3GPP TSG SA WG3 Security — SA3#33 **S3-040360**
10-14 May 2004
Beijing, China

Source: **Sarvar Patel, Lucent Technologies Inc.**
Title: **Eavesdropping without breaking the GSM encryption algorithm**

Document for: **Discussion and decision**
Agenda Item: **GERAN network access security**

Contact: Sarvar Patel sarvar@lucent.com
Michael Marcovici marcovici@lucent.com

Abstract: We describe an attack that allows an attacker to eavesdrop on conversations made by GSM phones without breaking the GSM encryption algorithm. Previous attacks were based on tampering the network's message to turn on encryption or depended on the use weak or breakable encryption algorithms. We assume that the handset has been made secure against these attacks for example by mandating that encryption has to be used and that only unbroken encryption algorithms (e.g. A5/3) are used. Yet we are able to show how a man-in-the-middle type (MITM) attacker can eavesdrop on conversations by exploiting weaknesses in the GSM protocol without breaking any algorithm.

We would like our solutions to this attack to also work against other MITM attacks like the one recently described in [BBK] which exploits weak encryption. First we describe a two steps MITM attack which is similar to the above attack but different from the [BBK] attack. Unfortunately, the proposed solutions against the [BBK] attack are not effective against this two steps attack. Finally, we build on the promising approach of [Brookson] to have the network cryptographically authenticate the cipher mode command message. Our solution protects against both types of attacks by augmenting the existing GSM challenge/response protocol to become a limited mutual authentication and session key agreement protocol.

1. Introduction

The GSM mobile (MS) and the basestation (BSS) protect a call or session's content whether voice, data, or signaling by encrypting them using a session key, Kc, which changes for each session. The session key, Kc, is derived from a root key, Ki, which is stored in the user's SIM card inserted in the handset and is also stored in the authentication center AuC in the network. The root key is also used by the network to authenticate the MS at the beginning of the call.

A MS is associated with one GSM network called the home network and the MS at a given point can be either in its home network or roam into a different visited network. The BSS/MSC/VLR, either in the home network or the visited network communicates with the AuC in the home network to get authentication and session key values. At the start of a call the BSS/MSC/VLR gets (RAND, SRES, Kc) triplets from the AuC, or it should have gotten a triplet values ahead of time (Fig 1). The BSS/MSC/VLR then selects one triplet and sends the RAND challenge value to the MS. The MS forwards the RAND value to the SIM where the A3 algorithm is executed using the root key Ki and the RAND value to create the SRES value (Fig 2). The SIM forwards this

value to the MS which in turn sends the SRES over the air to the BSS/MSC/VLR. The BSS/MSC/VLR compares the SRES value received over the air to the SRES value received from the AuC; if they match then the MS has been successfully authenticated.

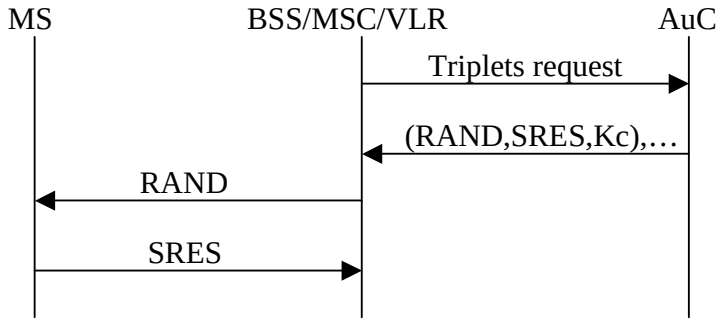


Fig 1: GSM Authentication Protocol
(e.g. at call origination)

A session key, Kc, associated with the RAND, is also sent by the AuC to the BSS/MSC/VLR as part of the triplet. This Kc value is computed in the AuC by executing the A8 algorithm with the root key Ki and the RAND value. The same Kc value is also computed by the SIM and is sent to the MS.

Once the authentication has been performed, the network sends the cipher mode command to commence encryption of the channel using the A5 encryption algorithm. The MS may support multiple A5 encryption algorithms, so the network informs the MS which specific version of A5 algorithm to use when it tells the MS to start encryption. Various A5 encryption algorithms have been specified by 3GPP including A5/1, A5/2 and A5/3. An extremely efficient attack has been recently shown against A5/2 [BBK]. A5/1 has also been attacked, although the attacks are less efficient. There are no known attacks on the A5/3 algorithm which is based on the Kasumi block cipher.

