Rainbow table attack on 3rd generation GSM Telephony secure algorithm - A5/3

T Anand, M Shanmugam, B Santhoshini

ABSTRACT--- ***GSM is a digital cellular network standard to send the customer's data or voice through the air in mobile communication. GSM standard spreads over more than 80% of population in all over the world. The security of customer's data is protected in the GSM by A5 family of cryptosystem. We are working on A5/3 cryptosystem used by the 3rd generation GSM for transmitting secured information through the air. The A5/3 cryptosystem is a stream cipher and having a key generator based on KASUMI block cipher. The A5/3 acccepts 64-bit input and gives a pair of 114-bit block output under the control of 128-bit key. Because of the large key space for the A5/3, we decided to work on a reduced version of the A5/3, called T5/3. The T5/3 accepts 32-bit input and gives a pair of 64-bit block output under the control of 64-bit keys. We are using TMTO(Time Memory Trade Off) technique to attack the T5/3 cipher. Rainbow table attack is a TMTO based technique and is feasible for the T5/3 cryptosystem. There are two phases, the offline precomputation phase and the online lookup phase in Rainbow table attack. The precomputation phase is a time consuming process and the lookup phase is a real time process which retrives the key used for the T5/3 cryptosystem. We have generated different sized Rainbow table and successfully attacked the T5/3 cipher. We have analyzed different parameters used in the Rainbow table attack like Distinguish Point(DP), Reduction Function(RF) and Collision. The Reduction function is a mapping from cipher text to a key in the keyspace. The Reduction function doesn't have much significance in the chainlength and the collision in Rainbow table. Distinguish points are certain conditions which allows the reduction in time for the searching of key in lookup phase. If the DP value is more then the chianlength, collision and time to generate the Rainbow table are also increases.***

Index Terms: GSM, Cryptosystem, Rainbow table attack, Ciphertext, A5/3

1. INTRODUCTION

GSM (Global System for Mobile Communications) is a wireless communication network standard and uses digital cellular technology for sending voice or data services through the air[1]. The inception of the GSM standard was done by the European Conference of Post and Telecommunication Administrations(CEPT) in 1982, proposed a new European cellular standard which can replace the existing cellular systems. In order to accomplish a new standard, they(CEPT) formed a group called Group Speciale Mobile(GSM).Finally, GSM have started the service in the European market in 1991, then more network operators are willing to accept the new standard and rechristened Group Speciale Mobile to Global System for Mobile Communication in 1992. Currently, around 4.4 billion subscribers across more than 219 countries using the GSM standard for mobile communication. The key features of the GSM communication are given below.

International Roaming Better Speech quality

New services : Short Message Service(SMS), Call waiting, Call forwarding, etc. High-level security mechanisms

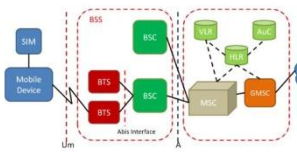
Packet Radio Service (GPRS).

GSM provides high-level security mechanisms. These security mechanisms take care of the privacy of customer's data or voice and the integrity of the cellular network. The authentication between customer and network provides the integrity of the network. The encryption procedure provides security to the subscriber's data or voice.

1.1. GSM Security Structure

The GSM provides a high-level security mechanism for both subscriber and network operator. Every subscriber has to authenticate to the network prior to doing services given by the GSM. Once the authentication process was successful, then the network operator and the subscriber are able to start generating the session key for the encryption and the decryption. The GSM security structure is given below Fig[1].

2



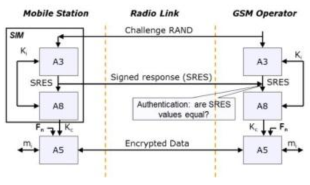


Fig. 1. GSM security structure

Fig. 2. GSM Authentication & Encryption

To enable the mobile communication, a cell phone should have an active SIM(Subscriber Identity Module) card. The SIM card consists of A3 and A8 algorithm along with an encryption key Ki and an IMSI(International Mobile Subscriber Identity).Ki is a 128 bit randomly generated number which known as root encryption key. Ki present in SIM card as well as Authentication Center(AuC)in the network operator also. A3 provides authentication between mobile device and network operator, and A8 is used for session key agreement. The session key is valid upto only one hour, so most of the voice calls generated from 3G mobiles disconnect every one hour to change the session key. The authentication process is based on the challenge-

Revised Manuscript Received on February 11 , 2019.

T Anand, CSE, VFSTR University, AP, India. (E-mail: aanand.t20@gmail.com)

M Shanmugam, CSE, VFSTR University, Guntur, AP, India. (E-mail: shaninfo247@gmail.com)

B Santhoshini, CSE, VFSTR University, AP, India (E-mail: santhoshini.banda@gmail.com)

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

response communication. Both algorithms are using symmetric key cryptography(same key is used for the encryption and the decryption of data between mobile device and network).Kc is used for encryption and decryptions which is generated by the A8 algorithm present in the SIM card with the help of the root encrytpion key Ki, and it is 64 bit in size. The overall mechanism is mentioned in Fig[2]. In GSM communication, every 4.6ms a sequences of 114-bit frames sent from Mobile to Network and vice versa. A 24-bit value given to every frame, known as Frame Number. Using session key (Kc), A5 generates an 114-bit keystream which is XORed with a 114-bit plaintext to obtain the ciphertext. The A5 algorithm implemented in the Mobile handest hardware. The A5 algorithm has three different variants namely A5/1, A5/2 and A5/3. A5/3 is a stream cipher based on a block cipher named as KASUMI whereas, A5/1 and A5/2 are LFSR(Linear Feedback Shift Register) based stream ciphers.

