# A new authentication and message encryption algorithm for roaming user in GSM network

Bum-Sik Kim, In-Chul Shin
Department of Electronics and Computer Engineering, Dankook University
San 8, Hannam-Dong , Yongsan-Gu,
Seoul, Korea 140-714
TEL : 82-02-709-2592
FAX : 82-02-796-6395
E-MAIL : kbs87@dankook.ac.kr

**Abstract:** With the advance of wireless communications technology, mobile communications has become more convenient than ever. However because of the openness of wireless communications, how to protect the privacy between communicating parties is becoming a very important issue. In this paper an authentication and encryption algorithms of the GSM network for roaming user is proposed. In the proposed algorithms, we use a new hash function for user authentication and LFSR (Linear Feedback Shift Register) based stream cipher for message encryption and decryption. Each algorithm is programmed with C language and simulated on IBM-PC system and we analyze the randomness properties of the developed algorithm using statistical analysis.

## 1. Introduction

Mobile communications has become more popular and easier for the past few years. Nowadays people can communicate with each other on any place at any time. However the openness of wireless communications poses serious security threats to communicating parties. How to provide secure communication channels is essential to the success of a mobile communication network. The GSM (Global System for Mobile communications) is the standard proposed by ETSI (European Telecommunication System Institute) for digital mobile communications [1]. One of the most important goal of GSM is to support personal mobility and terminal mobility through a world wide Personal Communication System. Personal mobility provided through the insertion of a subscriber identity module (SIM) card, allows a user to make and receive calls independently of both the network point of attachment and the specific terminal [2][3].

In the GSM system A3, A8 and A5 algorithms are used for security. A3 algorithm is used for subscriber authentication. A5 algorithm is used for ciphering/deciphering user data. This algorithm is standardized throughout all GSM network. A8 algorithm is used for cipher key generation. Algorithms A3 and A8 are not fully standardized by GSM and may be specified at the direction of PLMN (Public Land Mobile Network) operators. Different PLMN's may use different and proprietary versions of these algorithms. Algorithms A3 and A8 are always running together, in most cases these two are implemented as a single algorithm.

In this paper we present the study on the authentication and message encryption algorithm to support roaming service in GSM network. We implemented A3/A8 algorithm by using new hash function. Also we proposed the LFSR based stream cipher for message encryption and decryption. This stream cipher can't be used in A5 algorithm of the GSM system, but it may use in any cryptography application. Each algorithm is programmed with C language and simulated on IBM-PC system and we show the randomness properties of developed algorithms using statistical tests.

## 2. The Security architecture of GSM

The GSM authentication protocol for roaming users is carried out through the challenge/response mechanism that consists of asking a question that only the right user equipment (SIM) may answer. The returned answer computed internally in the user SIM card, will be compared in the authentication center. More precisely, a random number RAND is sent and the expected answer, called the signed result (SRES: Signed Response), is returned. The RAND is generated locally by HLR (Home Location Register)/AuC (Authentication Center) and then combined with the user's secret key *Ki* through the A3 algorithm to get the SRES. Fig.1 depicts the authentication procedure.

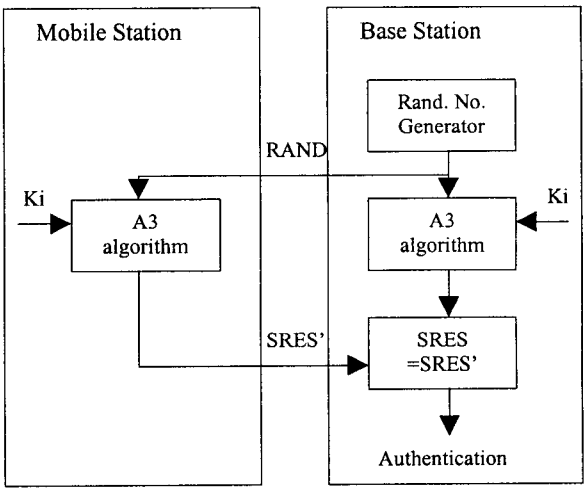


Fig. 1. Authentication procedure in GSM network

Each time a mobile station is authenticated, the network and MS (Mobile Station) has to compute the ciphering key *Kc* which is used for ciphering and deciphering of transmitted data. The ciphering key *Kc* computation is basically similar to the SRES, using the A8 algorithm. Fig.2 depicts that A8 algorithm uses the output from A3 algorithm to generate the key string for the A5 algorithm. In fig. 3 the A5 algorithm uses a proprietary algorithm for message encryption and decryption [5][6]. Table 1 summarizes the input/output parameter of security algorithm in GSM Network.

