0]}_{3,t+1}$ makes a candidate for values $x_{0,t+1}$ and $x'_{0,t+1}$ and consequently, using Eq. (1), a candidate for

$$\begin{gathered} x_{0,t+1} - x'_{0,t+1} = \\ (g_{0,t} + g_{7,t} \lll 16 + g_{6,t} \lll 16) - (g'_{0,t} + g'_{7,t} \lll 16 + g'_{6,t} \lll 16) = g_{0,t} - g'_{0,t} \end{gathered}$$

Since inputs to $g_{0,t}$ and $g'_{0,t}$ differ by $\pm 2^i$ for some unknown $i = 0, \ldots 31$, this constraint can be described by a set of g function additive differentials $\{(\pm 2^i, \delta) | i = 0, \ldots 31\}$.

Suppose now the attacker obtains two more faulted keystream words s''_{t+1} and s'''_{t+1}, derived from states S''_t and S'''_t differing from S_t on bits j and k of word $x_{0,t}$, where $k \neq j, k \neq i$, $j \neq i$. Since in all three cases, values $x^{[31..16]}_{5,t+1}$ and $x^{[15..0]}_{3,t+1}$ do not change, using s_{t+1}, s'_{t+1}, s''_{t+1} and s'''_{t+1} three sets of differentials $\{(\pm 2^i, \delta_1) | i = 0, \ldots 31\}$, $\{(\pm 2^i, \delta_2) | i = 0, \ldots 31\}$ and $\{(\pm 2^i, \delta_3) | i = 0, \ldots 31\}$ are obtained using the same guess in the way described above. As will be shown later, the probability that there exists an input x for the g function such that it satisfies all three sets of differentials at once, i.e., such that there exist mutually different i_1, i_2 and i_3 such that

$$\begin{aligned} g(x) - g(x \pm 2^{i_1}) &= \delta_1, \\ g(x) - g(x \pm 2^{i_2}) &= \delta_2, \\ g(x) - g(x \pm 2^{i_3}) &= \delta_3 \end{aligned}$$

is small if the guess above is not correct. Thus, the attacker is able to discard wrong guesses for $x_{5,t+1}^{[31..16]}$ and $x_{3,t+1}^{[15..0]}$. Also, if the guess is a correct one, the attacker obtains candidates for g input value $x_{0,t} + c_{0,t+1}$. In the following we provide a full internal state recovery algorithm.

4.2 Determining the Position of the Fault

In the attack proposed in this paper, the first step after inducing a fault is to make restrictions on the position where the fault took place. The induced bit flipping can happen at one of the bits of words $x_{0,t}, \ldots x_{7,t}$, $c_{0,t}, \ldots c_{7,t}$ as well as at the 1-bit value $\phi_{7,t}$.

In the following we provide a tool for deducing important information on the location at which the fault occurred. Based on difference among faulty and non-faulty keystreams, information on the difference among internal states S_t and S'_t is deduced. More precisely, only keystream words s_t and s'_t will be used and according to the fault model, it will be assumed that internal states S_t and S'_t differ exactly on one bit.

To express these differences in a convenient way, we introduce the function d_{ST}, describing differences on the internal states and the function d_{KS}, describing differences among faulty and non-faulty keystream words. Let

$$d_{ST}(S, S') = \begin{cases} 0, & \text{if a fault occured either at } x_{0,t},\ c_{0,t} \text{ or } \phi_{7,t} \\ 1 \le i \le 7, & \text{if a fault accured either at } x_{i,t} \text{ or } c_{i,t} \end{cases}$$

The function d_{ST} is defined for every pair of states (S, S') that differ exactly on one bit. If s and s' are two 128-bit keystream words at some step, then we define

$$d_{KS}(s, s') = \begin{cases} 0, & s^{(5)} = s'^{(5)}, s^{(6)} = s'^{(6)} \text{ and } s^{(i)} \ne s'^{(i)} \text{ for } i \ne 5, 6 \\ 1, & s^{(0)} = s'^{(0)}, s^{(5)} = s'^{(5)} \text{ and } s^{(i)} \ne s'^{(i)} \text{ for } i \ne 0, 5 \\ 2, & s^{(0)} = s'^{(0)}, s^{(7)} = s'^{(7)} \text{ and } s^{(i)} \ne s'^{(i)} \text{ for } i \ne 0, 7 \\ 3, & s^{(2)} = s'^{(2)}, s^{(7)} = s'^{(7)} \text{ and } s^{(i)} \ne s'^{(i)} \text{ for } i \ne 2, 7 \\ 4, & s^{(1)} = s'^{(1)}, s^{(2)} = s'^{(2)} \text{ and } s^{(i)} \ne s'^{(i)} \text{ for } i \ne 1, 2 \\ 5, & s^{(1)} = s'^{(1)}, s^{(4)} = s'^{(4)} \text{ and } s^{(i)} \ne s'^{(i)} \text{ for } i \ne 1, 4 \\ 6, & s^{(3)} = s'^{(3)}, s^{(4)} = s'^{(4)} \text{ and } s^{(i)} \ne s'^{(i)} \text{ for } i \ne 3, 4 \\ 7, & s^{(3)} = s'^{(3)}, s^{(6)} = s'^{(6)} \text{ and } s^{(i)} \ne s'^{(i)} \text{ for } i \ne 3, 6 \end{cases}$$

If a pair of 128 bit (s, s') words does not satisfy any of the conditions proposed by the right-hand side of the equation above, function $d_{KS}(s, s')$ is *undefined.*

To understand the motivation behind the above definition, assume that the injected fault affected the input to the function $g_{0,t}$. From Figure 1, it is clear that such fault *directly* affects the computation of $x_{0,t+1}$, $x_{1,t+1}$ and $x_{2,t+1}$. From Eq. (3), it follows that these three terms also *directly* affect the computation of all words on the output stream except ${s'}_{t+1}^{[95..80]} = {s'}^{(5)}$ and ${s'}_{t+1}^{[111..96]} = {s'}^{(6)}$ which explains the first line in the above definition. A similar argument applies to to rest of entries in the definition of $d_{KS}(s, s')$.

