Multiset Collision Attacks on Reduced-Round SNOW 3G and SNOW 3G$^{\oplus}$

Alex Biryukov, Deike Priemuth-Schmid, and Bin Zhang

University of Luxembourg
{alex.biryukov,deike.priemuth-schmid,bin.zhang}@uni.lu

Abstract. The stream cipher SNOW 3G designed in 2006 by ETSI/SAGE is a base algorithm for the second set of 3GPP confidentiality and integrity algorithms. In this paper we study the resynchronization mechanism of SNOW 3G and of a similar cipher SNOW 3G$^{\oplus}$ using multiset collision attacks. For SNOW 3G we show a simple 13-round multiset distinguisher with complexity of 2^8 steps. We show full key recovery chosen IV resynchronization attacks for up to 18 out of 33 initialization rounds of SNOW3G$^{\oplus}$ with a complexity of 2^{57} to generate the data and 2^{53} steps of analysis.

Keywords: Stream ciphers, SNOW 3G, Resynchronization attack.

1 Introduction

The SNOW 3G stream cipher is the core of the 3GPP confidentiality and integrity algorithms UEA2 and UIA2, published in 2006 by the 3GPP Task Force [5]. Compared to its predecessor, SNOW 2.0 [4], SNOW 3G adopts a finite state machine (FSM) of three 32-bit words and 2 S-Boxes to increase the resistance against algebraic attacks by Billet and Gilbert [2]. Full evaluation of the design by the consortium is not public, but a survey of this evaluation is given in [6]. SNOW 3G$^{\oplus}$ (in which the two modular additions are replaced by xors) is also defined and evaluated in this document. The designers and external reviewers show that SNOW 3G has remarkable resistance against linear distinguishing attacks [7,8], while SNOW 3G$^{\oplus}$ offers much better resistance against algebraic attacks.

In this paper we analyze the resynchronization mechanizm of SNOW 3G and SNOW 3G$^{\oplus}$ using multiset collision attacks. This technique has proved itself useful against AES [3] but to the best of our knowledge has not been used yet for the analysis of the key-IV setup of stream ciphers. It seems natural to apply this technique to SNOW 3G since its finite state machine (FSM) is essentially a 96-bit AES like cipher in which the LFSR plays a role of a key-schedule. This picture is complicated by the fact that there is a feedback from the FSM to the LFSR during the setup phase (a feature never present in block ciphers) and that the attacker sees only 32-bits of output at a time, while the internal state keeps changing constantly.

J. Zhou and M. Yung (Eds.): ACNS 2010, LNCS 6123, pp. 139–153, 2010.

We start by showing a very efficient multiset distinguisher for 13-round SNOW 3G with complexity of 2^8 steps. We then switch to the analysis of SNOW 3G$^{\oplus}$ which is a very good model for the main features of SNOW 3G, since its analysis is not blurred by the presence of carries. We have found an attack on up to 18-rounds (out of 33) with a complexity of 2^{57} to generate the data and 2^{53} steps of analysis. In this attack for the first 10 rounds the multiset propagates for free since we put it in the most significant byte of the IV word IV_0. It enters the FSM at the 11th round. Strong cancellations due to balanced properties of multisets stop to be useful after 15 rounds and we have to resort to *multiset collision* techniques which can allow us to go three more rounds deeper. Multisets still help us to cancel out the keystream words out of the keystream equation, which are an obstacle for a simple differential analysis at this depth. This attack is very technical and is more involved than attacks of similar type on block ciphers. We have experimentally verified the crucial parts of our attacks.

This paper is organized as follows. We give a description of SNOW 3G and SNOW 3G$^{\oplus}$ in Section 2. The multiset collision chosen IV attacks on round-reduced SNOW3G and SNOW 3G$^{\oplus}$ are presented in Section 3. Finally, some conclusions are given in Section 4.

2 Description of SNOW 3G and SNOW 3G$^{\oplus}$

The SNOW 3G stream cipher uses a 128-bit key and a 128-bit IV, considered as four 32-bit words vectors. It consists of a linear feedback shift register (LFSR) of 16 32-bit words and a finite state machine (FSM) with three 32-bit words, shown in Figure 1. Here '$\oplus$' denotes the bit-wise xor and '$\boxplus$' denotes the addition modulo 2^{32}. The feedback word of the LFSR is recursively computed as

$$s_{15}^{t+1} = \alpha^{-1} \cdot s_{11}^t \oplus s_2^t \oplus \alpha \cdot s_0^t,$$

where α is the root of the $GF(2^8)[x]$ polynomial $x^4+\beta^{23}x^3+\beta^{245}x^2+\beta^{48}x+\beta^{239}$ with β being the root of the $GF(2)[x]$ polynomial $x^8 + x^7 + x^5 + x^3 + 1$. The FSM has two input word s_5^t and s_{15}^t from the LFSR and is updated as follows.

$$R_3^t = S_2(R_2^{t-1}), \quad R_2^t = S_1(R_1^{t-1}), \quad R_1^t = R_2^{t-1} \boxplus (R_3^{t-1} \oplus s_5^{t-1}),$$

and output $F^t = (s_{15}^t \boxplus R_1^t) \oplus R_2^t$, where S_1 and S_2 are 32-bit to 32-bit S-boxes defined as compositions of 4 parallel applications of two 8-bit to 8-bit small S-boxes, S_R and S_Q, with a linear diffusion layer respectively. Here S_R is the well known AES S-box and S_Q is defined as $S_Q(x) = x \oplus x^9 \oplus x^{13} \oplus x^{15} \oplus x^{33} \oplus x^{41} \oplus x^{45} \oplus x^{47} \oplus x^{49} \oplus 0x25$ for $x \in GF(2^8)$ defined by $x^8 + x^6 + x^5 + x^3 + 1$. If we decompose a 32-bit word B into four bytes $B = B^0 \| B^1 \| B^2 \| B^3$ with B^0 being the most and B^3 the least significant bytes, then

$$S_i(B) = MC_i \cdot \begin{pmatrix} S_R(B^0) \\ S_R(B^1) \\ S_R(B^2) \\ S_R(B^3) \end{pmatrix} = \begin{pmatrix} 2\,1\,1\,3 \\ 3\,2\,1\,1 \\ 1\,3\,2\,1 \\ 1\,1\,3\,2 \end{pmatrix}_i \cdot \begin{pmatrix} S_R(B^0) \\ S_R(B^1) \\ S_R(B^2) \\ S_R(B^3) \end{pmatrix}, \ (i = 1, 2)$$

