INTERNAL STATE RECOVERY OF ESPRESSO STREAM CIPHER USING CONDITIONAL SAMPLING RESISTANCE AND TMDTO ATTACK

NISHANT SINHA
Bosch India (RBEI/ESY)
Bangalore, India

(Communicated by Renate Scheidler)

ABSTRACT. Espresso is a stream cipher proposed for the 5G wireless communication system. Since the design of this cipher is based on the Galois configuration of NLFSR, the cipher has a short propagation delay, and it is the fastest among the ciphers below 1500 GE, including Grain-128 and Trivium. The time-memory-data tradeoff (TMDTO) attack on this cipher and finding the conditional BSW sampling resistance are difficult due to its Galois configuration. This paper demonstrates the calculation of conditional BSW-sampling resistance of Espresso stream cipher, which is based on Galois configuration, and also mounts the TMDTO attack on the cipher by employing the calculated sampling resistance. It is also shown that the attack complexities of TMDTO attack are lower than those claimed by the designers of the ciphers.

1. Introduction

The security is always a challenge during communication over insecure channels where channels may be wired or wireless. The security of such communication systems heavily relies on ciphering schemes. Several ciphering schemes have been proposed to meet the challenges in secure communication. Today’s applications demand more efficient and lightweight cipher systems without compromising any security parameters. Espresso is a stream cipher proposed by Dubrova and Hell [7] to meet the security requirements of 5G wireless communication systems. In the 5G communication system, the volume of traffic over the channel is very high in comparison to the Long-Term Evolution (LTE) system. The designers of Espresso claim that its throughput is higher than other ciphers having below 1500 GE, including Grain [9] and Trivium [5], as well as being resistant to well-known attacks. The designs of the majority of stream ciphers are based on feedback shift registers (FSR), which may be linear or nonlinear. A nonlinear feedback shift register (NFSR) can be implemented using either Fibonacci configuration or Galois configuration. Both configurations have their advantages and disadvantages. In Galois configuration, there are several small feedback functions used to update the internal state of the register, while in Fibonacci configuration, a large feedback function is used to do the same. Due to a large feedback function in Fibonacci configuration, propagation delay is higher, while security evaluation is simple. In Galois configuration, propagation delay is very low because of small feedback functions, while the security

2010 Mathematics Subject Classification: Primary: 68P25, 94A60; Secondary: 06E30.
Key words and phrases: Stream cipher, cryptanalysis, Galois configuration, BSW-sampling resistance, time-memory-data tradeoff.
evaluation is complicated. However, Galois configuration can be transformed into Fibonacci configuration and vice-versa [6]. While surveying the literature related to cryptanalysis, we have been observed that the feedback shift registers of those cipher designs are predominantly based on the Fibonacci configuration. Thus, it is challenging to perform cryptanalysis of a stream cipher like Espresso, whose design is based on Galois configuration. The designers of Espresso have pointed out that it is difficult to attempt the time-memory-data tradeoff (TMDTO) attack on the cipher for internal state recovery primarily because of its Galois configuration. In this paper, we perform cryptanalysis of Espresso in its Galois configuration using conditional sampling resistance and TMDTO attacks.

2. The design of espresso

Espresso is a stream cipher proposed for the 5G wireless communication system. This stream cipher consists of two basic components, one is 256-bit NLFSR, and another is a nonlinear output function. The sizes of key and initialization vector are 128 bits and 96 bits, respectively. The complete description of the cipher is presented in [7]. The internal state bits are denoted by $x_i$ where $0 \leq i \leq 255$. The considered NLFSR in the cipher is in the Galois configuration. Therefore, the stages of the cipher are updated by using the following feedback functions:

\begin{align}
    x_{255} &= x_0 \oplus x_{41}x_{70} \\
    x_{251} &= x_{252} \oplus x_{42}x_{83} \oplus x_8 \\
    x_{247} &= x_{248} \oplus x_{44}x_{102} \oplus x_{40} \\
    x_{243} &= x_{244} \oplus x_{43}x_{118} \oplus x_{103} \\
    x_{239} &= x_{240} \oplus x_{46}x_{141} \oplus x_{11} \\
    x_{235} &= x_{236} \oplus x_{67}x_{90}x_{110}x_{137} \\
    x_{231} &= x_{232} \oplus x_{50}x_{159} \oplus x_{189} \\
    x_{217} &= x_{218} \oplus x_{3}x_{32} \\
    x_{213} &= x_{214} \oplus x_{4}x_{45} \\
    x_{209} &= x_{210} \oplus x_{6}x_{64} \\
    x_{205} &= x_{206} \oplus x_{5}x_{80} \\
    x_{201} &= x_{202} \oplus x_{8}x_{103} \\
    x_{197} &= x_{198} \oplus x_{29}x_{52}x_{72}x_{99} \\
    x_{193} &= x_{194} \oplus x_{12}x_{21}
\end{align}

The rest of the stages of the cipher are updated by the following equation:

\begin{equation}
    x_i = x_{i+1}.
\end{equation}

The nonlinear output function is a 14 variables function that computes the $i$-th keystream bit $z_i$ as follows:

\begin{align}
    z_i &= x_{i+80} \oplus x_{i+99} \oplus x_{i+137} \oplus x_{i+227} \oplus x_{i+222} \oplus x_{i+187} \oplus x_{i+243}x_{i+217} \\
    &\oplus x_{i+247}x_{i+231} \oplus x_{i+213}x_{i+235} \oplus x_{i+255}x_{i+251} \oplus x_{i+181}x_{i+239} \\
    &\oplus x_{i+174}x_{i+44} \oplus x_{i+164}x_{i+29} \oplus x_{i+255}x_{i+247}x_{i+243}x_{i+213}x_{i+181}x_{i+174}.
\end{align}
3. Computation of Conditional Sampling Resistance of Espresso

3.1. Background. This section, firstly, describes the TMDTO attack and sampling resistance in brief. Further, the computation of conditional sampling resistance for Espresso cipher is discussed. There are two phases of the TMDTO attack. One is the preprocessing phase, and another is the online phase. The primary goal of the TMDTO attack is to recover the internal state of a cipher using the preprocessing tables constructed in the preprocessing phase and the data received in the online phase. The preprocessing tables usually cover a certain number of possible internal states of ciphers such that there will be at least one collision with the internal states generating keystream bits intercepted in the online phase. Biryukov, Shamir, and Wagner [3] introduced the concept of sampling resistance that provides a way to reduce the number of required internal states covered by the preprocessing table. The sampling resistance assists in enumerating a set of special states such that the first $k$ bits of the keystream generated by them are of a particular pattern. The preprocessing tables cover only those special states. The maximum value of $k$ depends on the specific cipher design. Some internal state bits of the cipher are fixed with particular values for maximizing the value of $k$. This scenario leads to the concept of conditional time-memory-data tradeoff attack, and $R = 2^{-k}$ is known as conditional sampling resistance.