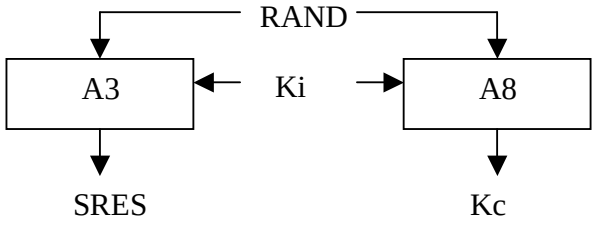


Fig 2: Authentication and session Key generation

The encryption algorithm A5 takes the session key Kc as the first input and the frame number, FN, as the second input, and outputs two 114 bit blocks, $Block1_{FN}$ and $Block2_{FN}$, of pseudorandom outputs (Fig 3). These pseudorandom bit strings are XORed with the plaintext in the uplink and the downlink respectively as shown in figure 4. The message block at the current frame, M_{FN}, and the pseudorandom stream block at the current frame, $BLOCK1_{FN,}$ in the uplink are used to create the ciphertext block $C_{FN} = M_{FN} \oplus BLOCK1_{FN}$ which is sent over the air. On the network side the $BLOCK1_{FN}$ is also generated and the received ciphertext is used to recover the

message $M_{FN} = C_{FN} \oplus BLOCK1_{FN}$. Similarly, in the downlink the message block M'_{FN} is XORed with the pseudorandom stream block, $BLOCK2_{FN}$. The same procedure is used for both dedicated signaling and for traffic channels. A session consisting of N+1 frames includes many messages, $M_{FN}, M_{FN+1}, \ldots, M_{FN+N}$ which are XORed with $BLOCK1_{FN}$, $BLOCK1_{FN+1}, \ldots$, $BLOCK1_{FN+N}$. We will simply call the pseudorandom stream $BLOCK1_{FN}$, $BLOCK1_{FN+1}, \ldots$, $BLOCK1_{FN+N}$ of the session as the BLOCK stream. The sequence of message blocks and ciphertext blocks will be referred to as the message or plaintext stream and the ciphertext stream respectively.

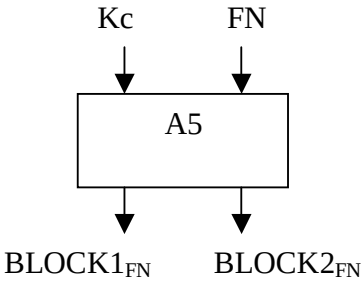


Fig 3: Pseudorandom bit stream generation for frame FN

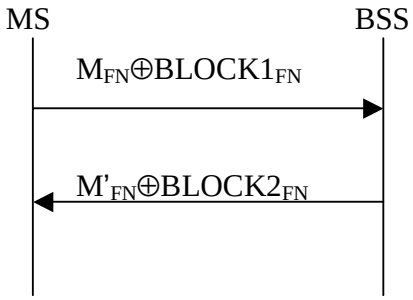


Fig 4: Encryption of uplink and downlink

It is clear that GSM uses a unilateral authentication and key agreement protocol rather than a mutual authentication and key agreement protocol. This was thought to be acceptable because from a service fraud point of view it is only necessary that the network authenticate the user and from a user privacy point of view it was thought that even if a BSS impersonator can trick the MS to engage in a conversation, the impersonator would not have the encryption key Kc and thus would not be able to understand the message from the user and nor would he be successful in sending meaningful messages to the user.

Previous attacks: There have been several attacks described on GSM security. Handsets may allow sessions without encryption if the network so requests. Unfortunately, the network's request is not authenticated and this results in an easy MITM attack with the attacker requesting that the MS never turn on encryption [Mitchell]. Note that this is a protocol only attack and does not require the breaking of any algorithm. The standard lets a handset give an encryption on/off indication to the user which may block this attack from working on more judicious callers. Also this attack can be handled if it is mandated that encryption has to be always used by a MS. Our attack works even when encryption is mandated or when the handset displays an encryption indication and the user checks it. Some protocol attacks on non-GSM networks were also previously described [Patel].

