Received November 23, 2021, accepted December 27, 2021, date of publication December 30, 2021,
date of current version January 7, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3139614

Improved Ciphertext-Only Attack on GMR-1

DONGJAE LEE[1], DEUKJO HONG[2], JAECHUL SUNG[3], SEONGGYEOM KIM[1], AND SEOKHIE HONG[1]

[1]Institute of Cyber Security and Privacy (ICSP), Korea University, Seoul 02841, South Korea
[2]Department of Information Technology and Engineering, Jeonbuk National University, Jeonju 54896, South Korea
[3]Department of Mathematics, University of Seoul, Seoul 02504, South Korea

Corresponding author: Seokhie Hong (shhong@korea.ac.kr)

This work was supported in part by the Military Crypto Research Center funded by the Defense Acquisition Program Administration (DAPA) and the Agency for Defense Development (ADD) under Grant UD210027XD.

ABSTRACT The GEO-Mobile Radio Interface-1 (GMR-1) is a satellite communication standard used in Thuraya, a United Arab Emirates-based regional mobile satellite service provider. The specification of the encryption algorithm used in GMR-1 was not disclosed until it was uncovered by Driessen *et al.* in 2012 through reverse engineering. Given that A5-GMR-1, a stream cipher used in GMR-1, is primarily based on A5/2, Driessen *et al.* presented a ciphertext-only attack from the attacks on A5/2. Their ciphertext-only attack recovers the session key from multiple sets of 24 ciphertexts in an average of 32.1 min and requires 400 GB of pre-computed data. This study enhances Driessen *et al.*'s ciphertext-only attack on A5-GMR-1 in all aspects of time, memory, and data. Our contributions are fourfold. First, we optimize the inefficient part of the previous attack. As a result, our ciphertext-only attack recovers the session key from multiple sets of 13 ciphertexts in less than 1 second and requires 400 MB of pre-computed data. Second, we propose novel memory-saving techniques. These techniques reduce the memory complexity to 216 $\sim$ 289 MB without increasing the time and data complexity. Third, we present several time-memory-data tradeoff techniques. Using these techniques, we can present an attack that meets the desired conditions, such as memory minimization or data minimization. Furthermore, while the complexity of the previous attack is presented vaguely as ''multiple sets'' of 24 ciphertexts, these techniques allow us to accurately calculate the time, memory, and data complexity of the attack. Finally, we demonstrate that A5-GMR-1 can be attacked without frame numbers. To find out the frame number of each ciphertext, it is necessary to analyze and synchronize multiple channels. We present a plaintext recovery attack that does not require these processes.

INDEX TERMS A5-GMR-1, ciphertext-only attack, cryptography, stream cipher.

I. INTRODUCTION

Nowadays, mobile communication systems allow people to communicate with each other and transmit data in most parts of the world. However, some areas, such as deserts, oceans and mountains, are not covered by mobile communication systems. A satellite communication system overcomes these regional limitations and allows people from a wider area to communicate with each other. The GEO-Mobile Radio Interface (GMR), a standard of European Telecommunication Standards Institute (ETSI), is now commonly used for satellite communications. There is the GMR-1 standard adopted by the Thuraya phone and the GMR-2 standard adopted by the Inmarsat phone.

The associate editor coordinating the review of this manuscript and approving it for publication was Md. Zakirul Alam Bhuiyan.

Telecommunication systems such as mobile communication systems and satellite communication systems are always exposed to the risk of eavesdropping. Therefore, robust encryption algorithms are essential to protect data. Encryption algorithms adopted for mobile communication systems include A5, ZUC, and SNOW. These algorithms have been sufficiently analyzed and evaluated over the past years [1]–[6]. On the other hand, the encryption algorithms adopted for GMR are not included in the officially published standards.

In 2012, Driessen *et al.* revealed the algorithm used in GMR through reverse engineering and named the encryption algorithm adopted by GMR-1 as A5-GMR-1, and the encryption algorithm adopted by GMR-2 as A5-GMR-2 [7], [8]. They also presented a known plaintext attack on A5-GMR-2. This attack was enhanced in 2018 by Jiao *et al.* [9].

A5-GMR-1 has a structure quite similar to that of A5/2 used in the global system for mobile communications (GSM). Both ciphers are composed of four linear feedback shift registers (LFSRs) and have a similar process of generating a keystream. The differences between them are the feedback polynomials of the LFSRs, keystream generation formula, and number of iterations. Owing to their similarities, existing attacks on A5/2 can be applied also to A5-GMR-1 with some modifications.

Briceno *et al.* revealed the specifications of A5/2 (and A5/1) through reverse engineering of GSM phones in 1999 [10]. After disclosing the specification, Goldberg *et al.* first conducted a cryptanalysis of A5/2 [11]. The main disadvantage of the attack is the requirement that two known plaintexts whose frame numbers have exactly 1326 differences are given. Barkan *et al.* proposed a ciphertext-only attack by exploiting that encryption is applied after encoding in GSM encrypted communication [12]. Furthermore, they optimized the attack by pre-computing some of the attack steps. Even in GMR-1 satellite communication, data is encoded and then encrypted. Based on this similarity, Driessen *et al.* revealed the specification of A5-GMR-1 through reverse engineering and then mounted a ciphertext-only attack against A5-GMR-1 by adjusting ciphertext-only attacks on A5/2.

In Driessen *et al.*'s attack on A5-GMR-1, a session key can be recovered from multiple sets of 24 ciphertexts in an average of 32.1 min, and 400 GB of pre-computed data, compressed using the LZ algorithm, is required. However, Barkan *et al.*'s attack on A5/2 recovers the session key in less than 1 second with 8 ciphertexts, and requires 4GB of memory complexity. The two attacks have different levels of complexity. This does not result from distinct target algorithms. It is because Driessen *et al.* missed several optimizations and allowed some parts of their attack to be inefficient.

Another problem is that the complexity of Driessen *et al.*'s attack is presented somewhat vaguely. In order to accurately calculate the memory complexity, it is necessary to consider the XOR-differences between the frame numbers of given ciphertexts. They neither take this into account nor present the memory complexity before and after applying the LZ compression algorithm. Moreover, the number of ciphertexts required is suggested as multiple sets of 24 ciphertexts, not the exact number.

In this paper, we present an improved ciphertext-only attack on A5-GMR-1. We address the problems of the previous work we pointed out and propose new methods to further improve the attack. The contributions of this study are fourfold. First, we enhance the attack proposed by Driessen *et al.* by optimizing inefficient processes. We exploit the ideas from Barkan *et al.*'s attack on A5/2. As a result, the time, memory, and data complexity of the ciphertext-only attack on A5-GMR-1 are significantly reduced. The time complexity is reduced from 32.1 minutes to less than 1 second. The memory complexity is reduced from 400 GB to 400 MB. The data complexity is reduced from multiple sets of 24 ciphertexts to multiple sets of 13 ciphertexts.

Second, we present novel methods to reduce memory complexity without increasing the time and data complexity. Specifically, optimizing the attack parameters can reduce the memory complexity. In the attack process, we express every bit of the keystream as a quadratic combination of session key related variables. Collecting quadratic combinations, we linearize the quadratic terms and generate a linearized system of equations. We pre-compute possible systems and store the matrices associated with them in the offline stage. Our memory-saving techniques minimize the size of the linearized system of equations and the size of system-related matrices to be stored.