3.2. Description of the Computation of Conditional Sampling Resistance.

Now we describe the computation of sampling resistance of Espresso in detail. To compute the sampling resistance, it is required that we have access to consecutive keystream bits $z_i$, for $i = 0, \ldots, 34$. As we know, usually, stream ciphers are proposed using Fibonacci configuration, and in such ciphers, the internal state of the FSR is updated only by using shifting operation except for the last state bit of FSR (which is updated using a feedback function). But due to the use of Galois configuration in Espresso cipher, more than one internal state bit of FSR is updated simultaneously using different feedback functions. For this reason, the authors of Espresso cipher claim that finding sampling resistance is difficult. In this section, we show that it is possible to calculate sampling resistance by using a sequence of equations. During the description of these equations, the internal state bits are usually denoted by $x_i$, where $0 \leq i \leq 255$. Whenever the feedback bits are involved in the equations, then the state bits are underlined so that one can easily distinguish between internal state bits and feedback bits.

In the following sequence of equations we compute the values of 35 internal state bits in terms of the other internal state bits by using the first 35 keystream bits and fixing the 39 internal state bits with 0 values. Let the set $\mathcal{F}$ represents the internal state bits which are fixed with value 0, where $\mathcal{F} = \mathcal{F}_1 \cup \mathcal{F}_2$ such that $\mathcal{F}_1 = \{ x_i = 0 : i = 29, \ldots, 36 \}$ and $\mathcal{F}_2 = \{ x_i = 0 : i = 46, \ldots, 76 \}$. The equation (3), which is used for keystream generation, can be rewritten as

$$
\begin{align*}
  x_{i+137} &= z_i \oplus x_{i+80} \oplus x_{i+99} \oplus x_{i+227} \oplus x_{i+222} \oplus x_{i+187} \oplus x_{i+243} x_{i+217} \\
  &\quad \oplus x_{i+247} x_{i+231} \oplus x_{i+213} x_{i+235} \oplus x_{i+255} x_{i+251} \oplus x_{i+181} x_{i+239} \\
  &\quad \oplus x_{i+174} x_{i+44} \oplus x_{i+164} x_{i+29} \\
  &\quad \oplus x_{i+255} x_{i+247} x_{i+243} x_{i+213} x_{i+181} x_{i+174}.
\end{align*}
$$

(4)

When we place $i = 0$ in equation (4) to recover $x_{137}$, the indices of internal state bit appearing on the right hand side of the equation are as follows:
Nishant Sinha

Thus to recover $x_{137}$, these internal state bits are required to be guessed, except $x_{29}$ and $x_{164}$ because $x_{164}$ appears as product term with $x_{29}$ and $x_{29}$ is fixed with 0 due to set $F_1$.

In what follows, we demonstrate how the consecutive internal state bits from $x_{138}$ to $x_{171}$ are recovered by using the keystream bits from $z_1$ to $z_{34}$ and equation (4). It should be noted that all state bits appearing in the equations used for recovery are either internal state bits or feedback bits with respect to $z_0$. Thus to make the descriptions of the equations more intelligible, it is required to rewrite the feedback functions (1) as follows:

\[
\begin{align*}
x_{i+256} &= x_{i+0} \oplus x_{i+41} x_{i+70} \\
x_{i+252} &= x_{i+252} \oplus x_{i+42} x_{i+83} \oplus x_{i+8} \\
x_{i+248} &= x_{i+248} \oplus x_{i+44} x_{i+102} \oplus x_{i+40} \\
x_{i+244} &= x_{i+244} \oplus x_{i+43} x_{i+118} \oplus x_{i+103} \\
x_{i+240} &= x_{i+240} \oplus x_{i+46} x_{i+141} \oplus x_{i+117} \\
x_{i+236} &= x_{i+236} \oplus x_{i+67} x_{i+90} x_{i+110} x_{i+137} \\
x_{i+232} &= x_{i+232} \oplus x_{i+50} x_{i+159} \oplus x_{i+189} \\
x_{i+218} &= x_{i+218} \oplus x_{i+3} x_{i+32} \\
x_{i+214} &= x_{i+214} \oplus x_{i+4} x_{i+45} \\
x_{i+210} &= x_{i+210} \oplus x_{i+6} x_{i+64} \\
x_{i+206} &= x_{i+206} \oplus x_{i+5} x_{i+80} \\
x_{i+202} &= x_{i+202} \oplus x_{i+8} x_{i+103} \\
x_{i+198} &= x_{i+198} \oplus x_{i+29} x_{i+52} x_{i+72} x_{i+99} \\
x_{i+194} &= x_{i+194} \oplus x_{i+12} x_{i+21}
\end{align*}
\]

When we place $i = 1, \ldots, 34$ to recover $x_{138}, \ldots, x_{171}$ in equation (4), the feedback bits will start to appear in the equations and correspondingly we have to select the equations from (5), \ldots, (18) to get the values of feedback bits. The recovery of internal state bits from $x_{138}$ to $x_{141}$ is simple. The problem arises when we attempt to recover the internal state bits from $x_{142}$ onwards. For instance, when we place $i = 5$ in equation (4) to recover $x_{142}$ and derive the following:

\[
x_{142} = x_5 \oplus x_{85} \oplus x_{104} \oplus x_{232} \oplus x_{i+227} \oplus x_{192} \oplus x_{248} x_{222}
\]

\[
\oplus x_{252} x_{236} \oplus x_{218} x_{240} \oplus x_{260} x_{256} \oplus x_{186} x_{244}
\]

\[
\oplus x_{169} x_{34} \oplus x_{260} x_{252} x_{248} x_{218} x_{186} x_{179}.
\]

We note that the feedback bit $x_{256}$ appears on the right-hand side of equation (19). It seems that it can be calculated using any of the equations out of (5), \ldots, (18) by placing corresponding values. But due to consequences of shifting operation of Feedback Shift Register (FSR) and Galois configuration based cipher design, it should be calculated by using only equations (5) and (6) as follows:

1. First calculate $x_{256}$ by placing $i = 0$ in the equation (5).
2. Then, we again calculate $x_{256}$ by placing $i = 4$ in the equation (6) and derive

\[
x_{256} = x_{256} \oplus x_{46} x_{87} \oplus x_{12}.
\]
We use the value of \( x_{256} \), calculated at Step 1, in place of \( x_{256} \) appearing in the right hand side of equation (20) and evaluate the new value of \( x_{256} \). This new value of \( x_{256} \) will be used in equation (19).

Thus we may conclude that a situation occurs in which required state bits (for calculating feedback bits) are itself feedback bits. For example, as earlier in step 2, when we calculate feedback bit \( x_{256} \) by placing \( i = 4 \) in equation (6), then \( x_{256} \) also appears on the right-hand side of the equation which is feedback bit itself, instead of internal state bit because of equation (5).