2. A5/3 ENCRYPTION PROCEDURE

A5 family of cryptosystem is used for the security of data used in GSM mobile communication. The A5/3 ciphersystem is used in 3G GSM mobile communication. It is a stream cipher with KASUMI as a keystream generator. The inputs to the A5/3 are 22-bit COUNT value and 64-bit key from the A8 algorithm. The GSM A5/3 algorithm produces two 114-bit keystream strings, one of them is used for the uplink encryption/decryption and the other is used for downlink encryption/decryption.

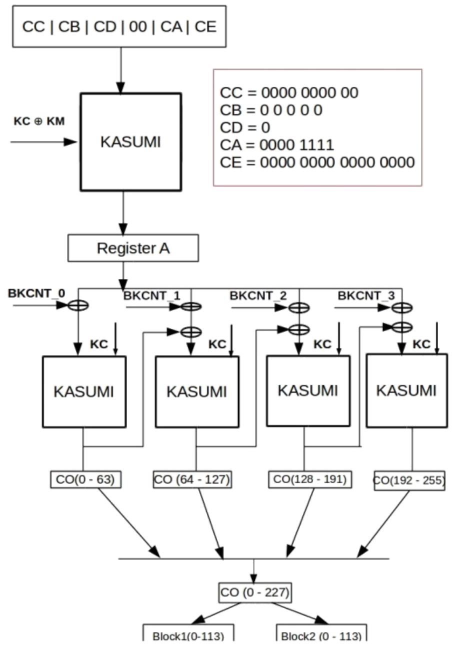


Fig. 3. Block Diagram of A5/3 using five KASUMI elements

2.1. KASUMI Cipher[1]

KASUMI is an eight round Feistel structure block cipher. Feistel ciphers is based on the principle that each state U_i is divided into two halves L_i and R_i.

The round function g has the following form:

$$g(L_{i\ \ 1}, R_{i\ \ 1}, K_i) = (L_i, R_i)$$

where

$$L_i \,{}^{=} R_{i\ 1}$$

$$R_i = L_{i\ \ 1} g(R_{i\ \ 1}, K_i)$$

KASUMI cipher produces a 64-bit output from a 64-bit input under the control of a 128-bit key. KASUMI contains mainly two functions FL and FO in each round. In odd rounds the data is passed through FL function first and then FO function whereas in even rounds FO function first then FL function. The function FO is again a three round Feistel structure and function FI is used in each round. The function FI has four S-box operations. The implementation result is verified with the KASUMI test data[2]. The block diagram of KASUMI is in Fig[4]

2.2. KASUMI

2.3. Key schedule[1]

KASUMI consists of 128-bit key K which is obtained from Kc by appending Kc to itself. Each round of KASUMI uses 128 bits of subkeys that are derived from K. The round subkeys are derived linearly from the original key K. Table I represents the each round subkeys used in the KASUMI. 128-bit key is subdivided into eight 16-bit values K1...K8 where,

$$K = K_1 \text{ jj } K_2 \text{ jj } K_3 \text{ jj } K_4 \text{ jj } K_5 \text{ jj } K_6 \text{ jj } K_7 \text{ jj } K_8$$

Algorithm 1 KASUMI

```
 1: Input : 64-bit plaintext (P) : $L_0$ $R_0$ and Key = K
    Output : 64-bit output (C) : $L_g$ $R_g$
 2: Derive sub keys $KL_i$, $KO_i$ and $KI_i$ for rounds (0 <
    i   8) from key K.
 3: for i = 1 to 8 do
 4:     if  i == odd then
 5:         $L_i = FO(FL(L_{i\ \ 1}, KL_i), KO_i, KI_i)\ \ R_{i\ \ 1}$
 6:         $R_i = L_{i\ \ 1}$
 7:     else
 8:         $L_i = FL(FO(L_{i\ \ 1}, KO_i, KI_i), KL_i)\ \ R_{i\ \ 1}$
 9:         $R_i = L_{i\ \ 1}$
10:     end if
11: end for
12: return  $(L_g, R_g)$
```

KASUMI key schedule derives the following operation:
for each integer j with 0 < i   8
$K^0_j = K_j \quad C_j \quad C_i$: known and fixed constants.
The round keys are derived from K_j and K^0_j in Table I.

Rounds	$KL_{i;1}$	$KL_{i;2}$	$KO_{i;1}$	$KO_{i;2}$	$KO_{i;3}$	$KI_{i;1}$	$KI_{i;2}$	$KI_{i;3}$
1	K_1 n 1	K_3^0	K_2 n 5	K_6 n 8	K_7 n 13	K_5^0	K_4^0	K_8^0
2	K_2 n 1	K_4^0	K_3 n 5	K_7 n 8	K_8 n 13	K_6^0	K_5^0	K_1^0
3	K_3 n 1	K_5^0	K_4 n 5	K_8 n 8	K_1 n 13	K_7^0	K_6^0	K_2^0
4	K_4 n 1	K_6^0	K_5 n 5	K_1 n 8	K_2 n 13	K_8^0	K_7^0	K_3^0
5	K_5 n 1	K_7^0	K_6 n 5	K_2 n 8	K_3 n 13	K_1^0	K_8^0	K_4^0
6	K_6 n 1	K_8^0	K_7 n 5	K_3 n 8	K_4 n 13	K_2^0	K_1^0	K_5^0
7	K_7 n 1	K_1^0	K_8 n 5	K_4 n 8	K_5 n 13	K_3^0	K_2^0	K_6^0
8	K_8 n 1	K_2^0	K_1 n 5	K_5 n 8	K_6 n 13	K_4^0	K_3^0	K_7^0