Third, we present two complexity-tradeoff techniques. The first technique considers the tradeoff between memory and data complexity. We analyze the effect of the XOR-differences between the frame numbers of given ciphertexts on the memory complexity. Through this technique, it is possible to determine how the number of given ciphertexts affects memory complexity. Furthermore, it provides us with the exact time, memory, and data complexity of the attack. Another complexity-tradeoff technique addresses the relation between memory complexity and the probability of success. When the number of ciphertexts increases to more than 28, the first technique cannot further reduce the memory complexity. In this case, our second technique provides an efficient way to reduce memory complexity. The most balanced attack considering both tradeoff techniques requires 28 ciphertexts, recovers the session key in less than 1 second, and requires 289 MB for pre-computed data. This setting provides an attack for the least data $\times$ time $\times$ memory.

Finally, we present a plaintext recovery attack on A5-GMR-1 which does not require a frame number. Frame numbers are transmitted without encryption. However, the frame number and the voice data are transmitted through different channels. Therefore, in order to determine the frame number of each ciphertext, we need to collect, analyze, and synchronize multiple channels. If the goal is to analyze the set of ciphertexts associated with just a few phone calls, these processes may not be a burden. However, if the goal is to analyze multiple phone calls in real time, our variant attack is very advantageous. Moreover, the data and memory complexity are the same as them of the attack using frame numbers, and only the time complexity increases several times. Under the same conditions as our most balanced attack (28 ciphertexts and 289 MB of pre-computed data are given), the time complexity of the plaintext recovery attack is 16 times that of the previous one, which is still less than 1 second.

The results are summarized in Table 1. The attack group $\mathcal{A}1$ presents the results of previous studies. $\mathcal{A}2$ represents the main results of applying the proposed optimization methods and tradeoff techniques. The first in $\mathcal{A}2$ represents the most balanced attack. The second and third in $\mathcal{A}2$ represent attacks for the least memory and data, respectively. $\mathcal{A}3$ represents

TABLE 1. Summary of cryptanalysis on A5-GMR-1.

Attack Group	Data	Time	Memory	Prob. of Success	Reference
$\mathcal{A}1$	24×n	32.1min	400GB	1	[8]
$\mathcal{A}2$	28	< 1sec	289MB	1	This Paper
	28	5sec	216MB	1	This Paper
	13	< 1sec	52GB	1	This Paper
$\mathcal{A}3$	13	< 1sec	11GB	0.9	This Paper
	13	< 1sec	6.7GB	0.75	This Paper
	13	< 1sec	3.4GB	0.5	This Paper
	13	< 1sec	1.1GB	0.25	This Paper
$\mathcal{A}4$	56	< 1sec	173MB	0.9	This Paper
	56	< 1sec	116MB	0.75	This Paper
	56	< 1sec	57.8MB	0.5	This Paper
	56	< 1sec	28.9MB	0.25	This Paper

the results of applying the memory/probability of the success tradeoff technique. $\mathcal{A}4$ represents the results of applying the memory/probability of the success tradeoff technique when more ciphertexts are available compared to that in a previous setting. We chose number 56 because a 10 second call has a ciphertext streak of up to 56 lengths according to [8].

The remainder of this paper is organized as follows. Section II defines the notations used in this paper with brief explanations of the technical background of the GMR-1 and the stream cipher A5-GMR-1 which are necessary to understand this study. Section III presents an improved ciphertext-only attack on A5-GMR-1. Section IV presents the methods used to reduce memory complexity. Section V presents several complexity tradeoff techniques that enable the selection of appropriate attacks under resource constraints. Section VI presents a plaintext recovery attack that does not require knowledge of the frame numbers. Finally, Section VII concludes the paper.

II. PRELIMINARY

A. NOTATIONS

In this paper, among the various channels of GMR-1, we are interested in Traffic Channel-3 (TCH3) through which encrypted voice data are transmitted. Therefore, notations defined in this subsection are focused on the TCH3 channel. The bitstream is considered as a column vector. The following notations are used throughout the paper.

- d : The data transmitted through one frame (160-bit). It is encoded, encrypted, multiplexed, modulated and then transmitted.
- G : The generating matrix (160×208). It represents part of encoding.
- N : The frame number (19-bit). Each frame is numbered by a frame number. The frame number of the i-th given ciphertext is denoted by N_i.
- s : The fixed scrambling bits (208-bit). Scrambling is the process of adding a binary-noise sequence to the input bit stream.
- z : The keystream (208-bit). It is the output of the stream cipher A5-GMR-1, with the session key and frame number as inputs.
- c : The ciphertext (208-bit). It is the bitstream after the data has been encoded and encrypted. $c = (G^\top \times d) \oplus s \oplus z$
- r : The syndrome (48-bit). It is defined as $r = H \times (c \oplus s)$.
- $0_{m\times n}$: $m \times n$ matrix, where all elements are zero.

A more detailed description of each notation is covered in the rest of paper. Each bitstream is indexed by writing the frame number as a superscript. For example, ciphertext, data, and syndrome having N_0 as a frame number are denoted by c^{N_0}, d^{N_0}, and r^{N_0}, respectively. Each bit of the column vector is separated by a subscript (starting from 0). For example, the i-th bit of c^{N_0} is denoted by $c_i^{N_0}$. The transpose of a matrix (or a vector) M is denoted by $M^\top$.

B. DESCRIPTION OF TCH3 IN GMR-1 STANDARD

We briefly introduce the process of signal generation in this subsection. According to the ETSI technical specification [13], a user unit delivers a bitstream comprising several blocks of 80 information bits to the encoder. Channel coding, interleaving, scrambling, encryption, and multiplexing are applied to the block in this order. They are mapped and transmitted in an NT3 burst. In summary, two blocks of 80 information bits are encoded into a 212-bit block. The multiplexing process can be reversed simply by removing 4 bits from specific positions in the 212-bit block. Therefore, we assume that the 208-bit ciphertexts are given after convolutional coding, puncturing, interleaving, scrambling, and encryption without multiplexing (the detailed procedure of these encodings can be found in [13]).

Convolutional coding, puncturing, and interleaving are considered as matrix multiplications. Therefore, a generating matrix G with the size of 160×208 can be obtained, which represents the entire encoding process as the matrix multiplication $G^\top \times d$, except for scrambling. Given that scrambling and encryption are executed by XORing the corresponding bitstreams s, z with the encoded data, the following condition is satisfied.

$$c = (G^\top \times d) \oplus s \oplus z.$$

In the encoding process, the multiplication $G^\top \times d$ includes 48-bit redundancy. Therefore, a parity-check matrix H with the size of 48×208 such that

$$H \times G^\top \times d = 0 \quad \forall d \in \{0,1\}^{160}.$$

can be found. Thus, the syndrome $r = H \times (c \oplus s)$ satisfies the following:

$$r = H \times ((G^\top \times d) \oplus z) = H \times z. \tag{1}$$

Therefore, we can simply remove the effect of d from ciphertexts and mount a ciphertext-only attack on A5-GMR-1 with $r = H \times z$ without the knowledge of d.

C. DESCRIPTION OF A5-GMR-1

The stream cipher A5-GMR-1 is used in the GMR-1 standard. Although its specification is not available in public,

TABLE 2. Parameters of the A5-GMR-1 registers.

LFSR	Length	Feedback polynomial
R1	19	$x^{19}+x^{18}+x^{17}+x^{14}+1$
R2	22	$x^{22}+x^{21}+x^{17}+x^{13}+1$
R3	23	$x^{23}+x^{22}+x^{19}+x^{18}+1$
R4	17	$x^{17}+x^{14}+x^{13}+x^{9}+1$

Driessen *et al.* in 2012 [7] uncovered A5-GMR-1 through reverse engineering.