The criterion for determining the position of the fault $d_{ST}(S_t, S'_t)$ based on the first keystream word can now be simply stated as follows:

- If $d_{KS}(s_{t+1}, s'_{t+1})$ is defined, put $d_{ST}(S_t, S'_t) = d_{KS}(s_{t+1}, s'_{t+1})$
- Otherwise, leave $d_{ST}(S_t, S'_t)$ undefined

During the attack, when after a fault $d_{ST}(S_t, S'_t)$ value is undefined, the fault will be discarded and the attacker proceeds by inducing another fault.

The successfulness of this criterion can be measured by two types of errors, p_{incorr} and p_{undef}. Error p_{incorr} is defined as the probability that the criterion returns a wrong $d_{ST}(S, S')$ value, while error p_{undef} is defined as the probability that the criterion will leave $d_{ST}(S, S')$ undefined. The probability that the criterion returns correct $d_{ST}(S, S')$ value will be denoted as p_{corr}.

According to the theoretical estimate of these three probabilities (see Appendix A), $p_{corr} \approx 0.98635$, $p_{undef} \approx 0.013645$ and $p_{incorr} \approx 0$. To confirm these theoretical estimates, the following experiment was conducted. A Rabbit internal state was randomly initialized and a random fault was induced. The criterion was applied and it was noted which of the three options happened: correct position of the fault returned, incorrect position of the fault returned or position of the fault left undefined. After repeating the experiment for 10^6 times, the probabilities were obtained as $p_{corr} = 0.98408$, $p_{undef} = 0.015924$, and $p_{incorr} = 0$. Hence, given, say 100 faults, correct position will be determined for around 98 faults and 2 faults will be discarded. For no faults the incorrect position will be returned. Thus, it can be concluded that the proposed criterion represents reliable means for determining the position of faults.

4.3 The Complete Attack

Before stating the complete attack we introduce following definitions. Throughout the following three definitions, let $k \in \{0, 1, 2\}$ and $\sigma \in \{-1, +1\}$ be fixed values and let x be restricted to the set $\mathbb{Z}_{2^{32}}$. By $(\sigma 2^i, \delta)_k$ we denote a g function additive differential where the input difference is $\sigma 2^i$ and the output difference δ, taken after rotating g function output for $8 \times k$ bits.

Definition 1. *An x value will be considered to* satisfy differential $(\sigma 2^i, \delta)_k$ *if*

$$[g(x) \lll (8k)] - [g(x + \sigma 2^i) \lll (8k)] = \delta$$

Definition 2. *A set of differentials*

$$(\pm 2^i, \delta)_k|_{i=0}^{31} := \{(\sigma 2^i, \delta)_k | i = 0..31, \sigma = -1, +1\}$$

will be called a generalized differential. *An x value will be considered to* satisfy *generalized differential $(\pm 2^i, \delta)_k|_{i=0}^{31}$ if it satisfies any of the differentials contained in the set.*

Definition 3. *A set of generalized differentials*

$$\Delta = \{(\pm 2^i, \delta_1)_k|_{i=0}^{31}, (\pm 2^i, \delta_2)_k|_{i=0}^{31}, \ldots (\pm 2^i, \delta_n)_k|_{i=0}^{31})\}$$

will be considered satisfiable *if at least one x value satisfies them all, i.e., if there exists an x value as well as distinct values $d_1, \ldots d_n$, chosen from the set $\{\pm 2^i | i = 0..31\}$, such that*

$$[g(x) \lll 8k] - [g(x + d_j) \lll 8k] = \delta_j, j = 1, \ldots n$$

For such an x, we shall say that it satisfies *set Δ.*

The following procedure, `flt_init(t)` induces a sufficient number of faults at the internal state of the cipher at step t and arranges faulty keystream words to appropriate sets, using the mechanism described in Section 4.2.

```
- Let $FLTS_i = \emptyset$, $i = \ldots 7$.
- While $|FLTS_i| < 3$ for any $i = 0 \ldots 7$
    - Reinitialize the cipher, forward to step $t$ and induce a fault. Obtain
      $s'_{t+1}$. If $d_{KS}(s'_{t+1}, s_{t+1})$ is defined, let $i = d_{KS}(s'_{t+1}, s_{t+1})$ and add $s'_{t+1}$
      to $FLTS_i$.
```

The procedure that follows, `derive_inf(i,k)`, utilizes information in $FLTS_i$ to deduce the set of possible values for $x_{i,t} + c_{i,t+1}$. Parameter k can take values 0,1 and 2 and it determines the way $x_{i,t} + c_{i,t+1}$ value will be recovered. Namely, as will be seen from the algorithm, there are three different ways to derive candidates for this value and the logic of these three ways is encoded through values of $\alpha_{i,k}, \beta_{i,k}$, $k = 0, 1, 2$ in Table 1. The values for $\alpha_{i,k}, \beta_{i,k}$ have been derived utilizing Eq. (3). For example, running `derive_inf(0,0)`, returns the set of candidates for $x_{0,t} + c_{0,t+1}$ by working on values $s_{t+1}^{(0)}$ and $s_{t+1}^{(1)}$, i.e., guessing values $x_{3,t+1}^{[15..0]}$ and $x_{5,t+1}^{[31..16]}$, creating the set of generalized differentials using $g_{0,t} - g'_{0,t} = x_{0,t+1} - x'_{0,t+1}$ and finally finding g-input values that satisfy it. On the other hand running `derive_inf(0,1)` aims to recover the same value $x_{0,t} + c_{0,t+1}$, but in a different way. Namely, in this case, the procedure operates on values $s_{t+1}^{(4)}$ and $s_{t+1}^{(7)}$, i.e., guesses $x_{4,t+1}^{[15..0]}$ and $x_{6,t+1}^{[31..16]}$, derives the generalized differential set by $g_{0,t} \lll 8 - g'_{0,t} \lll 8 = x_{1,t+1} - x'_{1,t+1}$ and then searches for g-input values that satisfy the set. The objective of obtaining the same value in three different ways is to take the intersection afterwards and hence minimize redundant candidates. Also, in the first case, a difference with no rotation was obtained and in the second, a difference after 8-bit rotations was found. The table is encoded so that whenever $k = 0$, $k = 1$ and $k = 2$, the number of rotations in the obtained difference will be 0, 8 and 16, respectively. This justifies the same value k present as index both for α, β and for generalized differentials themselves from $\Delta_i^k(A)$ sets in the procedure below.