To make the last description understandable entirely, we prepared Table 1, Table 2, and Table 3, which lists all the feedback bits appearing throughout the equations and the state bits appearing in the right-hand side of corresponding feedback functions. In each table, there are column numbers that indicate which feedback bits are calculated by using which feedback equations out of (5), . . . , (18). For instance, column 0 indicates feedback equation (5), column 1 indicates feedback equation (6) and so on. The feedback bits (which are to be calculated) as well state bits (which appear in the right-hand side of corresponding feedback equations) are superscripted to indicate which feedback equations are required to calculate them. The state bits which have no superscript indicate that they are the internal state bits.

In the tables, over-lined state bits indicate that they are fixed with 0 value, and underlined state bits indicate that they appear as product terms with over-lined bits. In the tables, over-lined state bits indicate that they are fixed with 0 value, and thus underlined state bits are required to calculate the corresponding feedback bits. For example, row 4 of column 1 of Table 1 represents that when we put \( i = 4 \) in feedback function (6) to calculate the value of \( x^1_{256} \), the state bits \( x^0_{256}, x_{46}, x_{87}, x_{12} \) appear in the right hand side of equation. Out of all those state bits, \( x_{46}, x_{87}, x_{12} \) are internal state bits, except \( x^0_{256} \) which is calculated using row 0 of column 0 of Table 1. At the same time, the internal state bit \( x_{46} \) is fixed with 0 value, and thus \( x_{87} \) is not required to calculate the value of \( x^1_{256} \). In the end, we conclude that it is required to know the value of all state bits as per entry of tables to calculate corresponding feedback bits except over-lined state bits and underlined state bits.

Thus for \( i = 1, 2, 3, 4 \), we rewrite the keystream generation equation (4) as equation (21), where superscripts of variables indicate the columns of Table 1, Table 2 and Table 3, used to calculate them. The internal state bits are those that appear without superscripts, except \( z_i \).

\[
x_{i+137} = z_i \oplus x_{i+80} \oplus x_{i+99} \oplus x_{i+227} \oplus x_{i+222} \oplus x_{i+187} \oplus x^3_{i+243} x^7_{i+217} \\
\quad \oplus x^2_{i+247} x^6_{i+231} \oplus x^8_{i+213} x^5_{i+235} \oplus x^0_{i+255} x^1_{i+251} \oplus x_{i+181} x^4_{i+239} \\
\quad \oplus x_{i+174} x_{i+44} \oplus x_{i+164} x_{i+29} \\
\quad \oplus x^0_{i+255} x^2_{i+247} x^3_{i+243} x^8_{i+213} x_{i+181} x_{i+174}.
\]

The internal state bit \( x_{138} \) is recovered by placing \( i = 1 \) in equation (21) and derive

\[
x_{138} = z_1 \oplus x_{81} \oplus x_{100} \oplus x_{228} \oplus x_{223} \oplus x_{188} \oplus x^3_{244} x^7_{218} \\
\quad \oplus x^2_{248} x^6_{232} \oplus x^5_{214} x^5_{216} \oplus x^0_{256} x^4_{252} \oplus x_{182} x^4_{240} \\
\quad \oplus x_{175} x_{45} \oplus x_{165} x_{30} \oplus x^0_{256} x^2_{248} x^3_{244} x^8_{214} x_{182} x_{175}.
\]

In the right-hand side of equation (22), internal state bits and feedback bits are present. The set \( A_i \) contains the indices of those internal state bits that appear directly as well as the internal state bits required to calculate feedback bits appear.
| Row | Feedback bit calculation | Feedback bit calculation | Feedback bit calculation | Feedback bit calculation | Feedback bit calculation |
|-----|--------------------------|--------------------------|--------------------------|--------------------------|--------------------------|
|     | because of (5)           | because of (6)           | because of (7)           | because of (8)           | because of (9)           |
|     | Feedback bits            | State bits               | Feedback bits            | State bits               | Feedback bits            | State bits               |
|     | bits appeared on RHS of (5) | bits appeared on RHS of (6) | bits appeared on RHS of (7) | bits appeared on RHS of (8) | bits appeared on RHS of (9) |
| 0   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 1   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 2   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 3   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 4   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 5   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 6   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 7   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 8   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 9   | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 10  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 11  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 12  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 13  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 14  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 15  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 16  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 17  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 18  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 19  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 20  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 21  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 22  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 23  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 24  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 25  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 26  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 27  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 28  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 29  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 30  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 31  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 32  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 33  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |
| 34  | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    | v_{10}                    |

on the right-hand side of equation (22). 
\[ \mathcal{A}_1 = \{81, 100, 228, 223, 188, (244, 43, 118, 103), (218, 2, 32), (248, 44, 102, 40), (232, 30, 159, 189), (214, 4, 45), (236, 67, 90, 110, 137), (0, 41, 70), (252, 42, 83, 8), 182, (240, 36, 141, 117), 175, 45, 165, 30 \}. \]
In the set $\mathcal{A}_1$, the over-lined state bits are those bits that belong to the set $\mathcal{F}$, i.e., are fixed, and the state bits enclosed in parenthesis are those state bits which are required to calculate the feedback bits appearing in equation (22). However, due
Table 3. State Bits required to calculate feedback bits

| Row | Feedback calculation because of (15) | Feedback calculation because of (16) | Feedback calculation because of (17) | Feedback calculation because of (18) |
|-----|------------------------------------|------------------------------------|------------------------------------|------------------------------------|
| Column 10 | State bits appeared on RHS of (15) | State bits appeared on RHS of (15) | State bits appeared on RHS of (15) | State bits appeared on RHS of (15) |
| Column 12 | State bits appeared on RHS of (17) | State bits appeared on RHS of (17) | State bits appeared on RHS of (17) | State bits appeared on RHS of (17) |
| Column 13 | State bits appeared on RHS of (18) | State bits appeared on RHS of (18) | State bits appeared on RHS of (18) | State bits appeared on RHS of (18) |