TABLE I
KASUMI KEY SCHEDULE IN A5/3

Note: X n i : X rotated to the left by i bits

3. EXISTING WORK

There are several popular attacks available nowadays such as brute force attack, codebook attack and algebraic attack. But none of these attacks are practically applicable to A5/3. Beacuse, the brute force attack checks every single possibility of the value,i.e., if K bits is the input size then brute force attacks checks 2^K 1 possibilities which are very expensive. A code book attack is a technique for cryptanalysis. The name comes from the attack on a block cipher; the attacker tries to build up a code book, a table saying which ciphertexts correspond to which plaintexts. There is also a variant usable against a stream cipher; the attacker attempts to build a listing of the entire output stream, until it repeats. For example, consider a block cipher with only an 8 bit block size. Assume the enemy is able to get or guess some plaintext; he makes a little table, his own code book showing which ciphertext blocks correspond to which plaintexts. If he can fill in all 256 entries, then the cipher is broken; he can read everything ever sent with that key. An algebraic attack is a very recent cryptanalytic technique which reduces the cryptanalysis of the attacked cryptosystem into problem of finding and solving a system of polynomial equations. Therefore the problem of solving a system of nonlinear equations over finite field is close to the algebraic attack. As we see, one attack requires a lot of time and another large memory. So, a trade-off between time and memory could solve our problem which commonly known as TMTO(Time-Memory Trade Off). The concept of TMTO is more like codebook attack but with a complex data structure.

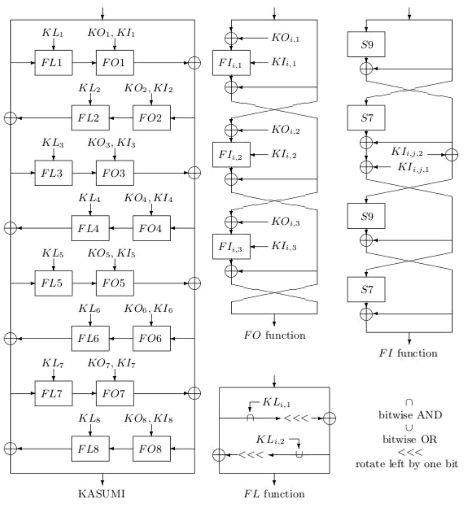


Fig. 4. KASUMI block diagram

3.1. Time Memory Trade-Off Attack

TMTO is a combination of brute force attack and codebook attack. Suppose a cryptosystem has k-bits of key and keyspace is N (2^k). In brute force attack, the attacker tries all the possible combination of keys to find a match with given ciphertext. The computational time T(time taken to perform operations) and memory M(required to store different solutions) for the above attack is (2^k) and 1 respectively. In the case of code book attack, the attacker precomputes the all possible pairs of key (K) and cipher text (C) and stored in a table. Whenever a ciphertext is given to it, the attacker can easily find the key for the corresponding ciphertext. The computational time and memory required for this attack are 1 and (2^k) respectively.

In 1980, Hellman [3] proposed the idea of TMTO technique to cryptanalysis a cryptosystem better than brute force attack and codebook attack. He proposed that TMTO attack can cryptanalyse any N key space cryptosystem with $N^{2=3}$ computational time and $N^{2=3}$ words of memory. The TMTO attack is a known plaintext attack, the attacker has the access to the cryptosystem and knows some part of the plain text for the encryption.

Precomputation phase and lookup phase are the two phases in TMTO attack. In precomputation phase, the attacker analyze the structure of the cryptosystem and summarizes results in a huge table. In precomputation, a

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

function operation(f) includes an encryption and a reduction func-tion(RF) operation. The reduction function is a mapping function which maps a ciphertext to a key in the keyspacet. In order to create table, chooses a random key from the keyspace called Starting Point(SP). Then, apply SP to the function(f) and operates for 't' operations to make a chain and final result is stored along with the SP. The final result of the chain is called End Point(EP). In order to reduce the memory requirements, the intermediate values in a chain are not storing in the table. The pair values(EP and SP) are sorted based on EP and stored in the huge table. In lookup phase, the attacker gets a cipher text and apply reduction function to it and then checks for any matching entry in the table. If it is not in the chain, applies another function operation(f) on the result and so on. If a matching occurs with the EP, then recomputes the corresponding chain from the SP and checks the key from the chain. If a matching occurs in the table with EP does not imply that he can find the key from the table. This situation is called false alarms. Creation of Hellman chain is shown in fig 5.

In 1982, Rivest [4] proposed some improvements to the Hellman TMTO method. He suggested the idea of distinguishing points(DP) as endpoints(EP). Distinguish points(DP) are certain conditions which generate the chain until satisfying with DP constraints. The chain generation stops when DP condition is matched so that there is no concept of fixed chain length. But we can fix the range of chain length by giving minimum and maximum chain lengths. Some chains may fall into loops and never reaches Distinguish point, then we can remove that chain from the table with the help of DP. The main advantage of DP is the reduction of time required for table lookups in the online phase. When a DP condition is satisfied then only the table lookup process be initiated. The idea of DP method reduces the computational time for the online phase to $N^{1=3}$.

In 2003, Oechslin [5] introduced a new TMTO method with Rainbow tables. He is using different reduction functions for each position in the chain. For example, if a chain length is t, then there should be t 1 reduction functions are used. In Hellman chain, if there is a collision occurred in a chain leads to merging. The merging of chains reduces the probability of success. But in Rainbow chain, merging occurs when two chains collide at the same position in the same table. If collisions occurring in different positions of the chain may not be lead to merging. If the chain size is 't' and the probability of collision leads to merging is 1=t. There are some advantages of Rainbow table, The number of table lookups reduced. Rainbow chains have no loops. Rainbow chains have fixed length. Merging of chains are easily detectable.