Other published attacks have focused on the A8 algorithm, COMP128, which has been successfully attacked. Also A5 algorithm cryptanalysis has seen some significant results culminating in [BBK] which also describe MITM attacks where the attacker forces the MS to use a weaker encryption algorithm which they show how to break quickly to recover the Kc key and then to use Kc to encrypt the conversation with the true BSS.

Our work: The results in this paper are divided into two parts. First we describe a novel MITM attack which allows an attacker to eavesdrop on the conversation of a MS originating a call without breaking the A5 encryption algorithm or even performing any cryptanalysis. We go into the details of how such an attack would work in GSM systems or GSM type systems. Also we explain why such attacks are successful.

In the second part we look at solutions which work not only against this attack, but also work against other MITM attacks like the [BBK] attack recently described which exploits weak encryption. Similar to the attack on strong encryption, we describe a two steps MITM attack, different from the [BBK] attack, when we have weak encryption. Unfortunately, the proposed solutions against the [BBK] attack are not effective against the two steps attack. Finally, we build on the promising approach of [Brookson] to have the network cryptographically authenticate the cipher mode command. Our solution protects against both types attacks by augmenting the existing GSM challenge/response protocol to become a limited mutual authentication and session key agreement protocol.

We describe in section 2 how we can eavesdrop without breaking the GSM encryption algorithm. In section 3 we give details of the attack working in GSM networks. In the section 4, we explore MITM attacks against weak encryption algorithms. In section 5 we propose a solution that works against described attacks. Finally, we conclude in section 6.

2. Overview of the attack

We describe a new "BLOCK stream replay MITM" attack which works in two phases. The first phase is the collection phase which has to be performed once with the MS that will be attacked. The second phase is the actual attack phase which can be repeated by a MITM attacker against the MS performing call origination as often as desired.

2.1 Collection Phase

The attacker uses a legitimate GSM mobile and places a legitimate phone call to the target MS[1] and carries a conversation with the user of the MS (Fig 5). In the meantime the attacker records the RAND that the GSM network used to set up the encryption key, Kc, for the MS. The attacker does not know the Kc, however, he records the entire ciphertext stream exchanged between the GSM network and the MS on the uplink and the downlink and their frame numbers. On the attacker's MS side the attacker records all the associated plaintext during the call. Since the attacker placed a call through his phone, the plaintext is available for the attacker to record.

Lets discuss the collection on the uplink side at the target MS, but analogous steps also apply to the downlink side. Furthermore, at the MS side, lets number each of the message frame by a frame number, FN. So at FN, M_{FN} is sent by MS which will eventually be forwarded to the attacker's MS. However, at the target MS side only the ciphertext C_{FN} is available over the air. This ciphertext as previously described was created as $C_{FN} = M_{FN} \oplus BLOCK1_{FN}$. Since both C_{FN}

[1] We will refer to the "target MS" simply as the "MS" and refer to the MS used by the attacker as the "attacker's MS".

and M_{FN} are collected by the attacker, he can recover the pseudorandom bits at each frame, $BLOCK1_{FN} = C_{FN} \oplus M_{FN}$ and thus can recover the entire BLOCK1 and BLOCK2 streams associated with the RAND[2]!