- Row 1: 0
- Row 2: 1
- Row 3: 2
- Row 4: 3
- Row 5: 4
- Row 6: 5
- Row 7: 6
- Row 8: 7
- Row 9: 8
- Row 10: 9
- Row 11: 10
- Row 12: 11
- Row 13: 12
- Row 14: 13
- Row 15: 14
- Row 16: 15
- Row 17: 16
- Row 18: 17
- Row 19: 18
- Row 20: 19
- Row 21: 20
- Row 22: 21
- Row 23: 22
- Row 24: 23
- Row 25: 24
- Row 26: 25
- Row 27: 26
- Row 28: 27
- Row 29: 28
- Row 30: 29
- Row 31: 30
- Row 32: 31
- Row 33: 32
- Row 34: 33
to the set \( \mathcal{F} \), many state bits are not required to recover the value of the internal state bit \( x_{138} \). For example, \( x_{67} \) is fixed with the value 0 because of the set \( \mathcal{F} \). Consequently, we do not require the value of \( x_{90}, x_{110}, x_{137} \) because these state bits appear in terms containing \( x_{67} \) in equation (10) (or Column 5 of Table 2). In the same manner, \( x_{165} \) appears in a term with \( x_{30} \) (which is fixed with value 0), and so, we do not require the value of \( x_{165} \). In Step 0 (during recovery of \( x_{137} \)), we have already guessed \( x_{44} \). Thus, \( x_{44} \) is also excluded. By considering the last fact, the indices of the internal state bits, which are required to guess for recovery of \( x_{138} \), are described using the following set:

\[
\mathcal{A}_2 = \{81, 100, 228, 223, 188, (244, 43, 118, 103), (218), (248, 102, 40), (232, 189), (214, 4, 45), (236), (0), (252, 42, 83, 8), 182, (240, 117), 175, 45\}. 
\]

Thus, \( x_{138} \) is recovered using the keystream bit \( z_1 \) and the internal state bits whose indices belong to the set \( \mathcal{A}_2 \). In the similar manner, we recover the internal state bits \( x_{139}, x_{140} \) and \( x_{141} \).

As we proceed further to recover more internal state bits, there is an increase in the number of feedback bits appearing in the equations used for recovery. Thus, we have to rewrite equation (21) as equation (23) to recover \( x_{142} \) and \( x_{143} \) and attach one more superscript to the state bits appearing on the right-hand side of the equation.

\[
x_{i+137} = z_i \oplus x_{i+80} \oplus x_{i+99} \oplus x_{i+227}^6 \oplus x_{i+222} \oplus x_{i+187} \oplus x_{i+243} x_{i+217}^7 \oplus x_{i+247} x_{i+231}^8 \oplus x_{i+123} x_{i+135}^8 \oplus x_{i+255} x_{i+251}^8 \oplus x_{i+181} x_{i+253}^1 \nonumber \\
\quad \quad \quad \oplus x_{i+174} x_{i+44} \oplus x_{i+164} x_{i+29} \nonumber \\
\quad \quad \quad \oplus x_{i+255} x_{i+247} x_{i+133} x_{i+181} x_{i+174}^7. 
\]

In the above equation we place \( i = 5 \) to recover \( x_{142} \) and derive

\[
x_{142} = z_5 \oplus x_{85} \oplus x_{104} \oplus x_{232}^6 \oplus x_{227} \oplus x_{192} \oplus x_{248} x_{222}^7 \nonumber \\
\quad \quad \quad \oplus x_{252} x_{236}^6 \oplus x_{218} x_{245}^8 \oplus x_{260} x_{256}^6 \oplus x_{186} x_{244}^4 \nonumber \\
\quad \quad \quad \oplus x_{179} x_{249} \oplus x_{169} x_{34} \oplus x_{200} x_{252} x_{248} x_{218} x_{186} x_{179}. 
\]

In the right-hand side of equation (24), internal state bits and feedback bits are present. The set \( \mathcal{A}_3 \) contains the indices of those internal state bits that appear directly as well as the internal state bits required to calculate feedback bits appearing on the right-hand side of the equation (24).

\[
\mathcal{A}_3 = \{85, 104, (232, 50, 159, 189), 227, 192, (248, 7, 122, 107), (222, 7, 36), (252, 45, 106, 44), (236, 5, 163, 193), (218, 8, 49), (240, 7, 94, 114, 141), (4, 45, 71), (256, 76, 87, 12), 186, (244, 50, 145, 221), 179, 76, 169, 34\}. 
\]

Again we have to rewrite the elements of the set \( \mathcal{A}_3 \) by excluding those elements which are not required to recover the value of \( x_{142} \) because over-lined state bits in set \( \mathcal{A}_3 \) are fixed with 0 value, and those state bits appear as product term with underlined state bits. Thus, the internal state bits which are required to guess to recover the value of \( x_{142} \) are as follows:

\[
\mathcal{A}_4 = \{85, 104, (232, 189), 227, 192, (248, 44, 102, 40, 107), (222), (252, 42, 83, 8, 44), (236, 193), (218), (240, 117), (4), (0, 41, 70, 12), 186, (244, 43, 118, 103, 221)\}. 
\]
The indices of internal state bits which we have already guessed at earlier steps are the following:

\[ \mathcal{A}_5 = \{(232, 189), 227, (248, 44, 102, 40), (222), (252, 42, 83, 8, 44), (236), (218), (240, 117), (4), (0), (244, 43, 118, 103)\} \]

Thus, to recover \( x_{142} \), we guess only those state bits whose indices belong to set \( \mathcal{A}_4 - \mathcal{A}_5 \).

In same manner we can recover \( x_{143} \). To recover \( x_{144}, x_{145} \) and \( x_{146} \), we have to rewrite equation (21) as follows:

\[
x_{i+137} = x_0 \oplus x_{i+80} \oplus x_{i+99} \oplus x_{i+227} \oplus x_{i+222} \oplus x_{i+187} \oplus x_{i+243} \oplus x_{i+217}
+ x_{2}^{2} + 247 \cdot x_{i+231} \oplus x_{i+213} \oplus x_{i+235} \oplus x_{i+255} \oplus x_{i+251} \oplus x_{i+181} \oplus x_{i+239}
+ x_{i+174} \oplus x_{i+4} \oplus x_{i+164} \oplus x_{i+29}
+ x_{0}^{2} + 255 \cdot x_{i+247} \oplus x_{i+234} \oplus x_{i+181} \oplus x_{i+174}. \tag{25}
\]

Now we put \( i = 7, 8, 9 \) in equation (25) to recover \( x_{144}, x_{145} \), and \( x_{146} \), respectively, and require to guess some more internal state bits which may appear directly or required to calculate feedback bits appear on the right-hand side of (25).

