Dynamic Key Generation During a Communication Instance Over GSM

Joseph Zalaket¹, Khalil Challita²*

¹Holy-Spirit University of Kaslik, Faculty of Engineering, 446, Jounieh, Lebanon
²Notre-Dame University, Louaize, Faculty of Natural & Applied Sciences, 72, Zouk Mikael, Zouk Mosbeh, Lebanon
*Corresponding Author: kchallita@ndu.edu.lb

Copyright © 2013 Horizon Research Publishing All rights reserved.

Abstract  The use of mobile phones became vital in our everyday life. This emergence has led many companies to allow new activities which were previously running strictly over the Internet to run over the mobile network such as the electronic payment. These circumstances make the security of mobile communication a priority to preserve the authentication, confidentiality and integrity of data sent between subscribers and mobile network. In this paper, we propose a dynamic key generation for the A5 GSM encryption algorithm to enforce the security and protect the transferred data. The improvement we made here is the generation and use of multiple encryption keys during a single phone communication, which makes it much harder for a cryptanalyst to eavesdrop on a phone conversation. Note that our algorithm can be implemented over any GSM generation GSM/3G/4G.

Keywords  Mobile Communication, Encryption, GSM, A5 Algorithm

1. Introduction

Mobile phone became the protagonist of the new electronic technology. If we compare it with that of other technologies, the infiltration rate of mobile phones in the world is extremely high, both in cities than rural communities of the most of the countries. According to estimates made by the International Telecommunication Union the access to mobile networks is growing much faster than the access to Internet. Most of the mobile phones are running through GSM (Global System for Mobile Communication). The security of the GSM is based on three main algorithms [1, 9]. Circumstances make the security of mobile communication a priority to preserve the authentication, confidentiality and integrity of data sent between subscribers and mobile network. In this paper, we propose a dynamic key generation for the A5 GSM encryption algorithm to enforce the security and protect the transferred data. Our algorithm can be implemented over any GSM generation GSM/3G/4G.

A3 is an authentication algorithm whereas A8 is a key generation one. These two algorithms are based on a hash function that takes as input a 128-bit key Ki stored on a SIM card (on the network’s side) and a random number ‘RAND’ of 128-bit to generate a 32-bit RESPONSE that will either verify the identity of the subscriber or deny it. These same inputs are given to A8 to generate a key of 64 bits that will be used to encrypt all the messages using an A5 algorithm. There are several versions of A5: A5/1, A5/2 and A5/3. The one implemented in the second generation and providing an acceptable level of security is A5/1.

Although there are several cryptographic algorithms used to provide security in GSM, this system is not completely secure since both A3/A8 algorithm and the A5/1, A5/2 and A5/3 algorithms were broken, and the key used in ciphering was retrieved sometimes in a few seconds [10, 12, 14].

To increase the security of the A5 family of algorithms, we propose a new scheme that consists in using multiple secret keys to encrypt the content of a single communication. As opposed to the single shared key originally stored on a SIM card, the keys will be created dynamically along with the communication itself. The algorithm we suggest may run on the extended SIM card architecture we next describe as well as the current SIM architecture.

2. Background

Several approaches have been adopted to allow mutual authentication from both the network and the Mobile Subscriber (MS) in order to secure the communication between two parties [10, 11, 12].

The authors in [2] used public key encryption to introduce a new mutual authentication method between the MS and the Visitor Resource Locator (VLR), called Proxy signature.

In this method, the Home Resource Locator (HLR) delegates the Mobile Station (MS, subscriber) the power to sign the nonce (random number generated only once) generated by the VLR, bearing in mind that the VLR can verify the signature based on the HLR’s public key. This
model enforces privacy and non-repudiation. In this case key management is easy since only one key (the public key of HLR) must be managed.

The authentication phase is divided into two parts: on-line authentication and off-line authentication. During the on-line authentication phase, the process requires that VLR connects to HLR whenever MS asks for an authentication. On the other hand, and to save authentication time and provide fault tolerance, off-line authentication is performed by VLR locally according to the parameters obtained from HLR in advance and without connecting to HLR. Note that the first authentication request must be performed online and the subsequent authentication requests can be continually performed off-line.

In [3], the researchers pointed out some weaknesses in the existing security system of GSM and proposed a better solution which provides an authenticated session key distribution protocol between the authentication centre (AUC) and the MS for every call attempt by the MS. At the end of an authentication session key distribution protocol, the identities are mutually verified (between the AUC and a SIM) and a session key for call encryption is distributed to the MS.