A5-GMR-1 is a stream cipher that accepts a 64-bit session key K and a 19-bit frame number N. It uses four LFSRs, namely, R1, R2, R3, and R4 of 19-, 22-, 23-, and 17-bit lengths, respectively, as the main building blocks (Figure 1). We denote each bit of the LFSR using a subscript. For example, $\mathsf{R1}_0$ represents the least significant bit (LSB) of R1 (the 0-th bit of R1), and $\mathsf{R4}_6$ represents the 6-th bit of R4. Table 2 lists the feedback polynomial of each LFSR. The nonlinear majority function $\mathcal{M}$ used for irregular clocking and keystream generation is defined as follows:

$$\begin{aligned}\mathcal{M} : \{0,1\}^3 &\rightarrow \{0,1\}\\ (x,y,z)_2 &\mapsto xy \oplus yz \oplus zx. \end{aligned} \tag{2}$$

In a cryptanalysis, we exploit the fact that the outputs of the majority function can be expressed by the quadratic monomials of the input bits.

A5-GMR-1 comprises initialization and keystream generation phases, wherein the former is as expressed follows:

- **Initialization Phase of A5-GMR-1**

(1) All bits of four LFSRs are initialized to 0.

(2) A 64-bit initialization vector α is derived from the frame number N and the session key K as follows:

$$\begin{aligned}\alpha = {} & K_{0,1,2}||K_3 \oplus N_6||K_4 \oplus N_7||K_5 \oplus N_8||K_6 \oplus N_9||\\ & K_7 \oplus N_{10}||K_8 \oplus N_{11}||K_9 \oplus N_{12}||K_{10} \oplus N_{13}||\\ & K_{11} \oplus N_{14}||K_{12} \oplus N_{15}||K_{13} \oplus N_{16}||K_{14} \oplus N_{17}||\\ & K_{15} \oplus N_{18}||K_{16\ldots21}||K_{22} \oplus N_4||K_{23} \oplus N_5||\\ & K_{24\ldots59}||K_{60} \oplus N_0||K_{61} \oplus N_1||K_{62} \oplus N_2||\\ & K_{63} \oplus N_3. \end{aligned}$$

(3) The bits of α are re-ordered to α' such that

$$\alpha'_{16i+j} = \alpha_{16(i+1)-(j+1)}$$

for all $i = 0, 1, \ldots, 3, j = 0, 1, \ldots, 15$.

(4) For each LFSR Rj (j $= 1, .., 4$), execute the following: For $i = 0, 1, \ldots, 63$,

a) Clock Rj

b) $\mathsf{Rj}_0 \leftarrow \mathsf{Rj}_0 \oplus \alpha'_i$.

(5) Each LSB of all four registers is set to 1; that is,

$$\mathsf{Rj}_0 \leftarrow 1 \quad \text{for j} = 1, \ldots, 4.$$

We denote the 81-bit initial state after the initialization phase for each frame number by β, and the initial states of Rj by $\mathsf{Rj}_{\mathsf{init}}$; that is,

$$\beta = \underbrace{\beta_0, \ldots, \beta_{18}}_{\mathsf{R1}_{\mathsf{init}}}, \underbrace{\beta_{19}, \ldots, \beta_{40}}_{\mathsf{R2}_{\mathsf{init}}}, \underbrace{\beta_{41}, \ldots, \beta_{63}}_{\mathsf{R3}_{\mathsf{init}}}, \underbrace{\beta_{64}, \ldots, \beta_{80}}_{\mathsf{R4}_{\mathsf{init}}}.$$

Upon completion of the initialization phase, the keystream generation phase works as follows:

- **Keystream Generation Phase of A5-GMR-1**

(1) R1, R2, and R3 are clocked according to the following conditions. (a)

a) If $\mathcal{M}(\mathsf{R4}_1, \mathsf{R4}_6, \mathsf{R4}_{15}) = \mathsf{R4}_{15}$, R1 is clocked.

b) If $\mathcal{M}(\mathsf{R4}_1, \mathsf{R4}_6, \mathsf{R4}_{15}) = \mathsf{R4}_6$, R2 is clocked.

c) If $\mathcal{M}(\mathsf{R4}_1, \mathsf{R4}_6, \mathsf{R4}_{15}) = \mathsf{R4}_1$, R3 is clocked.

(2) Each bit z_l of the keystream is created as follows.

$$\begin{aligned} z_l \leftarrow {} & \mathcal{M}(\mathsf{R1}_1, \mathsf{R1}_6, \mathsf{R1}_{15}) \oplus \mathcal{M}(\mathsf{R2}_3, \mathsf{R2}_8, \mathsf{R2}_{14})\\ & \oplus \mathcal{M}(\mathsf{R3}_4, \mathsf{R3}_{15}, \mathsf{R3}_{19}) \oplus \mathsf{R1}_{11} \oplus \mathsf{R2}_1 \oplus \mathsf{R3}_0. \end{aligned}$$

(3) R4 is clocked.

These processes are repeated 458 times, the first 250 bits generated are discarded, and the next 208 bits are used as a keystream. In a real environment, another 208-bit keystream is generated, and two 208-bit keystreams are used for encryption and decryption. The first 208-bit keystream is used on the side of the handset for decryption and on the side of the network provider for encryption. Therefore, we conduct a cryptanalysis in this study while considering only the first 208-bit keystream.

III. IMPROVED CIPHERTEXT-ONLY ATTACK ON A5-GMR-1

In this section, we present an improved ciphertext-only attack on A5-GMR-1. We optimize inefficient steps of Driessen *et al.*'s attack on A5-GMR-1 by exploiting optimization ideas from Barkan *et al.*'s attack on A5/2. Let c^{N_0}, c^{N_1}, ... be the given ciphertexts, where N_0, N_1, ...are the corresponding frame numbers. We assume that $N_i = N_0 + i$. According to the standard [14], the frame number increases by 1 every 40ms. Therefore, if given ciphertexts are collected for a continuous period of time, this assumption is natural.[1]

Our attack recovers β^{N_0}, the initial state of the first frame number N_0, and then finds the session key by reversing the initialization phase. Our attack exploits that, given $\mathsf{R4}_{\mathsf{init}}$, each bit of the keystream can be expressed as a quadratic combination of bits in $\mathsf{R1}_{\mathsf{init}}$, $\mathsf{R2}_{\mathsf{init}}$, and $\mathsf{R3}_{\mathsf{init}}$. This fact is independent of the frame number. Therefore, in the process of explaining how each bit of the keystream can be expressed as a quadratic combination, the indication of the frame number is omitted.

In keystream generation phase, a series of processes are repeated 458 times (c.f., (1) of **Keystream Generation Phase**). R4 is always clocked, and whether R1, R2, or R3 is clocked is determined by R4. Since, R4 does not depend

[1] It is natural to assume that a given ciphertext has been collected over a continuous period of time. This is because the target of our attack is phone calls.