The complete procedure `derive_inf(i,k)` follows:

```
- Let $Sat(\Delta_i^k) = \emptyset$
- For $A = 0, \ldots 2^{32} - 1$
```

Table 1. α and β index values used during the attack

i	0	1	2	3	4	5	6	7
$\alpha_{i,0}$	1	4	3	6	5	0	7	2
$\beta_{i,0}$	0	7	2	1	4	3	6	5
$\alpha_{i,1}$	4	3	6	5	0	7	2	1
$\beta_{i,1}$	7	2	1	4	3	6	5	0
$\alpha_{i,2}$	3	6	5	0	7	2	1	4
$\beta_{i,2}$	2	1	4	3	6	5	0	7

- Form the set of generalized differentials as follows:

$$\Delta_i^k(A) = \{ \ (\pm 2^l, ([s^{(\alpha_{i,k})}||s^{(\beta_{i,k})}] \oplus A) - ([s'^{(\alpha_{i,k})}||s'^{(\beta_{i,k})}] \oplus A))_k|_{l=0}^{31} \\ |s' \in FLTS_i\}$$

- Let $Sat(\Delta_i^k) = Sat(\Delta_i^k) \cup Sat(\Delta_i^k(A))$, where $Sat(\Delta_i^k(A))$ is the set of x values that satisfy $\Delta_i^k(A)$

where $\alpha_{i,k}$, $\beta_{i,k}$, $i = 0 \ldots 7$, $k = 0, 1, 2$ are defined by Table 1. The Derivation of $Sat(\Delta_i^k(A))$ sets is done using precomputation, as explained in Section 5.2. To recover g input values at step t, i.e., values $x_{i,t} + c_{i,t+1}$, the procedure `g_inp(t)` can be invoked, as follows:

- `flt_init(t)`
- For $i = 0, \ldots 7$
  - Call `derive_inf(i,0)`, `derive_inf(i,1)` and `derive_inf(i,2)` to find $Sat(\Delta_i^0)$, $Sat(\Delta_i^1)$ and $Sat(\Delta_i^2)$
  - $Cand(x_{i,t} + c_{i,t+1}) = Sat(\Delta_i^0) \cap Sat(\Delta_i^1) \cap Sat(\Delta_i^2)$

In the next section it will be shown that the probability that there will be more than one candidate for $x_{i,t} + c_{i,t+1}$, i.e., that there will be more than one element in the set `Cand`$(x_{i,t} + c_{i,t+1})$, is small.

Finally, the complete internal state at time $t = 1$ can be recovered by invoking the previous procedure for $t = 0$ which yields values $x_{i,0} + c_{i,1}$, $i = 0 \ldots 7$. This in turn yields $g_{i,0}$, $i = 0 \ldots 7$ values, which yield $x_{i,1}$, $i = 0 \ldots 7$, by Eq. 1. Invoking the previous procedure once again for $t = 1$ yields values $x_{i,1} + c_{i,2}$, $i = 0 \ldots 7$. Subtracting according values reveals $c_{i,2}$, $i = 0 \ldots 7$. Now $c_{i,1}$ values can be recovered by reversing the counter one step backward, according to the specification of counter update step. Whether $\phi_{7,1} = 0$ or $\phi_{7,1} = 1$ is found by mere trying both options and comparing the resulting keystream words.

5 Attack Success Probability and Complexity

5.1 Success Probability

In this section we show that the procedure from previous section determines the internal state uniquely. More precisely, it will be shown that $|Cand(x_{i,t} +$

$c_{i,t+1})| = 1$, for any $i = 0 \dots 7$ and $t \geq 0$ with high probability. This will be done by modelling g as a random function and then showing that if differences $([s^{(\alpha_{i,k})}||s^{(\beta_{i,k})}] \oplus A)$ - $([s'^{(\alpha_{i,k})}||s'^{(\beta_{i,k})}] \oplus A)$ are chosen uniformly randomly, i.e., not corresponding to the actual values produced by the attack procedure, this set of candidates will have 0 elements with high probability. Then, this probability can be taken as probability that $|Cand(x_{i,t} + c_{i,t+1})| = 1$ since following the procedure with actual differences and using the real g function guarantees existence of one correct candidate for $x_{i,t} + c_{i,t+1}$. According to the way complete internal state is recovered from g input values at times $t = 0$ and $t = 1$, as described by the last paragraph of the previous section, it is clear that from uniqueness and correctness of g input values, uniqueness and correctness of the recovered internal state at step $t = 1$ follows.

Since during the algorithm, $Cand(x_{i,t} + c_{i,t+1})$ is derived as the intersection of $Sat(\Delta_i^0)$, $Sat(\Delta_i^1)$ and $Sat(\Delta_i^2)$, the probability distribution of the number of elements in these three sets is first examined. Assume g is a randomly chosen function and differences $(s^{(\alpha_{i,k})}|s^{(\beta_{i,k})}) \oplus A - (s'^{(\alpha_{i,k})}|s'^{(\beta_{i,k})}) \oplus A$ are chosen randomly uniformly. Sets $Sat(\Delta_i^k)$, $k = 0, 1, 2$ are formed as follows, as described by `derive_inf(i,k)`:

$$Sat(\Delta_i^k) = Sat(\Delta_i^k(0)) \cup \dots \cup Sat(\Delta_i^k(2^{32} - 1))$$

For a given A, consider a generalized differential from $\Delta_i^k(A)$. The probability that random 32-bit x value will satisfy it is $63/2^{32}$. Since each set of generalized differentials $\Delta_i^k(A)$ contains at least three generalized differentials

$$P[\text{ x satisfies } \Delta_i^k(A)\ ] \leq (63/2^{32})^3 = 2^{-78.068}$$

The probability that among 2^{32} possible x values there exists at least one that will satisfy $\Delta_i^k(A)$, i.e., the probability that $\Delta_i^k(A)$ is satisfiable, is

$$P[\text{ Some } x \in \{0, \dots, 2^{32} - 1\} \text{ satisfies } \Delta_i^k(A)\ ] \leq 1 - (1 - 2^{-78.068})^{2^{32}} \approx 2^{-46}$$