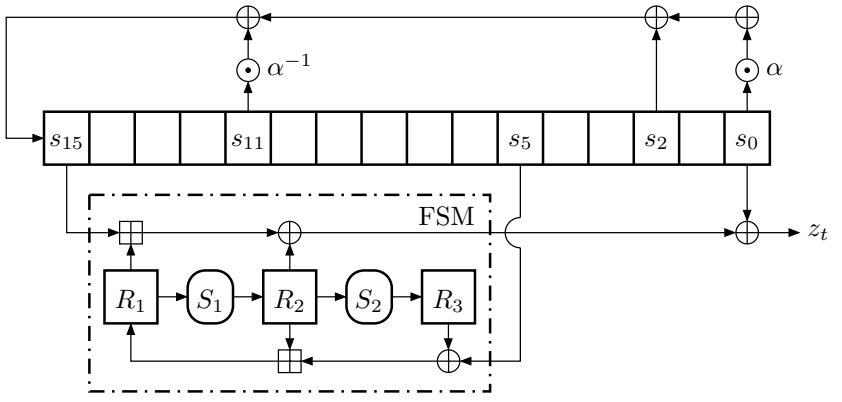

Fig. 1. Keystream generation of SNOW 3G

where MC_1 is the AES mix-column for S_1 over $GF(2^8)$ defined by $x^8 + x^4 + x^3 + x + 1$ and MC_2 is the similar operation for S_2 over $GF(2^8)$ defined by $x^8 + x^6 + x^5 + x^3 + 1$.

SNOW 3G is initialized with the key $K = (k_0, k_1, k_2, k_3)$ and the $IV = (IV_0, IV_1, IV_2, IV_3)$ as follows. Let $\mathbf{1}$ be the all-one word, first load the LFSR as follows.

$$\begin{array}{llll} s_{15} = k_3 \oplus IV_0 & s_{14} = k_2 & s_{13} = k_1 & s_{12} = k_0 \oplus IV_1 \\ s_{11} = k_3 \oplus \mathbf{1} & s_{10} = k_2 \oplus \mathbf{1} \oplus IV_2 & s_9 = k_1 \oplus \mathbf{1} \oplus IV_3 & s_8 = k_0 \oplus \mathbf{1} \\ s_7 = k_3 & s_6 = k_2 & s_5 = k_1 & s_4 = k_0 \\ s_3 = k_3 \oplus \mathbf{1} & s_2 = k_2 \oplus \mathbf{1} & s_1 = k_1 \oplus \mathbf{1} & s_0 = k_0 \oplus \mathbf{1} \end{array} .$$

The FSM is initialized with $R_1 = R_2 = R_3 = 0$. Then run the cipher 32 times with the FSM output F xored to the feedback of the LFSR and no keystream generated. After this, the cipher is switched into the keystream generation mode, but the first keystream word is discarded. Hence, there are 33 initialization rounds. The keystream word generated at clock t is

$$\text{SNOW 3G:} \qquad z^t = s_0^t \oplus F^t = (s_{15}^t \boxplus R_1^t) \oplus R_2^t \oplus s_0^t \qquad (1)$$

$$\text{SNOW 3G}^{\oplus}\text{:} \qquad z^t = s_0^t \oplus F^t = s_{15}^t \oplus R_1^t \oplus R_2^t \oplus s_0^t \qquad (2)$$

If we replace the two modulo additions in SNOW 3G by xors, we get SNOW 3G$^{\oplus}$.

3 Chosen IV Attacks on Reduced Round SNOW 3G and SNOW 3G$^{\oplus}$

In this section, we evaluate the security margin of SNOW 3G and SNOW 3G$^{\oplus}$ against chosen IV attacks. Our results are listed in Table 1.

Table 1. Our results on SNOW 3G and SNOW 3G$^{\oplus}$

Cipher	Round	Data	Time	Type
SNOW 3G	13	2^8	2^8	distinguisher
SNOW 3G$^{\oplus}$	14	2^8	2^8	distinguisher
SNOW 3G$^{\oplus}$	14	$2^{12.1}$	2^{27}	full key recovery
SNOW 3G$^{\oplus}$	15	$2^{32.1}$	$2^{32.4}$	partial state recovery
SNOW 3G$^{\oplus}$	18	2^{57}	2^{53}	full key recovery

3.1 Distinguishing Attack on 13-Round SNOW 3G

We first look at SNOW 3G with 13-round initializations. For each secret key K, we randomly choose an IV and make a multiset at the most significant byte IV_0^0 of the most significant word IV_0 such that it takes all the byte values in $[0, 255]$ exactly once. From the key/IV loading of SNOW 3G, we known that the multiset difference is introduced in the most significant byte of s_{15}. Now we trace the multiset difference propagation in the 19 registers during the 13 rounds of initialization, which is shown in Table 2. The differences at round i are the differences at the end of the corresponding round.

Here we actually have 256 IVs associated with the same key. Denote the first keystream word generated by (K, IV) when $IV_0^0 = i$ by $z_{i,0}$ and denote the corresponding content in the j-th LFSR cell by $s_{i,j}$, then we have

$$\bigoplus_{i=0}^{255} z_{i,0} = \bigoplus_{i=0}^{255}(s_{i,0} \oplus R_{i,2}) \oplus \bigoplus_{i=0}^{255}(s_{i,15} \boxplus R_{i,1}) = \bigoplus_{i=0}^{255}(s_{i,15} \boxplus R_{i,1}).$$

From Table 2, the least significant bit is always 0. To show that this property holds for the other 7 bits in the least significant byte, it suffices to note that the least significant bytes of R_1 forms an permutation set, while the least significant bytes of $s_{i,15}$ are the same, so by lemma 2 in [1], the least significant byte of $\bigoplus_{i=0}^{255} z_{i,0}$ is always 0. In experiments, we randomly choose 2^6 IVs to check this property. For each chosen IV, we make a multiset attack as above and calculate $\bigoplus_{i=0}^{255} z_{i,0}$. We found that the least significant byte of this sum is always 0. This gives a very simple distinguishing attack of complexity 2^8 IV's and key-stream words for 13-round SNOW 3G. We expect that this attack can be extended into a key recovery attack on 14-round SNOW 3G, but we preferred to concentrate on breaking more rounds of SNOW 3G$^{\oplus}$ instead.

3.2 Distinguishing Attack on 14-Round SNOW 3G$^{\oplus}$

The above distinguisher can be extended by several rounds in SNOW 3G$^{\oplus}$. For each secret key K, we also randomly choose an IV and make a multiset at the most significant byte IV_0^0 of the most significant word IV_0 such that it takes all the byte values in $[0, 255]$ exactly once. The multiset difference propagation is formally derived in Table 6 in Appendix A, where $\Delta_i = i$ denotes the difference in IV_0^0 for $i = 0, \cdots, 255$. From that table, we can see that until round 11, the

Table 2. Multiset sum propagation in 13-round initialization of SNOW 3G. (? indicates that the sum in this byte takes some random value and 0 means that the corresponding sum is 0).