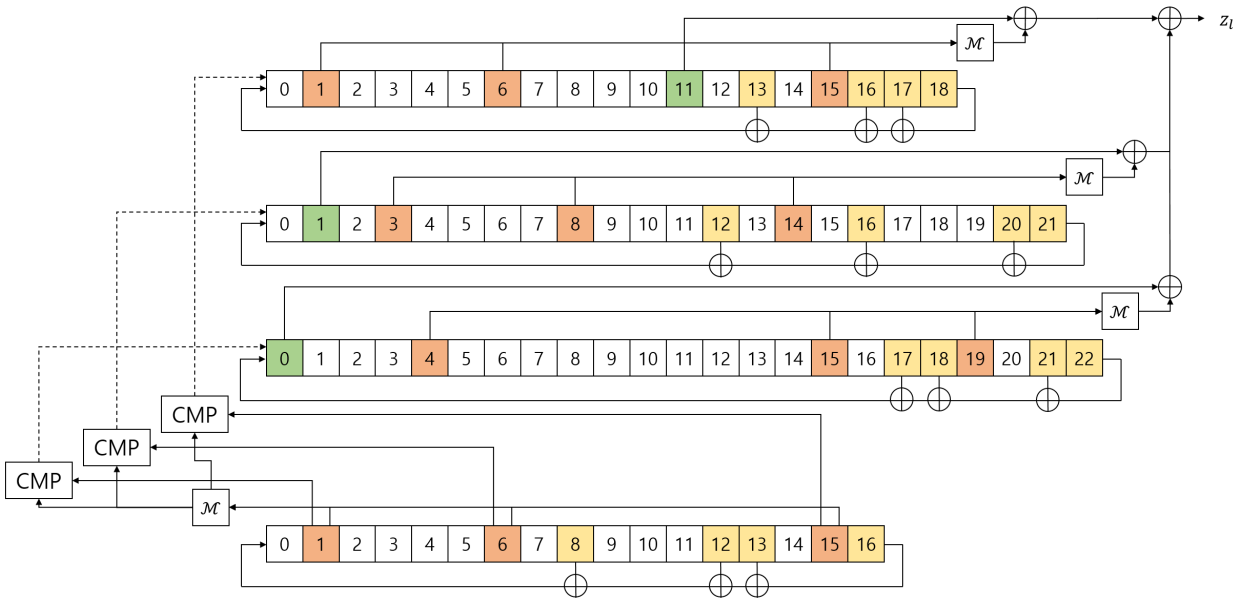

FIGURE 1. A5-GMR-1 structure.

on the other registers, if $\mathsf{R4}_{\mathsf{init}}$ is given, one can obtain all the values of R4 during the keystream generation phase. This derives the timings when R1, R2 and R3 are clocked, respectively. It implies that for every iteration, we can express all the bits in R1, R2, and R3 as a linear combination of bits in $\mathsf{R1}_{\mathsf{init}}$, $\mathsf{R2}_{\mathsf{init}}$, and $\mathsf{R3}_{\mathsf{init}}$.

Example 1: After R1 is clocked once from its initial state, each bit of R1 can be expressed as linear combinations of $\mathsf{R1}_{\mathsf{init}} = \beta_0, \ldots, \beta_{18}$;

$$\begin{aligned}\mathsf{R1}_0 &= \beta_{18} \oplus \beta_{17} \oplus \beta_{16} \oplus \beta_{13},\\ \mathsf{R1}_1 &= \beta_0 (= 1),\\ \mathsf{R1}_2 &= \beta_1,\\ &\cdots\\ \mathsf{R1}_{18} &= \beta_{17}.\end{aligned}$$

The output of the majority function can be represented as a quadratic form of the 3 input bits (c.f., Equation (2)). Therefore, each keystream bit can be expressed as a quadratic combination of bits in $\mathsf{R1}_{\mathsf{init}}$, $\mathsf{R2}_{\mathsf{init}}$, and $\mathsf{R3}_{\mathsf{init}}$.

Example 2: After R1 is clocked once from its initial state, $\mathcal{M}(\mathsf{R1}_1, \mathsf{R1}_6, \mathsf{R1}_{15})$ can be expressed as a quadratic combination of β;

$$\begin{aligned}\mathcal{M}(\mathsf{R1}_1, \mathsf{R1}_6, \mathsf{R1}_{15}) &= \beta_0\beta_5 \oplus \beta_5\beta_{14} \oplus \beta_{14}\beta_0\\ &= \beta_5 \oplus \beta_5\beta_{14} \oplus \beta_{14}.\end{aligned}$$

Since β_0 is fixed to 1, some quadratic terms can be evaluated as linear terms. Moreover, R1 is clocked further, the quadratic combination may also contain a constant 1. The other terms of the keystream generation formula can be similarly expressed as quadratic combinations.

In the keystream generation formula, the input of the majority function always comes from the same LFSR (c.f., (2) of **Keystream Generation Phase**). This indicates that quadratic terms derived from distinct LFSRs do not exist. For example, the term $\beta_1\beta_{43}$ never appears in quadratic combinations. Because the LSBs of $\mathsf{R1}_{\mathsf{init}}$, $\mathsf{R2}_{\mathsf{init}}$, and $\mathsf{R3}_{\mathsf{init}}$ (β_0, β_{19}, and β_{41}) are set to 1, the maximum number of terms used to represent the keystream bits is

$$18 + \binom{18}{2} + 21 + \binom{21}{2} + 22 + \binom{22}{2} = 655.$$

We define a column vector x comprising these 655 linear/quadratic terms and a constant term 1. In addition, we define $\overline{x}$ consisting of the linear terms of x; that is,

$$\begin{aligned}x = (&1, \beta_1\beta_2, \beta_1\beta_3, \ldots, \beta_{17}\beta_{18},\\ &\beta_{20}\beta_{21}, \beta_{20}\beta_{22}, \ldots, \beta_{39}\beta_{40},\\ &\beta_{42}\beta_{43}, \beta_{42}\beta_{44}, \ldots, \beta_{62}\beta_{63},\\ &\beta_1, \ldots, \beta_{18}, \beta_{20}, \ldots, \beta_{40}, \beta_{42}, \ldots, \beta_{63})\end{aligned}$$

$$\overline{x} = (\beta_1, \ldots, \beta_{18}, \beta_{20}, \ldots, \beta_{40}, \beta_{42}, \ldots, \beta_{63})$$

Now, we consider the frame numbers N_i and indicate them on each notation. We can express each keystream bit $z_l^{N_i}$ as a linear combination of terms in x^{N_i}. Therefore, a 208×656 matrix A^{N_i} such that

$$A^{N_i} \times x^{N_i} = z^{N_i} \tag{3}$$

can be found. The inputs in the initialization phase are the session key and frame number (i.e., the 64-bit α'), and all steps are linear operations, except the last step of setting the LSB of each LFSR to 1. Therefore, we can find a 81×19 matrix B such that

$$\beta^{N_i} = \beta^{N_0} \oplus (B \times \Delta N_i), \tag{4}$$

where $\Delta N_i = N_i \oplus N_0$. By Equation (4) and the definition of x, each terms of x^{N_i} can be expressed as a linear combination of terms in x^{N_0}. Therefore, we can find a 208×656 matrix M^{N_i} and rewrite Equation (3) as

$$M^{N_i} \times x^{N_0} = z^{N_i}.$$

Remind Equation (1) and multiply both sides by the parity-check matrix.

$$H \times M^{N_i} \times x^{N_0} = H \times z^{N_i} = r^{N_i}.$$

Concatenate all these equations for $i = 0, 1, \ldots$

$$M \times x^{N_0} = r^*, \tag{5}$$

where

$$M = ((HM^{N_0})^\top || (HM^{N_1})^\top || \ldots)^\top,$$
$$r^* = ((r^{N_0})^\top || (r^{N_1})^\top || \ldots)^\top.$$

It is noted that, in this linearized system of equations, we only need to find 61 terms representing the initial state $(\beta_1, \ldots, \beta_{18}, \beta_{20}, \ldots, \beta_{40}, \beta_{42}, \ldots, \beta_{63})$. We define the reduced row echelon form of M as M_{rref} and the matrix that represents the Gaussian elimination as M_{GE}.

$$M_{GE} \times M = M_{rref}. \tag{6}$$

We experimentally found that an average of 12.25 and a maximum of 13 ciphertexts are sufficient to determine the 61 terms via Gaussian elimination i.e., if 13 ciphertexts are given, M_{rref} must be of the form as follows:

$$\left\{\begin{array}{c|c} M_{(563-n)\times 656} & \\ \hline 0_{61\times 595} & I_{61\times 61} \\ \hline 0_{n\times 656} & \end{array}\right\} \begin{array}{l} \text{Upper Part} \\ \text{Middle Part} \\ \text{Lower Part} \end{array}$$

$*$ The upper part $M_{(563-n)\times 656}$ can be any form while the other parts must be the above forms.