If the key space of a cryptosystem is N. The computational time and memory required for Rainbow Table attack are $N^{2=3}$ and $N^{1=3}$ respectively.

In 2011 a mainstream desktop PC with a high-end graphics card and flash memory of size in some Terabytes was enough to break the A5/1 GSM algorithm in few seconds with a success rate over 90%. Traditionally, to reverse password hashes the Rainbow Tables are used but their application against A5/1 algorithm, have shown a new area of exploitation.

3.2. FPGA based Rainbow Table Attack

In 2013, Papantonakis [6] proposed an attack against A5=3 with fast FPGA-based Rainbow Table attack. They have created an FPGA-based architecture for the efficient creation of Rainbow Tables for A5=3 cipher. They cryptanalyses A5=3 under known plaintext attack. Unavailability of large memory requirements, they did the attack on a reduced version of A5=3.

3.3. Fault injection Attack on A5/3

This attack [7] is based on the fault assumption in [8]. Injecting some faults to the A5=3 system, it reduces the number of rounds in block cipher KASUMI of A5/3 and can recover the session key of A5/3 supporting a 64-bit session key. The A5/3 uses the 64 to 128- bit session key. In this paper, the attack applied to A5/3 supporting a 64-bit session key. Their attack can recover a session key of A5/3(64-bit version) by using a small number of fault injections. They required only one fault injection and about $2^{45:44}$ KASUMI encryptions in order to recover the session key of A5/3. This attack is a known plaintext attack.

3.4. Key classification attack on block ciphers

In this paper [9], they have performed the security analysis of block ciphers which has key length greater than the block length. When the key length is significantly greater than block length, there is more than one key which maps fixed input to fixed output. Using some classes of keys, their proposed algorithm would be able to recover the key of the block cipher with complexity $O(max[2^n,2^{k\ n}])$ where n is block length and k is the key length. They applied their algorithm to 2-round KASUMI block cipher as a sample block cipher. KASUMI block cipher is a 64-bit block length and accepts 128-bit key length. Based on birthday paradox, when a fixed plaintext encrypts with 232 different keys.There should be two equal cipher text with the probability of 1=2. For 6-round KASUMI, they found three keys with the same ciphertext. They find out more non-random properties of KASUMI and it can be the vulnerability for KASUMI block cipher.

3.5. Differential Attacks on KASUMI

Here based on differential characteristics of KASUMI cipher cryptographers tries to attack KA-SUMI.

Higher-Order Differential Attack [10] Higher order differential is a versatile attack on block ciphers. This attack is a chosen plaintext attack. This attack is based on the fact that the value of higher order differential of the output is independent of key. In this paper, they successfully cryptanalysed the 4-round KASUMI excluding FL function with 2nd order differentials. They determine 6-sub keys from 1416 chosen plaintext. The computational complexity is $2^{22:11}$ times function FO. In

2007, Nobuyuki Sugio [11] improved this attack to five round KASUMI with the complexity is equal to 2^{22} chosen plaintext and 2^{63} KASUMI encryption.

Impossible Differential of 6 round KASUMI [12] It was proposed by Ulrich Kuhn in 2001. In this paper, they present attacks on reduced-round variants of MISTY1 and MISTY2. They are considering above mentioned ciphers without and with the key-dependent linear functions FL. The attack is a combination of collision searching technique [13] and impossible differentials [14]. The round function involves a large amount of keying operations so that they pointed out properties of the round function. In order to improve the attacks on the cipher, they tried the divide and conquer techniques on the subkeys. Finally, they extended this attack to reduced 6 round KASUMI. The computational complexity is 2^{100} encryption time and 2^{55} data.

Impossible Differential of 7 round KASUMI [15]

In this paper, they improved the attack on two different version of KASUMI with first 7 rounds(rounds 1 - 7) of KASUMI and last 7 rounds(rounds 2 - 8) of KASUMI. In this attack, they treat the function FI as a big S-box and construct difference distribution table for 16-bit subkey KI. The new impossible differential attack on the last 7 rounds needs $2^{114:3}$ encryptions with $2^{52:5}$ chosen plaintexts. For the attack on the first 7 rounds, the data complexity is 2^{62} known plaintexts and the time complexity is $2^{115:8}$ encryptions.

3.6. A Single-Key Attack on 6-Round KASUMI

In 2010, a related-key attack on full KASUMI was reported [1]. The attack is in practical time complexity. The assumptions made for this attacks were impractical so that this was not direct a threat to full KASUMI. In order to generate a single key attack in practical time complexity, this paper [16] extends the single key attack to the six rounds of KASUMI. The attack applies a technique of higher order differential attacks and uses 48th order differential. In terms of time comlexity, this attack is better than six round Integral-Interpolation attack and impossible differential attack. The data and time required for this attack is $2^{60:8}$ data and $2^{65:4}$ encryption time. Summary of different Single Key attacks are tabulated in Table II

Number of Rounds	Complexity		Method used for
	Data	Time	
5	$2^{22:1}$	$2^{60:7}$	HOD attack
5	$2^{28:9}$	$2^{31:2}$	HOD attack
6(2-7)	2^{48}	$2^{126:2}$	II attack
6(2-7)	2^{55}	2^{100}	ID attack
6(2-7)	$2^{60:8}$	$2^{65:4}$	HOD attack