Finally, the probability that for at least one A there will exist an x that will satisfy $\Delta_i^k(A)$, i.e., the probability that $Sat(\Delta_i^k)$ is nonempty in a random model, is

$$P[\ Sat(\Delta_i^k) \text{ is empty }] \geq (1 - 2^{-46})^{2^{32}} = 1 - 2^{-14} \qquad (4)$$

The final set of candidates for $x_{i,t} + c_{i,t+1}$ in procedure *g_inp* is derived as an intersection of $Sat(\Delta_i^0)$, $Sat(\Delta_i^1)$ and $Sat(\Delta_i^2)$. The probability that, in a random model, the intersection of these three sets is non-empty is

$$P[\text{ Randomly modelled } Cand(x_{i,t} + c_{i,t+1}) \text{ nonempty }] \leq (2^{-14})^3 = 2^{-42} \qquad (5)$$

This can finally be taken as an upper bound for the probability that there will be an element other than the correct one in $Cand(x_{i,t} + c_{i+1,t})$. Since $Cand(x_{i,t} + c_{i,t+1})$ is calculated for $i = 0, \dots 7$ at times $t = 0, 1$ during the attack procedure, it can be concluded that there will be no redundant candidates for the internal state after the procedure is completed.

5.2 Attack Complexity

The attack complexity can be measured by the number of faults required, computational complexity as well as storage complexity. First, we examine the number of faults necessary to undertake an attack.

As described above, the input for the attack is a non-faulty keystream word s_{t+1} as well as certain number of faulty keystream words s'_{t+1}. Also, the set of faulty states from which s'_{t+1} values are produced needs to satisfy certain properties. More precisely, as specified by the `flt_init` procedure, at each of the following groups of bits

$$\begin{gathered} x_{0,t}, c_{0,t}, \phi_{7,t} \\ x_{1,t}, c_{1,t} \\ x_{2,t}, c_{2,t} \\ \vdots \\ x_{7,t}, c_{7,t} \end{gathered} \tag{6}$$

the attacker has to produce at least three different faults and obtain three corresponding s'_{t+1} values. It follows that the minimal number of required faults that will need to be induced is $3 \times 8 = 24$. However, since an attacker does not have the possibility to choose locations of faults he induces, the number of necessary faults will be higher.

Let n denote the overall number of induced faults. Let $p(n)$ denote the probability that that there will be at least 3 faults at each one of the 8 groups of bits above. Let A_i be the event that after inducing n random faults there will be at most 2 faults at $x_{i,t}, c_{i,t}$, or $x_{i,t}, c_{i,t}, \phi_{7,t}$ if $i = 0$. Then, $A_i = B_i^0 \cup B_i^1 \cup B_i^2$ where B_i^j, $j = 0, 1, 2$, $i = 0 \ldots 7$ is an event that at $x_{i,t}, c_{i,t}$ or $x_{i,t}, c_{i,t}, \phi_{7,t}$ if $i = 0$ there will be 0,1 or 2 different faults. Then, $p(n)$ can be approximated as follows:

$$\begin{aligned} & p(n) = \\ & 1 - P[A_0 \cup \ldots \cup A_7] = \\ & 1 - P[(B_0^0 \cup B_0^1 \cup B_0^2) \cup \ldots \cup (B_7^0 \cup B_7^1 \cup B_7^2)] \approx \\ & 1 - (\sum_{i=0}^{7} \sum_{j=0}^{2} P[B_i^j]) - (\sum_{i_1=0}^{7} \sum_{j_1=0}^{2} \sum_{i_2=i_1+1}^{7} \sum_{j_2=0}^{2} P[B_{i_1}^{j_1} \cap B_{i_2}^{j_2}]) \end{aligned}$$

where the fact that $P[B_i^{j_1} \cap B_i^{j_2}] = 0$ for $j_1 \neq j_2$ has been used. For $i = 0 \ldots 7$

$$\begin{aligned} & P[B_i^0] = (\frac{7}{8})^n, i = 0 \ldots 7 \\ & P[B_i^1] = \sum_{k_1+k_2=n-1} (\frac{7}{8})^{k_1} (\frac{1}{8}) (\frac{7 \times 32 + 1}{8 \times 32})^{k_2} \\ & P[B_i^2] = \sum_{k_1+k_2+k_3=n-2} (\frac{7}{8})^{k_1} (\frac{1}{8}) (\frac{7 \times 32 + 1}{8 \times 32})^{k_2} (\frac{31}{8 \times 32}) (\frac{7 \times 32 + 2}{8 \times 32})^{k_3} \end{aligned}$$

and the second order probabilities are provided in Appendix B.

Substituting the according values of n yields $p(64) = 0.900$, $p(96) = 0.997$ and $p(128) = 0.999$. The quality of the approximation above has been verified by the following experiment. For 10^5 times, a data structure equivalent to Rabbit internal state was initialized with zeros and n faults were simulated by writing 1 to a uniformly random chosen bit location. After each iteration, if there was at least three 1-bits at each of the groups of bits in question, a counter was incremented. At the end of the experiment, the probability was obtained by dividing the counter by 10^5. Obtained ratios for $n = 64, 96, 128$ were 0.900, 0.996, 0.999 respectively. Consequently, throughout the rest of the paper, we assume that 64-128 faults are practically sufficient to guarantee that there will be at least 3 faults at each one of the 8 groups of bits defined in Eq. (6).

Since during the attack, as described in Section 4.3, procedure `flt_init` is called two times, the number of necessary faults is around $128 - 256$.