Therefore, we assume that 13 ciphertexts are given hereafter. We further consider the sub-matrices of M_{rref} and M_{GE}.[2] We denote by M_{linear} the 61×656 matrix that consists of 61 rows from M_{GE}'s $(563 - n + 1)$-th row. With Equation (5), it is satisfied that

$$M_{linear} \times M \times x^{N_0} = \overline{x}^{N_0} = M_{linear} \times r^*.$$

Therefore, the corresponding entire initial state can be calculated by determining $\mathsf{R4}_{\mathsf{init}}$; that is, the possible values of β^{N_0} are only 2^{16} for the given 13 ciphertexts $c^{N_0}, c^{N_1}, \ldots, c^{N_{12}}$.

Among the 2^{16} candidates of initial state, we filter the impossible cases using M_{GE}. Suppose that M_{rref} has at least n rows of all zeros in the lower part. We denote by M_{zero} the $n \times 656$ matrix that consists of n rows in the lowest part of M_{GE}. Equation (5) and (6) give

$$M_{zero} \times M = 0_{n\times 656}, \text{ and}$$
$$M_{zero} \times r^* = 0_{n\times 1}.$$

[2]The sizes of M_{rref} and M_{GE} are 624×656 and 624×624, respectively. We reduce their size in Section IV

Therefore, one can filter impossible candidates of $\mathsf{R4}_{\mathsf{init}}$ by finding the corresponding matrix M_{zero} and comparing $M_{zero} \times r^*$ with $0_{n\times 1}$.

Barkan *et al.*'s attack used $n = 16$. Thus, two of the 2^{16} possibilities survived on average, after filtering. We reduce memory complexity by optimizing n in Section IV. From $\mathsf{R4}_{\mathsf{init}}$ that has passed the filtering, the entire initial state can be found using M_{linear}, as described previously. We generate keystreams $k_1, k_2, \ldots$ from the initial states and compare $H \times (c^{N_0} \oplus k_i \oplus s)$ with $0_{48\times 1}$. When the recovered initial state is correct, the results will match. However, with an incorrect initial state, the probability of matching is 2^{-48}; therefore, this probability can be ignored. The session key can be recovered by reversing the initialization phase with the recovered initial state.

We divide the session key recovery into an offline phase and an online phase to significantly reduce the time complexity. Because the matrix M^{N_i} depends only on the initial state of R4 and ΔN_i, we can pre-compute the matrix M^{N_i} in the offline phase. In this phase, we first determine which $(\Delta N_0, \ldots, \Delta N_{12})$ to consider. This decision is covered in Section V. For each $(\Delta N_0, \ldots, \Delta N_{12})$, we construct 2^{16} M corresponding to all the values of $\mathsf{R4}_{\mathsf{init}}$. We calculate M_{linear} and M_{zero} from M and store them. Figure 2 presents the summary of the entire attack process.

When $n = 16$, 2^{16} possibilities of M_{linear} and M_{zero} should be stored in the off-line phase for each $(\Delta N_0, \ldots, \Delta N_{12})$. Therefore, we need $2^{16} \times (61 \times 656 + 16 \times 656) \approx 400MB$ for each $(\Delta N_0, \ldots, \Delta N_{12})$.

IV. MINIMIZING MEMORY COMPLEXITY

In Section III, we present an efficient ciphertext-only attack on A5-GMR-1. In this section, we present novel methods to further reduce memory complexity by optimizing several parameters. These methods reduce memory complexity required for each $(\Delta N_0, \ldots, \Delta N_{12})$. Compared with the attack presented in Section III, the memory complexity can be further reduced by 28% to 46%.

A. REDUCING THE COLUMN SIZE OF M_{GE}

The first method optimizes the column size of M_{GE}. Since M_{linear} and M_{zero} both consist of some rows of M_{GE}, memory complexity can be reduced by minimizing the column size of M_{GE}. The column size of M_{GE} is equal to the number of equations included in the linearized system of equations (row size of M). Therefore, minimizing the number of equations in system M can reduce the memory space for pre-computed M_{linear} and M_{zero}.

Provided that the system $M \times x^{N_0} = r^*$ satisfies the following two conditions, our attack can work.

1) It should be possible to calculate 61 linear terms of x^{N_0} using Gaussian elimination (row operation).
2) The reduced row echelon form of M must have at least n all zero rows.

- **Off-line Phase**

(1) For a given $(\Delta N_0, \ldots, \Delta N_{12})$ and the 2^{16} possible values of $\mathsf{R4}_{\mathsf{init}}$, compute the corresponding matrices $M^{N_0}, \ldots, M^{N_{12}}$ such that

$$M^{N_i} \times x^{N_0} = z^{N_i}.$$

(2) Multiply both sides by the H, and concatenate all HM^{N_i} in row-wise to form matrix M such that

$$M \times x^{N_0} = r^*,$$

where $r^* = ((r^{N_0})^\top || (r^{N_1})^\top || \ldots || (r^{N_{12}})^\top)^\top$.

(3) Find M_{GE} such that

$$M_{GE} \times M = M_{rref},$$

where M_{rref} is a reduced echelon form of M.

(4) Choose $n + 61$ rows of M_{GE} for M_{zero} and M_{linear} such that

$$M_{zero} \times r^* = 0$$
$$M_{linear} \times r^* = x^{N_0'},$$

and store both matrices.

- **Online Phase**

(1) Calculate syndromes from given ciphertexts and concatenate syndromes in row-wise to form r^*.

$$r^{N_i} = H \times (c^{N_i} \oplus s)$$
$$r^* = ((r^{N_0})^\top || (r^{N_1})^\top || \ldots || (r^{N_{12}})^\top)^\top$$

(2) For all 2^{16} possible values of $\mathsf{R4}_{\mathsf{init}}$, do the following.

(a) Load corresponding M_{zero} and calculate $M_{zero} \times r^*$. If the result equals $0_{n\times 1}$, go to step (b). Otherwise, go to the next value of $\mathsf{R4}_{\mathsf{init}}$.

(b) Load corresponding M_{linear} and derive an initial state β^{N_0} from $\overline{x}^{N_0} = M_{linear} \times r^*$. Then, verify the initial state β^{N_0} – generate keystream z with β^{N_0} and check whether $H \times (c^{N_0} \oplus s \oplus z) = 0_{48\times 1}$. If β^{N_0} is verified, the session key is recovered from β^{N_0} by reversing the initialization phase. Otherwise, go to the next value of $\mathsf{R4}_{\mathsf{init}}$.

FIGURE 2. Ciphertext-only attack on A5-GMR-1.