TABLE II
SUMMARY OF SINGLE KEY ATTACK

HOD : Higher Order Differential
II : Integral Interpolation
ID : Impossible Differential

3.7. Related Key Attacks on KASUMI

Related-key attacks were introduced by Biham [17]in 1993. This attack is a Chosen Plaintext attack, the attacker is allowed to choose a plaintext and encryption of selected plaintext. In this attack they are using two keys, one key related other by some known way. The attacker is able to request the encryptions of plaintexts under two related keys. The attacker uses the relations between the keys and finds various weaknesses of the cipher to derive information about the two keys. In the related-key differential attack[18], they generated a four round differential which helped them to extend this attack to six round KASUMI. In this attack, the attacker selected different pairs of plaintexts which hold a common difference under two related keys. They have chosen 2^{19} plaintext pairs and common difference for the related keys is differ by only one bit. They attacked five round KASUMI with 2^{15} chosen plaintext encrypted under the original key, 2^{19} chosen plaintext encrypted under the related key. The time complexity required for five round KASUMI is 2^{33} encryptions. They extended this attack to six round KASUMI. The complexity of this attack was encryption of 3 $2^{18:6}$ chosen plaintext under a chosen related key, 3 2^{13} chosen plaintexts under the original key and $2^{113:5}$ trial encryptions of 6 round(1 -6) KASUMI.

The related-key boomerang attack was presented by Kim[19] and independently by Biham[20]. This attack is a combination of the boomerang attack and related-key differential attack. In this attack, the attacker treated the cipher as a cascade of two sub-ciphers $E = E_0\ E_1$. Then, they find the related key differentials of E_0 and E_1. They combined the above result with adaptive chosen plaintext and ciphertext to form a distinguisher for E.

In 2013, Orr Dunkelman [1] introduced a practical time attack on full round KASUMI. In this attack, the cipher is treated as a cascade of three sub-ciphers. They introduced a new middle layer called the sandwich. The arrangement of cipher is as follows; $E = E_0\ M\ E_1$. They constructed a simple distinguisher for seven rounds KASUMI with the high probability of $2^{\ 14}$. The attack required four related keys, 2^{26} data, 2^{30} memory and 2^{32} encryption time. They recovered the 96 bits of keys in few minutes and 128 bits of keys in less than two hours on single core PC.

The summarized related key attack is tabulated in Table[III]

Attack	Number of	Number of	Complexity

	Rounds	Keys	Data	Time
RK Differential	6	1	$2^{18:6}$ RK-CP	$2^{113:6}$
RK Boomerang Distinguisher	6	4	768 RK-ACPC	1
RK Boomerang Key Recovery	6	34	2^{13} RK-ACPC	2^{13}
Basic RK Attack	8	4	2^{53} RK-CP	2^{102}
Improved Basic RK Attack	8	4	$2^{54:6}$ RK-CP	$2^{76:1}$
RK Boomerang	8	4	$2^{45:2}$ RK-ACPC	$2^{78:7}$
RK Sandwich	8	4	2^{48} RK-ACPC	2^{32}

TABLE III
SUMMARY OF RELATED KEY ATTACK

RK: Related Key CP : Chosen Plaintext
ACPC :Adaptive Chosen Plaintext and Ciphertext

4. PROTOTYPE IMPLEMENTATION

We have implemented the A5/3 prototype and verified with the test cases[21]. The A5/3 accepts 22-bit frame number produces a pair of 114-bit blocks under the control of 128-bit key(Kc). The keyspace for A5/3 is 2128. The time and memory required to cryptanalyse the A5/3 with Rainbow table attack are 2^{85} and 2^{42} respectively. These complexities are impractical for an academic project, so we planned to work on a reduced version of the A5/3 which is called T5/3. We have made some changes in the parameters like input, keyspace, KASUMI and output which is given in Table VI. We did not make any changes in the internal structure design of the original A5/3 cryptosystem. Operation of T5/3 is similar to A5/3 and the changes applied only to the values used in T5/3.

Change in field	**A5/3**	**T5/3**
Input	64–bit	32–bit
Key	128–bit	64–bit
Output	Two blocks with 114–bits	Two blocks with 64–bits
Fixed input values	CA 8-bit(00001111)	CA 4-bit(0011)
	CB 5-bit(00000)	CB 2-bit(00)
	CC 10-bit(all zeroes)	CC 5-bit(all zeroes)
	CD 1-bit(0)	CD 1-bit(0)
	CE 16-bit(all zeroes)	CE 8-bit(all zeroes)
Input format	CC CB CD 00 CA CE	CC CB CD 0 CA CE
Count	22-bit	11-bit
Change in field	A5/3	T5/3
Input	64–bit	32–bit
Key	128–bit	64–bit
Output	64–bits	32–bits
Key schedule	$KO_{i;1}$: n 5	$KO_{i;1}$: n 2
	$KO_{i;2}$: n 8	$KO_{i;2}$: n 3
	$KO_{i;3}$: n 13	$KO_{i;3}$: n 4
Fixed Constant in Keyschedule	16-bit	8- bit
S-box	Bit logic and Lookup table	Prime field
S-box size	9-bit and 7-bit	5-bit and 3-bit

TABLE IV
CHANGES IN THE MAIN FUNCTION FOR T5/3

4.1. S boxes

The input to the S-box is X and output from S-box are Y. There are two S-boxes with 5 bits and 3 bits are used. The S-box design is based on the Galois field. The S-box design is given below

$$Y = (MI(input)\ \&\ AM\ ) \quad C \bmod N$$

where,
MI : Multiplicative Inverse in prime field N
AM: Affine Matrix
C: Fixed constant value N : Prime Number