	s_{15}	s_{14}	s_{13}	s_{12}	s_{11}	s_{10}	s_9	s_8	$s_{j:\ (0\le j\le 7)}$	R_1	$R_{i:\ (i=2,3)}$
0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
5	0	0	0	0	0	0	0	0	0	0	0
6	?000	0	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
7	?000	?000	0	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
8	?000	?000	?000	0	⋮	⋮	⋮	⋮	⋮	⋮	⋮
9	?000	?000	?000	?000	0	⋮	⋮	⋮	⋮	⋮	⋮
10	?000	?000	?000	?000	?000	0	⋮	⋮	⋮	⋮	⋮
11	??00	?000	?000	?000	?000	?000	0	⋮	⋮	⋮	⋮
12	??00	??00	?000	?000	?000	?000	?000	0	⋮	0	⋮
13	0?00	??00	??00	?000	?000	?000	?000	?000	0	???0	0

difference Δ_i will not affect the memory registers R_i $(i = 1, 2, 3)$. Hence, at the end of round 10, the contents in R_i $(i = 1, 2, 3)$ are three unknown constants not depending on IV_0 and the difference Δ_i. Let the unknown constant in R_i be c_i for $i = 1, 2, 3$. Table 3 shows the contents evolution process in the three memory registers. We have the following theorem:

Theorem 1. *If there are* 14 *initialization rounds in SNOW 3G$^{\oplus}$ and the multiset is taken at IV_0^0, then $\bigoplus_{i=0}^{255} z_{i,0} = (2a, 3a, a, a)$ with $a \in [0, 255]$.*

Proof. From the line 14 of Table 3 and the keystream equation (2), we have

$$\bigoplus_{i=0}^{255} z_{i,0} = \bigoplus_{i=0}^{255}(s_{i,0} \oplus R_{i,2} \oplus s_{i,15} \oplus R_{i,1}) = \bigoplus_{i=0}^{255}(R_{i,1} \oplus R_{i,2})$$
$$= \bigoplus_{i=0}^{255}(B_{i,13} \oplus C_{i,13} \oplus s_{18} \oplus \Delta_i) \oplus \bigoplus_{i=0}^{255} MC_1[S_1(A_{13} \oplus \Delta_i)].$$

Note that there is only one active byte in IV_0, so we have $\bigoplus_{i=0}^{255} B_{i,13} = 0$, $\bigoplus_{i=0}^{255} C_{i,13} = 0$ by Lemma 2 in [1]. Since $\forall i$, $s_{i,18} = s_{18}$, we have $\bigoplus_{i=0}^{255} s_{18} = 0$. Thus, $\bigoplus_{i=0}^{255} z_{i,0} = \bigoplus_{i=0}^{255} MC_1[S_1(A_{13} \oplus \Delta_i)] = MC_1 \cdot \bigoplus_{i=0}^{255}[S_1(A_{13} \oplus \Delta_i)]$. Expanding $\bigoplus_{i=0}^{255}[S_1(A_{i,13} \oplus \Delta_i)]$, we have

Table 3. The contents evolution process of the memory registers during the $10-17$ initializations round of SNOW 3G$^{\oplus}$. (A_i, B_i and C_i are the contents in R_1, R_2 and R_3 for $i \geq 11$ with A_i being the content when $\Delta_i = 0$. * denotes permutation property of the values, c denotes constant property, ? denotes property to be determined, b denotes balanced property).

	R_1	R_2	R_3
10	c_1	c_2	c_3
11	$c_2 \oplus c_3 \oplus k_3 \oplus IV_0 \oplus \Delta_i$	$MC_1[S_1(c_1)]$	$MC_2[S_2(c_2)]$
	$(*, c, c, c)$	(c, c, c, c)	(c, c, c, c)
12	$B_{11} \oplus C_{11} \oplus s_{16} \oplus \Delta_i$	$MC_1[S_1(A_{11} \oplus \Delta_i)]$	$MC_2[S_2(B_{11})]$
	$(*, c, c, c)$	$(*, *, *, *)$	(c, c, c, c)
13	$B_{12} \oplus C_{12} \oplus s_{17} \oplus \Delta_i$	$MC_1[S_1(A_{12} \oplus \Delta_i)]$	$MC_2[S_2(B_{12})]$
	$(b, *, *, *)$	$(*, *, *, *)$	(b, b, b, b)
14	$B_{13} \oplus C_{13} \oplus s_{18} \oplus \Delta_i$	$MC_1[S_1(A_{13} \oplus \Delta_i)]$	$MC_2[S_2(B_{13})]$
	(b, b, b, b)	$(?, ?, ?, ?)$	(b, b, b, b)
15	$B_{14} \oplus C_{14} \oplus s_{19} \oplus \Delta_i$	$MC_1[S_1(A_{14} \oplus \Delta_i)]$	$MC_2[S_2(B_{14})]$
16	$B_{15} \oplus C_{15} \oplus s_{20} \oplus \Delta_i \oplus \alpha^{-1}\Delta_i$	$MC_1[S_1(A_{15} \oplus \Delta_i)]$	$MC_2[S_2(B_{15})]$
17	$B_{16} \oplus C_{16} \oplus s_{21} \oplus \Delta_i$	$MC_1[S_1(A_{16} \oplus \Delta_i \oplus \alpha^{-1}\Delta_i)]$	$MC_2[S_2(B_{16})]$

$$\bigoplus_{i=0}^{255} S_1(A_{13} \oplus \Delta_i) = \bigoplus_{i=0}^{255} S_1\big(s_{17} \oplus C_{12} \oplus \Delta_i \oplus MC_1[S_1(A_{11} \oplus \Delta_i)]\big), \quad (3)$$

where $A_{11} = c_2 \oplus c_3 \oplus k_3 \oplus IV_0$. From (3) and Table 4, we can see that s_{17}, C_{12} and A_{11} do not dependent on Δ_i. Let $A_{11} = A_{11}^0 \| A_{11}^1 \| A_{11}^2 \| A_{11}^3$ and $s_{17} \oplus C_{12} = m^0 \| m^1 \| m^2 \| m^3$, then we have the byte equations:

$$\bigoplus_{i=0}^{255} S_R[2S_R(A_{11}^0 \oplus \Delta_i) \oplus S_R(A_{11}^1) \oplus S_R(A_{11}^2) \oplus 3S_R(A_{11}^3) \oplus m^0 \oplus \Delta_i] = a \quad (4)$$

$$\bigoplus_{i=0}^{255} S_R[3S_R(A_{11}^0 \oplus \Delta_i) \oplus 2S_R(A_{11}^1) \oplus S_R(A_{11}^2) \oplus S_R(A_{11}^3) \oplus m^1] = 0 \quad (5)$$

$$\bigoplus_{i=0}^{255} S_R[S_R(A_{11}^0 \oplus \Delta_i) \oplus 3S_R(A_{11}^1) \oplus 2S_R(A_{11}^2) \oplus S_R(A_{11}^3) \oplus m^2] = 0 \quad (6)$$

$$\bigoplus_{i=0}^{255} S_R[S_R(A_{11}^0 \oplus \Delta_i) \oplus S_R(A_{11}^1) \oplus 3S_R(A_{11}^2) \oplus 2S_R(A_{11}^3) \oplus m^3] = 0 \quad (7)$$

It is easy to see that (5), (6) and (7) equal to 0 for any value of its inputs, while the value of (4) is dependent on the input value. Let $(3) = a$, by passing the vector $(a, 0, 0, 0)$ through the MC_1 we finish the proof. □

Theorem 1 shows that the four sub-bytes of $\bigoplus_{i=0}^{255} z_{i,0}$ are correlated. In experiments, we randomly choose 2^6 IVs to verify it. We found that Theorem 1 holds all the time. This property is a distinguisher with a complexity of 2^8 steps, given 1 keystream word for each IV.