As for computational and storage complexity, the `flt_init` procedure can make use of precomputation. In particular, 32 tables $T_0^+, \ldots T_{31}^+$ can be created, such that cell $T_i^+[j]$ contains all the x values such that $j = g(x) - g(x + 2^i)$. Another 32 tables $T_0^-, \ldots T_{31}^-$ can be created, such that cell $T_i^-[j]$ contains all the x values such that $j = g(x) - g(x - 2^i)$. Analogous sets of tables can be created for $[g(x) \lll 8] - g(x \pm 2^i) \lll 8]$ and $[g(x) \lll 16] - g(x \pm 2^i) <$ $< 16]$. Thus, the storage complexity is given by $3 \times 64 \times 2^{32} = 2^{39.6}$ words, i.e., $2^{41.6}$ bytes, and now the computational complexity for a query for x such that it satisfies a generalized differential is $O(1)$. Since around $2 \times 8 \times 3 \times 2^{32}$ such queries are made, the computational complexity of the attack is about 2^{38} steps.

To summarize, the proposed attack requires around $128 - 256$ faults, precomputed table of size $2^{41.6}$ bytes, and recovers the cipher internal state in about 2^{38} steps. Further refitment for the attack complexity and success probability is provided in Appendix C.

It should be noted that our proposed attack does not work if the induced faults have a Hamming weight > 1. Extending the ideas presented in this paper to the case where this assumption is relaxed seems to be a challenging research problem.

References

1. Anderson, R., Kuhn, M.: Low Cost Attacks on Tamper Resistant Devices. In: Christianson, B., Crispo, B., Lomas, M., Roe, M. (eds.) Security Protocols 1997. LNCS, vol. 1361, pp. 125–136. Springer, Heidelberg (1998)
2. Aumasson, J.P.: On a bias of Rabbit. In: Proc. of the State of the Art of Stream Ciphers, SASC (2007)
3. Biham, E., Shamir, A.: Differential Fault Analysis of Secret Key Cryptosystems. In: Kaliski Jr., B.S. (ed.) CRYPTO 1997. LNCS, vol. 1294, pp. 513–525. Springer, Heidelberg (1997)

4. Boesgaard, M., Vesterager, M., Pedersen, T., Christiansen, J., Scavenius, O.: Rabbit: A new high-performance Stream Cipher. In: Johansson, T. (ed.) FSE 2003. LNCS, vol. 2887, pp. 307–329. Springer, Heidelberg (2003)
5. Boneh, D., Demillo, R.A., Lipton, R.J.: On the importance of checking cryptographic protocols for faults. In: Fumy, W. (ed.) EUROCRYPT 1997. LNCS, vol. 1233, pp. 37–51. Springer, Heidelberg (1997)
6. Biham, E., Granboulan, L., Nguyen, P.Q.: Impossible Fault Analysis of RC4 and Differential Fault Analysis of RC4. In: Gilbert, H., Handschuh, H. (eds.) FSE 2005. LNCS, vol. 3557, pp. 359–367. Springer, Heidelberg (2005)
7. Cannière, C., Preneel, B.: TRIVIUM: A stream cipher construction inspired by block cipher design principles. In: Katsikas, S.K., López, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) ISC 2006. LNCS, vol. 4176, pp. 171–186. Springer, Heidelberg (2006)
8. Cryptico A/S, Algebaric Analysis of Rabbit (2003), `http://www.cryptico.com`
9. Cryptico A/S, Analysis of the key setup function in Rabbit (2003), `http://www.cryptico.com`
10. Cryptico A/S, Hamming weights of the g-function (2003), `http://www.cryptico.com`
11. Cryptico A/S, Periodic properties of Rabbit (2003), `http://www.cryptico.com`
12. Cryptico A/S, Second degree approximations of the g-function (2003), `http://www.cryptico.com`
13. Cryptico A/S, Security Analysis of the IV-setup for Rabbit (2003), `http://www.cryptico.com`
14. Cryptico A/S, Mod n analysis of Rabbit (2003), `http://www.cryptico.com`
15. Dusart, P., Letourneux, G., Vivolo, O.: Differential fault analysis on AES. In: Zhou, J., Yung, M., Han, Y. (eds.) ACNS 2003. LNCS, vol. 2846, pp. 293–306. Springer, Heidelberg (2003)
16. Hoch, J., Shamir, A.: Fault Analysis of Stream Ciphers. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 240–253. Springer, Heidelberg (2004)
17. Kocher, P., Jaffe, J., Jun, B.: Differential Power Analysis. In: Wiener, M. (ed.) CRYPTO 1999. LNCS, vol. 1666, pp. 388–397. Springer, Heidelberg (1999)
18. Kocher, P.: Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems. In: Koblitz, N. (ed.) CRYPTO 1996. LNCS, vol. 1109, pp. 104–113. Springer, Heidelberg (1996)
19. Lu, Y., Wang, H., Ling, S.: Cryptanalysis of Rabbit. In: Wu, T.-C., Lei, C.-L., Rijmen, V., Lee, D.-T. (eds.) ISC 2008. LNCS, vol. 5222, pp. 204–214. Springer, Heidelberg (2008)
20. Hojsík, M., Rudolf, B.: Differential fault analysis of Trivium. In: Nyberg, K. (ed.) FSE 2008. LNCS, vol. 5086, pp. 158–172. Springer, Heidelberg (2008)
21. Hojsík, M., Rudolf, B.: Floating fault analysis of Trivium. In: Chowdhury, D.R., Rijmen, V., Das, A. (eds.) INDOCRYPT 2008. LNCS, vol. 5365, pp. 239–250. Springer, Heidelberg (2008)
22. Maximov, A., Johansson, T.: Fast computation of large distributions and its cryptographic properties. In: Roy, B. (ed.) ASIACRYPT 2005. LNCS, vol. 3788, pp. 313–332. Springer, Heidelberg (2005)
23. Skorobogatov, S.P., Anderson, R.J.: Optical fault induction attacks. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 2–12. Springer, Heidelberg (2003)

24. Zenner, E.: A Cache Timing Analysis of HC-256. In: Avanzi, R., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381. Springer, Heidelberg (2009)

A Estimating p_{corr}, p_{incorr} and p_{undef}

In this section, we provide an analytical estimate for the probabilities that the criterion defined in Section 4.2 will return a correct, incorrect or undefined value.