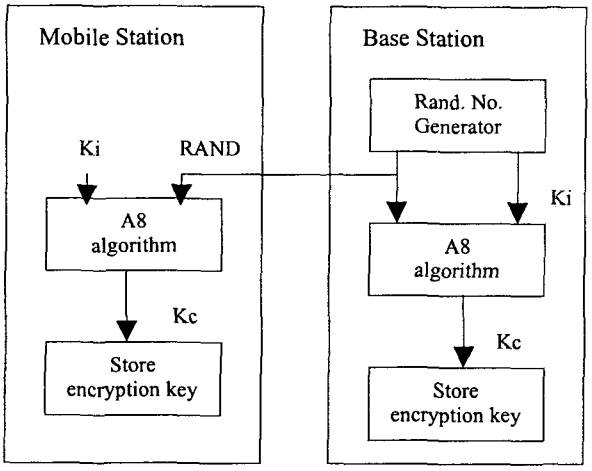


Fig. 2. Generation of encryption key (*Kc* )

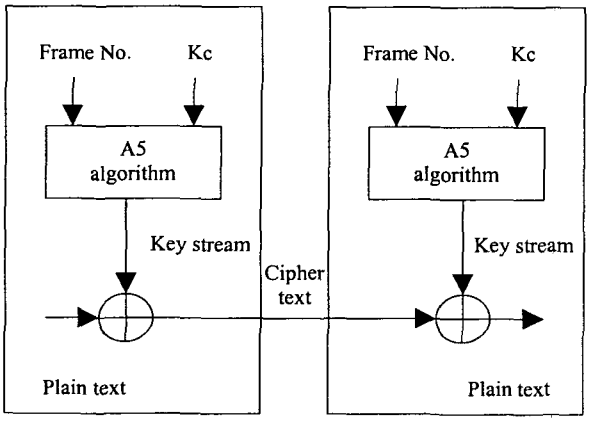


Fig. 3. Mesaage encryption/decryption

Table 1. The parameters of security algorithm in GSM

| Algorithm | Input/output parameter (bit size) |
|---|---|
| A3 | Input : RAND (128), *Ki* (128)<br>Output : SRES (32) |
| A5 | Input : Encryption key *Kc* (64)<br>Output : Key stream (114) |
| A8 | Input : RAND (128), *Ki* (128)<br>Output : Encryption key *Kc* (64) |

## 3. The proposed algorithm

### 3.1 The definition of the hash function

Hash function (more exactly cryptographic hash function) are functions that map bit-strings of arbitrary finite length into strings of fixed length. This output is commonly called a hash value, a message digest, or a fingerprint. Given $h$ and an input $x$, computing $h(x)$ must be easy. A one-way hash function must satisfy the following properties [7][8][10].

- Preimage resistance: it is computationally infeasible to find any input that hashes to any pre-specified output.
- $2^{nd}$ preimage resistance: it is computationally infeasible to find any second input which has the same output as any specified input.

A cryptographically useful hash function must satisfy the following additional property.

- Collision resistance: it is computationally infeasible to find a collision. That is, it is computationally infeasible to find a pair of two distinct inputs $x$ and $x'$ such that $h(x)=h(x')$.

### 3.2 The proposed A3/A8 algorithm

In the proposed hash function all computations are on 32-bit words, in little-endian, 2's complemnet representation. To run quickly it uses primary operations of the CPU such as addition, subtraction, multiplication, and exclusived-or. MD family hash function uses Boolean function but the proposed hash function uses $x^{-1}$ operation for destroying linearity property. Generally a computation to find inverse element requires a lot of operation time. So we uses lookup-table which consists with a precomputed inverse element to find inverse elements in GF($2^8$). A prominent feature of the proposed hash function is the use of lookup-table (called s_box) for running quickly and security.

Note that we use the notation of the C progrmming language, where ^ denotes the XOR operator. We use six 32-bit registers called a, b, c, d, e, and f as the intermediate hash values. These registers are initiated to $h_0$ which is:

```
a=0x01234567;   b=0xEFCDAB89;
c=0x98BACDEF;   d=0x10325476;
e=0xC3D2E1F0;   f=0x5A3CF01D;
```

Each successive 256-bit message block is divided into eight 32-bit words x0, x1, x2, ... , x7, and the following computation is performed to update $h_i$ to $h_{i+1}$.

