Cryptography in GSM

Hagen Fritsch
<fritsch+aui@in.tum.de>

CSC3355: Computer Security
Al Akhawayn University, Morocco

October 8th, 2007

Contents

1 Introduction to GSM / Relevance of Encryption

The 1st generation of mobile communication did not make any use of encryption or authentication, thus imposing the wireless network to a lot of possible attacks, some of them practically misusing the network. When designing GSM – that is the 2nd generation (2G) – security was at least worried about. Therefore GSM comes with a "moderate level of security" (Wikipedia, 2007b). Algorithms were developed secretly since it was still believed that *security by obscurity* reduces possible impacts to the network, which later turned out not to be true at all.

This paper describes how authentication and encryption work in GSM by describing protocols and algorithms. In Chapter 3, I will briefly describe weaknesses of the algorithms and cryptanalysis of them. A recent approach to eavesdrop any GSM communication will close the circle.

2 GSM Techniques

2.1 Authentication

GSM uses a challenge-response method to authenticate the client to the network using a pre-shared key (K_i, 128bit) that is saved in the client's SIM-card and in the provider's Home Location Register (HLR). The net sends a random-number (the *challenge*: RAND, 128bit) to the client. Both the provider and the client compute the response (SRES, 32bit) using the *secret* A3-algorithm. The client sends this value back to the net, which compares both SRES-values and either authorizes the client to use the network or rejects it.

At the same time at which the SRES is computed, both, client and provider, compute the session key (K_c, 64bit) using the A8-algorithm. That key is then used for en- and decryption of the data as will be seen in section 2.3.

Using that technique the client is authenticated to the network, but not vice versa, which allows some of the attacks described in section 3.3. The K_i

is never transmitted over the network and therefore kept safe.

2.2 A3 and A8

A3 and A8 are one-way algorithms (i.e. one can consider them as hash-functions) that need only to be implemented on the SIM-card and on the provider's HLR. Therefore different providers can use different algorithms. However most use the reference implementation of COMP128 which has been kept secret for a long time, but has been reverse engineered in 1998. Goldberg and Wagner (1998) demonstrate an attack on those algorithms that allows to recover the K_i given access to the SIM-card, by exploiting some weakness in the algorithm. This basically works by using certain combinations of challenges, which, when giving the same result, leak certain bytes of the key.

The Chaos Computer Club (2001) demonstrates that attack and is able to recover the K_i with around 150000 challenges within 12 hours.

Goldberg and Wagner point out that even over-the-air-attacks might be pratical, which poses a serious risk to the GSM network. However, according to them, GSM industry is taking steps to repair the weaknesses by replacing COMP128 by COMP128-2.

2.3 Encryption

In contrast to the A3 and A8 algorithms which can be chosen differently by each provider, the A5 algorithms are standardized, though not officially public, but cryptoanalyzed and reverse-engineered and therefore publicly available (see Briceno et al., 1998). The A5/1 and A5/2 algorithms are very similar stream-ciphers. The latter one is just a modified weaker version of A5/1.

2.3.1 A5/1

The A5/1 algorithm is a pseudo-random-keystream generator consisting of three clocked linear feedback shift registers (LFSR) which are registers whose input when being shifted is a linear function of its current content. The output of these registers is XOR-combined resulting in the keystream used

for encryption and decryption. As the figure illustrates the registers have different sizes (19, 22 and 23 respectively) and different feedback-functions.

In the **initialisation phase** registers are zero initialized and fed with the 64bit-key one bit each clock. After 14, 21 and 8 clocks the LFSR starts to work for the registers respectively. Afterwards COUNT[1] is loaded into the registers exactly the same way. Any output generated during this process is discarded. After initialisation the algorithm is in state S(0). Though there are $2^{64+22} = 2^{86}$ possible inputs (K_i+COUNT) the registers themselves only have $19+22+23 = 64$ bits. Therefore there are only 2^{64} possible S(0)-states.

In the **operation phase** the clocking starts to work, which scrambles timing of the shifting of registers adding a non-linear-part to the algorithm. Otherwise the algorithm could be broken easily by solving linear equations. Each register has a clocking-bit (the orange colored blocks in figure 1: $\tau_1 =$

[1]Each GSM frame carries a publicly visible frame-number. For each frame the A5 is initialized with K_i and COUNT whereby COUNT is a linear function of the frame-number (see Barkan et al., 2006, Appendix B).

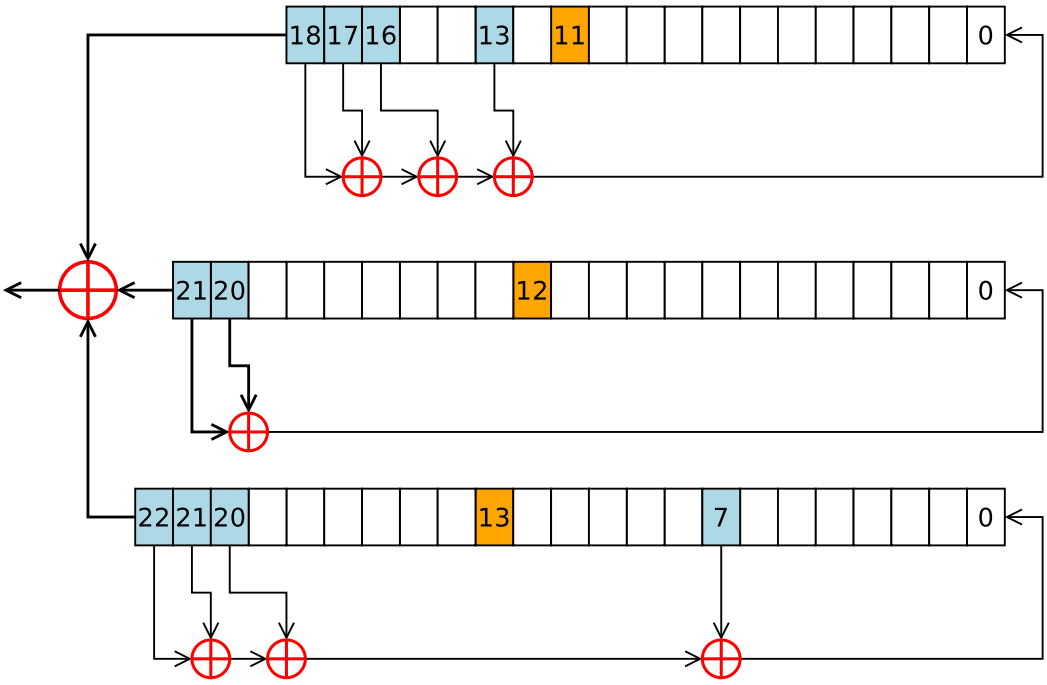

Figure 1: Structure of A5/1

11, $\tau_2 = 12$ and $\tau_3 = 13$ respectively according to Golic, 1997[2]). A register is clocked if its clocking-bit complies with the majority all the clocking-bits. Therefore the probability of a register to be clocked is 75%.

For **application in GSM** the algorithm is initialized as described (S(0)) and dry-run (discarding output bits) for 100 clock-cycles. Afterwards another 228 clock-cycles produce 114 bits used to XOR the outgoing plaintext (encrypt) and 114 bits used to XOR the incoming ciphertext (decrypt).

2.3.2 A5/2

Due to export restrictions on *strong ciphers*, a weakened version of A5/1 has been developed which is used in other countries besides Europe and the USA (Wikipedia, 2007b). A5/2 works very much like A5/1, except that it introduces a fourth LFSR (R_4: 17bits) which is initialized as the other registers. After initialisation, the 10th bit of R_4 is forced to 1. Three bits of the fourth register (instead of one bit of each of the other registers) are used to determine if the register is to be clocked. R_4 will always be clocked. For further details see Barkan et al. (2006).

2.4 Further Development and A5/3

Since A5/1 and A5/2 were known to be weak (and it had been learned that security by obscurity won't work), GSM changed the development-process and let A5/3 (also known as *KASUMI*) be a public cipher. It supports a key size of 128bit though in GSM due to the authentication protocol described earlier, there is only $64 + 22 = 84$bits available for use. KASUMI is used for 3rd generation mobile services such as UMTS. It is a feistel-structure 8-round block-cipher in which some weaknesses have been found though none of pratical impact.

[2]Though according to Wikipedia (2007a) and Briceno et al. (1998) those are supposed to be $\tau_1 = 8$, $\tau_2 = 8$ and $\tau_3 = 10$ respectively

3 Breaking GSM

A lot of flaws have been discovered in the algorithms, that allow practical attacks on GSM-encryption. As soon as in 1998 it was discovered, that all implementations of COMP128 produce 64bit keys with ten of the bits always set to zero, reducing the key-space to now only 2^{54} (Briceno et al., 1998). Another problem in GSM is, that error-protection to voice data is applied before encryption (instead of the other way around as it should be). This is only possbile due to the fact that the plaintext is only XORed with the key-stream and bit errors do not propagate as it would be the case for any block-cipher (see Barkan et al., 2006, 4, p. 13). Therefore an attacker knows about certain relations within the plaintext.

Due to the linearity of the initialisation process, once the initial state S(0) and the publicly visible COUNT are known, the session key can be easily recovered and used to decrypt previous and following frames.

3.1 Attacks on A5/2

The goal is therefore to discover S(0). Since R_4 only consists of 16bits[3] one can guess all 2^{16} possible values for R_4 and reconstruct the S(0) once any internal state is known. This works due to the fact that R_4 determines how many times a register is clocked and therefore every bit of the registers' internal state can be expressed as a linear combination of its original internal state's bits. This results in an overdetermined Gaussian equation system, which will eliminate most of the invalid solutions.

Barkan et al. (2006) point out that COUNT for frames being 1326 frames apart (which is roughly 6 seconds) differs in the very bit that is set to 1 in R_4 anyway. Thus both R_4 have the same value (though the other registers are still different) and therefore the algorithm for both of the frames follows the same clocking, which results in a only linear difference in the key-stream. Together with a known plaintext it is possible to recover the internal state and also the session-key K_c.

[3]It is actually 17bits, but since $R_4[10]$ is always set to 1, there remain only 2^{16} possible values for this register.

Due to the previously mentioned error-correction being applied before encryption Barkan et al. modify their attack so that it won't require any more known-plaintext. Adding a time-memory tradeoff makes it possible to break A5/2 within seconds with low-end equipment.

3.2 Attacks on A5/1

A5/1 seems to be stronger than A5/2, but still a lot of attack points have been found. As Golic (1997) describes in his paper, thanks to the relatively low number of bits in the registers, there are only 2^{64} possible internal states of the algorithm and thus a time-memory tradeoff can be used by precomputing the states. When the actual output bits are generated a set of possible progressions through the internal state can be used to find intersections with the precomputed states which eventually leads to identification of an actual state of the algorithm.

Another approach was shown by Biryukov et al. (2001), who state that the first $\log(n)$ bits of the cipher uniquely identify the algorithm's state. Therefore their attack precomputes the first bits for some special states and compares eavesdropped keystream bits in order to reconstruct the internal state. This however has to be combined with an approach to inverse-run the algorithm in order reconstruct S(0) which according to Biryukov et al. can be done efficiently since (due to the register-clocking) states have from zero up to three predecessors (one on average). Thus backtracking dead-ends will eventually lead to a set of possible S(0)-states. As already stated for A5/2, S(0) can be used to reconstruct the session-key K_c.

All attacks need to know at least some plaintext (and thus automatically revealing the keystream). That seems to make attacks a lot more impractical. But as Golic (1997, p. 242) points out, knowing the plaintext is not as unlikely as one might expect:

> "If two users want to communicate to each other via their base station(s), the same messages get encrypted twice which makes the known plaintext cryptanalytic attack possible, provided a co-operative insider user can be established."

Furthermore, certain patterns of communication are not unlikely to occur such as speech-pauses.

3.3 Circumvent use of stronger ciphers

Though the attacks on A5/1 are to be considered seriously, they are not really practical in application. However, Barkan et al. (2006) show that by setting up a fake (man-in-the-middle) base-station, one can force the phone and the real base-station to talk in A5/2 by having the fake base-station claim to the real base-station only to understand A5/2[4]. That attack can easily be prevented on the provider's side by not accepting A5/2 at all, because every phone is supposed to understand A5/1. Further attacks rely on the fact that the same RAND can be used as often as wished. So if an attacker managed to eavesdrop on a conversation using RAND_1, he can again use his fake base-station to initiate communication with the phone using A5/2 with the very same RAND_1, which will result in the phone using the very same K_c. Once the attacker recovers K_c, he can use that key to decrypt the recorded conversation. It even makes no difference if that conversation used to be A5/1 or A5/3 since both use the same K_c as a key.

4 Conclusions

Though only drawing an excerpt on the cryptanalysis of the ciphers being used in GSM, the previous sections showed that there are so many issues with the protocols and algorithms, that it would be impossible to draw a positive conclusion here. However the approach to use encryption and authentication at all is at least promising. The GSM designers certainly learned from previous mistakes and employ the stronger and publicly developed A5/3 cipher nowadays. The attacks involving base-station man-in-the-middle are certainly fun to play with, but can be easily avoided by using mutual authentication instead of only authenticating the client. That technique is used in

[4]The phone sends a class-mark message stating which encryption protocols it supports. That message can then be intercepted and changed by the fake base-station

UMTS thus 3rd generation mobile service offer a higher level of security over the air.

But still, all conversations are routed in plaintext through the PSTN and the air-link between base-stations and BSC. In order to defend against sophisticated eavesdroppers and wiretapping, another layer of encryption needs to be added to provide end-to-end encryption. Unfortunately this does not comply with most mobile-phone companies' or government's policies, therefore there is no easy way to accomplish this by software only (except for using VoIP) and the only remaining way seems to be the cryptophone on whose integrity one would have to trust in that case.

References

Barkan et al. 2006

Barkan, Elad ; Biham, Eli ; Keller, Natan: Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication. In: *Technion* (2006). `http://www.cs.technion.ac.il/users/wwwb/cgi-bin/tr-get.cgi/2006/CS/CS-2006-07.pdf`

Biryukov et al. 2001

Biryukov, Alex ; Shamir, Adi ; Wagner, David: Real Time Cryptanalysis of A5/1 on a PC. In: *Lecture Notes in Computer Science* Vol. 1978 (2001)

Briceno et al. 1998

Briceno, Marc ; Goldberg, Ian ; Wagner, David: *A pedagogical implementation of A5/1.* `http://jya.com/a51-pi.htm`. Version: 1998. – [Online; accessed 6-October-2007]

Chaos Computer Club 2001

Chaos Computer Club: *CCC klont D2 Kundenkarte.* `http://www.ccc.de/gsm/`. Version: 2001. – [Online; accessed 6-October-2007]

COMP128

COMP128 ; Briceno, Marc (Hrsg.) ; Goldberg, Ian (Hrsg.) ; Wag-

NER, David (Hrsg.): *Implementation of COMP128*. http://www.iol.ie/~kooltek/a3a8.txt. – [Online; accessed 6-October-2007]

Goldberg and Wagner 1998

GOLDBERG, Ian ; WAGNER, Dave: *GSM cloning*. http://www.isaac.cs.berkeley.edu/isaac/gsm.html. Version: 1998. – [Online; accessed 5-October-2007]

Golic 1997

GOLIC, Jovan D.: Cryptanalysis of Alleged A5 Stream Cipher. In: *Proceedings: International Conference on the Theory and Application of Cryptographic Techniques*. Konstanz, Germany : Springer, 1997, p. 239–255

Wikipedia 2007a

WIKIPEDIA: *A5/1 — Wikipedia, The Free Encyclopedia*. http://en.wikipedia.org/w/index.php?title=A5/1&oldid=146578652. Version: 2007. – [Online; accessed 5-October-2007]

Wikipedia 2007b

WIKIPEDIA: *GSM — Wikipedia, The Free Encyclopedia*. http://en.wikipedia.org/w/index.php?title=GSM&oldid=162524057. Version: 2007. – [Online; accessed 5-October-2007]