A self-concealing mechanism is introduced in [5]. The purpose here is to reduce the entries of the GSM databases (VLR and HLR) by discarding the bulky database and creating valuable improvements for portable communication systems. In this approach, the shared secret is concealed by the authentication server and only the authentication server has the private key to open the shared secret.

The new concept initiates several positive changes. First, the sensitive and large database can be discarded. Consequently, this prevents hacker attacks to the database and reduces maintenance demand for the server.

A warrant is used to guarantee the user’s access rights; an issue that is not addressed in the conventional challenge-response scheme.

Our protocol uses a nonce and variables that already exist in the GSM but not in the security algorithms such as TMSI, LAC.

3. Dynamic Key Generation

Note that any change arising in the algorithms currently implemented on the SIM card requires the generation of new SIM cards for all users, which could be inconvenient for many of them. That is why the protocol we suggest supports two versions of SIM cards: the old one and the new generated SIMs whose users will benefit from a higher security that we describe in the following sections.

3.1. Key Generation Process

The GSM network sends a 128-bit RAND number to the MS. They both apply the A3 (authentication algorithm) that will accept this RAND and a 128-bit key Ki stored on the SIM and known to the GSM as input to generate a 32-bit RESponse number. The MS sends this RES to the GSM network that compares it to the response number it generated. If they match, the user is authenticated and the A8 algorithm is applied: it takes Ki and RAND as input and generates the 64-bit key Kc that will be used in the ciphering algorithm A5.

Now that the Kc has been generated, new core elements take part in the new protocol: 32-bit TMSI, 64-bit IMSI, 16-bit LAC and a new number NEWRAND that is randomly generated by the GSM.

A GSM conversation is sent as a sequence of frames every 4.6 millisecond [6] and therefore we chose to generate the NEWRAND every 10 x 4.6 ms, which means every 46 milliseconds.

Once this new random number is generated, the timestamp (dd/mm/hh/mm/ss) will be divided by the NEWRAND.

This time variable has a 24-hour format to avoid having the same timestamp twice a day. A random interval was chosen to decide whether or not to use this NEWRAND. If the division result was within the interval the NEWRAND is sent for use by both the MS and the GSM otherwise, the old value of NEWRAND is used. This interval can be left for the operator to decide its boundaries for better security and to avoid having any recurring pattern. But in our simulation the range of values was chosen to be within the interval [20790, 977890].

Once the NEWRAND is generated and sent, the main function called - KeyGen() will use it with other parameters to generate an enhanced security key.

![IMSI blocks](figure1.png)

**Figure 1. IMSI blocks**

**IMSI(64 bits / 15 digits)**

<table>
<thead>
<tr>
<th>8 bits</th>
<th>8 bits</th>
<th>8 bits</th>
<th>8 bits</th>
<th>8 bits</th>
<th>8 bits</th>
<th>8 bits</th>
<th>8 bits</th>
<th>8 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td></td>
</tr>
</tbody>
</table>
The main function KeyGen() takes seven parameters: TMSI, LAC, IMSI, NEWRAND, Kc, version, NewKeyGenerated.

- TMSI is the 32-bit or 8 digits value that changes each time the VLR is modified, or if the interval of time set for it expires.
- 16-bit LAC or a 4 digits, is the location area code that will be updated at each location change. A Location Update Procedure is triggered.
- IMSI is a 15 digits number (64-bits). It is the main identifier for the MS. In our protocol we divide it (logically) into 4 blocks and, based on the NEWRAND, we select the appropriate block to be used to generate the new key as shown in Figure 1.
- NEWRAND is generated as described in the previous section
- Kc is the original 64-bit key generated using the A8 algorithm.
- Version is set to 1 in this protocol because it is the new version of SIM card, for the previous cards it will be set to 0. This function will only run if the version is set to 1.
- NewKeyGenerated is the new 64-bit key generated during each execution of the function KeyGen() and is used as input to the A5 ciphering algorithm.

All the values used as identifiers such as TMSI, LAC, IMSI and the key are originally sent in hexadecimal format. So each one of them is converted into a binary format and stored in a vector of bytes.

Example: If the TMSI is composed of 8 hexadecimal digits: A1 09 34 E7, then it will be converted to binary and stored in a 4 bytes vector as shown in Figure 2:

![TMSI Array](image)

Figure 2. TMSI array

So, in memory A1 will be stored as (10100001). The same applies for all these variables in term of conversion from hexadecimal to binary.

As we can see, TMSI is an array consisting of 4 cells. The possibilities of arranging these cells in different order are 4! = 24. For this purpose, we have created a static matrix that consists of 24 rows and 4 columns. In each column we store an index from 0 to 3. This matrix plays the role of an index to the TMSI array in order to accelerate the generation.

The remainder of the division of NEWRAND by 23 will determine the index of the row of the matrix to be chosen. Let us say that the NEWRAND modulus 23 gives 5 as a result. Looking at the row of index 5, the columns contain the values 0, 3, 1, and 2. Therefore, the rearranged TMSI array will have as elements: TMSI[0], TMSI[3], TMSI[1] and TMSI[2], respectively. And thus the rearranged TMSI will be A1 E7 09 34.

The LAC array is of 16 bits (2 hexadecimal digits), for example: F2 56.

The IMSI is divided into 4 blocks of 16 bits. Based on this NEWRAND, we decide which block to use. The remainder of NEWRAND divided by 4 determines the number of the block. If NEWRAND modulo 4 gives 2 as a remainder, the block of index 2 of IMSI is chosen, which means that IMSI[4] and IMSI[5] will be used.

A new array of 4 bytes is considered, which is the combination of the LAC array and the two chosen cells of the IMSI. This 32-bit array is XORed bit by bit with the rearranged TMSI array.

Since NEWRAND is even, the New Output will be swapped with block 1 of Kc and therefore the new key generated consists of the block 0 of Kc concatenated with the new output (32-bits) as described in Figure 3.

This whole process is repeated:
- Every 46 milliseconds,
- Whenever the TMSI value is updated,
- Once a Location Update Procedure is taking place.

So we have multiple keys that will be generated every while. Each time a new key is generated, the A5 will call its main function using the newly generated key as input.

### 3.2. New Algorithm for Supporting Existing SIM Cards

Now that the multiple keys generation is solved, we still have to make the new protocol compatible with both versions of SIMs. This must be taken into consideration from the GSM network view given that it would be the one responsible of generating the NEWRAND and sending it to the MS. Since the implementation of the global function grouping all three algorithms (i.e. A3, A8, A5) is not made clear by the GSM community, the following is a theory on how the protocol should be put into practice.

We assume that the major function that combines all the elements is a function called F( ) and has several parameters: key, TMSI, LAC, IMSI. So if the values of the last three parameters are not sent, then we know that we are using the old version of SIM cards and therefore the usual protocols will be followed. If these values are sent as parameter arguments then we are using a new version of SIM cards and therefore the KeyGen() function early described will be applied.

The below pseudo-code explains our idea and how a default value of null will be assigned to the parameters that were not sent in order to be compatible with the current SIMs:

```plaintext
F(char Key, char TMSI=Null, char LAC=Null, char IMSI=Null)

{If no values are set for TMSI, LAC and IMSI
Use the old protocol and algorithms e.g. call g(key)
Else
Use A3/A8 and then call the new function KeyGen
e.g. call h(Key, TMSI, LAC, IMSI)}
```
3.3. New SIM Card Architecture

In addition to the functions that are newly embedded in the system, we need to take into consideration the elements that were added and were not available in the previous version. For example, NEWRAND is used in our new protocol and therefore we need to store it on the SIM. For this purpose, the new SIMs will hold a new register for the NEWRAND value; each new value will override the previous one.

The LAI value is updated using a broadcast control channel (BCCH) with the ongoing conversation. For this reason we will be using this same channel to send the NEWRAND after it has been generated.

The function that is responsible for generating and sending the NEWRAND must be implemented on the network side only, while the function that generates the key should be available on both the SIM and the network.

4. Implementation and Attack

We implemented our algorithm using the C language. The simulation was done on a PC having the following specifications:

Intel Core i3 CPU, 2.40 GHz - 3 GB of DDR2 RAM - Windows 7 32-bit OS - 500 GB Hard Disk

Note that the capacity of the used machine is not important as the algorithm uses trivial system resources such as: only 30 bytes of RAM in addition to which is used by the A5/1 algorithm, 2 bytes of physical memory (register to store the random number on the SIM card side) and about 0.001ms of CPU time to generate a new key.
Even though, a new implementation of this prototype is currently prepared to run on mobile phones using Android OS.

We simulated 3 phone calls and the results are listed in Table 1.

Then we tested our algorithm against 2 types of attacks: 1- A brute-force attack with 264 keys to test, and 2- The Biham-Dunkelman attack [15].

None of the attacks was able to reveal any of the above conversations. Even after relaxing the problem (e.g. by feeding some known information to the attacker such as the timestamp), the Biham-Dunkelman attack was unable to disclose in real-time the actual key used during a conversation.

We already know that a brute force attack needs to test 264 keys and that the Biham-Dunkelman attack requires testing approximately 240 keys. In order to be successful, the latter attack requires that the same key be used during about 2.36 minutes of conversation time. Such an attack would fail against our algorithm since a key would be changed during at most 0.5s. Actually, and based on our experimental results, the longest time a key was changed is 312.5 milliseconds.

5. Discussion

Using a single key during a whole mobile communication or even for a long period of time made it easy for cryptanalysts to reveal the key and get all the information shared between the MS and the network. Whereas in our protocol we have more than one key that is being used during a single conversation. The location update depends on whether the user is moving or not, the TMSI change depends also on the network implementation.

The timestamp varies every second, and its XORed result with the NEWRAND cannot be detected since it is done on the network side. Also the range in which this result is tested can be set by the operators making it harder for the attackers to recover the keys. The generation of the number is done randomly, so guessing this number is practically very difficult to achieve.

As we know, potential weaknesses in GSM were never published, and therefore the security level of GSM was not evaluated. Due to the fact that our protocol uses NEWRAND numbers, an attacker has to detect the key that was originally generated and then to track and capture all the random numbers that are sent in order to decrypt a message.

A NEWRAND number is generated every 46 ms but the time execution of the algorithm is still the same. So in case we decided to generate this random less frequently, catching it becomes harder for an attacker while generating this number in less than 46ms will make it simpler for the process.

This new algorithm is not mandatory. So the users are not obliged change their SIM cards. They can keep the old ones and benefit from the same security features already offered by GSM.

<table>
<thead>
<tr>
<th>Duration of the phone call</th>
<th>Number of generated keys</th>
<th>Number of times the same key is repeated for more than one session</th>
<th>Number of Random numbers generated (NewRAND)</th>
<th>Longest period the same key is used during the same conversation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5s</td>
<td>8</td>
<td>1/ (2 sessions)</td>
<td>1</td>
<td>125ms</td>
</tr>
<tr>
<td>3mn</td>
<td>2878</td>
<td>243/(2 sessions) 85/(3 sessions) 2/(4 sessions)</td>
<td>3913</td>
<td>150ms</td>
</tr>
<tr>
<td>10mn</td>
<td>9584</td>
<td>811/(2 sessions) 428/(3 sessions) 36/(4 sessions) 7/(5 sessions)</td>
<td>13043</td>
<td>312.5ms</td>
</tr>
</tbody>
</table>
6. Conclusion

The GSM community pretends that its used algorithms are safe and secure. By contrast, many successful attacks have been done on each of these algorithms and this is due in the first place to the fact that the implementation of these algorithms was originally kept secret and then was successfully reversed engineered.

Several attempts were made to enhance these algorithms and several propositions were made for both authentication and encryption algorithms.

As for us, we proposed in this paper a new protocol for generating multiple keys during the same conversation, so each stream might be encrypted using a different session key than the previous one. The strength of our protocol lies in the fact that we depend on several variables that are very hard to be detected or guessed by an eavesdropper: the encrypted TMSI, the LAC changing at each location and a random number generated by the network but that is not constantly sent to avoid any recurrent pattern. Also the new protocol supports two versions of SIM cards. The subscribers holding the old versions will profit from the usual security features while those using the new version will benefit from a higher level of security.

We have simulated these functions on a standard PC using the C programming language, the same language used in the other algorithms implementation.

Our strategy was not to hide the proposed algorithm from cryptanalysts, but rather to use a combination of dynamic parameters that makes it very hard for them to decrypt a message (e.g. the technique of public key or asymmetric key cryptography)

The A5/1 and A5/2 algorithms were broken, and the key Kc used in ciphering was revealed. To overcome these weaknesses, the GSM engineers implemented a 3rd generation system, but also kept it secret. They claim enhancing the security authentication with their new Authentication and Key Agreement protocol (AKA). To conclude, note that the new block cipher algorithm A5/3 [8] based on KASUMI which has been successfully attacked [12, 13].

REFERENCES