This computation consists of three passes, and between each of them there is a key schedule - an invertible transformation of the input data which prevents

an attacker forcing sparse inputs in all three rounds. Finally there is a feedforward stage in which the new values of a, b, c, d, e, and f are combined with their initial values to generate $h_{i+1}$:

```
Save_abcdef( );
Pass(a,b,c,d,e,f,5);
Key_schedule;
Pass(a,b,c,d,e,f,5);
Key_schedule;
Pass(a,b,c,d,e,f,5);
Feedforward( );
```

Where

(1) Save_abcdef saves the value $h_i$

```
aa=a; bb=b; cc=c; dd=d; ee=e; ff=f;
```

(2) Pass(a, b, c, d, e, f, X, mul) is

```
Round(a, b, c, d, e, f, x0, mul);
Round(b, c, d, e, f, a, x1, mul);
Round(c, d, e, f, a, b, x2, mul);
Round(d, e, f, a, b, c, x3, mul);
Round(e, f, a, b, c, d, x4, mul);
Round(f, a, b, c, d, e, x5, mul);
Round(a, c, e, b, d, f, x6, mul);
Round(f, b, d, a, c, e, x7, mul);
```

where round(a, b, c, d, e, f, X, mul) is

```
f ^= X; a -= s_box(f0, f1, f2, f3);
f ^= a; b += s_box(f0, f1, f2, f3); b *= mul;
f ^= b; c += s_box(f0, f1, f2, f3); c *= mul;
f ^= c; d += s_box(f0, f1, f2, f3); d *= mul;
f ^= d; e += s_box(f0, f1, f2, f3); e *= mul;
```

and where $f_i$ is the $i$ th byte of f ($0 <= i <= 3$).

(3) Key_shcedule( ) is

```
x0 -= x7 ^ 0xA5A5A5A5;
x1 ^= x0;          x2 += x1;
x3 -= x2 ^ ((~x1) << 7);
x4 ^= x3;          x5 += x4;
x6 -= x5 ^ ((~x4) >> 23);
x7 ^= x6;          x0 += x7;
x1 -= x0 ^ ((~x7) << 7);
x2 ^= x1;          x3 += x2;
x4 -= x3 ^ ((~x2) >> 23);
x5 ^= x4;          x6 += x5;
x7 ^= x6 ^ 0x01234567;
```

where << and >> are logical shift operators.

(4) feedforward( ) is

```
a ^= aa; b -= bb; c += cc; d ^= dd; e -= ee; f += ff;
```

The resultant register a, b, c, d, e, and f are the 192 bits of the (intermediate) hash value $h_{i+1}$.

(5) result is

SRES = a^b^c^d; $Kc$ = ef;

### 3.1.1 Generation of s_box

The proposed hash function uses $x^{-1}$ operation for destroying linearity property. Generally a computation to find inverse element requires a lot of operation time. So we uses lookup-table which consist with precomputed inverses element to find inverse elements in GF($2^8$). It is not secure in cryptographic aspect because a inverse element of 0 does not exist. For this reason, each of inverse elements is exclusive-ORed with 0xa5.

### 3.2 The proposed message encryption algorithm

For essentially all possible secret keys, the output sequence of an LFSR-based key stream generator should have the following properties; long period, large linear complexity and good statistical properties[9][10].

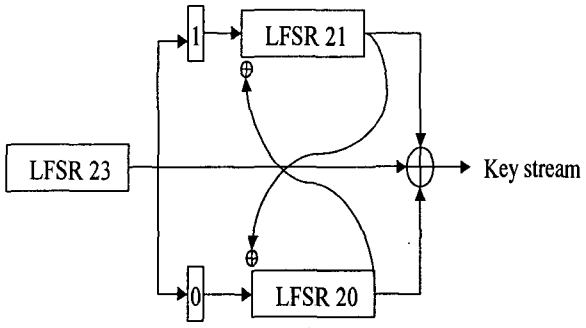


Fig. 4. The proposed stream cipher

The new desinged stream chiper is composed of three LFSRs. The connection polynomials of each LFSR are $x^{23} + x^5 + 1$, $x^{21} + x^2 + 1$, and $x^{20} + x^2 + 1$. The period of a a proposed algorithm is $2^{23} \times (2^{21} - 1) \times (2^{20} - 1) \approx 2^{64}$. The linear complexity is $(21 \times 20) \times 2^{23} \approx 2^{31.7}$.

The following steps are repeated until a key stream of desired length is produced.