The operations included in S-box generation are based on GF(2). The Affine Matrix used for S-box generation is a rotational mtirx based on value 1316. The Multiplication is done in GF(2)with logical AND operation and Addition in GF(2) with xor operation. The Affine Matrix is given below.

$$\text{AffineMatrix(AM)} = \begin{array}{llllll} 2_1 & 1 & 0 & 0 & 1 & 3 \\ 1 & 0 & 0 & 1 & 1 & \\ 6 & & & & & 7 \\ 1 & 1 & 1 & 0 & 0 & \\ 6_0 & 1 & 1 & 1 & 0 & 7 \\ 6_0 & 0 & 1 & 1 & 1 & 7 \\ 6 & & & & & 7 \\ 4 & & & & & 5 \end{array}$$

The S box generation is as follows. The input is given to find the inverse. The inverse value is arranged in such way that the LSB bit comes first. This value is multiplied(logical AND) with each row in the affine matrix. The logical AND operation is performed between arranged inverse value and each row in the Affine Matrix. Finally, all bits are XORed against each other within that row to generate the transformed bit for that row. Lookup table for S_3 and S_5 S boxes are given below.

S3 = f 6, 4, 2, 7, 3, 5, 0, 1g;
S5 = f 16, 19, 22, 13, 28, 25, 11, 0,
8, 26, 2, 10, 7, 27, 17, 30,
1,14, 4, 24, 21, 29, 5, 23,
31, 20, 6, 3, 18, 9, 12, 15g;

We have verified the S-box with the design criteria of S-box with [22]. Some properties of S-box is given below

Each S-box is a permutation of integers with a corresponding range

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

of values. S-box is not a linear or affine function of the input.

Changing one input bit to an S-box results in changing at least half of the bits in output. The success percentage is equal to 87.5%.

S(x) and S(x 01100) or S(x 00110) must differ by at least two bits. The success percentage is equal to 87.5%.

The S-boxes were chosen to minimize the difference between the number of 1's and 0's when an input bit is held constant.

5. EMPIRICAL ANALYSIS & RESULTS

The Rainbow Table attack is a TMTO method and consists of a precomputation phase and real time lookup phase. In precomputation phase, we are using different parameters like Reduction Function(RF), Distinguish Points(DP), Staring Point(SP)and End Point(EP).

5.1. Precomputation Phase

The precomputation alogrithm is working as follows. We are initially setting the RF function(32 values[0 - 31]), DP function and number of entries required in the Rainbow Table(M). Select a Random value(32-bit) from the key space and feeds this into T5/3 algorithm. The result of T5/3 is exored with RF function, stores in a temporary register and cheking the register value with DP function. If a match occurs, then change the RF function. The 32nd DP value is the end point(EP). SP, EP and chainlength values are stored in a file and once M entries generated then the algorithm stops. This file is sorted based on the EP values to make the lookup phase fast.

Algorithm 2 Rainbow Table Precomputation Phase

```
 1: Input : Reduction Function(RF), No of entries
    required in Rainbow Table(M), Distinguish
    Point(DP) Output :Rainbow Table with M
    entries
 2: Set $C_1$ = 1 and $C_2$ = RF index = 0.
 3: while $C_1$    M do
 4:    Set SP = Z = rand(), chain length = 0;
 5:    while  $C_2$ != 32 do
 6:       Z = T5/3(Z)   RF($C_2$);
 7:       chain length ++;
 8:       if Z == DP then
 9:          $C_2$ ++;
10:       end if
11:    end while
12:    Store values SP, Z, chain length to Rainbow
       Table;
13:    $C_1$ ++;
14: end while
```

5.2. Look Up Phase

In look up phase, we are searching the key for the given ciphertext in practical time complexity. We have generated a large Rainbow Table with SP, EP and chainlength entries and sorted based on EP values. Whenever a ciphertext is given to this phase, apply Reduction Function to the ciphertext to generate the key in the keyspace. We are using the Reduction Function reversely. Apply the 32nd Reduction Function initially then 31st and 32nd and so on. The cipher text is exored with the RF function and checks for matching with DP condition. If a matching occurs with DP and RF index value is 32 then start searching in the Rainbow Table for the DP value. If not, then reduce the RF index value and repeat the process. We are using binary search to reduce the time required for the DP searching with EP . If a matching occurs with EP then starts recomputing the corresponding chain by getting SP from the Rainbow Table. If a match with EP occurs, we can conclude that the key may or may not be present in the chain. After getting SP, starts generating the chain and this time after each operation in the chain generation process check for a match with the ciphertext. If it occurs then store the previous value, which will be the key. The algorithm for the Lookup is given in algorithm[3]. The algorithm for the regeneration of chain is given in algorithm[4].

5.3. Analysis

In this phase, we have analysed the behaviour of RF, DP, EP, SP, collison and chainlength under different conditions. Collision is an important parameter in the rainbow table attack. In Hellman chain[3], there is only one reduction function and if there is any collision happens in the chain, it leads to a merging and results in false alarm. If a EP has more than one preimage of SP's, then it is called false alarm. The collision which leads to false alarm reduces the search space in the table. In Oechslin chain [5], there are different set of reduction functions are used in every point of the chain. The probability of collision leads to false alarm is less in rainbow chain. If the collision happens at the same position of two chains lead to flase alarm. But, if the collision happens at the different positions in the chain may not be lead to merging in rainbow tables. Different set of reduction functions are used in the rainbow table so that it maps the collisions in different positions to different values. If the collisions lead to flase alarm is more in rainbow chain, results in reduction of the searching space for the key.