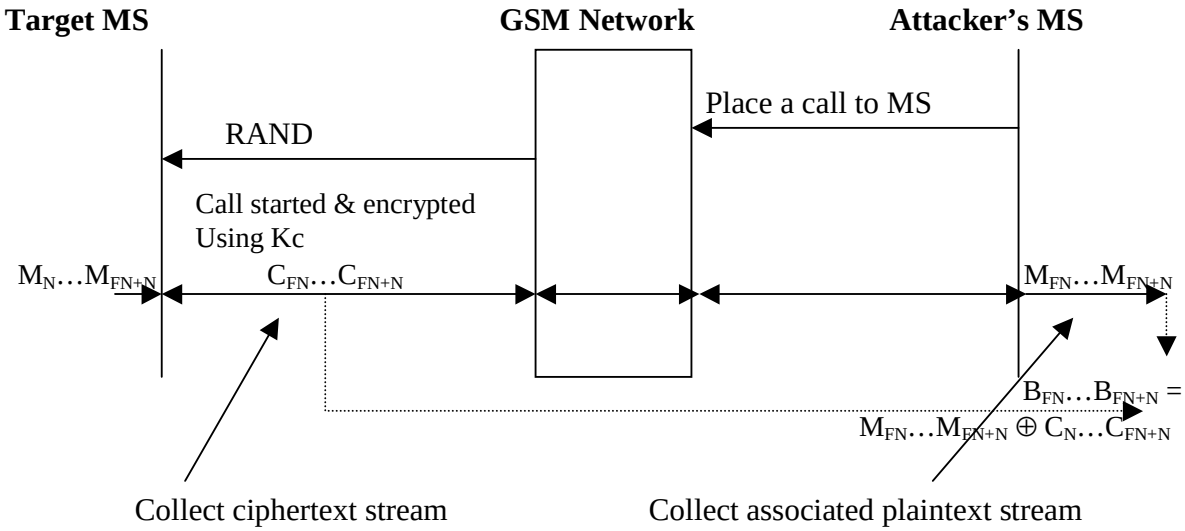


Fig 5: Collection Phase; the attacker collects uplink and downlink ciphertext on the target MS side and collects associated plaintext on his side.

2.2 MITM Attack Phase

Now the attacker is prepared to perform a MITM attack on the MS in phase 2. Although the attacker does not possess the session encryption key, Kc, for the MS associated with a RAND value, he has something essentially equivalent. He knows the pseudorandom bit streams BLOCK1 and BLOCK2 that are associated with the key Kc for the duration of a call. The second phase of the attack is illustrated in Figure 6. In step 1 the network impersonator continues to broadcast the current frame number, FN, on the sync channel that a MS should use as the frame number when communicating with the cell. The attacker sets this number to the beginning frame numbers that was used in the collection phase for which the attacker knows the pseudorandom stream values $BLOCK1_{FN}$ and $BLOCK2_{FN}$.

[2] The pseudorandom bit masking happens after the speech is coded for error correction/detection, hence, the message M in reality is not just the speech frame but the error coded speech frame.

In step 2, the radio channel is set up and in steps 3 the challenge response authentication protocol is performed; the network impersonator sends the RAND value that was used in the collection phase. In step 4 the network sends the cipher mode command to tell the MS which encryption algorithm to use and to start the encryption; the mobile replies with the cipher mode complete acknowledgement. At this point encryption has been turned on. In step 5 the MS sends a call origination set up message including the dialed digits of the called party.

At this point the network impersonator can send encrypted messages to the MS and decrypt ciphertext received from the MS. This is so because the MS is using frame numbers from the collection phase for which the attacker has the pseudorandom outputs $BLOCK1_{FN}$ and $BLOCK2_{FN}$. Thus to decrypt ciphertext C_{FN} received in the uplink, the network impersonator calculates $M_{FN}=C_{FN}\oplus BLOCK1_{FN}$ and to encrypt a message M_{FN} to send on the downlink the network impersonator sends $C_{FN}=M'_{FN}\oplus BLOCK2_{FN}$.

In step 6a, the attacker is able to decrypt the setup message with the dialed digits received from the MS in step 5 and use its phone to place a call to the called party. In step 6b the call is setup and data is sent on the traffic channel between the called party and the attacker's phone. Also in step 7 the MS and the network impersonator exchange data on the traffic channel. In step 8 the attacker connects the two calls so that the two ends, the MS and the called party are not aware that the MITM attacker is eavesdropping on the call. This is surprising since even though encryption is on and even if an unbreakable encryption algorithm is used the GSM encryption can be defeated without any cryptanalysis.