3.3 Key Recovery Attack on 14-Round SNOW 3G$^{\oplus}$

The above distinguisher can be converted into a key recovery attack on 14-round SNOW 3G$^{\oplus}$. It works as follows. From Theorem 1, we have

$$\bigoplus_{i=0}^{255} S_R[2S_R(A_{11}^0 \oplus \Delta_i) \oplus S_R(A_{11}^1) \oplus y(0) \oplus \Delta_i] = \bigoplus_{i=0}^{255} z_{i,0}^3, \tag{8}$$

where $y(0) = S_R(A_{11}^2) \oplus 3S_R(A_{11}^3) \oplus m^0$. To solve it, we randomly choose two other IVs, IV' and IV'', such that

1. $IV_r' = IV_r'' = IV_r$ for $r = 1, 2, 3$.
2. $IV_0'^r = IV_0^r$ for $r = 0, 2, 3$.
3. $IV_0'^1 = IV_0^1 \oplus \beta_1$.
4. $IV_0''^r = IV_0^r$ for $r = 0, 2, 3$.
5. $IV_0''^1 = IV_0^1 \oplus \beta_2$.

with $\beta_i \in GF(2^8)$ for $i = 1, 2$. For IV' and IV'', we also make a multiset at the corresponding most significant byte. Our observation is that for such chosen IVs, we can derive similar equations to (8) due to the linearity of $A_{11} = c_2 \oplus c_3 \oplus k_3 \oplus IV_0$, $A_{11}' = c_2 \oplus c_3 \oplus k_3 \oplus IV_0'$ and $A_{11}'' = c_2 \oplus c_3 \oplus k_3 \oplus IV_0''$:

$$\bigoplus_{i=0}^{255} S_R[2S_R(A_{11}^0 \oplus \Delta_i) \oplus S_R(A_{11}^1 \oplus \beta_1) \oplus y(\beta_1) \oplus \Delta_i] = \bigoplus_{i=0}^{255} z_{i,0}'^3 \tag{9}$$

$$\bigoplus_{i=0}^{255} S_R[2S_R(A_{11}^0 \oplus \Delta_i) \oplus S_R(A_{11}^1 \oplus \beta_2) \oplus y(\beta_2) \oplus \Delta_i] = \bigoplus_{i=0}^{255} z_{i,0}''^3, \tag{10}$$

where $y(\beta_1) = y(\beta_2) = y(0)$ according to the conditions 2 and 4. From (8) − (10), we can derive A_{11}^1 with 2^{24} steps. It is interesting to note that we cannot restore A_{11}^0 and $y(0)$ together with A_{11}^1 from (8) − (10). The reason is that (8) − (10) cannot be regraded as random equations, which is supported by extensive experiments. Note that the information we recovered is the byte where we introduce the difference β_i. In order to determine other bytes of A_{11}, we just shift the byte position where the difference β_i is introduced to $A_{11}^r (r = 2, 3)$. Thus, we will get equations looking like

$$\bigoplus_{i=0}^{255} S_R[2S_R(A_{11}^0 \oplus \Delta_i) \oplus S_R(A_{11}^1) \oplus S_R(A_{11}^2 \oplus \gamma_j) \oplus 3S_R(A_{11}^3) \oplus m^0 \oplus \Delta_i] = \bigoplus_{i=0}^{255} z_{i,0}^3$$

$$\bigoplus_{i=0}^{255} S_R[2S_R(A_{11}^0 \oplus \Delta_i) \oplus S_R(A_{11}^1) \oplus S_R(A_{11}^2) \oplus 3S_R(A_{11}^3 \oplus \delta_j) \oplus m^0 \oplus \Delta_i] = \bigoplus_{i=0}^{255} z_{i,0}^3$$

for randomly chosen $\gamma_j, \delta_j \in GF(2^8)$ $(j = 1, 2, 3)$, from which we can recover A^2_{11} and A^3_{11}. We can determine A^0_{11} by shifting the multiset position to another byte and introduce the byte differences at A^0_{11}. Thus, we will get

$$\bigoplus_{i=0}^{255} S_R[3S_R(A^0_{11} \oplus \xi_j) \oplus 2S_R(A^1_{11} \oplus \Delta_i) \oplus S_R(A^2_{11}) \oplus S_R(A^3_{11}) \oplus m^1 \oplus \Delta_i] = \bigoplus_{i=0}^{255} z^0_{i,0}$$

for randomly chosen $\xi_j \in GF(2^8)(j=1,2,3)$. In this case, $\bigoplus_{i=0}^{255} z_{i,0}=(a, 2a, 3a, a)$ with $a \in GF(2^8)$.

Next, we can restore $s_{17} \oplus C_{12}$ by substituting A^i_{11} $(i = 0, 1, 2, 3)$ into the solution set of $(8)-(10)$, identifying the corresponding variable and determining m^i. To make a full key recovery, we need to look at the second keystream word and derive the following byte equations:

$$\bigoplus_{i=0}^{255} S_R[s^0_{18} \oplus \Delta_i \oplus \underbrace{2f_0 \oplus f_1 \oplus f_2 \oplus 3f_3}_{MC_2} \oplus 2S_R(A^0_{12} \oplus \Delta_i) \tag{11}$$
$$\oplus S_R(A^1_{12} \oplus \alpha_j) \oplus S_R(A^2_{12}) \oplus 3S_R(A^3_{12})] = \bigoplus_{i=0}^{255} z^0_{i,1} \oplus \bigoplus_{i=0}^{255} z^3_{i,0}$$

$$\bigoplus_{i=0}^{255} S_R[s^1_{18} \oplus \alpha_j \oplus \underbrace{3f_0 \oplus 2f_1 \oplus f_2 \oplus f_3}_{MC_2} \oplus 3S_R(A^0_{12} \oplus \Delta_i) \tag{12}$$
$$\oplus 2S_R(A^1_{12} \oplus \alpha_j) \oplus S_R(A^2_{12}) \oplus S_R(A^3_{12})] = \bigoplus_{i=0}^{255} z^1_{i,1}$$

$$\bigoplus_{i=0}^{255} S_R[s^2_{18} \oplus \underbrace{f_0 \oplus 3f_1 \oplus 2f_2 \oplus f_3}_{MC_2} \oplus S_R(A^0_{12} \oplus \Delta_i) \tag{13}$$
$$\oplus 3S_R(A^1_{12} \oplus \alpha_j) \oplus 2S_R(A^2_{12}) \oplus S_R(A^3_{12})] = \bigoplus_{i=0}^{255} z^2_{i,1}$$