This implies that one can only consider the reduced version of M, rather than the entire column size. Thus, storing distinct column sizes depending on $\mathsf{R4}_{\mathsf{init}}$ and $(\Delta N_0, \ldots, \Delta N_{12})$ becomes more efficient. We apply Gaussian elimination each time we add an equation to the system. Consequently, it was experimentally shown that the average number of equations required for M to satisfy these two conditions is 588 (<624) when $n = 16$.

B. REDUCING THE ROW SIZE OF M_{linear}

The second method optimizes the row size of M_{linear} to reduce memory complexity. α'_{63} $(= \alpha_{48})$ does not influence the cipher owing to the last step in the initialization phase, which sets LSBs of LFSRs to 1 [15]. We define $\overline{\alpha}$ as a 63×1 column vector by excluding α_{48} from α and $\overline{\beta}$ as a 77×1 column vector by excluding the LSBs of Rj from β; that is,

$$\overline{\alpha} = \alpha_0, \ldots, \alpha_{47}, \alpha_{49}, \ldots, \alpha_{63},$$
$$\overline{\beta} = \beta_1, \ldots, \beta_{18}, \beta_{20}, \ldots, \beta_{40}, \beta_{42}, \ldots, \beta_{63}, \beta_{65}, \ldots, \beta_{80}.$$

Then, each bit of $\overline{\beta}$ can be expressed as a linear combination of bits in $\overline{\alpha}$. Therefore, we can find the 77×63 matrix M_{init} that satisfies

$$\overline{\beta} = M_{init} \times \overline{\alpha}.$$

We define the upper 63×63 of M_{init} as $M_{init,up}$, and the upper 63×1 of $\overline{\beta}$ as $\overline{\beta}_{up}$. Because $M_{init,up}$ has full rank, we can find the inverse matrix of $M_{init,up}$ that satisfies

$$\overline{\beta} = M_{init} \times \overline{\alpha} = M_{init} \times (M_{init,up})^{-1} \times \overline{\beta}_{up}.$$

Therefore, from 63-bit of the initial state $\overline{\beta}_{up}$, we can find the entire 81-bit initial state β. Moreover, because the

initial state of R4 is guessed during the attack, the entire initial state can be calculated after only the remaining 47-bit is additionally found. Thus, the row size of M_{linear} can be reduced from 61 to 47.

C. REDUCING THE ROW SIZE OF M_{zero}

The third method minimizes the parameter n. This method trades the time spent in two steps in the online phase. The two steps are (a) and (b) of (2) of Online Phase presented in Figure 2. In step (a), an $n \times 588$ matrix and a 588×1 column vector are multiplied 2^{16} times. In step (b), keystreams that correspond to the surviving values for $\mathsf{R4}_{\mathsf{init}}$ are generated, and the correct initial state can be identified using the parity check matrix.

As n decreases, the memory complexity also decreases owing to the deduction in the row size of M_{zero}. Meanwhile, the time taken for step (a) is shortened, but the time taken for step (b) is increased. In our implementation, the time complexity (*Time*) and memory complexity (*Memory*) according to n are shown in Figure 3. Compared with the case of $n = 16$, $Time \times Memory$ is smaller when $n = 8 \sim 15$. $Time \times Memory$ is minimized when $n = 10$. For $n \geq 3$, the average time required for an attack is less than 1 s. When $n = 0$, the attack takes approximately 5 seconds on average to find the session key.

D. GET THEM ALL TOGETHER

When using all three optimizations, the required memory complexity for each frame number XOR-differences is $2^{16} \times (47 \times 588 + n \times 588) \approx 216 \sim 289MB$ for $0 \leq n \leq 16$, which amounts to 54%∼72% of the previous result. In the case of the third optimization, the complexity analysis could only be performed experimentally because the optimal value of n may vary depending on the implementation (specifically depending on the elapsed times in steps (a) and (b) of (2) of Online Phase in Figure 2). Therefore, we do not consider the third optimization in the rest of the paper because it can be used after the application of the tradeoff techniques.

V. TRADEOFF TECHNIQUES

In this section, we present several complexity tradeoff techniques. Based on these techniques, we can determine the number $(\Delta N_0, \ldots, \Delta N_{12})$ that should be considered in the offline phase. We can then evaluate the exact complexity of our attack. These tradeoff techniques are analyzed under two assumptions. First, the obtained ciphertexts have consecutive frame numbers. We call such ciphertexts a *ciphertext streak*. Second, the frame numbers are uniformly distributed. According to the standard, the frame number has a value between 0 and 313343. Thus, under the second assumption, the probability that the first frame number is i is deduced as

$$P[N_0 = i] = 1/313344 \quad \forall i \in \{0, 1, \ldots, 313343\}.$$

A. MEMORY/DATA TRADE-OFF

Our attack in Section III requires 13 ciphertexts with consecutive frame numbers. Therefore, the 184 XOR-differences

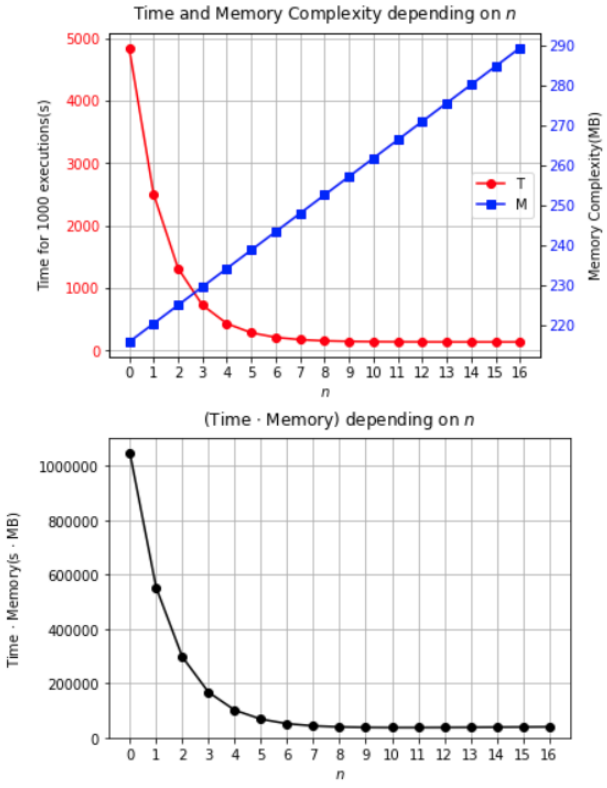

FIGURE 3. Time and Memory complexities depending on n.

of frame number are possible; that is, $|D| = 184$, where $D = \{(\Delta N_1, \ldots, \Delta N_{12}) \mid \Delta N_i = N_0 \oplus (N_0 + i)$ for $N_0 = 0, .., 313343, i = 1, \ldots, 12\}$. Assume that we consider $D' \subset D$ for the offline phase. The total memory complexity is $|D'| \times 289MB$, according to subsection IV-D. Let l be the length of the given ciphertext streak. Our attack can succeed as long as any 13 XOR-differences derived from the l consecutive frame numbers are included in the prepared set D'. Given D' and l, the probability of attack success can be calculated by Algorithm 1. For example, let $D' = \{(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)\}$ and, $l = 27$. The probability of success is 90%.