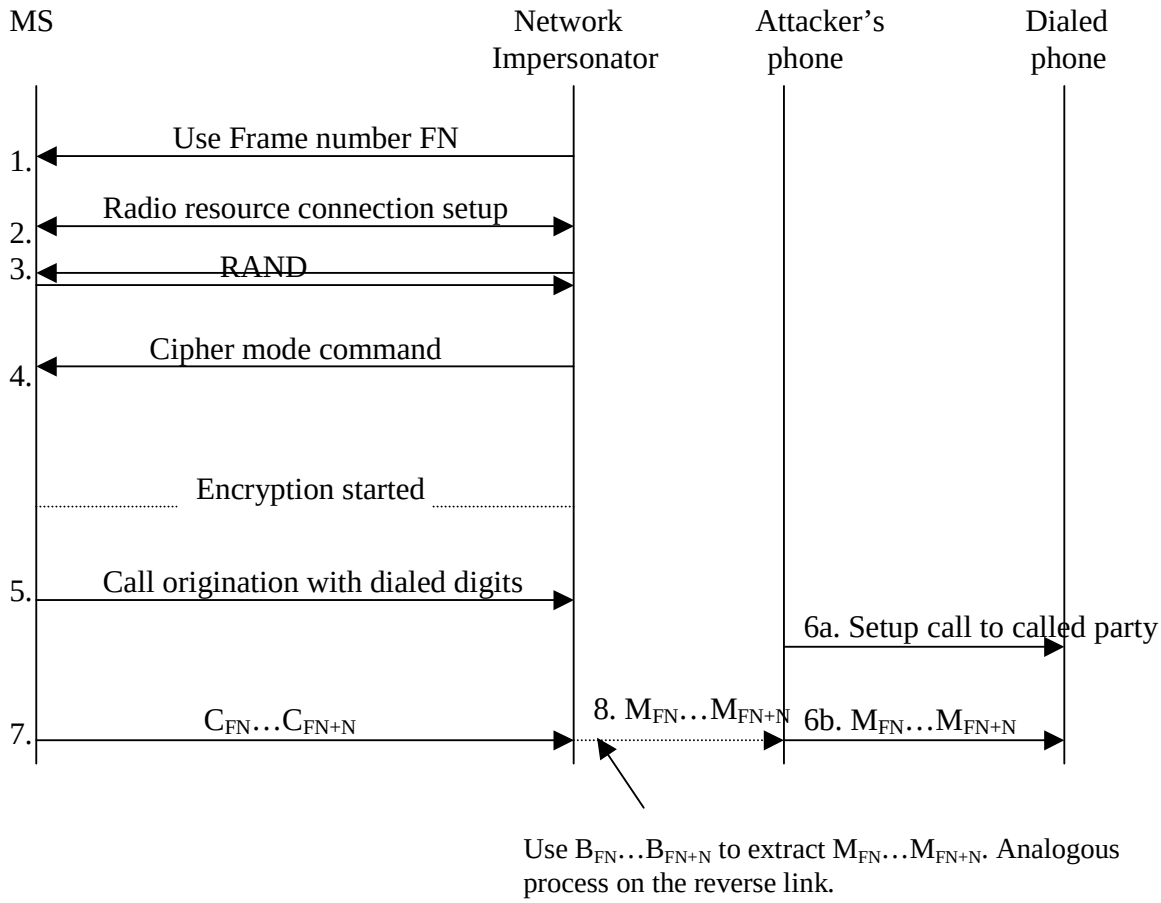


Fig 6: Phase 2 of the Session replay MITM attack

3. Details of the attack

The principles of the stream replay MITM attack described above are rather straightforward, however, certain details of the GSM system can in practice make the attack more involved; in this section we explore some of these details. Also a GSM system can make configuration choices from various options (e.g. codec selection) available. These choices can render the attack easy or in some cases practically infeasible. We do not explore all the possible configurations and their impacts, but rather work with examples of how the attack can proceed in a real system.

The GSM security/encryption layer is believed to provide protection against eavesdropping and message insertion independent of what is happening at upper layers or what applications are running. The upper layer applications could be voice, data, SMS, different layers of signaling, email, etc, it shouldn't matter; the air link should be independently protected by the GSM security/encryption layer. This was believed to be true even though GSM only used unilateral authentication. The stream replay MITM attack shows that this belief is false. The goal of giving example collection phases described below is to give at least one concrete example of how the attack could happen, it isn't suppose to be the best way or the way that works for all operators, but just an example of higher layer application/configurations for which the GSM security/encryption layer does not provide protection and thus to establish the principle at hand.

There are certain theoretical limitations to the attack. First the eavesdropping is successful only when the MS performs a call origination, it is not successful during a call termination or page response because in that case the challenge RAND is chosen by the legitimate network and not by the network impersonator. In practice, this need not be a serious limitation because an attacker can usually force a call terminated at the MS in to a call origination. Suppose the MS has received a call, the attacker can add interference and eventually cause the call to drop. Usually, both sides of the call would try to call each other assuming that there was “inadequate signal” which caused the call to drop. If the target MS attempts call origination, the attacker can succeed with the stream replay MITM attack.