$$\bigoplus_{i=0}^{255} S_R[s^3_{18} \oplus \underbrace{f_0 \oplus f_1 \oplus 3f_2 \oplus 2f_3}_{MC_2} \oplus S_R(A^0_{12} \oplus \Delta_i) \tag{14}$$
$$\oplus S_R(A^1_{12} \oplus \alpha_j) \oplus 3S_R(A^2_{12}) \oplus 2S_R(A^3_{12})] = \bigoplus_{i=0}^{255} z^3_{i,1},$$

where

$$f_0 = S_Q(2S_R(A^0_{11} \oplus \Delta_i) \oplus S_R(A^1_{11} \oplus \alpha_j) \oplus S_R(A^2_{11}) \oplus 3S_R(A^3_{11})) \tag{15}$$
$$f_1 = S_Q(3S_R(A^0_{11} \oplus \Delta_i) \oplus 2S_R(A^1_{11} \oplus \alpha_j) \oplus S_R(A^2_{11}) \oplus S_R(A^3_{11})) \tag{16}$$
$$f_2 = S_Q(S_R(A^0_{11} \oplus \Delta_i) \oplus 3S_R(A^1_{11} \oplus \alpha_j) \oplus 2S_R(A^2_{11}) \oplus S_R(A^3_{11})) \tag{17}$$
$$f_3 = S_Q(S_R(A^0_{11} \oplus \Delta_i) \oplus S_R(A^1_{11} \oplus \alpha_j) \oplus 3S_R(A^2_{11}) \oplus 2S_R(A^3_{11})) \tag{18}$$

for randomly chosen α_j $(j = 1, 2, 3)$. Since A_{11} is known, we can recover A^0_{12}, A^1_{12} and $S_R(A^2_{12}) \oplus 3S_R(A^3_{12}) \oplus s^0_{18}$ from (11) by the three equations corresponding to α_j $(j = 1, 2, 3)$. Shifting the byte position of α_j and the multiset position

Δ_i to the other positions, we can derive A_{12}^2, A_{12}^3 in a similar manner. After obtaining A_{12}, we can restore s_{18} by (11) − (14), which is a linear combination of the internal states after initialization. Then we proceed in the same way as above to look at the next 15 keystream words and derive 16 linear equations on the internal states of LFSR after the key/IV setup. Solving this linear system will yield the initial internal state of the LFSR. The values of the three memory registers can be recovered from A_{11}, A_{12} and $s_{17} \oplus C_{12}$ according to Table 4. Then we can run the cipher backwards to recover the secret key since all the steps here are invertible.

The total complexity of the above attack is $4 \cdot 2^{24} + 4 \cdot 2^8 + 4 \cdot 2^{24} + 4 \cdot 2^8 \approx 2^{27}$ steps and 17 keystream words for each IV. We also made experiments to verify the attack. The experiments show that there are exactly 256 solutions to (7) − (9) with a common A_{11}^1 and (10) − (13) behaves like random equations. From (10) − (13), we always recover A_{12} and s_{18} correctly.

3.4 Key Recovery Attack on 15-Round SNOW 3G$^{\oplus}$

In this and the following subsections, we extend previous ideas and combine them with the Gilbert-Minier [3] like ideas of functional collisions in order to cover more rounds of SNOW 3G$^{\oplus}$.

For 15-round SNOW 3G$^{\oplus}$, from the first keystream word we have:

$$\bigoplus_{i=0}^{255} MC_1[S_1(s_{17} \oplus C_{12} \oplus \Delta_i \oplus MC_1[S_1(A_{11} \oplus \Delta_i)])] \oplus \bigoplus_{i=0}^{255} MC_1[S_1(s_{18} \oplus \Delta_i \oplus MC_2[S_2(MC_1[S_1(A_{11} \oplus \Delta_i)])] \oplus MC_1[S_1(A_{12} \oplus \Delta_i)])] = \bigoplus_{i=0}^{255} z_{i,0}.$$

Note that the first term $B_{14} = \bigoplus_{i=0}^{255} MC_1[S_1(s_{17} \oplus C_{12} \oplus \Delta_i \oplus MC_1[S_1(A_{11} \oplus \Delta_i)])]$ has a special pattern $(2a, 3a, a, a)$ with unknown a. Denote the inverse of MC_1 by MC_1^{-1}, we have

$$\bigoplus_{i=0}^{255} S_1(s_{18} \oplus \Delta_i \oplus MC_2[S_2(B_{12})] \oplus MC_1[S_1(A_{12} \oplus \Delta_i)]) = MC_1^{-1}(\bigoplus_{i=0}^{255} z_{i,0} \oplus B_{14}). \tag{19}$$

Expanding (19) to byte equations, we have

$$\bigoplus_{i=0}^{255} S_R[s_{18}^0 \oplus \Delta_i \oplus \underbrace{2f_0 \oplus f_1 \oplus f_2 \oplus 3f_3}_{MC_2} \oplus 2S_R(A_{12}^0 \oplus \Delta_i) \oplus S_R(A_{12}^1 \oplus \eta_j) \oplus S_R(A_{12}^2) \oplus 3S_R(A_{12}^3)] = kc_j^0 \oplus a \tag{20}$$

$$\bigoplus_{i=0}^{255} S_R[s_{18}^1 \oplus \eta_j \oplus \underbrace{3f_0 \oplus 2f_1 \oplus f_2 \oplus f_3}_{MC_2} \oplus 3S_R(A_{12}^0 \oplus \Delta_i) \oplus 2S_R(A_{12}^1 \oplus \eta_j) \oplus S_R(A_{12}^2) \oplus S_R(A_{12}^3)] = kc_j^1 \tag{21}$$

$$\bigoplus_{i=0}^{255} S_R[s_{18}^2 \oplus \underbrace{f_0 \oplus 3f_1 \oplus 2f_2 \oplus f_3}_{MC_2} \oplus S_R(A_{12}^0 \oplus \Delta_i) \oplus 3S_R(A_{12}^1 \oplus \eta_j) \oplus 2S_R(A_{12}^2) \oplus S_R(A_{12}^3)] = kc_j^2 \tag{22}$$

$$\bigoplus_{i=0}^{255} S_R[s_{18}^3 \oplus \underbrace{f_0 \oplus f_1 \oplus 3f_2 \oplus 2f_3}_{MC_2} \oplus S_R(A_{12}^0 \oplus \Delta_i) \oplus S_R(A_{12}^1 \oplus \eta_j) \oplus 3S_R(A_{12}^2) \oplus 2S_R(A_{12}^3)] = kc_j^3, \tag{23}$$

where $kc_j = MC_1^{-1} \cdot \bigoplus_{i=0}^{255} z_{i,0}$ corresponding to η_j, f_i $(0 \le i \le 3)$ defined in $(15)-(18)$ and η_j $(1 \le j \le t)$ are randomly chosen byte differences with t to be determined. Our first observation is that there is no unknown variables on the right hand of $(21)-(23)$, so these equations can be used directly to restore the involving variables. However, if we try to solve (21) by exhaustively searching all the possible values of A_{11}, A_{12}^0, A_{12}^1 and $s_{18}^1 \oplus S_1(A_{12}^2) \oplus S_1(A_{12}^3)$, we need to choose $t = 7$ and the time complexity is 2^{56} steps. In order to get an efficient attack, we proceed as follows.