Firstly, we estimate the probability that a fault in one of the $c_{i,t}$, $i = 0..6$ values propagates to $c_{i+1,t+1}$ and not only to $c_{i,t+1}$ during the update step, via carry transfer mechanism implemented by auxiliary $\phi_{i,t+1}$ value. Suppose the fault occurred at position $c_{i,t}$, the probability that $c_{i,t} + a_i + \phi_{i-1,t+1}$ will have a carry at 32-nd bit place is approximately equal to

$$p_{cr} \approx \frac{1}{32} \sum_{i=1}^{32} \frac{1}{2^i} = 0.03125.$$

This probability is given by the event that that addition of $\pm 2^i$, $i = 0..31$ to a random 32-bit number x changes value of $1_{\mathbb{Z}/2^{32}\mathbb{Z}}(x)$. While the a_i values in the actual cipher are fixed ($a_i \in \{4D34D34D,\ D34D34D3, 34D34D34D\}$), our experimental results confirmed the accuracy of the above approximation.

Then, values p_{corr}, p_{incorr}, p_{undef} are estimated in what follows. Suppose the a random fault was induced in the internal state of the cipher. Then, the position of the bit-flip can be at

- $x_{i,t}, i \in \{0, \dots 7\}$ with probability $\frac{256}{513}$. In this case, our criterion will return a correct $d_{ST}(S, S')$ value if $g(x_{i,t} + c_{i,t+1}) \neq g(x'_{i,t} + c_{i,t+1})$, i.e., with probability $\frac{2^{32}-1}{2^{32}}$. In case that is not true, the criterion leaves $d_{ST}(S, S')$ undefined.
- $\phi_{7,t}$ with probability $\frac{1}{513}$. In this case, $c'_{0,t+1} \neq c_{0,t+1}$ with probability 1. Let $z \in \{0, \dots 7\}$ such that $c'_{j,t+1} \neq c_{j,t+1}$ for $j = 0, ..z$ and $c'_{j,t+1} = c_{j,t+1}$ for $j = z + 1, ..7$. If
  - $z = 0$, which happens with probability $\frac{2^{32}-1}{2^{32}}$, and $g(x_{0,t} + c_{0,t+1}) \neq g(x'_{0,t} + c_{0,t+1})$, for which the probability is $\frac{2^{32}-1}{2^{32}}$, then our criterion returns a correct value. If, however, in this case $g(x_{0,t} + c_{0,t+1}) = g(x'_{0,t} + c_{0,t+1})$ which happens with probability $\frac{1}{2^{32}}$, the criterion leaves $d_{ST}(S, S')$ undefined.
  - $z = 1$ which happens with probability $\frac{1}{2^{32}}$. In this case, we consider only the case $g(x_{i,t} + c_{i,t+1}) \neq g(x'_{i,t} + c_{i,t+1})$, $i = 0, 1$, probability being $(\frac{2^{32}-1}{2^{32}})^3$ and in this case again our criterion leaves $d_{ST}(S, S')$ undefined. Other possibilities within the case $z = 1$ are highly improbable and hence do not have any practical implications on the success probability of our attack.
  - $z \geq 2$ occurs with probability $(\frac{1}{2^{32}})^2 \times \frac{2^{32}-1}{2^{32}}$. We do not go into further consideration since these events are highly improbable.

- $c_{i,t}, i \in \{0, \ldots 7\}$, with probability $\frac{256}{513}$. Again, let $z \in \{0, \ldots 7\}$ such that $c'_{j,t+1} \neq c_{j,t+1}$ for $j = i, \ldots i + z$ and $c'_{j,t+1} = c_{j,t+1}$ for $j = i + z + 1, ..7$. If
  - $z = 0$, which happens with probability $1 - p_{cr}$ if $i \leq 6$ and with probability 1 if $i = 7$, the same analysis as with a fault on $x_{i,t}$ values applies. Namely, the correct $d_{ST}(S, S')$ value will be returned with probability $\frac{2^{32}-1}{2^{32}}$ and otherwise criterion value in question will be left undefined
  - $z = 1$, which happens with probability $p_{cr} \times \frac{2^{32}-1}{2^{32}}$ if $i \leq 5$, with probability p_{cr} if $i = 6$ and with probability 0 if $i = 7$, the following analysis applies. If values $g(x_{j,t} + c_{j,t+1})$ and $g(x_{j+1,t} + c_{j+1,t+1})$ are both equal to, or both different than $g(x'_{j,t} + c_{j,t+1})$ and $g(x'_{j+1,t} + c_{j+1,t+1})$, respectively, the criterion leaves $d_{ST}(S, S')$ undefined and the probability for this to happen is $(\frac{2^{32}-1}{2^{32}})^2 + (\frac{1}{2^{32}})^2$. However, if $g(x_{j,t} + c_{j,t+1}) = g(x_{j,t} + c'_{j,t+1})$ and $g(x_{j+1,t} + c_{j+1,t+1}) \neq g(x_{j+1,t} + c'_{j+1,t+1})$ the criterion returns the wrong value as an answer. The probability for this to happen is $\frac{1}{2^{32}} \times \frac{2^{32}-1}{2^{32}}$. In case $g(x_{j,t} + c_{j,t+1}) \neq g(x_{j,t} + c'_{j,t+1})$ and $g(x_{j+1,t} + c_{j+1,t+1}) = g(x_{j+1,t} + c'_{j+1,t+1})$, which occurs with the same probability, the criterion returns the right answer.
  - $z = 2$, which happens with probability $p_{cr} \times \frac{1}{2^{32}} \times \frac{2^{32}-1}{2^{32}}$ if $i \leq 4$, with probability $p_{cr} \times \frac{1}{2^{32}}$ if $i = 5$ and with probability 0 if $i \geq 6$, the following consideration applies. In the case where all three g values are changed, $d_{ST}(S, S')$ value is left undefined and the probability for this case is $(\frac{2^{32}-1}{2^{32}})^3$. Other cases are highly improbable and we do not consider them.
  - $z \geq 3$ occurs with probability $p_{cr} \times \frac{1}{2^{32}} \times \frac{2^{32}-1}{2^{32}}$. We do not go into consideration of further cases since their corresponding probabilities are negligible.