3.1 Call or message spoofing

There are two damaging things an attacker can do to the MS. First as we discussed during call origination the attacker can do a stream replay MITM attack and eavesdrop. Secondly, the attacker can place a call to the MS (i.e. call termination) and pretend that a call from some other phone had been placed. For a voice call this attack is of a limited value since it is difficult to impersonate another speaker in the speech and conversation patterns. Nevertheless, this attack would be more successful if data was sent by the attacker. Previously, it was assumed that an attacker (i.e. network impersonator) should not be able to send meaningful data to a user because since the attacker would not possess the encryption key, Kc, any ciphertext the attacker sends to the MS would be decrypted with high probability in to a meaningless message. Now that we are able to effectively possess a Kc, by possessing a pseudo random stream, we can send meaningful messages to the MS in a call or session.

3.2 Displaying of Calling party number

Another limitation of the session replay MITM attack is that when the called party answers the phone, the calling party number that might be displayed is not of the MS but of the attacker's phone. However, this is not a serious limitation for multiple reasons. For this limitation to be effective it must be the case that the calling party ID must be an authenticated or protected value in a call made from any telephone network. Even if the MS's home network or the current network protect the calling party ID, the attacker only needs to place his call through some telephone network which does not protect the calling party ID.

Even if all the networks do secure the calling party ID, in practice the called party would accept calls in most cases. Firstly many phones, especially land lines, do not have the equipment or the service to display the calling party's phone number; against them the attack would have no limitations. Secondly, humans do not always call from only one number rather they make calls from many different phones, including home numbers, mobile numbers, work numbers, visited location, etc. Thus it is common to accept phone calls from a person that originates from a new number as long as the voice and the speaker are recognizable. Hence, it is the voice that is ultimately used to authenticate the call rather than the displayed calling number. In the end, the calling party number cannot be always relied upon by the receiver because most systems allow the caller, for privacy reasons, to disable the sending of the calling number.

Removing the Limitation: However, for the purist who wants an attack that works with the right calling party ID, there is a way for the attacker to do this by doing double the work. Suppose the attacker wants to hear future calls between MS A and MS B, but also wants to properly convey the calling party ID during the call. Then the attacker can first do the collection phase against both MS A and MS B. Later the attacker uses two network impersonators, one near MS A and one near MS B. When MS A places a call to MS B, the first impersonator intercepts the call and then relays the call to the second impersonator who places a call to MS B. Both impersonators have pseudo random streams from the collection phase which will be used to properly decrypt and encrypt the speech. Since all information sent from MS A to MS B is forwarded, hence the proper calling party ID information is also properly forwarded. Since the telecommunication networks are not involved in either side they will not be checking for any caller ID verifications.

3.3 Working with small amounts of plaintext

Another potential limitation is the difficulty in collecting a large amount of plaintext. Since the attacker (possibly a stranger) is placing a call to the MS it would be difficult for the attacker to carry a long conversation with the owner of the MS. This would mean that the session replay MITM attack length is limited to the length of the session in the collection phase. If the attacker is successful in carrying a long conversation in the collection phase then this is not a limitation which may be the case if the attacker knows the owner of the MS (e.g. an employee wanting to eavesdrop on his supervisor), but in general it could be difficult even though only one long conversation is needed.

Fortunately for the attacker there is a simple solution when the attacker was only able to carry on a short conversation with the MS owner. When the attacker is about to run out of frame numbers for which he has BLOCK values in the attack phase, the network impersonator can try to reset the current frame number to the earliest value it had used. One way the network impersonator can get the MS to use a previous frame number is to force a handover to a second BS which is also

impersonated by the attacker. The second BS would be broadcasting the earlier frame number on its sync channel. Thus when the MS does complete its handover, it will start using the earlier frame number and the attacker has now doubled the length of the conversation he can support in phase 2. By repeating this handover and resetting the frame value, the attacker can carry a conversation for an indefinite time even though he had collected only a very limited amount of speech or plaintext in the collection phase. The amount of stream needed for replaying is thus dependent upon the duration it takes, under good condition, for a handover to take place. There may be easier ways to reset the frame number for example when a synchronization burst is sent.