To recover more internal state bits from \( x_{147} \) to \( x_{171} \), we again rewrite the equation (21) by adding more superscripts to state bits appearing in the right-hand side of equations. For this, we also need to guess some more internal state bits. The index of internal state bits required to guess during this recovery method are from 0 to 28, 37 to 45, 77 to 136 and 172 to 255. The details of all equations used for recovery of internal state bits are listed in Table 4. In Table 4, over-lined state bits are state bits fixed with value 0.

| Step/Row | Equations used for recovery |
|----------|----------------------------|
| 0        | \( x_{137} = x_0 \oplus x_{80} \oplus x_{99} \oplus x_{227} \oplus x_{222} \oplus x_{187} \oplus x_{243} \oplus x_{231} \oplus x_{217} \oplus x_{255} \oplus x_{251} \oplus x_{174} \oplus x_{164} \oplus x_{253} \oplus x_{247} \oplus x_{234} \oplus x_{181} \oplus x_{174} \) |
| 1        | \( x_{138} = x_1 \oplus x_{81} \oplus x_{100} \oplus x_{228} \oplus x_{223} \oplus x_{188} \oplus x_{244} \oplus x_{218} \oplus x_{248} \oplus x_{232} \oplus x_{214} \oplus x_{236} \) |
| 2        | \( x_{139} = x_2 \oplus x_{82} \oplus x_{101} \oplus x_{229} \oplus x_{224} \oplus x_{189} \oplus x_{245} \oplus x_{219} \oplus x_{249} \oplus x_{233} \oplus x_{215} \oplus x_{237} \) |
| 3        | \( x_{140} = x_3 \oplus x_{83} \oplus x_{102} \oplus x_{230} \oplus x_{225} \oplus x_{190} \oplus x_{246} \oplus x_{220} \oplus x_{250} \oplus x_{234} \oplus x_{216} \oplus x_{238} \) |
| 4        | \( x_{141} = x_4 \oplus x_{84} \oplus x_{103} \oplus x_{231} \oplus x_{226} \oplus x_{191} \oplus x_{247} \oplus x_{221} \oplus x_{251} \oplus x_{235} \oplus x_{217} \oplus x_{239} \) |
| 5        | \( x_{142} = x_5 \oplus x_{85} \oplus x_{104} \oplus x_{232} \oplus x_{227} \oplus x_{192} \oplus x_{248} \oplus x_{222} \oplus x_{252} \oplus x_{236} \oplus x_{218} \oplus x_{240} \) |
| 6        | \( x_{143} = x_6 \oplus x_{86} \oplus x_{105} \oplus x_{233} \oplus x_{228} \oplus x_{193} \oplus x_{249} \oplus x_{223} \oplus x_{253} \oplus x_{237} \oplus x_{219} \oplus x_{241} \) |
| Step/Row | Equations used for recovery |
|----------|----------------------------|
| 7        | \[ x_{144} = 27 \oplus x_{147} \oplus x_{166} \oplus x_{234} \oplus x_{229} \oplus x_{194} \oplus x_{226} \oplus x_{247} \oplus x_{254} \oplus x_{238} \oplus x_{220} \oplus x_{242} \] |
| 8        | \[ x_{145} = 28 \oplus x_{148} \oplus x_{167} \oplus x_{235} \oplus x_{230} \oplus x_{195} \oplus x_{251} \oplus x_{225} \oplus x_{255} \oplus x_{239} \oplus x_{221} \oplus x_{243} \] |
| 9        | \[ x_{146} = 29 \oplus x_{149} \oplus x_{168} \oplus x_{236} \oplus x_{231} \oplus x_{196} \oplus x_{252} \oplus x_{226} \oplus x_{222} \oplus x_{244} \] |
| 10       | \[ x_{147} = 30 \oplus x_{150} \oplus x_{169} \oplus x_{237} \oplus x_{232} \oplus x_{197} \oplus x_{253} \oplus x_{237} \oplus x_{257} \oplus x_{241} \oplus x_{223} \oplus x_{245} \] |
| 11       | \[ x_{148} = 31 \oplus x_{151} \oplus x_{170} \oplus x_{238} \oplus x_{233} \oplus x_{198} \oplus x_{254} \oplus x_{228} \oplus x_{258} \oplus x_{242} \oplus x_{224} \oplus x_{246} \] |
| 12       | \[ x_{149} = 32 \oplus x_{152} \oplus x_{171} \oplus x_{239} \oplus x_{234} \oplus x_{199} \oplus x_{255} \oplus x_{239} \oplus x_{259} \oplus x_{243} \oplus x_{225} \oplus x_{247} \] |
| 13       | \[ x_{150} = 33 \oplus x_{153} \oplus x_{172} \oplus x_{240} \oplus x_{241} \oplus x_{200} \oplus x_{260} \oplus x_{224} \oplus x_{226} \oplus x_{248} \] |
| 14       | \[ x_{151} = 34 \oplus x_{154} \oplus x_{173} \oplus x_{242} \oplus x_{241} \oplus x_{201} \oplus x_{262} \oplus x_{225} \oplus x_{226} \oplus x_{248} \] |
| 15       | \[ x_{152} = 35 \oplus x_{155} \oplus x_{174} \oplus x_{243} \oplus x_{244} \oplus x_{202} \oplus x_{264} \oplus x_{226} \oplus x_{227} \oplus x_{249} \] |
| 16       | \[ x_{153} = 36 \oplus x_{156} \oplus x_{175} \oplus x_{245} \oplus x_{245} \oplus x_{203} \oplus x_{266} \oplus x_{227} \oplus x_{228} \oplus x_{251} \] |
| 17       | \[ x_{154} = 37 \oplus x_{157} \oplus x_{176} \oplus x_{247} \oplus x_{246} \oplus x_{204} \oplus x_{268} \oplus x_{228} \oplus x_{229} \oplus x_{252} \] |
| 18       | \[ x_{155} = 38 \oplus x_{158} \oplus x_{177} \oplus x_{248} \oplus x_{249} \oplus x_{205} \oplus x_{270} \oplus x_{230} \oplus x_{231} \oplus x_{253} \] |
| 19       | \[ x_{156} = 39 \oplus x_{159} \oplus x_{178} \oplus x_{250} \oplus x_{251} \oplus x_{206} \oplus x_{271} \oplus x_{232} \oplus x_{233} \oplus x_{254} \] |
| 20       | \[ x_{157} = 40 \oplus x_{160} \oplus x_{179} \oplus x_{252} \oplus x_{253} \oplus x_{207} \oplus x_{272} \oplus x_{234} \oplus x_{235} \oplus x_{255} \] |
| 21       | \[ x_{158} = 41 \oplus x_{161} \oplus x_{180} \oplus x_{254} \oplus x_{255} \oplus x_{208} \oplus x_{273} \oplus x_{236} \oplus x_{237} \oplus x_{256} \] |
| 22       | \[ x_{159} = 42 \oplus x_{162} \oplus x_{181} \oplus x_{256} \oplus x_{257} \oplus x_{209} \oplus x_{274} \oplus x_{238} \oplus x_{239} \oplus x_{256} \] |
| 23       | \[ x_{160} = 43 \oplus x_{163} \oplus x_{182} \oplus x_{258} \oplus x_{259} \oplus x_{210} \oplus x_{275} \oplus x_{240} \oplus x_{241} \oplus x_{257} \] |
| 24       | \[ x_{161} = 44 \oplus x_{164} \oplus x_{183} \oplus x_{260} \oplus x_{261} \oplus x_{211} \oplus x_{276} \oplus x_{242} \oplus x_{243} \oplus x_{258} \] |
3.3. Procedure for choosing the set of equations for recovery. Table 4 represents a set of equations that are used to recover 35 bits of the internal state by fixing the specific 39 bits of state with value zero. We adopt a heuristic procedure to find out such a set of equations. Such a heuristic procedure was reported earlier to mount TMDTO attack on the stream cipher Lizard by Maitra et al. [12]. The heuristic procedure adopted in this draft is slightly different from the procedure reported in [12] because of the design principles of Espresso cipher. The FSR of the Lizard cipher is based on the Fibonacci configuration, while the FSR of Espresso cipher is based on the Galois configuration. The main contribution of this paper is finding a set of equations by using a heuristic that can be employed to recover the internal state bits of Galois configuration based stream ciphers. The primary fact that guides the decisions during the selection of such equations is to maximize the number of internal state bits recovered and minimize the number of internal state bits fixed. While adopting this heuristic procedure, we have followed specific guidelines which are as follows:

- Firstly, try to choose such internal state bits for recovery that appear as linear terms in the keystream generation equation.

| Step/Row | Equations used for recovery |
|----------|----------------------------|
| 25       | \( x_{162} = x_{25} \oplus x_{105} \oplus x_{124} \oplus x_{252} \oplus x_{247} \oplus x_{212} \oplus x_{268}x_{242} \oplus x_{272}x_{256} \oplus x_{238}x_{260} \) |
| 26       | \( x_{163} = x_{26} \oplus x_{106} \oplus x_{125} \oplus x_{253} \oplus x_{248} \oplus x_{213} \oplus x_{269}x_{243} \oplus x_{273}x_{257} \oplus x_{238}x_{261} \) |
| 27       | \( x_{164} = x_{27} \oplus x_{107} \oplus x_{126} \oplus x_{254} \oplus x_{249} \oplus x_{214} \oplus x_{270}x_{244} \oplus x_{274}x_{258} \oplus x_{240}x_{262} \) |
| 28       | \( x_{165} = x_{28} \oplus x_{108} \oplus x_{127} \oplus x_{255} \oplus x_{260} \oplus x_{215} \oplus x_{273}x_{245} \oplus x_{275}x_{259} \oplus x_{241}x_{263} \) |
| 29       | \( x_{166} = x_{29} \oplus x_{109} \oplus x_{128} \oplus x_{256} \oplus x_{251} \oplus x_{216} \oplus x_{272}x_{246} \oplus x_{276}x_{260} \oplus x_{242}x_{264} \) |
| 30       | \( x_{167} = x_{30} \oplus x_{110} \oplus x_{129} \oplus x_{257} \oplus x_{252} \oplus x_{217} \oplus x_{273}x_{247} \oplus x_{277}x_{261} \oplus x_{243}x_{265} \) |
| 31       | \( x_{168} = x_{31} \oplus x_{111} \oplus x_{130} \oplus x_{258} \oplus x_{253} \oplus x_{218} \oplus x_{274}x_{248} \oplus x_{278}x_{262} \oplus x_{244}x_{266} \) |
| 32       | \( x_{169} = x_{32} \oplus x_{112} \oplus x_{131} \oplus x_{259} \oplus x_{254} \oplus x_{219} \oplus x_{275}x_{249} \oplus x_{279}x_{263} \oplus x_{245}x_{267} \) |
| 33       | \( x_{170} = x_{33} \oplus x_{113} \oplus x_{132} \oplus x_{260} \oplus x_{255} \oplus x_{220} \oplus x_{276}x_{250} \oplus x_{280}x_{264} \oplus x_{246}x_{268} \) |
| 34       | \( x_{171} = x_{34} \oplus x_{114} \oplus x_{133} \oplus x_{261} \oplus x_{256} \oplus x_{221} \oplus x_{277}x_{251} \oplus x_{281}x_{265} \oplus x_{247}x_{269} \) |
Further, try to choose such internal state bits for recovery that are part of low degree terms in the keystream generation equation.

Let the set $\mathcal{A}$ represents the internal state bits decided for recovery. While adding a new element to the set $\mathcal{A}$, the feedback equations of cipher should be verified such that the internal state bits that appear in the feedback equations are not present in set $\mathcal{A}$. In case they are present in set $\mathcal{A}$, they should be recovered before employing such feedback equations for recovery.

If internal state bits decided for recovery appear as a linear term in the feedback function and are not recovered yet, then such internal state bits should be removed from the set $\mathcal{A}$.

If internal state bits decided for recovery appear as part of a low degree term of the feedback function and are not recovered yet, then we have to fix other internal state bits appearing in that low degree term of the feedback function with value zero. If that low degree is of degree more than two, then try to exclude the internal state bits from the set $\mathcal{A}$ unless such internal state bits are very promising to increase the number of recovered internal state bits.

To minimize the number of internal state bits fixed, choose such internal state bits for recovery, which always appear as a low order degree term in the keystream generation equations.

Following the above guidelines, the main challenge is handling the feedback functions that contain the internal state bits decided for recovery. In the case of Fibonacci configuration based cipher, the handling of such feedback functions is much easier than Galois configuration based cipher. The Galois configuration based cipher has more than one feedback function, which makes this heuristic approach challenging. This paper does not claim that the proposed heuristic procedure always outputs the optimal result. We only demonstrate that for particular ciphers, such as Espresso, in our case, our strategy provides values that lead to a slightly more efficient attack than that proposed by the designers.

4. Time-Memory-data tradeoff parameters

In the previous section, it is shown that the 35 internal state bits can be recovered using the first 35 bits of the keystream and fixing the specific 39 bits of internal state with all zero patterns. However, to perform those recoveries, we need to guess the remaining 182 internal state bits. To store all the pattern of 182 internal state bits, i.e. $2^{182}$, we employ the conditional Time-Memory tradeoff attack.

The Time-Memory tradeoff attack was initially proposed by Hellman for block ciphers [10] and later adopted by Babbage and Golic independently for stream ciphers [1, 8]. Biryukov and Shamir proposed the TMDTO attacks by combining the approaches of Hellman, Babbage, and Golic [2]. The TMDTO attack consists of two phases. One is the preprocessing phase, and the other is the online phase. In the preprocessing phase, we prepare the processing tables, and in the online phase, we use those tables to launch the attack. The size of memory and time required in the preprocessing phase are denoted by $M$ and $P$, respectively. The data and time required in the online phase are denoted by $D$ and $T$, respectively. However, the size of memory required in the preprocessing phase is almost the same as in the online phase. Thus, $M$ represents the required memory in both phases. In [2], a TMDTO attack on stream ciphers has been reported for which parameters of attack satisfy the relation $P = N/D$ and $TM^2D^2 = N^2$ for any $D^2 \leq T \leq N$. The size of memory and time required in the preprocessing phase are governed by the search space, i.e.
the size of the internal state of the cipher. To reduce the search space, some internal state bits are recovered directly by using the first $k$ bits of the keystream, which is related to the term “sampling resistance” introduced in [2] and provides more flexibility to choose the tradeoff parameters. These $k$ bits of keystream can be of any pattern, but it is required to fix $k$ bits of keystream with a particular pattern for preparing the preprocessing table. Further, conditional TMDTO appeared in the literature [13,14] in which some internal state bits are fixed to a particular value to recover more internal state bits, and ultimately search space is reduced again by some more factors.

In our case, the total size of the internal state of the cipher, total internal state bits recovered, and internal state bits fixed are 256, 35, and 39, respectively. Thus, the size of the reduced search space is $2^{256-35-39} = 2^{182}$. Let the reduced search space be denoted by the set $A^{(K)}$ where $K$ denotes the particular pattern of keystream bits which is used to recover 35 bits of the internal state and $|K| = 35$. Each element of the set $A^{(K)}$ consists of 182 bits. The conditional TMDTO operates only over the states from $A^{(K)}$. Now we apply TMDTO paradigm as discussed in [2] with $TM^2D^2 = N^2 = 2^{364}$ for any $D^2 \leq T \leq N$ and $P = N/D$. Accordingly, we can select $M = 2^{110}$, $D = 2^{46}$ and $T = 2^{72}$ with $P = 2^{146}$. By birthday paradox, it can be ensured that there will be negligible collisions in different rows of preprocessing tables for $t = 2^{36}$ and $m = 2^{110}$ where $t$ is the number of iterations in each row and $m$ is the number of rows in each preprocessing tables. The total number of tables required to cover all the $2^{182}$ states, which is denoted by the set $A^{(K)}$ is one. The above calculations are based on the idea of matrix stopping rule discussed in [2].

4.1. Preprocessing. In this section, we discuss the algorithm to construct the preprocessing table. First, we fixed a particular pattern of 35 bits and considered these fixed 35 bits as first 35 bits of the keystream, denoted as $K$ earlier. According to the above parameters, the number of rows in the table is $m = 2^{110}$, and the number of iterations required to construct any row is $t = 2^{36}$. To construct any rows of the table, we select an element randomly (say $s_i$) from the set $A^{(K)}$ and perform the following operation:

$$s_i = p_{i,1} \rightarrow p_{i,2} = f(p_{i,1})$$

$$\rightarrow \quad \ldots$$

$$\rightarrow p_{i,t-1} = f(p_{i,t-2})$$

$$\rightarrow p_{i,t} = f^*(p_{i,t-1}) = e_i.$$  \hspace{1cm} (26)

After the above operation, we store only initial $s_i$ and final $e_i$. The functions $f$ and $f^*$ are defined as follows:

- $f(x)$ maps $\{0,1\}^{182} \rightarrow \{0,1\}^{182}$ as follows:
  - First calculate 35 bits of the internal state using $x$, $K$ and fixed 39 bits of the internal state. Consequently, 256 bits of internal state are in hand and load the internal state of Espresso cipher with these 256 bits and generate 217 output bits according to Espresso algorithm.
  - Take the last 182 bits from the previous step and consider them as input for next iteration.

- $f^*(x)$ is another function which maps $\{0,1\}^{182} \rightarrow \{0,1\}^{256}$.
  - First calculate 35 bits of internal state using $x$, $K$ and fixed 39 bits of internal state. Consequently, 256 bits of the internal state are in hand
and load the internal state of Espresso cipher with these 256 bits and generate 256 output bits according to Espresso algorithm.

**Algorithm of Construction of the Preprocessing Table**

- **Input:** A fixed pattern of first 35 bits of keystream denoted by \( K \), \( m = 2^{110} \) and \( t = 2^{36} \).

- **Pre-Processing Steps**
  1. Set the counter: \( i = 0 \).
  2. Increment: \( i \rightarrow i + 1 \); If \( i + 1 > m \), stop the pre-processing.
  3. Randomly select previously unused \( s_i \in \mathcal{A}(K) \).
  4. Set \( s_i = p_{i,1} \).
  5. Employing equation (26) generate \( e_i \).
  6. Store the pair. 

\[(s_i, e_i) = (\text{internal state, output segment}),\]

and go to the Step 2.

- **Output:** Two-column table \( M \) with \( 2^{110} \) rows which are used in the online phase for recovery of the internal state of Espresso cipher.

4.1.1. **Online Phase.** Now we demonstrate an algorithm that recovers the internal state of Espresso cipher by employing the table constructed in the preprocessing phase and the TMDTO approach.

**Algorithm for Internal State Recovery**

- **Input:**
  - A fixed pattern of first 35 bits of keystream denoted by \( K \) which is used in the preprocessing phase, the table \( M \) constructed in the preprocessing phase and the number of iterations required to construct any row of the table \( M \) which is \( t = 2^{36} \).
  - Samples consisting of different 384-bit segments of the keystream generated by Espresso with the first 35 bits identical to \( K \).

- **Processing Steps:** For each sample segment do the following
  1. Check that the first 256 bits of considered sample appear in the second column of preprocessing table \( M \). If the result of check operation is positive and considered 256 bits of the sample exists in the \( i \)-th row of table \( M \) then evaluate the corresponding candidate for the internal state employing the \( s_i \) as defined in equation (26) and perform Step 2; Otherwise, perform Step 3.
  2. Load the internal state of cipher with the considered candidate and generate 384 bits and match them with the given sample segment:
    - If the result of the match is positive, then accept the candidate internal state as the true one, and go to the Output (a);
  3. Select the first 182 bits of the considered sample segment after skipping the first 35 bits of sample segment. Now the considered 182 bits are an argument of both function \( f(\cdot) \) and \( f^*(\cdot) \) as defined in equation (26) and do the following:
    (a) Set the counter: \( i = 0 \).
    (b) Increment: \( i \rightarrow i + 1 \); If \( i + 1 > t \), go to Step 4.
    (c) Apply function \( f^* \) and check that the 256-bit output of function \( f^* \) matches to a certain element of the second column of any row of table \( M \). If a match occurs to the \( i \)-th row of the second column of table \( M \)
then consider \( s_i \), perform iterative chaining using function \( f(\cdot) \) until the considered sample segment appears and accept the argument of \( f(\cdot) \) which has generated the sample segment as the candidate for the internal state, and perform the Step 2;

(d) Apply function \( f \), generate 182-bit output according to the definition of \( f \), and go to Step 3(b).

4. Otherwise, consider the next sample. If all segments are exhausted, go to the Output (b).

- **Output**
  - (a) The recovered internal state of Espresso;
  - (b) A flag indicating the failure of the algorithm.

4.1.2. *The Complexities of TMDTO parameters.* During the preparation of the preprocessing table in the last section, we have fixed the first 35 bits of keystream to a particular pattern, denoted by \( K \), as well as 39 bits of internal state with 0 pattern. Thus, the probability of occurrence of this event in online phase is \( 2^{-(35+39)} = 2^{-74} \). By following the TMDTO curve \( TM^2D^2 = N^2 \) for any \( D^2 \leq T \leq N \) and \( P = N/D \), we have selected earlier \( M = 2^{110} \), \( D = 2^{36} \) and \( T = 2^{72} \) with \( P = 2^{146} \).

So, the required data complexity is \( D' = \frac{1}{2^{\delta}} \cdot D = 2^{74-\delta} \cdot 2^{36} = 2^{110} \) and total time required in online phase is \( T' = \frac{1}{2^{\delta}} \cdot T = 2^{39+2\delta} \cdot 2^{72} = 2^{111} \). Accordingly, the cryptanalytic complexities can be summarized as follows:

- **Required Sample for Cryptanalysis:** \( D' = 2^{110} \).
- **A Time Complexity of Processing:** \( T' = 2^{111} \).
- **Space Complexity of Pre-Processing & Processing:** \( M = 2^{110} \).
- **A Time Complexity of Pre-Processing:** \( P = 2^{146} \).

In a similar manner, other possible tradeoffs can be derived by choosing different values of \( D \). We set \( D = 2^\delta \) and \( T = D^2 \) to derive other tradeoffs. By following the TMDTO curve \( TM^2D^2 = N^2 \), we obtain \( D' = 2^{74+\delta} \), \( T' = 2^{39+2\delta} \), \( M = 2^{182-2\delta} \) and \( P = 2^{182-3\delta} \). Table 5 lists some of possible tradeoffs for different values of \( \delta \).

| \( \delta \) | \( D' \)   | \( T' \)   | \( M \)   | \( P \)   |
|-----------|------------|------------|------------|------------|
| 30        | \( 2^{104} \) | \( 2^{109} \) | \( 2^{122} \) | \( 2^{152} \) |
| 32        | \( 2^{106} \) | \( 2^{103} \) | \( 2^{118} \) | \( 2^{150} \) |
| 34        | \( 2^{108} \) | \( 2^{107} \) | \( 2^{114} \) | \( 2^{148} \) |

5. **Conclusion**

This paper describes a strategy to compute sampling resistance of stream ciphers that employ feedback shift registers in Galois configuration. A set of equations using a heuristic approach have been derived, which are employed in calculating the conditional sampling resistance of the stream cipher Espresso. Further, it has
been proved that the conditional sampling resistance of the cipher is $2^{-35}$, and
the TMDTO attack has been mounted on the cipher by utilizing the calculated
sampling resistance. The TMDTO attacks on a stream cipher can be employed
using two different techniques. The first technique is concerned with such TMDTO
attacks in which one-way functions used to construct preprocessing tables are based
on the size of internal states [2]. Another technique is concerned with such TMDTO
attacks in which the one-way functions used to construct preprocessing tables are
based on the total size of key and initialization vector (IV) [11]. The parameters for
TMDTO attack for key recovery of Espresso provided by its designers [7] employing
the technique mentioned in [11] are $P = 2^{168}$, $D' = 2^{56}$ and $T' = M = 2^{112}$. In this
paper, we have employed the approach mentioned in [2,4] and mounted conditional
TMDTO attack to recover the internal state of Espresso. The preprocessing tables
of this TMDTO attack are based on the one-way functions which are defined over the
internal state rather than key and IV. The attack complexities of TMDTO attack
on Espresso cipher reported in this paper for $D = 2^{36}$ are slightly better than the
attack complexities reported in [7], except data complexity. In our opinion, research
in this direction may lead to the development of a strategy to compute the minimum
value of sampling resistance which may become a useful tool to evaluate feedback
shift register based stream ciphers.

ACKNOWLEDGMENTS

The author acknowledges and thanks to the anonymous reviewers for the detailed
technical and editorial comments that significantly improved the quality of the paper.

REFERENCES

[1] S. Babbage, A space/time tradeoff in exhaustive search attacks on stream ciphers, European
Convention on Security and Detection, 408 (1995).
[2] A. Biryukov and A. Shamir, Cryptanalytic time/memory/data tradeoffs for stream ciphers,
ASIACRYPT 2000, Lecture Notes in Computer Science, 1976 (2000), 1–13.
[3] A. Biryukov, A. Shamir and D. Wagner, Real time cryptanalysis of A5/1 on a PC, Fast
Software Encryption 2000, Lecture Notes in Computer Science, 1978 (2001), 37–44.
[4] T. E. Bjørstad, Cryptanalysis of grain using time/memory/data tradeoffs, (2008). Available
from: http://www.ecrypt.eu.org/stream/grain3.html.
[5] C. Cannière and B. Preneel, Trivium, new stream cipher designs: The eSTREAM finalists,
Lecture Notes in Computer Science, 4986 (2008), 244–266.
[6] E. Dubrova, A transformation from the Fibonacci to the Galois NLFSRs, IEEE Transactions
on Information Theory, 55 (2009), 5263–5271.
[7] E. Dubrova and M. Hell, A stream cipher for 5G wireless communications systems, Cryptog-
raphy and Communications, 9 (2017), 273–289.
[8] J. Golić, Cryptanalysis of alleged A5 stream cipher, EUROCRYPT 1997, Lecture Notes in
Computer Science, 1233 (1997), 239–255.
[9] M. Hell, T. Johansson, A. Maximov and W. Meier, The Grain family of stream ciphers, new
stream cipher designs: The eSTREAM finalists, Lecture Notes in Computer Science, 4986
(2008), 17–190.
[10] M. E. Hellman, A cryptanalytic time-memory trade-off, IEEE Transactions on Information
Theory, 26 (1980), 401–406.
[11] J. Hong and P. Sarkar, New applications of time memory data tradeoffs, ASIACRYPT 2005,
Lecture Notes in Computer Science, Springer, Berlin, 3788 (2005), 353–372.
[12] S. Maitra, N. Sinha, A. Siddhanti, R. Anand and S. Gangopadhyay, A TMDTO attack against
Lizard, IEEE Transactions on Computers, 67 (2018), 733–739.
[13] M. J. Mihaljević, S. Gangopadhyay, G. Paul and H. Imai, Internal state recovery of Grain-v1
employing normality order of the filter function, JET Information Security, 6 (2012), 55–64.
[14] M. J. Mihaljević, S. Gangopadhyay, G. Paul and H. Imai, Generic cryptographic weakness of $k$-normal Boolean functions in certain stream ciphers and cryptanalysis of Grain-128, *Periodica Mathematica Hungarica*, 65 (2012), 205–227.

Received November 2019; revised January 2020.

*E-mail address:* nishantsinha.iitr@gmail.com