We regard the left part of (21) as a function of the following variables: A_{11}, A_{12}^0, A_{12}^1 and $s_{18}^1 \oplus S_R(A_{12}^2) \oplus S_R(A_{12}^3)$. Note that there are 7 bytes involved here and if these bytes take the same value for two independent IVs, the outputs of (21) should be equal. In order to detect such an internal collision, we randomly choose a series of byte differences η_j $(1 \le j \le t)$ and compare the corresponding output kc_j^1. If a pair of IV, IV and IV', passes all the t tests, i.e., the outputs of (21) remain the same for η_j $(1 \le j \le t)$, we can conclude with high probability that the 7 bytes involved in the two equations have the same value for IV and IV'.

More precisely, given 2^{28} $(K, \overline{IV_i})$s such that

$$\forall i \ne j, (\overline{IV_i})_r = (\overline{IV_j})_r \quad \text{for} \quad r = 2, 3. \tag{24}$$

$$\forall i \ne j, (\overline{IV_i})_r \ne (\overline{IV_j})_r \quad \text{for} \quad r = 0, 1. \tag{25}$$

will guarantee that there exists such a pair. To filter out the wrong candidates, we choose $t = 8$. A wrong candidate will pass 8 consecutive tests with probability $2^{56} \cdot 2^{-64} = 2^{-8}$ which is less than 1, while the correct candidate will always pass the tests. We use the standard birthday paradox argument to detect such a pair, the time complexity is about $2^{28} \cdot 8 = 2^{31}$ steps. Now we have two IVs, IV and IV', that generate the same input values for (21), i.e.,

- $A_{11} = A'_{11}$, $A_{12}^0 = A_{12}^{'0}$, $A_{12}^1 = A_{12}^{'1}$.
- $s_{18}^1 \oplus S_R(A_{12}^2) \oplus S_R(A_{12}^3) = s_{18}^{'1} \oplus S_R(A_{12}^{'2}) \oplus S_R(A_{12}^{'3})$.

We need to investigate the value evolution process of the memory registers in the first 10 rounds of initialization to derive the state information, which is shown in Table 4. In Table 4, c_i $(i = 1, 2, 3)$ are the same variables as those in Table 3.

We have the following facts on Table 4 when the $(K, \overline{IV_i})$ pair are chosen according to the conditions (24) and (25):

Table 4. The value evolution process of the memory registers in the first 10-round initialization of SNOW 3G$^{\oplus}$ (h_i ($i = 1, 2, 3$) are known constants)

	A_i	B_i	C_i
0	0	0	0
1	k_1	h_1	h_2
2	$k_2 \oplus h_1 \oplus h_2$	$MC_1[S_1(k_1)]$	h_3
3	$h_3 \oplus k_3 \oplus MC_1[S_1(k_1)]$	$MC_1[S_1(A_2)]$	$MC_2[S_2(B_2)]$
4	$B_3 \oplus C_3$ $\oplus k_0 \oplus \mathbf{1}$	$MC_1[S_1(A_3)]$	$MC_2[S_2(B_3)]$
5	$B_4 \oplus C_4$ $\oplus k_1 \oplus \mathbf{1} \oplus IV_3$	$MC_1[S_1(A_4)]$	$MC_2[S_2(B_4)]$
6	$B_5 \oplus C_5$ $\oplus k_2 \oplus \mathbf{1} \oplus IV_2$	$MC_1[S_1(A_5)]$	$MC_2[S_2(B_5)]$
7	$B_6 \oplus C_6$ $\oplus k_3 \oplus \mathbf{1}$	$MC_1[S_1(A_6)]$	$MC_2[S_2(B_6)]$
8	$B_7 \oplus C_7$ $\oplus k_0 \oplus IV_1$	$MC_1[S_1(A_7)]$	$MC_2[S_2(B_7)]$
9	$B_8 \oplus C_8 \oplus k_1$	$MC_1[S_1(A_8)]$	$MC_2[S_2(B_8)]$
10	$\underbrace{B_9 \oplus C_9 \oplus k_2}_{c_1}$	$\underbrace{MC_1[S_1(A_9)]}_{c_2}$	$\underbrace{MC_2[S_2(B_9)]}_{c_3}$
11	$c_2 \oplus c_3 \oplus k_3$ $\oplus IV_0 \oplus \Delta_i$	$MC_1[S_1(c_1)]$	$MC_2[S_2(c_2)]$
12	$B_{11} \oplus C_{11}$ $\oplus s_{16} \oplus \Delta_i$	$MC_1[S_1(A_{11} \oplus \Delta_i)]$	$MC_2[S_2(B_{11})]$
	R_1	R_2	R_3

1. $A_4 = A'_4$, $B_4 = B'_4$ and $C_4 = C'_4$, which are only determined by K.
2. $A_i = A'_i$, $B_i = B'_i$ and $C_i = C'_i$ for $i = 5, 6, 7$.
3. $A_8 \oplus A'_8 = IV_1 \oplus IV'_1$, $B_8 = B'_8$ and $C_8 = C'_8$.
4. $A_9 = A'_9$, $C_9 = C'_9$.
5. $A_{10} \oplus A'_{10} = c_1 \oplus c'_1 = MC_1[S_1(A_8)] \oplus MC_1[S_1(A'_8)]$.
6. $B_{10} = B'_{10}$, i.e., $c_2 = c'_2$.

From Table 4, $A_{11} = c_2 \oplus c_3 \oplus k_3 \oplus IV_0$ and $A_{11} = A'_{11}$, we have $c_3 \oplus c'_3 = IV_0 \oplus IV'_0$, i.e.,

$$S_2(MC_1[S_1(A_8)]) \oplus S_2(MC_1[S_1(A'_8)]) = MC_2^{-1} \cdot (IV_0 \oplus IV'_0). \tag{26}$$

We can determine A_8 and A'_8 from (26) and $A_8 \oplus A'_8 = IV_1 \oplus IV'_1$ with 2^{32} steps. Knowing A_8 and A'_8, we can derive c_3 and c'_3. So far, we have partially recovered the internal states corresponding to IV and IV'.