We also consider the target probability of success, denoted by p. We aim to find D' that, given l and p, makes $\texttt{ProbSucc}(D', l) \geq p$. We present a method to find D' based on the greedy algorithm, as provided in Algorithm 2. Our method does not guarantee that the output D' is the smallest set that satisfies the conditions. For example, let $l = 20$ and $p = 90\%$. Algorithm 2 outputs D' with the size of $|D'| = 15$. This indicates that, if a ciphertext streak of 20 length is given, the attack can be successful with 90% with the output D'. However, this does not imply that it is impossible with a smaller set D' such that $|D'| < 15$. To verify if the size of the output is minimal, we must calculate the probability of success of approximately $\binom{184}{14}$ cases, which is computationally infeasible. The results of Algorithm 2 based

Algorithm 1 Obtaining the Probability of Attack Success, $\texttt{ProbSucc}(D', l)$

INPUT: $D' = \{\Delta\mathbf{N}_1, \Delta\mathbf{N}_2, \ldots\}$, length of ciphertext streak l
OUTPUT: Probability of attack success for given D', l
$S \leftarrow \{\}$
for $N_0 = 0, 1, \ldots, 313343$ **do**
  for $i = 0, 1, \ldots, l - 13$ **do**
    for $j = 1, 2, \ldots, |D'|$ **do**
      if $((N_0 + i) \oplus (N_0 + i + 1), \ldots, (N_0 + i) \oplus (N_0 + i + 12)) = \Delta\mathbf{N}_j$ **then**
        $S \leftarrow S \cup \{\Delta\mathbf{N}_j\}$
      end if
    end for
  end for
end for
RETURN $|S|/313344$

on the length of the ciphertext streak and the target probability of success are shown in Figure 4. *Memory* $\times$ *Data* $\times$ $(1/p)$, which is $|D'| \times l/p$, is minimized when $l = 28, p = 100\%$ and $|D'| = 1$. We use these values as the default settings for our attack parameters.

Algorithm 2 Set of Frame Number XOR-Differences to Pre-Computate

INPUT: length of ciphertext streak l, target probability of success p, set of all possible frame number XOR-differences $D = \{\Delta\mathbf{N}_1, \ldots, \Delta\mathbf{N}_{184}\}$
OUTPUT: D'
$D' \leftarrow \{\}$
while $\texttt{ProbSucc}(D', l) < p$ **do**
  $i = \text{argmax}_i\, \texttt{ProbSucc}(D' \cup \{\Delta\mathbf{N}_i\}, l)$
  $D' \leftarrow D' \cup \{\Delta\mathbf{N}_i\}$
end while
return D'

B. MEMORY/PROBABILITY OF SUCCESS TRADEOFF

According to Driessens *et al.*, a call during 10 seconds provides a ciphertext streak of up to 56 length. Assuming that we obtain a ciphertext streak of length 56, an additional memory/probability of success tradeoff is possible. Let the frame numbers of 56 ciphertexts be $N_0, N_1, \ldots, N_{55}$ and $D' = \{(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12)\} = \{\Delta\mathbf{N}\}$. Then, there are 44 13-length ciphertext streaks. Among them, the number of streaks whose frame number XOR-difference matches $\Delta\mathbf{N}$ is 2 with a 25% probability and 3 with a 75% probability i.e.,

$$\#\{i | (N_i \oplus N_{i+1}, \ldots, N_i \oplus N_{i+12}) = \Delta\mathbf{N}, i = 0, 1, \ldots, 43\} = 2 \text{ or } 3.$$

Therefore, even if only 50% of the result of the pre-computation is stored, the probability of success is

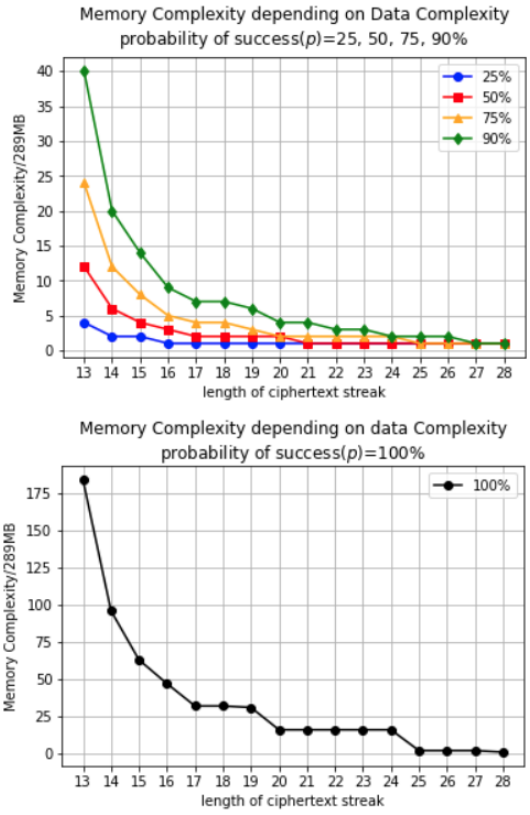

FIGURE 4. Memory/Data complexity for a given probability of success *p*.

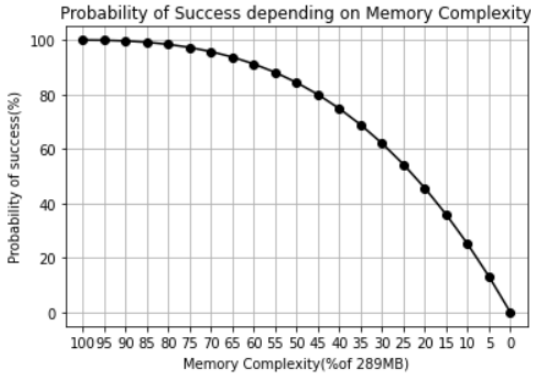

FIGURE 5. Probability of success based on M.

$25\% \times 3/4 + 75\% \times 7/8 = 84\%$ (much more than 50%). Figure 5 shows the probability of success according to the memory complexity.

VI. PLAINTEXT RECOVERY ATTACK NOT USING FRAME NUMBERS

The frame number is not encrypted between satellite communications. However, it is transmitted through a different channel from the encrypted voice data. In the GMR-1 standard, different channels encode data differently. In addition, obtaining the frame numbers of given ciphertexts requires

analyzing multiple channels followed by frame synchronization. The detailed process of obtaining frame number can be found in [14], [16], and [17]. Therefore, a variant attack that does not use frame numbers allows the attacker to concentrate on TCH3 only. It is advantageous when cryptanalyzing a large number of phone calls in real time, and can be an alternative if the frame number is not obtained.

This attack only uses the fact that the obtained ciphertexts have consecutive frame numbers. The original attack selects a 13-length ciphertext streak whose frame number XOR-differences match one element of $D' = \{\Delta\mathbf{N}_1, \Delta\mathbf{N}_2, \ldots\}$. However, this step cannot be performed if the frame numbers of the given ciphertexts are unknown. We repeat the original attack for every 13-length ciphertext streak and element in D'. If the frame number XOR-differences of the ciphertext streak do not match $\Delta\mathbf{N}_i$, none of the initial states of R4 survive after the online phase. Otherwise, the initial state of the first ciphertext in a 13-length streak is recovered.

However, because the frame numbers are unknown, the initialization phase cannot be reversed. Therefore, we cannot find the session key. Instead, using the matrix B defined in Equation (4), the initial state of all ciphertexts can be calculated when the XOR-differences of the frame numbers are known. Keystream can then be easily generated from its initial state. We can find the XOR-differences between frame numbers because the frame numbers of the obtained ciphertexts are consecutive. The frame number is an integer between 0 and 313343. Therefore, 19 XOR-differences are possible between two consecutive frame numbers.