Using the probabilities from the discussion above, but ignoring parts that are less than $\frac{1}{2^{32}}$, provides a practically accurate estimate for the probability that the criterion will return a correct $d_{ST}(S, S')$ value as follows:

$$\begin{aligned} p_{corr} \approx\ & \frac{256}{513}\frac{2^{32}-1}{2^{32}} + \frac{1}{513}(\frac{2^{32}-1}{2^{32}})^2 + \\ & \frac{7 \times 32}{513}(1 - p_{cr})\frac{2^{32}-1}{2^{32}} + \frac{32}{513}(1 \times \frac{2^{32}-1}{2^{32}}) = 0.98635 \end{aligned}$$

Again, ignoring terms that are less than $\frac{1}{2^{32}}$, probability of $d_{ST}(S, S')$ being left undefined is given by

$$\begin{aligned} p_{undef} \approx\ & \frac{6 \times 32}{513} \times p_{cr} \times \frac{2^{32}-1}{2^{32}}((\frac{2^{32}-1}{2^{32}})^2 + (\frac{1}{2^{32}})^2) + \\ & \frac{32}{513} \times p_{cr} \times ((\frac{2^{32}-1}{2^{32}})^2 + (\frac{1}{2^{32}})^2) = 0.013645 \end{aligned}$$

Finally, ignoring terms less than $\frac{1}{2^{32}}$ yields probability for the criterion to return a false $d_{ST}(S, S')$ value is $p_{incorr} \approx 0$.

B Second Order Terms in $p(n)$ Probability

As defined in section 5.2, $p(n)$ denotes the probability that there will be at least 3 faults at each one of the eight groups of bits defined in Eq. (6). In the following we provide formulas for second-order terms participating in the equation for $p(n)$:

$$P[B^0_{i_1} \cap B^0_{i_2}] = (\frac{6}{8})^n$$

$$P[B^0_{i_1} \cap B^1_{i_2}] = \sum_{k_1+k_2=n-1} (\frac{6}{8})^{k_1} \frac{1}{8} (\frac{6 \times 32 + 1}{8 \times 32})^{k_2}$$

$$P[B^0_{i_1} \cap B^2_{i_2}] = \sum_{k_1+k_2+k_3=n-2} (\frac{6}{8})^{k_1} \frac{1}{8} (\frac{6 \times 32 + 1}{8 \times 32})^{k_2} \frac{31}{8 \times 32} (\frac{6 \times 32 + 2}{8 \times 32})^{k_3}$$

$$P[B^1_{i_1} \cap B^2_{i_2}] =$$
$$3 \times \sum_{k_1+k_2+k_3+k_4=n-3} (\frac{6}{8})^{k_1} \frac{1}{8} (\frac{6 \times 32 + 1}{8 \times 32})^{k_2} \frac{1}{8} (\frac{6 \times 32 + 2}{8 \times 32})^{k_3} \frac{31}{8 \times 32} (\frac{6 \times 32 + 3}{8 \times 32})^{k_4}$$

$$P[B^2_{i_1} \cap B^2_{i_2}] =$$
$$6 \times \sum_{k_1+k_2+k_3+k_4+k_5=n-4} (\frac{6}{8})^{k_1} \frac{1}{8} (\frac{6 \times 32 + 1}{8 \times 32})^{k_2} \frac{1}{8} (\frac{6 \times 32 + 2}{8 \times 32})^{k_3}$$
$$\frac{31}{8 \times 32} (\frac{6 \times 32 + 3}{8 \times 32})^{k_4} \frac{31}{8 \times 32} (\frac{6 \times 32 + 4}{8 \times 32})^{k_5}$$

C On the Non-surjectiveness Property of the Function Used to Derive Differences

In this appendix, we examine the expression

$$([s^{(\alpha_{i,j})} || s^{(\beta_{i,j})}] \oplus A) - ([s'^{(\alpha_{i,j})} || s'^{(\beta_{i,j})}] \oplus A)$$

which is used to derive differences in `derive_inf(i,j)` procedure. One set of generalized differentials is created by fixing A and applying the expression to pairs of groups of bits of non-faulty and faulty keystream words (s, s'), (s, s''). If the function described by the expression above was $1-1$, the number of such different sets when A goes from $0, \ldots 2^{32} - 1$ would be 2^{32}. However, since the expression above is not a $1-1$ function, the question of what is the degree of repetition of generalized differential sets when A changes arises. In this section, we give an answer to this question and show how this can be used to provide some minor improvements in the computational complexity of the attack. Better lower bounds for the attack success probabilities are also provided.

Isolating the function given by the expression above yields

$$\Phi_{x,y}(A) = x \oplus A - y \oplus A$$

where x, y and A are 32-bit values. Clearly, the function is not $1-1$. For example, for A and A' such that $d(A, A') = \{31\}$, $\Phi_{x,y}(A) = \Phi_{x,y}(A')$ where $d(\cdot, \cdot)$ denotes the set of bit positions on which the enclosed bit strings differ, least significant bit being bit 0 (e.g., $d(0000, 1010) = \{0, 2\}$). Rephrasing the question from previous paragraph yields the problem of how many different sets

$$\{\Phi_{s,s'}(A), \Phi_{s,s''}(A), \dots \Phi_{s,s'\cdots'}(A)\} \tag{7}$$

is expected to be constructed when A goes from 0 to $2^{32} - 1$.

To start, we are interested for which A, A' will $\Phi_{x,y}(A) = \Phi_{x,y}(A')$ hold. Let $z = \Phi_{x,y}(A)$ and $z' = \Phi_{x,y}(A')$. Then, $z_i = x_i \oplus A_i \oplus y_i \oplus A_i \oplus c_i$, where z_i denotes i-th bit of z. Cancelling out A_i and same reasoning for z' gives

$$z_i = x_i \oplus y_i \oplus c_i \tag{8}$$

$$z'_i = x_i \oplus y_i \oplus c'_i \tag{9}$$

where

$$c_i = \begin{cases} 0 \text{ if } (x_{i-1} \oplus A_{i-1}, y_{i-1} \oplus A_{i-1}, c_{i-1}) \in \\ \qquad \{(1,0,0),\ (1,0,1),\ (1,1,0),\ (0,0,0)\} \\ 1 \text{ if } (x_{i-1} \oplus A_{i-1}, y_{i-1} \oplus A_{i-1}, c_{i-1}) \in \\ \qquad \{(0,1,0),\ (0,1,1),\ (0,0,1),\ (1,1,1)\} \end{cases} \tag{10}$$

and c'_i is defined analogously, with A' instead of A.