5.3.1. Time Analysis

Timing analysis was carried out on generation of Rainbow Table and lookup on T5/3. We have generated three Rainbow Table with different DP functions and with different set of Reduction

Algorithm 3 Rainbow Table Lookup Phase

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

```
1: Input : Ciphertext, Rainbow Table(SP, EP)
   Output :Key 64-bit
2: Set $C_2$ = RF index = 32, RF = Reduction
   Function, DP = Distinguish Point, Z =
   ciphertext
3: X = Z   RF($C_2$)
4: while flag EP == 0 do
5:     temp index = RF index, flag DP = 0;
6:     while  flag DP == 0 do
7:         Z = T5/3(Z)   RF($C_2$);
8:         chain length ++;
9:         if X == DP then
10:            flag DP = 1;
11:        end if
12:        X = T5/3(X)   RF(temp index);
13:        if flag DP == 1 && temp index ¡ 32 then
14:            flag DP = 0;
15:            temp index ++;
16:        end if
17:    end while
18:    Check X in Rainbow Table with EP;
19:    $C_1$ ++;
20:    if X == EP then
21:        get corresponding SP;
22:        flag EP = 1;
23:    else
24:        RF index –;
25:        if temp index == 32 then
26:            X = Z   RF index ;
27:        end if
28:    end if
29: end while
30: Recompute chain(SP, ciphertext);
```

Algorithm 4 Rainbow Table Recomputation Phase

```
1: Input : SP, ciphertext Output :Key 64bitInput :
   SP, ciphertext Output :Key 64bit
2: Set Z = SP , $C_2$ = RF index = 0;
3: while  $C_2$ != 32 do
4:     temp key = Z;
5:     Z = T5/3(Z);
6:     if Z == ciphertext then
7:         key = temp key;
8:     else
9:         Z = Z   RF($C_2$);
10:        if Z == DP then
11:            $C_2$ ++;
12:        end if
13:    end if
14: end while
15: return  (key)
```

Functions. The different DP functions are last 8 bits, 12 bits and 16 bits equal to zero. We have noted down the time required to generate the Rainbow Table in Table V.

DP function	No of Entries	Time(Sec)	Size(MB)
DP_8	10000	646	1
	100000	6468	10
	1000000	62354.08	104
	4000000	258592.16	416
DP_{12}	10000	8414	1
	100000	85144.06	10
	1000000	761108.02	60.8
DP16	10000	151771.5	1
	80000	1214172.08	8.4

TABLE V
TIME REQUIRED TO GENERATE RAINBOW TABLE

Three Rainbow Tables, generated with different DP function and RF function are tabulated in Table V. If the size of the DP function increases then the the time required to generate the Rainbow Table is increasing. We have generated the Rainbow Table with an i7 processor with 8 cores. Multithreading gives a reduction in the time required to generate the Rainbow Table.

5.3.2. RF Analysis

The RF function is used for generating a key from the cipher text. We have taken three different types of RF functions and analysed its significance in the chainlength and time to generate a chain. This work is done on Rainbow Table with 1000 entries. The different RF functions are 5-bits, 6-bits and 7-bits values. Only 32 RF values are used for each RF function. Time to generate Rainbow Table with different types of RF functions are given in Table VI.

DP function	Time(Sec)		
	RF_5	RF_6	RF_7
DP_8	64	65.10	63.53
DP_{12}	905.65	911.76	927.06
DP_{16}	12985.41	12128.60	12776.32

TABLE VI
TIME REQUIRED FOR THE GENERATION OF RT TABLE WITH DIFFERENT RF FUNCTIONS

The change in RF functions does not make much difference in the time required to generate Rainbow Table and the average chainlength for the Rainbow Table.

5.3.3. DP Analysis

We have generated Rainbow Table with three different DP functions. The different DP functions are DP_8(last 8-bits zero), DP_{12}(last 12 bits zero) and DP_{16}(last 16-bits zero). Time required for the generation of Rainbow Table with different DP functions, size of Rainbow Table and number of entries are tabulated in Table V. From Table V, it is clear that if DP functions are more then the time required to generate the Rainbow Table is more. There is a higher probability of getting the DP value for DP_8 than the others like DP_{12} and

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

DP_{16}. The chainlength for different DP functions are tabulated in Table VIII.

DP function	chainlength	
	Min	Max
DP_8	3031	14567
DP_{12}	51133	277252
DP_{16}	900403	4010675

TABLE VIII
CHAINLENGTH FOR DIFFERENT DP FUNCTIONS

The maximum and minimum chainlength for different DP functions are given in the above table. If the DP function has less bits then the chainlength also less. The size of the DP function is directly depend on the chainlength and time required to generate the Rainbow Table. The DP function has much significance on the chainlength and the time required to generate the Rainbow Table.

5.3.4. Collision Analysis

In our experiment, there are 32 differenrt RF functions are used. The 32nd DP value is EP. Whenever a DP condition satisfied then change the RF function. If a collision occurs in the same position in different chains lead to false alarm. If an EP has more than one SP then it is called false alarm. These false alarm reduces the key space for the lookup. If a collision occurs in the different position in the chains may not lead to false alarm. This is due to the different RF functions are used in the chains. We have conducted collision analysis on different Rainbow Table with different DP functions and the result is tabulated in Table IX.