3.5 Key Recovery Attack on 18-Round SNOW 3G$^{\oplus}$

Now we skip the 16 and 17 rounds case, since they are similar and go directly to the 18-round. Let us denote $F = MC_1[S_1(\cdot)]$, $G = MC_2[S_2(\cdot)]$ and $H = G[F(\cdot)]$, then from the first keystream word, we have:

$$\bigoplus_{i=0}^{255} F[s_{20} \oplus \alpha^{-1}\Delta_i \oplus \Delta_i \oplus H(s_{17} \oplus C_{12} \oplus \Delta_i \oplus F(A_{11} \oplus \Delta_i)) \oplus F(s_{18} \oplus H(A_{11} \oplus \Delta_i) \oplus F(A_{12} \oplus \Delta_i)] \oplus \bigoplus_{i=0}^{255} H(s_{19} \oplus \Delta_i \oplus H(A_{12} \oplus \Delta_i) \oplus F(s_{17} \oplus C_{12} \oplus \Delta_i \oplus F(A_{11} \oplus \Delta_i)) \oplus \bigoplus_{i=0}^{255} F(s_{21} \oplus \Delta_i \oplus H(s_{18} \oplus \Delta_i \oplus H(A_{11} \oplus \Delta_i) \oplus F(A_{12} \oplus \Delta_i)) \oplus F(s_{19} \oplus \Delta_i \oplus H(A_{12} \oplus \Delta_i)) \oplus F(s_{17} \oplus C_{12} \oplus \Delta_i \oplus F(A_{11} \oplus \Delta_i)) = \bigoplus_{i=0}^{255} z_{i,0}. \quad (27)$$

Here by using multisets, we get rid of the LFSR words, s_{33} and s_{18}, involved in the keystream equations. Of these, s_{33} is the main obstacle to a differential analysis of the keystream equation. To have an intuitive view, we color the repeating patterns of variables in the left side of (23) in the same color. Note that we can control the values of s_{17}, s_{18}, s_{19}, s_{20} and s_{21} by properly choosing the IVs. From the following equations (28) – (33),

$$s_{16} = \alpha^{-1}(k_3 \oplus \mathbf{1}) \oplus (k_2 \oplus \mathbf{1}) \oplus \alpha(k_0 \oplus \mathbf{1}) \oplus k_3 \oplus IV_0 \quad (28)$$

$$s_{17} = \alpha^{-1}(k_0 \oplus IV_1) \oplus (k_3 \oplus \mathbf{1}) \oplus \alpha(k_1 \oplus \mathbf{1}) \oplus k_1 \oplus s_{16} \oplus h_1 \quad (29)$$

$$s_{18} = \alpha^{-1}k_1 \oplus k_0 \oplus \alpha(k_2 \oplus \mathbf{1}) \oplus k_2 \oplus F(k_1) \oplus h_1 \oplus h_2 \oplus s_{17} \quad (30)$$

$$s_{19} = \alpha^{-1}k_2 \oplus k_1 \oplus \alpha(k_3 \oplus \mathbf{1}) \oplus k_3 \oplus F(k_1) \oplus F(k_2 \oplus h_1 \oplus h_2) \oplus h_3 \oplus s_{18} \quad (31)$$

$$s_{20} = \alpha^{-1}(k_3 \oplus IV_0) \oplus k_2 \oplus \alpha k_0 \oplus F(k_2 \oplus h_1 \oplus h_2) \oplus H(k_2) \oplus (k_0 \oplus \mathbf{1}) \oplus F(k_3 \oplus h_3 \oplus F(k_1)) \oplus s_{19} \quad (32)$$

$$s_{21} = \alpha^{-1}s_{16} \oplus k_3 \oplus \alpha k_1 \oplus k_1 \oplus \mathbf{1} \oplus IV_3 \oplus F(k_3 \oplus F(k_1)) \oplus H(k_2) \oplus F(A_4) \oplus s_{20}. \quad (33)$$

we know that if we choose IV and IV' such that: $IV_3 = IV_3'$ and

$$\alpha^{-1}(IV_1 \oplus IV_1') \oplus (IV_0 \oplus IV_0') = 0,$$

then, we have $s_{17} = s_{17}'$, $s_{18} = s_{18}'$, $s_{19} = s_{19}'$, $s_{21} = s_{21}'$ and $s_{20} \oplus s_{20}' = \alpha^{-1}(IV_0 \oplus IV_0')$. If we undo the MC_1 on both sides of (27), we can see that in order to have a collision on the involved variables of the left side of (27), we need the following 32-bit conditions:

$$C_{12} = C_{12}'. \quad (34)$$

$$A_{11} = A_{11}'. \quad (35)$$

$$A_{12} = A_{12}'. \quad (36)$$

There are 96 bits involved here, so 2^{48} random (K, IV) pairs satisfying the above specified conditions will ensure that there exists such a collision with high probability. Then injecting the byte differences λ_j $(1 \leq j \leq 8)$ into the second most significant byte of the involved variables will preserve the collision. Such a collision can be detected by changing these byte differences and comparing the most and second most significant bytes of the corresponding keystream words xors. The time complexity of this detection is $2^{48} \cdot 8 \cdot 2 = 2^{52}$ steps. Then from Table 4 and (34) − (36), we have $c_1 = c'_1$, i.e.,

$$k_2 \oplus F(A_8) \oplus G(B_8) = k_2 \oplus F(A'_8) \oplus G(B'_8). \tag{37}$$

From Table 4, B_8 is determined by B_6 and C_6 which are the same when the key and IV_3 are fixed, and thus $B_8 = B'_8$. Further, from (37), we get $A_8 = A'_8$, i.e.,

$$\begin{aligned} F(A_6) \oplus F(A'_6) = IV_1 \oplus IV'_1 \Rightarrow F(k_2 \oplus \mathbf{1} \oplus IV_2 \oplus F(A_4) \oplus G(B_4)) \oplus \\ F(k_2 \oplus \mathbf{1} \oplus IV'_2 \oplus F(A_4) \oplus G(B_4)) = IV_1 \oplus IV'_1. \end{aligned} \tag{38}$$

Let $k_2 \oplus F(A_4) \oplus G(B_4) \oplus \mathbf{1} = V$ which is an unknown constant, then from (38) we get V in 2^{32} steps. Then we know $A_6 = V \oplus IV_2$ and $A'_6 = V \oplus IV'_2$.

From $A_{12} = A'_{12}$, $c_1 = c'_1$ and $B_{11} = B'_{11}$, we have $G(c_2) \oplus G(c'_2) = IV_0 \oplus IV'_0$, see Table 4. Again from Table 4, we have

$$H(k_1 \oplus B_8 \oplus H(A_6)) \oplus H(k_1 \oplus B_8 \oplus H(A'_6)) = IV_0 \oplus IV'_0. \tag{39}$$

So we can derive $k_1 \oplus B_8$ from (39) in 2^{32} steps. Combining $k_1 \oplus B_8$ with A_6 and A'_6, we get c_2 and c'_2. Then from $A_{11} = A'_{11}$, we get $c_3 \oplus c'_3 = IV_0 \oplus IV'_0 \oplus c_2 \oplus c'_2$, i.e.,

$$\begin{aligned} H(IV_1 \oplus F(A_6) \oplus k_0 \oplus G(B_6)) \oplus H(IV'_1 \oplus F(A'_6) \oplus k_0 \oplus G(B_6)) \\ = IV_0 \oplus IV'_0 \oplus c_2 \oplus c'_2. \end{aligned} \tag{40}$$

From (40), we can get $k_0 \oplus G(B_6)$ in 2^{32} steps. Thus, we know A_8 and A'_8, c_3 and c'_3. So far, the information restored is shown in Table 5, where ♣ means recovered and ♢ means partially recovered.

Table 5. The register values restored

	5	6	7	8	9	10	11
R_1		♣		♣	♣		
R_2		♢	♣	♢	♣	♣	
R_3				♣		♣	♣

In order to recover the key, we recall the above attack once with a different value of $IV_3 = IV'_3$, i.e., we choose another set of 2^{48} random $(K, \overline{IV})$ pairs such that the key K is the same as before and the $\overline{IV}$s satisfy the same conditions, but with a different set of values. Then the above analysis process also applies to the second set. Our observation is that the values of the registers in the FSM

in the second set case are highly correlated to those in the first set case. For example, $k_2 \oplus F(A_4) \oplus G(B_4) \oplus \mathbf{1} = V$ is the same, since it only depends on the key K. From the values of $k_0 \oplus G(B_6)$, we have

$$k_0 \oplus G(B_6) = const1. \tag{41}$$

$$k_0 \oplus G(\overline{B_6}) = const2. \tag{42}$$

From (41) and (42), we have

$$H(IV_3 \oplus \mathbf{1} \oplus k_1 \oplus B_4 \oplus C_4) \oplus H(\overline{IV_3} \oplus \mathbf{1} \oplus k_1 \oplus B_4 \oplus C_4) = const3. \tag{43}$$

From (43), we can restore $k_1 \oplus B_4 \oplus C_4$ in 2^{32} steps, which in turn gives us A_5 and $\overline{A_5}$, B_6 and $\overline{B_6}$. Combining these values with A_8 and $\overline{A_8}$ which are known from the corresponding individual analysis, we can recover k_0 successfully. We can use similar procedures to recover the other key words. The total time complexity is $2^{52} \cdot 2 = 2^{53}$ steps and the data complexity is $2^8 \cdot 2^{48} \cdot 2 = 2^{57}$ keystream words.

4 Conclusions

In this paper, we have shown chosen IV resynchronization attacks on SNOW 3G and SNOW 3G$^{\oplus}$. We show full key-recovery attacks on up to 18 out of 33 initialization rounds of SNOW 3G$^{\oplus}$ using a *multiset collision* idea. We also show 13-round distinguisher of 2^8 complexity for the actual SNOW 3G. Practical parts of all these attacks have been verified experimentally on a PC. Our results show that about half of the initialization rounds of SNOW 3G might succumb to chosen IV resynchronization attacks. The remaining security margin however is quite significant and thus these attacks pose no threat to the security of SNOW 3G.

Acknowledgements. We would like to thank the anonymous reviewers for very helpful comments. Bin Zhang was with State Key Laboratory of Information Security, Institute of Software, Chinese Academy of Sciences, Beijing, 100190, China and supported by the key programm of the National Natural Science Foundation of China (Grant No. 60833008) and the general programm of the National Natural Science Foundation of China (Grant No. 60603018).

References

1. Biryukov, A., Shamir, A.: Structural Cryptanalysis of SASAS. In: Pfitzmann, B. (ed.) EUROCRYPT 2001. LNCS, vol. 2045, pp. 394–405. Springer, Heidelberg (2001)
2. Billet, O., Gilbert, H.: Resistance of SNOW 2.0 Against Algebraic Attacks. In: Menezes, A.J. (ed.) CT-RSA 2005. LNCS, vol. 3376, pp. 19–28. Springer, Heidelberg (2005)
3. Gilbert, H., Minier, M.: A Collision Attack on 7 Rounds of Rijndael. In: AES Candidate Conference 2000, pp. 230–241 (2000)

4. Ekdahl, P., Johansson, T.: A New Version of the Stream Cipher SNOW. In: Nyberg, K., Heys, H.M. (eds.) SAC 2002. LNCS, vol. 2595, pp. 37–46. Springer, Heidelberg (2003)
5. ETSI/SAGE. Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2 & UIA2. Document 2: SNOW 3G Specification, version 1.1 (September 2006), `http://www.3gpp.org/ftp/`
6. ETSI/SAGE. Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2 & UIA2. Document 5: Design and Evaluation Report, version 1.1 (September 2006), `http://www.3gpp.org/ftp/`
7. Nyberg, K., Wallén, J.: Improved Linear Distinguishers for SNOW 2.0. In: Robshaw, M.J.B. (ed.) FSE 2006. LNCS, vol. 4047, pp. 144–162. Springer, Heidelberg (2006)
8. Watanabe, D., Biryukov, A., De Canniére, C.: A Distinguishing Attack of SNOW 2.0 with Linear Masking Method. In: Matsui, M., Zuccherato, R.J. (eds.) SAC 2003. LNCS, vol. 3006, pp. 222–233. Springer, Heidelberg (2004)

A Multiset Difference Propagation Table

Table 6. Multiset difference propagation in 10-round initialization of SNOW 3G$^{\oplus}$

	s_{15}	s_{14}	s_{13}	s_{12}	s_{11}	s_{10}	s_9	s_8	s_7	s_6	s_5	s_j $(0\leq j\leq 4)$	R_i $(i=1,2,3)$
0	Δ_i	0	0	0	0	0	0	0	0	0	0	0	0
1	Δ_i	Δ_i	0	0	0	0	0	0	0	0	0	⋮	⋮
2	Δ_i	Δ_i	Δ_i	0	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
3	Δ_i	Δ_i	Δ_i	Δ_i	0	0	0	0	0	0	0	⋮	⋮
4	Δ_i	Δ_i	Δ_i	Δ_i	Δ_i	0	⋮	⋮	⋮	⋮	⋮	⋮	⋮
5	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	Δ_i	Δ_i	Δ_i	Δ_i	0	⋮	⋮	⋮	⋮	⋮	⋮
6	Δ_i	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	Δ_i	Δ_i	Δ_i	Δ_i	0	⋮	⋮	⋮	⋮	⋮
7	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	Δ_i	Δ_i	Δ_i	Δ_i	0	⋮	⋮	⋮	⋮
8	Δ_i	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	Δ_i	Δ_i	Δ_i	Δ_i	0	⋮	⋮	⋮
9	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	Δ_i	Δ_i	Δ_i	Δ_i	0	⋮	⋮
10	$\Delta_i \oplus \alpha^{-2}\Delta_i$	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	$\Delta_i \oplus \alpha^{-1}\Delta_i$	Δ_i	Δ_i	Δ_i	Δ_i	Δ_i	0	0