① LFSR23 is clocked.
② If the output of LFSR23 is 1 then LFSR21 is clocked, else LFSR20 is clocked.
③ The output bits of all LSFR are XORed, the resulting bit is part of the key stream.
④ The LSB of LFSR and the MSB of opposite LFSR is XORed.

## 4. Statistical test and test result

FIPS 140-1 specifies four statistical tests for randomness and are provided explicit bounds that the computed value

of a statistic must satisfy in following tests. The output bitstring of length 20000 bits generated from the proposed algorithm is subjected to each of the following tests. If any of the tests fail, then the algorithms fails the test[10][11]. The test results are shown in the following tables.

(1) Frequency/Serial/Poker test

Table 2. The result of freq., serial and poker tests

| Test | FIPS140-1 | New A3/A8 | New A5 |
|---|---|---|---|
| Frequency | 9654<X<10346 | 9998 | 10003 |
| Serial | 5.9915 | 2.968 | 1.822 |
| Poker | 1.03<X<57.4 | 26.009 | 8.243 |

(2) Run test

Table 3. The result of run test

| Length of run | FIPS140-1 | New A3/A8 | New A5 |
|---|---|---|---|
| 1 | 2267-2273 | 2543 | 2549 |
| 2 | 1079-1421 | 1221 | 1268 |
| 3 | 502-748 | 658 | 613 |
| 4 | 223-402 | 307 | 322 |
| 5 | 90-223 | 150 | 145 |
| 6 | 90-223 | 151 | 147 |

(3) Autocorrelation test

Table 4. The result of autocorrelation test

| D | Threshold value | New A3/A8 | New A5 |
|---|---|---|---|
| 4 | -1.96 – 1.96 | -0.50915 | 0.63646 |
| 8 | -1.96 – 1.96 | 1.51351 | 0.50921 |
| 16 | -1.96 – 1.96 | -0.9903 | 0.45272 |
| 32 | -1.96 – 1.96 | -0.1273 | -0.1132 |

## 5. Conclusions

In this paper we proposed algorithms for authentication and encryption key generation in GSM, called A3 and A8 respectively. And we implemented A3/A8 algorithm by using new hash function. We used primary operations of CPU and lookup-table which consists with a precomputed inverse elements for quick computation of hash function. Also we proposed the LFSR based stream cipher for message encryption and decryption. This stream cipher can't be used in A5 algorithm of the GSM system, but it may use in communication system and any other cryptography application.

The output bit string of the proposed algorithms pass all tests which FIPS 140-1 specified. But it is emphasized that these properties are only necessary conditions for the proposed algorithms to be considered cryptographically secure. Since mathematical proofs of security of them are not known, the proposed algorithms can only be deemed computationally secure after having withstood sufficient public scrutiny.

For the further research, we are considering the analysis of the security and the optimization of the implementation for the proposed algorithms. Futhermore, we need to keep going on more systematic approaches in hardware implementation.

## References


[1] Charles Brookson, "GSM Security : A description of the reasons for security and the techniques," IEEE, 1994.

[2] Luis M. Correia and Ramjee Prasad, "An Overview of Wireless Broadband Communications Services," IEEE communications Magazine, pp.28-33, Jan. 1997.

[3] Moe Rahnema, "Overview of the GSM system and protocol Architecture," IEEE Personal Communications Magazine, pp.92-100, April 1993.

[4] Dan Brown, "Techniques For Privacy and Authentication in Personal Communication systems," IEEE Personal Communications, vol. 2, no. 4, pp.6-10, Aug. 1995.

[5] Seshadri Mohau, "Privacy and Authentication Protocols for PCS," IEEE Personal Communications, pp34-38, Oct. 1996.

[6] Asha Mehrotra, Leonard s. Golding, "Mobility and security management in the GSM system and some proposed future improvements," Proceedings of the IEEE, vol. 86, no. 7, July 1998.

[7] B. Preneel, "Analysis and design of cryptographic hash functions," Doctoral Dissertation, Kathokieke Univeriteit Leuven, 1993.

[8] I.B. Damgrd, "A design principle for hash functions," Advanes in Cryptology Crypto'89, LNCS, vol. 435, pp.416-427, 1990.

[9] Rueppel, "Analysis and Design of Stream cipher," Springer-Verlag, 1986.

[10] A.J. Menezes, P.C. van Oorschot, S.A. Vanstone, "Handbook of Applied Cryptography," CRC press, 1997.

[11] NIST, "Secure hash Standard," FIPS 140-1, US Department of Commerce, Washington D.C., April 1995.