Number of entries	Number of collisions		
	DP_8	DP_12	DP_16
1000	0	1	4
2000	0	2	15
5000	0	4	41
10000	0	10	89
50000	0	14	485
100000	0	31	707
200000	1	73	1185
500000	3	153	2437
4000000	27		

TABLE IX
COLLISION IN RAINBOW TABLE WITH DIFFERENT DP FUNCTIONS

If the DP condition is more (more bits zero) then the chainlength is also more and this leads more time to generate the Rainbow Table. If the chainlength is more then the probablity occuring collision is also more. For 1000 entries, the number of collision occured in DP_8 is zero but in DP_{12} and DP_{16} collision occured is 10 and 89 respectively. If the DP condition is more then the number of collision occurs in Rainbow Table is more.

6. CONCLUSION

Various parameters of rainbow table were analysed in this project. Three Rainbow Tables gener-ated with different DP functions(k bits zero). If the k bits of DP function is large then the chainlength is also increases and the time to generate Rainbow Table is also large. In order to search a large key space, large DP function is required. But, if the DP function is large then the probability of collision is also increases. This reduces the key space to search the key. Our aim is to generate the Rainbow Table with less collision rate so that we chose a good DP function which results more chainlength and less collision. From the RF function analysis, it can be observed that RF function does not have much significance on chainlength, Rainbow Table generation time and collision rate. Due to this, it can be concluded that RF function does not have muh impact on the keyspace.

The Rainbow Table generation is a time consuming process. We have generated the Rainbow Table with multithreading to execute the chain generation process parallely. We have worked on a system with 12 cores and the time required to generate the Rainbow Table reduces by a factor of eight, compared to without multithreading. In order to reduce the time required to generate the Rainbow Table, we can use CUDA programming. CUDA is a parallel computing platform and programming model which contains a Graphics processing Unit(GPU) created by NVIDIA.

REFERENCES

1. Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the kasumi cryptosystem used in gsm and 3g telephony. In Advances in Cryptology–CRYPTO 2010, pages 393–410. Springer, 2010.
2. Specification of the 3GPP Confidentiality and Integrity Algorithms; Version: 1.0;Document 3: Implementors Test Data;.
3. Martin E Hellman. A cryptanalytic time-memory trade-off. Information Theory, IEEE Transactions on, 26(4):401–406, 1980.
4. Dorothy Elizabeth Robling Denning. Information warfare and security, volume 4. Addison-Wesley Reading, 1999.
5. Philippe Oechslin. Making a faster cryptanalytic time-memory trade-off. In Advances in Cryptology-CRYPTO 2003, pages 617–630. Springer, 2003.
6. Panagiotis Papantonakis, Dionisios Pnevmatikatos, Ioannis Papaefstathiou, and Charalampos Manifavas. Fast, fpga-based rainbow table creation for attacking encrypted mobile communications. In Field Programmable Logic and Applications (FPL), 2013 23rd International Conference on, pages 1–6. IEEE, 2013.
7. Kitae Jeong, Yuseop Lee, Jaechul Sung, and Seokhie Hong. Fault injection attack on a5/3. In Parallel and Distributed Processing with Applications (ISPA), 2011 IEEE 9th International Symposium on, pages 300–303. IEEE, 2011.
8. Hamid Choukri and Michael Tunstall. Round reduction using faults. FDTC, 5:13–24, 2005.

9. Maghsood Parviz, Seyed Hassan Mousavi, and Saeed Mirahmadi. Key classification attack on block ciphers. arXiv preprint arXiv:1305.4229, 2013.
10. Hidema Tanaka, Chikashi Ishii, and Toshinobu Kaneko. On the strength of kasumi without fl functions against higher order differential attack. In Information Security and CryptologyICISC 2000, pages 14–21. Springer, 2000.
11. Nobuyuki Sugio, Sadayuki HONGO, and Toshinobu KANEKO. A study on higher order differential attack of kasumi. IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, 90(1):14–21, 2007.
12. Ulrich Kuhn¨. Cryptanalysis of reduced-round misty. In Advances in CryptologyEUROCRYPT 2001, pages 325–339. Springer, 2001.
13. Eli Biham. A fast new des implementation in software. In Fast Software Encryption, pages 260–272. Springer, 1997.
14. Eli Biham, Alex Biryukov, and Adi Shamir. Miss in the middle attacks on idea and khufu. In Fast Software Encryption, pages 124–138. Springer, 1999.
15. Keting Jia, Leibo Li, Christian Rechberger, Jiazhe Chen, and Xiaoyun Wang. Improved cryptanalysis of the block cipher kasumi. In Selected Areas in Cryptography, pages 222–233. Springer, 2013.
16. Teruo Saito. A single-key attack on 6-round kasumi. IACR Cryptology ePrint Archive, 2011:584, 2011.
17. Eli Biham. New types of cryptanalytic attacks using related keys. Journal of Cryptology, 7(4):229–246, 1994.
18. Mark Blunden and Adrian Escott. Related key attacks on reduced round kasumi. In FSE, volume 2355, pages 277–285. Springer, 2001.
19. Jongsung Kim, Guil Kim, Seokhie Hong, Sangjin Lee, and Dowon Hong. The related-key rectangle attack–application to shacal-1. In Information Security and Privacy, pages 123–136. Springer, 2004.
20. Eli Biham, Orr Dunkelman, and Nathan Keller. A related-key rectangle attack on the full kasumi. In Advances in Cryptology-ASIACRYPT 2005, pages 443–461. Springer, 2005.
21. 3GPP TS 55.217 V6.1.0 (2002-12):3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; 3G Security; Specification of the A5/3 Encryption Algorithms for GSM and ECSD, and the GEA3 Encryption Algorithm for GPRS;Document 2: Implementors' Test Data.
22. L Goubin Nicolas T Courtois and G Castagnos. what do des s-boxes say to each other. 2002.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication