Security in the GSM system

By Jeremy Quirke | Last updated 1st May 2004

Introduction

Mobile phones are used on a daily basis by hundreds of millions of users, over radio links. Due to the fact that unlike a fixed phone, which offers some level of physical security (i.e. physical access is needed to the phone line for listening in), with a radio link, anyone with a receiver is able to passively monitor the airwaves.

Therefore it is highly important that reasonable technological security measures are taken to ensure the privacy of user's phone calls and text messages (and data), as well to prevent unauthorized use of the service.

As a result of recent mainstream controversy involving David Beckham's alleged text messages, some articles have been published criticizing GSM's security, even though it is highly unlikely in this case that any messages were in fact intercepted over the radio link (if at all).

As a result of some of these articles, I have decided to publish a technical article dealing with only the facts of how security measures are implemented in the GSM system, the world's most commonly used mobile telephony system. There is also a section summarizing the known flaws in these measures, and what has been done in subsequent revisions of the standard (i.e. by 3GPP) to resolve them.

Security features offered by GSM

GSM specification 02.09 identifies three areas of security that are addressed by GSM.

- Authentication of a user – this deals with the ability for a mobile phone to prove that it has access to a particular account with the operator
- Data and signaling confidentiality – this requires that all signaling and user data (such as text messages and speech) are protected against interception by means of ciphering
- Confidentiality of a user – this deals with the fact that when the network needs to address a particular subscriber, or during the authentication process, the unique IMSI (international mobile subscriber identity) should not be disclosed in plaintext (unciphered). This means someone intercepting communications should not be able to learn if a particular mobile user is in the area.

These 3 areas are covered in detail below.

Authentication

Authentication is needed in a cellular system to prohibit an unauthorized user from logging into the network claiming to be a mobile subscriber. If this were possible, it would be easily possible to "hijack" someone's account and impersonate that person (or simply making that person pay for the services). In fact, this was possible in some earlier cellular systems.

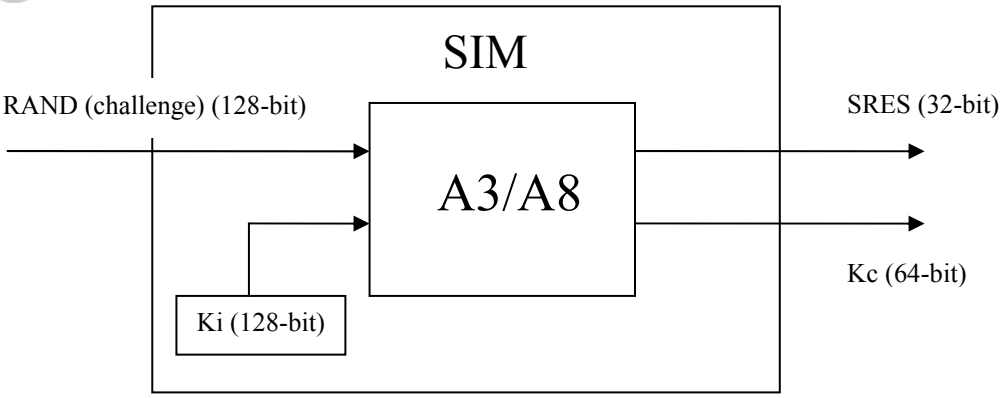
In order to solve this problem, some sort of challenge needs to be issued by the network which the mobile phone (MS) must respond to correctly.

The SIM card

Many users of GSM will be familiar with the SIM (Subscriber Identity Module) – the small smartcard which is inserted into a GSM phone. On its own, the phone has no association with any particular network. The appropriate account with a network is selected by inserting the SIM into the phone.

Therefore the SIM card contains all of the details necessary to obtain access to a particular account. These details come down to just 2 items of information.

- The IMSI – International Mobile Subscriber Identity – a unique number for every subscriber in the world. It includes information about the home network of the subscriber and the country of issue. This information can be read from the SIM provided there is local access to the SIM (normally protected by a simple PIN code). The IMSI is a sequence of up to 15 decimal digits, the first 5 or 6 of which specify the network and country (i.e. 50501 for Telstra, Australia)
- The Ki – the root encryption key. This is a randomly generated 128-bit number allocated to a particular subscriber that seeds the generation of all keys and challenges used in the GSM system. The Ki is highly protected, and is only known in the SIM and the network's AuC (Authentication Centre). The phone itself never learns of the Ki, and simply feeds the SIM the information it needs to know to perform the authentication or generate ciphering keys. **Authentication and key generation is performed in the SIM**, which is possible because the SIM is an intelligent device with a microprocessor.



The SIM authentication concept

Additional local security in the SIM

The SIM itself is protected by an optional PIN, much like an ATM PIN protects your ATM card. The PIN is entered on the phone's keypad, and passed to the SIM for verification. If the code does not match with the PIN stored by the SIM, the SIM informs the user (via the phone) that code was invalid, and refuses to perform authentication functions until the correct PIN is entered.

To further enhance security, the SIM normally "locks out" the PIN after a number of invalid attempts (normally 3). After this, a PUK (PIN UnlocK) code is required to be entered, which must be obtained from the operator. If the PUK is entered incorrectly a number of times (normally 10), the SIM refuses local access to privileged information (and authentication functions) **permanently**, rendering the SIM useless.

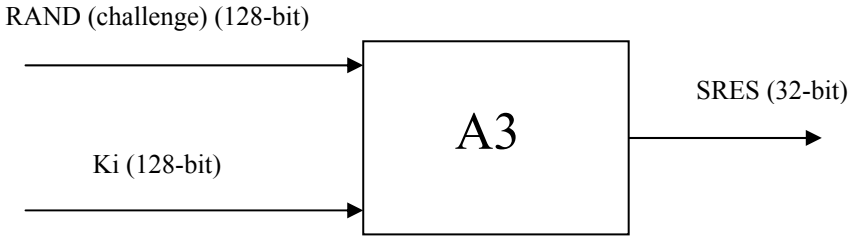
The A3 algorithm and authentication procedure

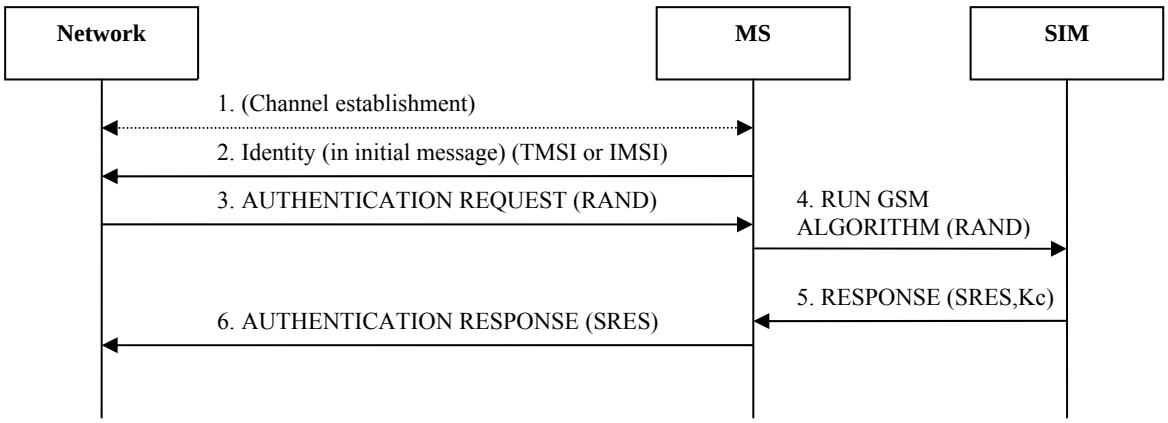
Now that we have established that there is a 'secret' Ki known only in the SIM and the network, the authentication procedure simply has to involve the SIM (via the phone) proving knowledge of the Ki. Of course, we could simply submit the Ki to the network for comparison when the network asks for it, but this is highly insecure, since the Ki could be intercepted.

Instead, the network generates a 128-bit random number, known as the RAND, which it then uses the A3 algorithm (see figure) to mathematically generate an authentication token known as the SRES. It then sends the RAND to the phone for the phone to do the same. The SIM generates the 32-bit SRES, which is returned to the network for comparison. If the received SRES matches the network's generated SRES, then the Ki's

must be the same (to a high mathematical probability), and the phone has proved knowledge of the Ki and is thus authenticated.

The RAND must obviously be different every time. Otherwise, if it were the same, an attacker could impersonate the user by sending the same SRES.





The above procedure can be summarized as follows:

(prior) – the network pre-generates a RAND and pre-calculates the SRES for that subscriber.

1. Some connection is attempted between the phone and the network.
2. The phone submits its identity. All potential messages used at the start of a connection contain an identity field. Where possible, it avoids sending its IMSI in plaintext (to prevent eavesdroppers knowing the particular subscriber is attempting a connection). Instead, it uses its TMSI (Temporary Mobile Subscriber Identity). This will be discussed later in this article.
3. The network sends the AUTHENTICATION REQUEST message containing the RAND.

4. The phone receives the RAND, and passes it to the SIM, in the RUN GSM ALGORITHM command.
5. The SIM runs the A3 algorithm, and returns the SRES to the phone.
6. The phone transmits the SRES to the network in the AUTHENTICATION RESPONSE message.
7. The network compares the SRES with its own SRES. If they match, the transaction may proceed. Otherwise, the network either decides to repeat the authentication procedure with IMSI if the TMSI was used, or returns an AUTHENTICATION REJECT message.

Authentication failure

If authentication fails the first time, and the TMSI was used, the network may choose to repeat the authentication with the IMSI. If that fails, the network releases the radio connection and the mobile should consider that SIM to be invalid (until switch-off or the SIM is re-inserted).

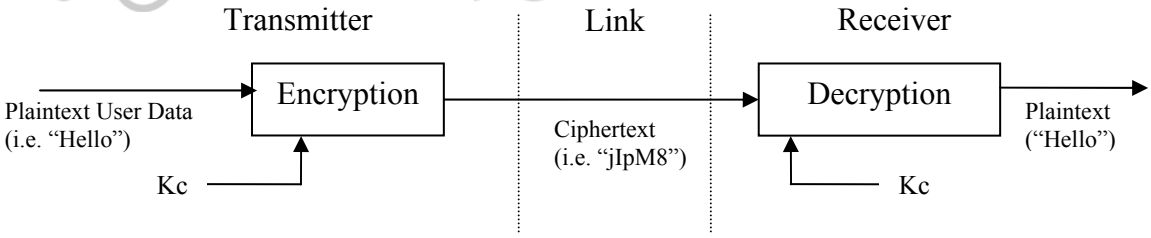
More on the A3 algorithm

The A3 algorithm does not refer to a particular algorithm, rather the algorithm the operator has chosen to be implemented for authentication. The most common implementations for A3 are COMP128v1 and COMP128v2. In fact, both of these algorithms perform the function of both A3 and A8 (the ciphering key generation algorithm – discussed later) in the same stage.

Whenever the SIM is asked to compute the SRES (with the RUN GSM ALGORITHM command) it also computes a new Kc (ciphering key – discussed later). Thus not only is the authentication procedure used to verify a user, it is also used whenever the network wishes to change keys.

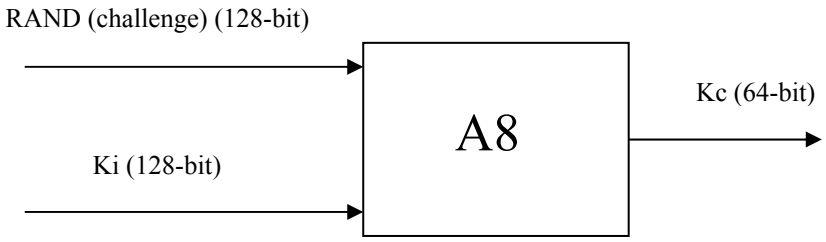
Ciphering

Ciphering is highly important to protect user data and signaling data from interception. The GSM system uses symmetric cryptography - the data is encrypted using an algorithm which is ‘seeded’ by the ciphering key – the Kc. This same Kc is needed by the decryption algorithm to decrypt the data. The idea is that the Kc should only be known by the phone and the network. If this is the case, the data is meaningless to anyone intercepting it.



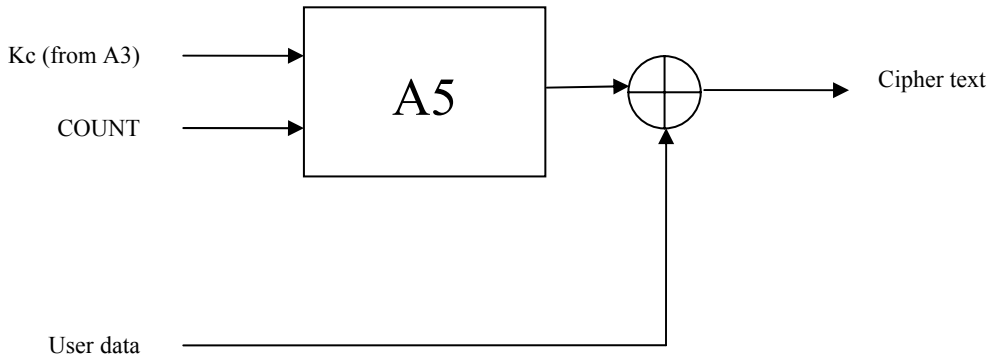
The Kc should also frequently change, in case it is eventually compromised. The method of distributing the Kc to the phone is closely tied in with the authentication procedure discussed above.

Whenever the A3 algorithm is run (to generate SRES), the A8 algorithm is run as well (in fact the SIM runs both at the same time). The A8 algorithm uses the RAND and Ki as input to generate a 64-bit ciphering key, the Kc, which is then stored in the SIM and readable by the phone. The network also generates the Kc and distributes it to the base station (BTS) handling the connection.



At any time, the network can then order the phone to start ciphering the data (once authenticated) using the Kc generated. The network can pick from a number of algorithms to use, as long as the phone supports the one chosen (this is indicated to the network earlier in a classmark message, which specifies the phone's capabilities).

The ciphering algorithm works by generating a stream of binary data (the cipher block), which is modulo-2 added (XORed) with the user data, to produce the ciphered text which is transmitted over the air. The data is decrypted by XORing the received data with the cipher block, which should be the same if the Kc is the same.



The algorithm is also 'seeded' by the value COUNT, which is based on the TDMA frame number, sequentially applied to each 4.615ms GSM frame. Internal state of the algorithm is flushed after each burst (consisting of 2 blocks of 57 bits each). In the case of multislot configurations, different cipher contexts are maintained for each timeslot. The same base Kc is used, however it is manipulated for each timeslot by XORing bits 32-34 of the Kc with the 3-bit timeslot number (0-7).

Ciphering algorithms

As mentioned above, the network can choose from up to 7 different ciphering algorithms (or no ciphering), however it must choose an algorithm the phone indicates it supports.

Currently there are 3 algorithms defined – A5/1, A5/2 and A5/3. A5/1 and A5/2 were the original algorithms defined by the GSM standard and are based on simple clock-controlled LFSRs. A5/2 was a deliberate weakening of the algorithm for certain export regions, where A5/1 is used in countries like the US, UK and Australia.

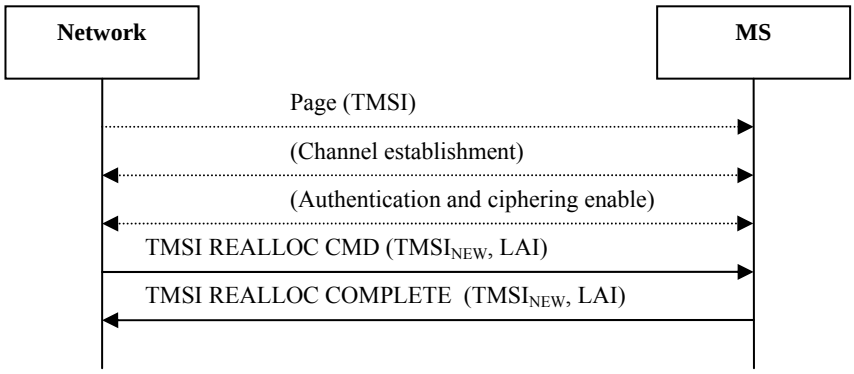
A5/3 was added in 2002 and is based on the open Kasumi algorithm defined by 3GPP.

Anonymity

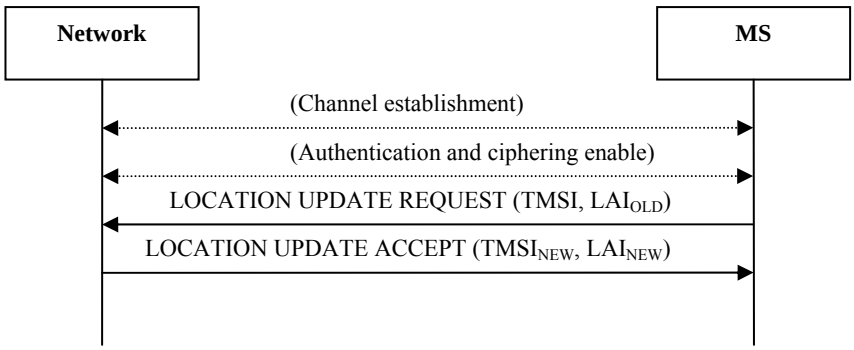
As mentioned above, one of the main goals of GSM security was to avoid having to use the IMSI (International Mobile Subscriber Identity) in plaintext over the radio link, thus stopping an eavesdropper from determining if a particular subscriber was in an area and what services they were using.

This is avoided by addressing the phones by a 32-bit TMSI (Temporary Mobile Subscriber Identity), which is only valid in a particular Location Area (i.e. one paging domain). The subscriber addresses itself or is paged by the 32-bit TMSI from then on.

The TMSI is updated at least during every location update procedure (i.e. when the phone changes LA or after a set period of time). The TMSI can also be changed at any time by the network. The new TMSI is sent in ciphered mode whenever possible so an attacker cannot maintain a mapping between an old TMSI and a new one and "follow" a TMSI.



Allocating a new TMSI (when not location updating or location change during connection)



Allocating a new TMSI (when moving into a new location area in idle mode)

The phone must store the TMSI in non-volatile memory (so it is not lost at switch off). It is normally stored in the SIM.

Initially of course the phone will have no TMSI, and thus is addressed by its IMSI. Once ciphering has commenced the initial TMSI is allocated. The VLR controlling the LA in which the TMSI is valid maintains a mapping between the TMSI and IMSI such as that the new VLR (if the MS moves into a new VLR area) can ask the old VLR who the TMSI (which is not valid in the new VLR) belonged to.

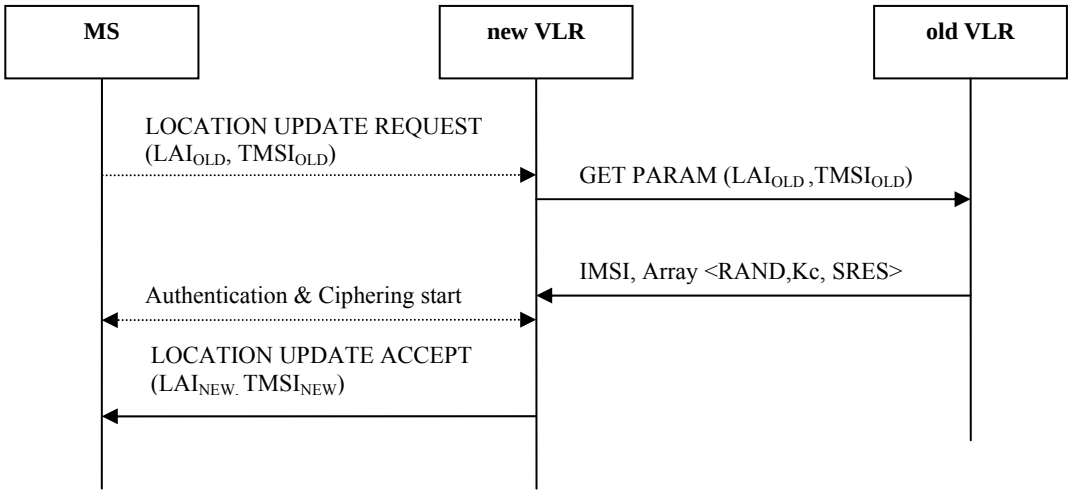
Distribution of the authentication and ciphering information throughout the network

As discussed above, the root key of all ciphering key generation and authentication is the Ki, stored only in the SIM and the network's AuC (considered to be part of the HLR – Home Location Register).

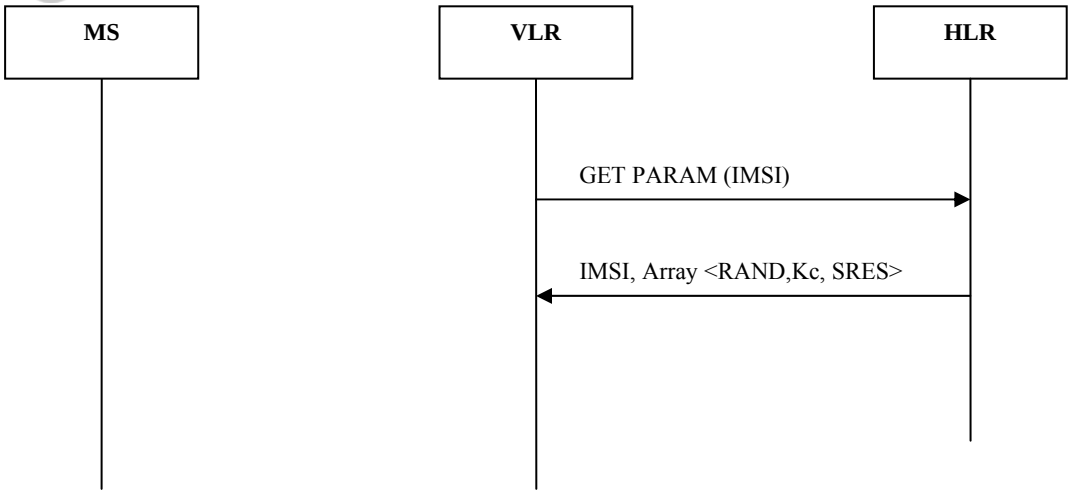
Therefore, when a particular VLR (Visitor Location Register), which serves one of more MSC (Mobile Switching Centres) and handles the mobility management for that area needs to authenticate a user, it must obtain information from the HLR. However, distributing the Ki to the VLR is a security threat, particularly if the subscriber is roaming, the foreign network would learn the Ki.

Furthermore, constantly asking for information from the HLR every time authentication is needed would cause a large signaling load on the HLR.

Instead, the HLR distributes authentication vectors, consisting of a valid SRES, Kc and RAND for the particular IMSI the VLR has specified. Normally multiple sets are distributed to the VLR for it to use. This reduces signaling load as well as keeping the Ki secret. Furthermore, the new VLR may be able to retrieve the unused vectors from the previous VLR when moving VLR areas.



Passing of authentication information between VLRs



Distribution of authentication vectors to the VLR

Implementations of A3, A8

Although the design of the GSM system allows an operator to choose any algorithm they like for A3 & A8, many decided on the one that was developed in secret by the GSM association, COMP128.

COMP128 eventually ended up in public knowledge due to a combination of reverse engineering and leaked documents, and serious flaws were discovered (as discussed below).

Some GSM operators have moved to a newer A3/A8 implementation, COMP128-2, a completely new algorithm which was also developed in secret. This algorithm for now seems to have addressed the faults of the COMP128 algorithm, although since it has yet to come under public scrutiny it may potentially be discovered via reverse-engineering and any possible flaws could be learned.

Finally, the COMP128-3 algorithm can also be used, it is simply the COMP128-2 algorithm, however all 64-bits of the Kc are generated, allowing maximal possible strength from the A5 ciphering algorithm (COMP128-2 still sets the 10 rightmost bits of the Kc to 0), deliberately weakening the A5 ciphering.

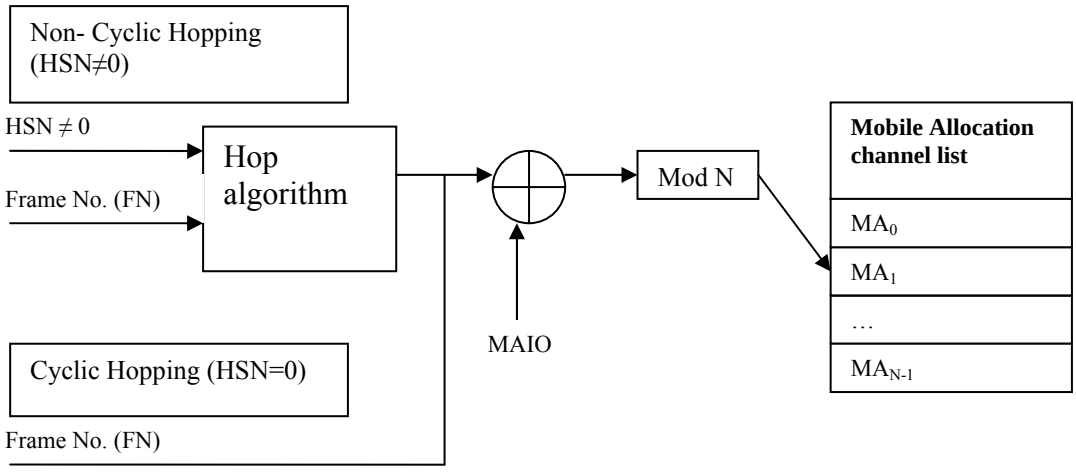
Frequency hopping

Finally, slow frequency hopping is used in GSM, where the transceiver changes physical carrier every frame (4.615ms), or about 217 times per second.

The hopping sequence is defined by two parameters – the HSN (Hopping Sequence Number), and the MAIO (Mobile Allocation Index Offset). There are 2 modes of hopping – cycling hopping and non cyclic hopping. In both modes, the MAIO simply chooses the phase in the hopping sequence that is to be used. If the HSN is 0, cyclic hopping is used where the mobile station simply steps through a set of frequencies (called the mobile allocation).

In non-cyclic hopping, the (publicly known) frame number is used to seed a more complex hopping algorithm.

In both cases, the hopping adds a layer of security. In order to decipher the stream with no knowledge (at the time of eavesdropping) of the hopping sequence, the entire bandwidth needs to be sampled. In both GSM900 and GSM1800 this can amount to tens of megahertz (although can be reduced if the operator's frequency allocation for that area is known).



Of course, if the channel information is allocated to the phone in plaintext, as it is at the start of the connection, then that sequence can be easily followed. But, typically, when setting up a data or voice (TCH) channel, an additional channel is allocated by sending messages on the initial channel when encryption has been enabled, thus making it significantly more difficult for an attacker to learn the TCH sequence.

The main problem security-wise with the hopping sequence is that the hopping sequence parameters on a particular base station are usually fairly static, and given the attacker has access to the network (easily achievable even with a prepaid kit), the typical hopping parameters of that base station can be learned quickly. Thus, there is not a great deal of security to be found in the hopping sequence, it simply adds just another layer of complexity for the attacker.

Flaws with these measures

Network does not authenticate itself to a phone

This is the most serious fault with the GSM authentication system. The authentication procedure described above does not require the network to prove its knowledge of the Ki, or any other authentication context to the phone.

Thus it is possible for an attacker to setup a false base station with the same Mobile Network Code as the subscriber's network. Since the authentication procedure initiation is up to the network's discretion, the false network may choose not to authenticate at all, or simply send the RAND and ignore the response. It does not have to activate ciphering either. The attacker can set the cell reselection parameters of his false base station to values that will highly encourage his 'victims' to camp on it – such as a high CELL_RESELECT_OFFSET.

The subscriber could then unknowingly be making calls or sending text messages that could be intercepted using this man-in-the-middle attack (as the false network could then route the calls back to the public telephone network).

Common implementation of A3/A8 is flawed – contains a narrow pipe

The most common implementation of the A3 and A8 algorithms is rolled into a single algorithm, COMP128, which generates the 64-bit Kc and the 32-bit SRES from the 128-bit RAND and the 128-bit Ki input.

This algorithm is seriously flawed, in that carefully chosen values for the input RAND will provide enough information to determine the Ki in significantly less than the ideal number of attempts (a brute force on the order of 2^{128} values). The flaw exists in the fact that in the second round of the algorithm, a narrow pipe exists (such that invididual bytes in isolated groups of 4 bytes in the second round output depend only on unique groups of 4 bytes in the input (2 of which are in the Ki, 2 are in the RAND)), and thus a collision attack can be performed.

Earlier attacks based on repeated 2R attacks could typically crack a SIM in approximately 2^{17} RANDs.

Dejan Kaljevic has written a utility (Sim Scan) which uses 2R, 3R, 4R and 5R attacks to obtain various bytes of the Ki, which he estimates can recover the Ki in around 18,000 RAND values on average. In my experience with this utility, it has recovered Ki's with between 7000 and 35,000 RAND attempts, i.e. around $2^{13} – 2^{15}$ attacks.

With this sort of attack, a Ki can be extracted within an hour (depending on the speed of a SIM – many smartcard readers can 'overclock' the SIM to higher clock rates such as 10MHz for faster extraction). At 6MHz I was able to achieve around 7 RAND attempts per second on some of the SIMs I tested.

Many have argued that this is not a serious attack, as it still requires physical access to the SIM (and the PIN, if one is used). Furthermore, some users have found it useful in order to 'clone' several of their SIMs onto a single programmable smartcard (to enable convenient switching between accounts).

However when paired with the previous concern (the network does not authenticate) the potential seriousness of this vulnerability is more clear, as described in a later section.

Common implementation of A3/A8 is flawed – reduces strength of ciphering key Kc

The common implementation of A3/A8, COMP128 has yet another "flaw", however this is almost certainly a deliberate weakening. When generating the 64-bit Kc, it always sets the least significant 10 bits of the Kc to 0. This effectively reduces the strength of the data ciphering algorithm to 54 bits (a reduction by a factor of 1024), regardless of which ciphering algorithm is used.

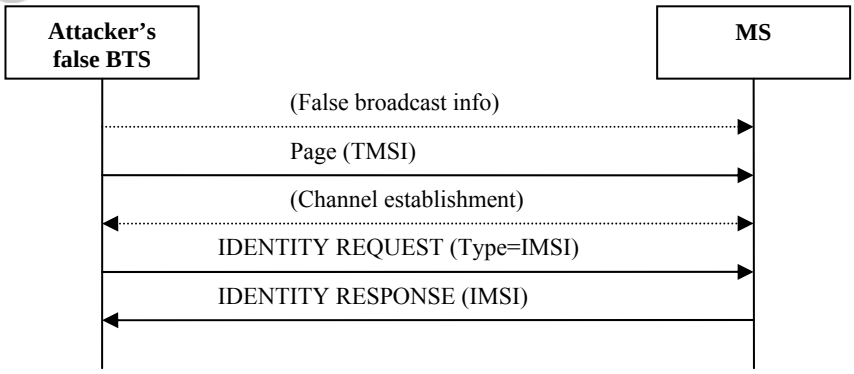
This same deliberate weakening is also present in the other algorithm of choice, COMP128-2.

Vulnerabilities in the subscriber identity confidentiality mechanism

As described earlier, the GSM specifications have gone to great length to avoid phone's being addressed (i.e. paged) or identifying themselves in plaintext by their IMSI. This is supposed to prevent an eavesdropper listening in on the initial plaintext stage of the radio communication learning that a particular subscriber is in the area (and what they are doing – i.e. the nature of communication can be known prior to ciphering – SMS, voice call, location update, etc).

Thus where possible the network pages users by their TMSI (Temporary Mobile Subscriber Identity) and maintains a database in the VLR mapping TMSIs to IMSIs.

If the network somehow loses track of a particular TMSI, and therefore cannot determine who the user is, it must then ask the subscriber its IMSI over the radio link, using the IDENTITY REQUEST and IDENTITY RESPONSE mechanism. Obviously, the connection cannot be ciphered if the network does not know the identity of the user, and thus the IMSI is sent in plaintext.



Combined with the previously described flaw that the network does not have to authenticate itself to a phone, an attacker can use this to map a TMSI to its IMSI. The attacker accomplishes this by imitating a legitimate base station of the subscriber's network, and paging that subscriber by its IMSI. The subscriber's phone will then establish a radio connection, and then the attacker can simply send the subscriber the IDENTITY REQUEST(Identity Type=IMSI) message, and the phone will respond with the IMSI.

Over the air cracking of Ki

The previous mentioned flaws combined can result in a more serious attack.

By imitating a legitimate GSM network, the attacker can then use the authentication procedure many times to exploit the vulnerabilities in COMP128.

To accomplish this, the attacker would imitate a valid base station with the same mobile network code as the customer's network. It would then page the mobile phone, either by its IMSI (or TMSI), to establish a radio (RR) connection to it.

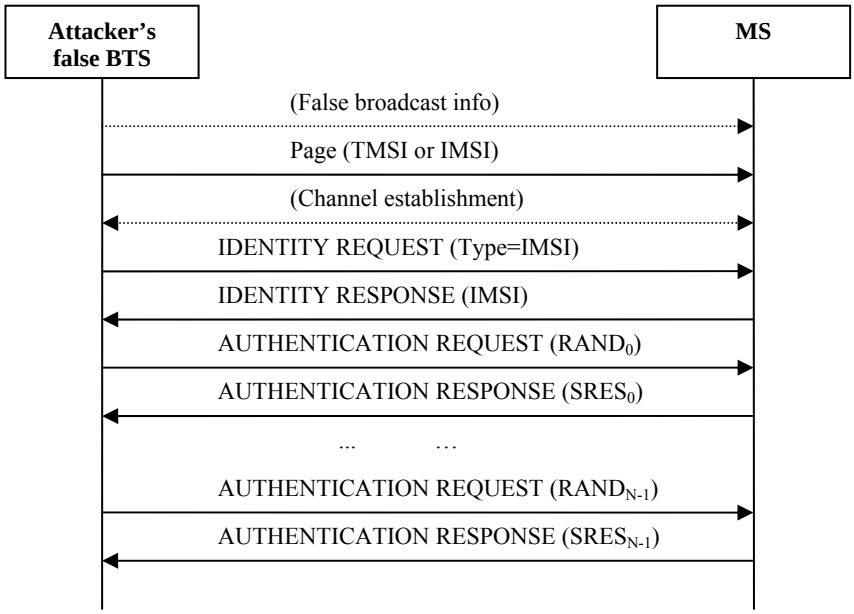
Once the connection is established, if the phone was paged by its TMSI, the IMSI can easily be found out by sending the phone the IDENTITY REQUEST command (to which the phone must respond at any time).

Following this, the attacker can keep choosing RANDs as such to exploit the COMP128 algorithm flaws and submitting them to the phone via AUTHENTICATION REQUEST messages (imitating a legitimate network asking the phone to authenticate itself). The phone, as required, simply returns the SRES. The attacker could then repeat the authentication requests many times, collecting the SRESes until he/she has gained enough information to learn the Ki.

Once the Ki and IMSI are known the attacker can impersonate that user, and make and receive calls and SMS messages in their name. It can also be used to eavesdrop, since

RANDs from the legitimate network to the legitimate user can be monitored, and thus combined with the known Ki can be used to determine the Kc used for the encryption.

This attack will work on any GSM phone, without any previous access to the phone (or even knowledge of the IMSI) – a random phone's TMSI can be chosen by monitoring radio traffic. Being an over-the-air attack, it can even be performed from many kilometers away.



We will initially pick conservative values for our calculations, because even then it can be shown that GSM is highly vulnerable to over-the-air attacks. For the COMP128 flaws, we will use repeated 2R attacks, resulting in approximately 2^{17} RANDs needed to find the Ki.

We must now determine the rate at which the phone can be challenged with RANDs over the air. Authentication typically happens on the SDCCH (Standalone Dedicated Control Channel), which in either mode (SDCCH/4, SDCCH/8) allows 1 radio-block (consisting of 4 bursts) in every 51-muliframe period (defined to be 120*51/26 milliseconds, or 235ms). The transmit and receive timing (transmit lagging) is skewed by 15 frames + 3 timeslots, equating to 15*120/26 + 3*120/26/8 or about 71ms. The authentication messages AUTHENTICATION REQUEST and AUTHENTICATION RESPONSE will each fit into their own SDCCH block. This means if the phone wishes to respond to the challenge in the next available SDCCH slot it has about 71ms – 4*120/26 = 52ms to decode and process the information, pass it to the SIM to run A3/A8, receive the SRES from the SIM and code and transmit it back to the network. This would certainly be an unrealistic expectation for some equipment. The GSM specifications allow up to 12 seconds for authentication processing, however we will be more realistic and decide the phone can process it by the next SDCCH transmit period, or 52ms + 235ms or around 288ms, a more reasonable expectation.

Using these figures, a challenge is delivered in the first 235ms period, and the response is sent back in the next 235ms period (at the same time as a new challenge is delivered). Thus around 4 challenges can be computed per second. Thus the expected time to recover the Ki in this method would be around 2^{17} RANDs / 4 RAND/s = 2^{15} seconds or 9 hours.

Now most terminals are battery powered and would be unable to sustain constant transmission for 9 hours (or may move out of range of the attacker). However, the attack can simply be broken into many time periods, so long as the attacker retains the 'context' of the challenges he has sent and received.

Furthermore, these estimates as stated were based on the 2R attack. By using more sophisticated attacks (such as used by Dejan Kaljevic's Sim Scan, which uses 2R, 3R, 4R and 5R attacks) this time could be reduced even more (Dejan estimates his Sim Scan can recover the Ki in an expected time of 18,000 RANDs). If this were the case, in this example, the key could be recovered in just over an hour.

In Australia, the 2 largest operators (Telstra and Optus) which have approximately 80 per cent of the mobile market share use COMP128 and are vulnerable to this attack.

Ciphering occurs after FEC

In the GSM system, like all digital radio communications systems, forward error correction (FEC) is used over the radio link to assist in the correction from errors caused by noise or signal fading.

FEC works by adding redundancy to the data stream, thus increasing the amount of bits to transfer.

The problem in GSM is that ciphering occurs after FEC, meaning the redundant stream of bits is then modulo-2 (XORed) added with the ciphering stream, meaning the known redundancy patterns could be used to assist in a crypt-analytical attack. In the most simple example, suppose the error correction codes consist of the convolutional-code polynomials $G_1 = 1$ and $G_2 = 1$, corresponding to a simple twice repetition of bits. Suppose a received 6-bit sequence is 101110. Thus, it can be concluded that the 6-bit ciphering stream consisting of the bits abcdef is such that:

a=b', c=d, and e=f'

Of course it is far more complicated than that; in GSM the data is interleaved over many blocks and the coding on most encrypted channels is a ½ rate (i.e. non punctured) convolutional code generated by polynomials with m from 5 to 7. Furthermore, on voice channels, only certain bits are protected with the convolutional code (unequal error protection).

However combined with knowledge of the A5 ciphering algorithm, a cryptanalyst could use this information to reduce the number of keys to search from the ideal (2^{64}) to something less.

Flaws in A5/1 and A5/2 algorithm

The A5/1 output is based on the modulo-2 summed output of 3 LFSRs whose clock inputs are controlled by a majority function of certain bits in each LFSR.

However, Alex Biryukov, Adi Shamir and David Wagner demonstrated in a paper that A5/1 could be cracked in under 1 second on a typical PC (however large precomputed tables are required, amounting to about 2^{36} bytes or 64G).

The attack exploits flaws in the algorithm when storing these tables utilizing a combination of what has been learnt through statistical analysis of the states the algorithm steps through, as well as exploiting the poor single-bit taps used to control the LFSR clocks.

A5/2 is a deliberately weakened version of A5/1, which has been demonstrated to be also flawed. A5/2 can be cracked on the order of about 2^{16}, and thus is even weaker than A5/1, if that is possible to believe!

Details of the A5/1 algorithm are listed in Appendix A.

Measures taken to address these flaws

The GSM specifications are not static, their have been many revisions to the standard, to add technologies such as GSM1800, HSCSD, GPRS and EDGE. In fact, the standard still evolves today into 3rd generation ("3G") technologies such as UMTS. In recent versions of the standard, particularly UMTS, much has been done to improve upon the security flaws discussed in this paper. They are described below.

GSM - Newer A3/A8 implementation

As discussed earlier, newer implementations of A3/A8 have been introduced, namely COMP128-2 and COMP128-3.

So far these algorithms have held up reasonably well however they are still developed in secret. COMP128-2 still has the deliberate 10-bit weakening of the ciphering Kc however. COMP128-3 is the same basic algorithm without this weakening (i.e. a truly 64-bit Kc).

COMP128-2 and COMP128-3 have stopped SIM cloning and also make the serious over the air Ki extraction unfeasible, even if they don't approach the ideal strength of 2^{128}.

Finally, 3GPP have defined completely new, open, authentication algorithms for use with the UMTS system, finally putting to rest the COMP128 nightmare.

GSM - A5/3 ciphering

As mentioned previously, GSM supports up to 7 different algorithms for A5 (ciphering). Until recently, only the A5/1 and A5/2 algorithms were used. In 2002, GSM added a much stronger algorithm, A5/3 which is based on the Kasumi core (the core encryption algorithm for UMTS). Only few networks and handsets support this algorithm currently, however.

GPRS – GEA3 ciphering

Similarly to A5/3, a new GPRS ciphering algorithm based on Kasumi has been added to the GPRS system, called GEA3.

GPRS/UMTS – Ciphering before FEC

In GPRS the ciphering is performed at a higher layer in the protocol stack – the LLC (Logical Link Control) layer. RLC/MAC messages are not ciphered. The FEC then applies at the physical layer.

Similarly in UMTS ciphering occurs at the RLC/MAC layer, which sits just above the physical layer, at which FEC is performed.

UMTS – Network authentication to phone

In UMTS the possibility of an attacker imitating the network has been removed by means of a 2-way authentication procedure.

The procedure for which the mobile authenticates itself to the network is largely unchanged from GSM, however the network now sends an Authentication Token (AUTN) along with the RAND. The AUTN consists of a sequence number (SQN) encrypted using the RAND and the root key (K). It also consists of the MAC code, which works much like the GSM SRES but in the opposite direction.

Thus if the XMAC does not match the MAC calculated by the SIM then phone then send an authentication reject message to the network and the connection is over.

Finally, to stop an attacker simply "replaying" the legitimate network's authentication request the SIM keeps track of the sequence numbers used (i.e. the same sequence number cannot be used twice, each sequence number must be newer).

UMTS – Improved, stronger algorithms

Completely new algorithms are used in UMTS for all security purposes.

Authentication and key generation

In much the same way as the GSM model, all authentication and key generation functions are performed in the SIM and the network's AuC. However, the algorithms typically have stronger input parameters.

The algorithms all still operate on a hidden 128-bit root key now known as K, known only in the SIM and AuC.

They are briefly described below:

f1 – is used to generate the authentication token (MAC) which has a similar purpose to SRES in GSM, but is used to authenticate the network to the mobile (i.e. the mobile tests if the network has knowledge of the root key K. It's inputs are RAND (128-bit), K (128-bit), the sequence number SQN (48-bit) and AMF (a value that can be used for implementation specific purposes).

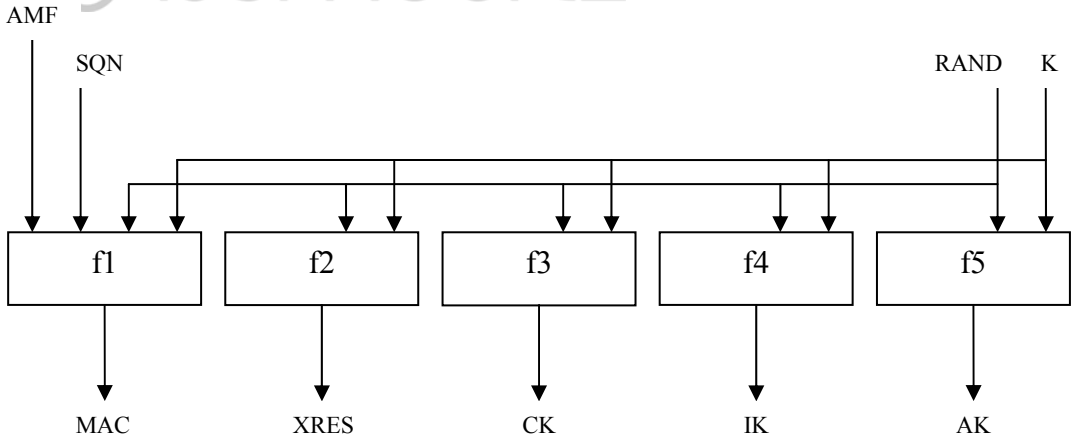
f2 – is used to generate the XRES, similar to the SRES but 128-bits long. Inputs are K and RAND

f3 – Used to generate the 128-bit ciphering key CK. Inputs are K and RAND.

f4 – used to generate the 128-bit integrity key IK. Inputs are K and RAND. The IK is used to 'sign' radio control messages, discussed later.

f5 – used to generate the 128-bit authentication key AK (48-bit) , which is used to encrypt (XORed with) the Sequence number SQN when it is sent to the mobile station.

Like GSM, these algorithms can be operator specific, but similarly to GSM it is expected operator's will use a common implementation. Such an implementation has already been drafted by 3GPP, and is known as MILENAGE, which is fully open to the public (it's based on AES).



Ciphering and integrity

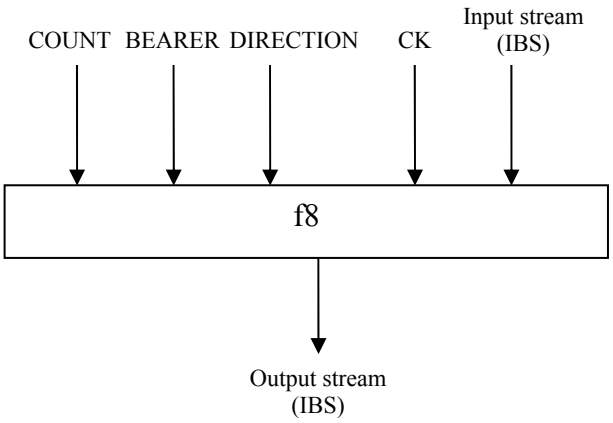
UMTS radio links are ciphered in a similar way to GSM. The network may choose from predefined ciphering algorithms, that are supported by the phone. The ciphering algorithm set is known as the f8 function.

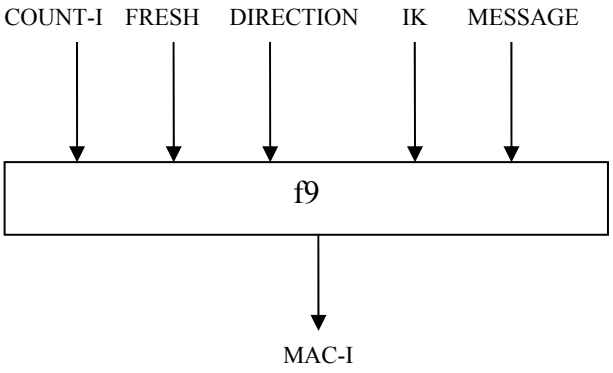
The inputs and outputs of f8 are similar to GSM, in that the 128-bit ciphering key (CK) is input to the algorithm, as well as new additional parameters DIRECTION (the direction of data flow), COUNT-C (ciphering sequence number) and BEARER (unique value for each multiplexed radio link for a user). A cipher stream block is produced and XORed with the data to be transmitted. Currently only one implementation of f8 is defined – known as UEA1, based on the Kasumi algorithm.

Additionally UMTS uses integrity checking on RRC (Radio Resource Control protocol) signaling messages between the base station (Node B) and the phone. Integrity checking consists of including a 32-bit field to each message called the MAC-I, which is essentially a digital ‘signature’ proving that the MS sent the message.

The signature is generated as a function (called f9) of the 128-bit integrity key (IK), DIRECTION (direction of data flow), the entire radio message, COUNT-I (based on the RRC sequence number and frame number), and FRESH, a unique value used for the duration of a connection (to stop the user being able to replay messages in future connections).

Both current implantations of f8 and f9 are based on Kasumi – an open standard algorithm developed by 3GPP, which has been subject to much analysis. Kasumi is a 8-round Feistel cipher, whose S-boxes can be easily implemented using combinational gate logic (i.e. easy hardware implementation).





Conclusion

It can be seen that although GSM had security in mind when drafting the original specifications, GSM fails to deliver on most of the criteria described in GSM 02.09.

GSM's faults result from a combination of designing algorithms in secret (security through obscurity) and deliberate weakening of the system (i.e. A5/2 and COMP128). Most will agree that keeping algorithms protected is a bad idea, as it prevents public scrutiny (until it is too late), and eventually the algorithm will be exposed anyway.

Fortunately for most users however, the concerns are not great. None of these exploits are easily carried out, so the casual telephone user is safe from people 'snooping' in on

random conversations. Those using GSM for highly sensitive information (i.e. Government or military) should think twice however.

And thus it seems the "undocumented" goals of GSM security have been accomplished – that is, the system is safe from general busybodies however not secure enough to stop organizations like Government spy agencies to intercept communications.

References

1. "GSM Cloning" (Goldberg & Briceno) - http://www.isaac.cs.berkeley.edu/isaac/gsm-faq.html
2. "Reducing the collision probability of alleged Comp128" (Handschuh & Paillier) - http://www.gemplus.com/smart/r_d/publications/pdf/HP00comp.pdf
3. "An implementation of the GSM A3A8 algorithm. (Specifically, COMP128.)" (Briceno, Goldberg & Wagner) - http://www.mirrors.wiretapped.net/security/cryptography/hashes/a3a8/a3a8.c
4. "A pedagogical implementation of the GSM A5/1 and A5/2 'voice privacy' encryption algorithms" (Goldberg, Briceno & Wagner) - http://kiwibyrd.chat.ru/gsm/a512.htm
5. "Real Time Cryptanalysis of A5/1 on a PC" (Biryukov, Shamir & Wagner) http://www.cs.berkeley.edu/~daw/papers/a51-fse00.ps
6. GSM 02.09 - Digital cellular telecommunications system (Phase 2+); Security aspects (GSM 02.09 version 8.0.1 Release 1999)
7. GSM 03.20 - Digital cellular telecommunications system (Phase 2+); Security related network functions (GSM 03.20 version 8.1.0 Release 1999)
8. GSM 04.08 - Digital cellular telecommunications system (Phase 2+); Mobile radio interface layer 3 specification (GSM 04.08 version 6.11.0 Release 1997)
9. GSM 05.02 - Digital cellular telecommunications system (Phase 2+); Multiplexing and multiple access on the radio path (GSM 05.02 version 8.5.1 Release 1999)
10. GSM 05.03 - Digital cellular telecommunications system (Phase 2+); Channel coding (GSM 05.03 version 8.5.1 Release 1999)
11. GSM 05.08 - Digital cellular telecommunications system (Phase 2+); Radio subsystem link control (GSM 05.08 version 8.5.0 Release 1999)
12. GSM 11.11 - Digital cellular telecommunications system (Phase 2+); Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface (GSM 11.11 version 8.3.0 Release 1999)
13. 3GPP TS 33.102 – Technical Specification Group Services and Security Aspects; 3G Security; Security architecture (Release 6)
14. 3GPP TS 35.201 - Technical Specification Group Services and System Aspects; 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 1: f8 and f9 Specification (Release 5)
15. 3GPP TS 25.202 - 3G Security; Specification of the 3GPP Confidentiality and Integrity Algorithms; Document 2: KASUMI Specification (Release 5)

16. 3GPP TS 35.205 - 3G Security; Specification of the MILENAGE Algorithm Set: An example algorithm set for the 3GPP authentication and key generation functions f1, f1*, f2, f3, f4, f5 and f5*; Document 1: General (Release 5)
17. GPP TS 35.206 - 3G Security; Specification of the MILENAGE Algorithm Set: An example algorithm set for the 3GPP authentication and key generation functions f1, f1*, f2, f3, f4, f5 and f5*; Document 2: Algorithm Specification (Release 5)
18. 3GPP TS 55.216 – 3G Security; Specification of the A5/3 encryption algorithms for GSM and EDGE, and the GEA3 encryption algorithm for GPRS; Document 1: A5/3 and GEA3 specification (Release 5)

Appendix A – A5/1 implementation

Basic algorithm

The A5/1 algorithm is simply the modulo-2 addition of the output of 3 LFSRs, which are defined by the polynomials:

(shift register A) $X^{19} + X^5 + X^2 + X + 1$
(shift register B) $X^{22} + X + 1$
(shift register C) $X^{23} + X^{15} + X^2 + X + 1$

However, the algorithm is slightly complicated by controlling the clock inputs of each of the LFSRs. An LFSR is only clocked if a certain bit (the control bit) of it 'agrees' with the majority of the control bits from each register.

Let R_i denote the i'th bit of shift register R (from the right, i.e. input to the shift register)

Let S (the sum of the controlling bits) = $A_8 + B_{10} + C_{10}$

Let M = 1 if S >= 2 (i.e. the majority of controlling bits are 1) and 0 if S<=1 (i.e. the majority of the controlling bits are 0). Thus M represents what the majority of the control bits values.

Let CLK_R be the clock input to register R.

$CLK_A = (A_8 \text{ XOR } M)`$
$CLK_B = (B_{10} \text{ XOR } M)`$
$CLK_C = (C_{10} \text{ XOR } M)`$

where ` indicates the bitwise complement.

This clocking mechanism makes divide-and-conquer style attacks less feasible, but not impossible.

Initial state

The algorithm is 'seeded' by the 64-bit Kc (ciphering key) and 22-bit COUNT (frame number) as follows:

Each bit of the 64-bit Kc is modulo-2 added to the input (bit 0) of each shift register and then the register is clocked, regardless of the majority function, thus each register is clocked 64 times in total – one for each bit of the Kc (the least significant bit of the Kc is clocked first).

The same procedure is repeated for the 22-bit COUNT, i.e. each bit of the count is clocked in (least significant bit first) with the majority function disabled, thus the registers are clocked 22 times each.

The algorithm is then run for 100 clock cycles in the normal manner with output discarded.

Generating cipher stream

The algorithm is initialized for each burst, generating 114 bits of information for each direction. Thus, 228 bits are produced (except in the case of EDGE, where 2*348 = 696 bits are produced).

The algorithm is run for 114 clock cyles with the output producing the cipher stream used for BLOCK1 – for encrypting the network to MS data.

The algorithm is then run for 100 clock cycles in the normal manner with output discarded.

The algorithm is then run again for 114 clock cycles with the ouput producing the cipher stream used for BLOCK2 – for encrypting the MS to network data.

Appendix B – Algorithms used by Australian operators

GSM/GPRS

Operator	**A3/A8 (auth/key**	**A5 (ciphering)**	**Kc strength**	**GPRS ciphering**

	generation)			
Telstra	COMP128	A5/1	54-bit	GEA1
Optus	COMP128	A5/1	54-bit	GEA1
Vodafone	COMP128-2	A5/1	54-bit	GEA1

UMTS

Operator	**f1-f5 (auth/key generation)**	**f8 (ciphering)**	**f9 (integrity)**	**CK strength (PS and CS)**	**IK strength (PS and CS)**
Three	MILENAGE	UEA1 (Kasumi)	UIA1 (Kasumi)	128-bit	128-bit