We have assumed above that once encryption is turned on it cannot be turned off by a cipher mode setting command during the call or at handover. If this isn't the case then the attacker can simply use the small amount of downlink pseudo random stream collected to start the call and then turn off encryption even if encryption use is mandated at the beginning of the call. This is different from the [Mitchell] attack where encryption is never turned on even at the beginning; but this is not a completely satisfactory attack because the handset could give an indication to the user that encryption is turned off. The advantage of not changing the cipher mode is that the attacker is able to eavesdrop without detection while the user may be informed (via the handset indication) that encryption is still on.

3.4 Collecting undistorted plaintext

During the collection phase we presented the attacker as simply getting the plaintext by recording the speech played at his end. However, depending on the specifics of the GSM network some distortion may be introduced into the speech. Suppose the attacker had placed a call from one mobile to the target MS; in this case the GSM network can introduce speech distortion if it converts the GSM coded speech in to A,μ Law speech when the MS's GSM network interfaces to the circuit switch world and one more time from A,μ Law to GSM coded speech by the GSM network of the receiving mobile. Thus up to two conversions may be required in mobile to mobile calls. From a land line to a mobile only one conversion is needed. It is not clear how the distortion caused by these conversion affect the final speech parameters. If values of many parameters remain essentially unaffected and only few parameters are effected then the attacker assumes that at each frame in the collection phase, instead of knowing all 114 bits of the plaintext and hence the 114 bits of BLOCK pseudorandom bits, the attacker will only know part of the 114 bits. This increases the attackers work significantly.

Fortunately, the attacker can sidestep this issue, for the GSM standards in Release 98 have specified a tandem free operation (TFO) mode where GSM-coded speech is sent end-to-end without any transcoding in the middle. If the current MS's GSM network supports TFO then the attacker only has to place the call to the MS from another mobile on that GSM network. If the current GSM network does not support TFO then the attacker can wait till the MS roams into a GSM network that does support TFO to complete the collection phase.

Voice activation detection (VAD) and discontinuous transmission (DTX) can also make the collection phase difficult because frames would not be transmitted during the silent period, hence, for some frames the pseudo random stream would not be collected. There are few things an attacker can do to get around this problem. First, the attacker can wait till the handset roams in to a network that does not turn on DTX. Or instead of waiting for the handset to roam, the attacker can place his partner in the network which doesn't turn on DTX and relay the call data to the partner so that the handset appears as if it has roamed in to the appropriate visited network (see

subsection 5.2 “Location independent attacks” below). The same could have been done with the TFO issue discussed above.

The decision to use DTX can be taken by the network on a per call basis and the mode can be different in the uplink and downlink part of the conversation. In the collection phase, the attacker can make sure that effectively DTX is not used for both the uplink and downlink. Recall that the attacker places a call to the target MS using the attacker’s MS. The attacker can make sure that the attacker’s MS has DTX disabled in the uplink and hence the target MS will always be receiving a frame on its downlink. Also the attacker can (via network impersonation) tell the target MS not to use DTX on its uplink during set up, so that the MS will always be sending frames and the attacker’s MS will be receiving them. Thus the pseudo random streams for both the uplink and downlink can be recovered. If for some reason this is not possible and there are gaps in the known uplink streams then they can be filled as described in subsection 3.3.3.

3.3.1 Working in a non-TFO environment

As we mentioned that if a visited network does not use TFO then the attacker can wait till the MS roams to a network that does use TFO. But it may be the case that none of a operator’s network implement TFO including all the allowed networks a MS can roam in to. In this case, attacker can still discover the pseudorandom stream in a non-TFO environment, but has to go about it differently. We will see that the attacker can still discover the target MS’s downlink stream directly, but cannot discover the uplink stream without doing additional work.

During the collection phase the attacker would place a call to the target MS using an ISDN phone, this is just so that the attacker knows the digital bits sent to the GSM network and doesn't have to worry about the unknown analog to digital conversion processes distortion that might be introduced if an analog link was used. The attacker knows all of his speech bits which will be sent to the GSM network where they will be converted to GSM coding from A,μ law. But this is a deterministic process and hence the attacker also know what bits the GSM BSS will send to the target MS. Since the attacker also recorded the encrypted speech bits sent over the air, he can XOR them out to recover the pseudo random stream for the downlink. It is only the uplink pseudorandom stream from the target MS that the attacker would have difficulty finding out. To find out the uplink pseudorandom stream, the attacker would have to do more work in the collection phase. An example of discovering the uplink pseudorandom values is given in subsection 3.3.3.