Therefore, if we identify the initial state β^{N_i}, we can calculate 19 possible values for $\beta^{N_{i+1}}$ (and $\beta^{N_{i-1}}$). Among the 19 values, we can use a parity-check matrix to find the correct one, as in the original attack. Similarly, $\beta^{N_{i+2}}$ (and $\beta^{N_{i-2}}$) can be found. By repeating this process, we can recover all the plaintexts.

VII. CONCLUSION

In this paper, we proposed an improved ciphertext-only attack on A5-GMR-1. We successfully adjusted the attack on A5/2 in [12] to A5-GMR-1. Moreover, we presented novel techniques for optimizing attack complexities. To the best of our knowledge, these optimizations allow the best attacks on A5-GMR-1. We also showed that plaintexts can be recovered without frame numbers. However, in this study, we only focused on encrypted voice data transmitted through TCH3. We expect that our novel techniques can also be applied to other channels of the GMR-1 standard, such as FACCH.

REFERENCES

[1] A. Biryukov, A. Shamir, and D. Wagner, "Real time cryptanalysis of A5/1 on a PC," in *Proc. 7th Int. Workshop Fast Softw. Encryption*, London, U.K., Apr. 2000, pp. 1–18.
[2] O. Dunkelman, N. Keller, and A. Shamir, "A practical-time attack on the A5/3 cryptosystem used in third generation GSM telephony," Cryptol. ePrint Arch., Tech. Rep. 2010/013, Feb. 2010.
[3] A. Kircanski and A. Youssef, "On the sliding property of SNOW 3G and SNOW 2.0," *IET Inf. Secur.*, vol. 5, no. 4, pp. 199–206, Dec. 2011.
[4] L. Li, X. Liu, Z. Wang, and F. Li, "An improved attack on clock-controlled shift registers based on hardware implementation," *Sci. China Inf. Sci.*, vol. 56, no. 11, pp. 1–10, Nov. 2013.
[5] H. Wu, T. Huang, P. H. Nguyen, H. Wang, and S. Ling, "Differential attacks against stream cipher ZUC," in *Proc. 18th Int. Conf. Theory Appl. Cryptol. Inf. Secur.*, vol. 7658, 2012, pp. 262–277.
[6] C. Zhou, X. Feng, and D. Lin, "The initialization stage analysis of ZUC v1.5," in *Cryptology and Network Security* (Lecture Notes in Computer Science), vol. 7092. Springer, 2011, pp. 40–53.
[7] B. Driessen, R. Hund, C. Willems, C. Paar, and T. Holz, "Don't trust satellite phones: A security analysis of two satphone standards," in *Proc. IEEE Symp. Secur. Privacy*, May 2012, pp. 128–142.
[8] B. Driessen, "Practical cryptanalysis of real-world systems," Ph.D. dissertation, Dept. Elect. Eng., Ruhr-Univ., Bochum, Germany, 2013.
[9] J. Hu, R. Li, and C. Tang, "A real-time inversion attack on the GMR-2 cipher used in the satellite phones," *Sci. China Inf. Sci.*, vol. 61, no. 3, pp. 1–18, Mar. 2018.
[10] M. Briceno, I. Goldberg, and D. Wagner. (1999). *A Pedagogical Implementation of the GSM A5/1 and A5/2 'Voice Privacy' Encryption Algorithms*. [Online]. Available: http://cryptome.org/gsma512.htm
[11] I. Goldberg, D. Wagner, and L. Green, "The real-time cryptanalysis of A5/2," in *Rump Session of Crypto*, vol. 99, 1999, p. 16.
[12] E. Barkan, E. Biham, and N. Keller, "Instant ciphertext-only cryptanalysis of GSM encrypted communication," in *Proc. 23rd Annu. Int. Cryptol. Conf.*, Santa Barbara, CA, USA, vol. 2729, Aug. 2003, pp. 600–616.
[13] *GEO-Mobile Radio Interface Specifications; Part 5: Radio Interface Physical Layer Specifications; Sub-Part 3: Channel Coding; GMR-1 05.003*, Standard ETSI TS 101 376-5-3 V1.2.1 (2002-04), ETSI, 2002.
[14] *GEO-Mobile Radio Interface Specifications; Part 5: Radio Interface Physical Layer Specifications; Sub-Part 2: Multiplexing and Multiple Access; Stage 2 Service Description; GMR-1 05.002*, Standard ETSI TS 101 376-5-2 V1.2.1 (2002-04), ETSI, 2002.
[15] V. Bhartia and L. Simpson, "Initialisation flaws in the A5-GMR-1 satphone encryption algorithm," in *Proc. Australas. Comput. Sci. Week Multiconf.*, Feb. 2016, pp. 1–7.
[16] *GEO-Mobile Radio Interface Specifications; Part 4: Radio Interface Protocol Specifications; Sub-Part 8: Mobile Radio Interface Layer 3 Specifications; GMR-1 04.008*, Standard ETSI TS 101 376-4-8 V1.2.1 (2002-04), ETSI, 2002.
[17] *GEO-Mobile Radio Interface Specifications; Part 5: Radio Interface Physical Layer Specifications; Sub-Part 7: Radio Subsystem Synchronization; GMR-1 05.010*, Standard ETSI TS 101 376-5-7 V1.2.1 (2002-04), ETSI, 2002.
[18] S. Petrovic and A. Fuster-Sabater, "Cryptanalysis of the A5/2 algorithm," *IACR Cryptol. ePrint Arch.*, to be published. [Online]. Available: https://eprint.iacr.org/2000/052.pdf
[19] A. D. Dwivedi, P. Morawiecki, R. Singh, and S. Dhar, "Differential-linear and related key cryptanalysis of round-reduced scream," *Inf. Process. Lett.*, vol. 136, pp. 5–8, Aug. 2018.
[20] H. Luo, W. Chen, X. Ming, and Y. Wu, "General differential fault attack on PRESENT and GIFT cipher with nibble," *IEEE Access*, vol. 9, pp. 37697–37706, 2021.

DONGJAE LEE is currently pursuing the Ph.D. degree with the Graduate School of Cyber Security, Korea University. His research interests include symmetric cryptography and post-quantum cryptography.

DEUKJO HONG received the B.S. and M.S. degrees in mathematics and the Ph.D. degree in information security from Korea University, in 1999, 2002, and 2006, respectively. From 2007 to 2015, he worked at ETRI. He is currently working as an Associate Professor with the Department of Information Technology and Engineering, Jeonbuk National University. His research interest includes symmetric cryptography.

JAECHUL SUNG received the Ph.D. degree in mathematics from Korea University, in 2002. From July 2002 to January 2004, he worked as a Senior Researcher at the Korea Information Security Agency (KISA). He is currently a Professor with the Department of Mathematics, University of Seoul. His research interests include cryptography, symmetric cryptosystems, hash functions, and MACs.

SEONGGYEOM KIM received the M.S. degree in information security from Korea University, in 2018, where he is currently pursuing the Ph.D. degree with the Graduate School of Cyber Security. His research interests include symmetric cryptography and random-number generators.

SEOKHIE HONG received the M.S. and Ph.D. degrees in mathematics from Korea University, in 1997 and 2001, respectively. He worked with Security Technologies Inc., from 2000 to 2004. Subsequently, he was a Postdoctoral Researcher with COSIC, KU Leuven, Belgium, from 2004 to 2005, after which he joined the Graduate School of Cyber Security, Korea University. His research interests include cryptography, public and symmetric cryptosystems, hash functions, and MACs.