As already noted, if $d(A, A') = \{31\}$, $\Phi_{x,y}(A) = \Phi_{x,y}(A')$ for every x and y. Let $S_1 = \{i | x_i = y_i, i \neq 31\}$.

Lemma 1. $\Phi_{x,y}(A) = \Phi_{x,y}(A') \Leftrightarrow d(A, A') \subseteq S_1 \cup \{31\}$

Proof. Let $z = \Phi_{x,y}(A)$ and $z' = \Phi_{x,y}(A')$.
($\Rightarrow$:) Assuming that the negation of the right side of equation is true, the goal is to prove that $z \neq z'$. According to (8) and (9), it is sufficient to show that $c_i \neq c'_i$ for some $i = 0, \dots 30$. Let $i_0 = \min\{i | i \in d(A, A'), i \notin S_1\}$. According to (10), since if $c_{i+1} = 1$, then $c'_{i+1} = 0$ and vice versa. Thus, $z_{i_0+1} \neq z'_{i_0+1}$ and $z \neq z'$.
($\Leftarrow$:) It is sufficient to prove that $c_i = c'_i$ for $i = 0, \dots 30$. We do this by induction. Let $i_0 = min(d(A, A'))$. Obviously, $c_i = c'_i$ for $i = 0, \dots i_0 - 1$. According to (10) and using that $c_{i_0-1} = c'_{i_0-1}$ yields $c_{i_0} = c'_{i_0}$. Now let $i_n \in d(A, A')$. By induction hypothesis, $c_{i_n-1} = c'_{i_n-1}$. Again, using (10) yields $c_{i_n} = c_{i_n}$, which proves this part of the Lemma.

In other words, for any given A, changing bits from $S_1 \cup \{31\}$ will not change $\Phi_{x,y}(A)$. On the other hand, changing any other bits in A changes $\Phi_{x,y}(A)$. Hence

$$|\mathrm{Im}(\Phi_{x,y})| = 2^{32-(|S_1|+1)} \tag{11}$$

Since the expected value for $|S_1|$ is 15.5, the expected number for $E(|\mathrm{Im}(\Phi_{x,y})|) = 2^{32-15.5-1} = 2^{15.5}$. Analogous analysis can be used to estimate the expected number of different sets (7) when A goes from $0, \ldots 2^{32} - 1$. Denote $s^{'}$ by $s^{[1]}$, $s^{''}$ by $s^{[2]}$, etc. Define the set S_n as follows

$$S_n = \{i | s_i = s_i^{[1]} = \ldots = s_i^{[n]}, i \neq 31\}$$

A generalization of the previous lemma can be used:

Lemma 2

$$\begin{aligned}&(\Phi_{s,s^{[1]}}(A), \ldots, \Phi_{s,s^{[n]}}(A)) = (\Phi_{s,s^{[1]}}(A'), \ldots, \Phi_{s,s^{[n]}}(A'))\\ &\Leftrightarrow d(A, A') \subseteq S_n \cup \{31\}\end{aligned}$$

To calculate $E(|S_n|)$, let X_i, $i = 0, ..30$ denote a random variable which takes value 1 if $s_i = s_i^{[1]} = \ldots = s_i^{[n]}$ and value 0 otherwise. It is easy to see that $E(X_i) = \frac{1}{2^n}$, $i = 0, \ldots 30$. Then, for some particular instantiation of s values, the number of mutually equal bits is given by $X = X_0 + \ldots X_{30}$, the most significant bit being excluded. Hence $E(X) = E(X_0 + \ldots + X_{30}) = E(X_0) + \ldots + E(X_{30}) = 31 \times \frac{1}{2^n}$.

Now the question from the beginning of this appendix can be answered by

$$\begin{aligned}&E_n = E(|\{\Phi_{s,s^{[1]}}(A), \Phi_{s,s^{[2]}}(A), \ldots \Phi_{s,s^{[n]}}(A)\}_{A=0}^{2^{32}-1}|) \approx\\ &E(|(\Phi_{s,s^{[1]}}(A), \ldots, \Phi_{s,s^{[n]}}(A))_{A=0}^{2^{32}-1}|) = 2^{32-(|S_n|+1)} = 2^{32-((\frac{31}{2^n})+1)}\end{aligned}$$

Table 2 shows the expected number of different $\Delta_i^j(A)$ sets when A varies from 0 to $2^{32} - 1$, for different $|FLTS_i| = n$ sizes.

Thus the complexity and success probability of the attack can now be further refined as follows:

- Computational complexity: in procedure `derive_inf(i,j)`, in the main loop, it is possible not to go through the whole set of A values, by fixing bit positions from determined by set S_n, i.e., bit positions on which $s, s', \ldots$, are mutually equal. If $|FLTS_i| = 3$, the loop will not have 2^{32} steps, but $2^{27.125}$ steps on average. Consequently, the overall attack complexity is now reduced to $2^{32.71}$ steps on average.
- Success probability: Eq. (4) now becomes

$$P[Sat(\Delta_i^j) \text{ is empty }] \geq (1 - 2^{-46})^k$$

where instead of $k = 2^{32}$, k takes values from the table, depending on $|FLTS_i| = n$. For $|FLTS_i| = 3$, the above value becomes equal to $1 - 2^{-18.9}$ instead of $1 - 2^{-14}$ and probability that the algorithm will return more than one element in set $Cand(x_{i,t} + c_{i,t+1})$ given by (5) will be 2^{-56} instead of 2^{-42}.

Table 2. Expected number of different $\Delta_i^j(A)$ sets

n	3	4	5	6	7	8
E_n	$2^{27.125}$	$2^{29.062}$	$2^{30.031}$	$2^{30.516}$	$2^{30.758}$	$2^{30.879}$