3.3.2 Working with the AMR Codec

The attacker can try to first avoid working with AMR codec by either waiting till the MS roams to a visited network that uses an older codec or the attacker can try to do a location independent attack as long as one network supports an older codec. If it is the case that all of the networks a MS can roam to are only using AMR codec then the collection phase has to work with it. The AMR codec does not use a fixed source rate which introduces variability that the attacker has to work to overcome in the collection phase. The first variability is due to the ability to choose either a full rate or half rate traffic channel based on the cell load. We assume that during the collection phase no switching between the half rate and full rate traffic channel happens. The second variability is due to code rate adaptation; if the channel is bad the speech coder uses a low source rate and more bits are allocated for forward error correction, and if the channel is good then a higher source rate is used and less bits are allocated for error protection. There are 8 different source bit rates used by the AMR codec.

We here assume the non-TFO mode, so our goal is to find out the downlink pseudorandom stream values. Knowing the downlink pseudorandom stream, the target MS's uplink pseudorandom stream can be discovered using techniques described in subsection 3.3.3. During the collection phase, an attacker knows the speech bits that will be received by the GSM network assuming the attacker has used an ISDN line. The mapping from the A,μ law to a GSM codec is deterministic and known. So the attacker knows for each 20 msec what the value of the bit string is that will be encrypted by the GSM BSS, however, with an AMR codec there are 8 possible source rates that could be used. So for a 20 msec duration the attacker knows the 8 different bit strings that the GSM BSS may have sent, but does not know which one of the 8 bit strings was encrypted and transmitted by the BSS. We briefly explore some potential ways to narrow down the choices, but this requires further investigation.

If the channel condition was constant and known to the attacker then the attacker can narrow the choice for the entire pseudorandom stream. But this is difficult in general. An attacker can try to add interference between the target MS and the true BSS to force the AMR codec to use the lowest source rate for the entire call during the collection phase. This way the attacker knows what the bits look before and after encryption and hence can recover the pseudorandom stream. Another possibility is for the attacker to insert a false basestation close to the target MS and a handset impersonator close to the true basestation and have them linked with a good link and relay the bits send between the true basestation and the target MS. The hope is that this may keep the channel in good condition so that the attacker's choice is narrowed for the entire stream. For those parts of the pseudorandom stream which are not correct, the attacker can try to narrow the choice down by performing an added step after the call in the collection phase. The attacker, acting as a network impersonator, makes a guess out of the 8 possible string at a frame number and sends a message to the target MS and sees if the target MS receives the frame correctly by observing the reaction of the MS. As an example if a message is sent to drop the channel, if the correct encryption was used then the result should be observable. This can be repeated to narrow the choice down to a single one for a frame and can also be repeated for different frames. Since this observation can happen at the signaling layer also, the user does not need to be aware of this.

3.3.3 Discovering uplink stream given downlink stream

It is sometimes useful for an attacker to discover the uplink pseudorandom stream given that only downlink pseudorandom stream is known. This can be thought of as part of the collection phase that happens after the downlink pseudorandom stream has been discovered using a legitimate call. We briefly explore some potential ways to do this, but this requires further investigation. On a channel when no messages are being sent if the channel is filled with known fill bits then the encrypted bits and the known fill bits can be XORed to recover the uplink pseudorandom stream. Since the attacker knows whether it has send messages to the MS, the attacker likely knows when the MS is not transmitting new information on the uplink and only using fill bits.

Also if the above is not the case then the attacker, impersonating as the network, can ask queries to the target MS on the signaling layer, whose answers the attacker already knows, and gets encrypted answers back from the MS. Then the attacker can XOR the encrypted response with the expected plaintext response to recover the pseudorandom stream for that frame. This can be repeated as often as desired with many frame values; since it's happening at the signaling layer, the user would not be aware of it. This works because the same pseudorandom stream for a frame is used whether traffic or signaling bits are sent. The attacker is able to query the target MS

because the attacker knows the downlink pseudorandom stream and can send encrypted commands which will be properly decrypted by the MS and hence answered back.

An example of a query could be for the GSM BSS after establishing the radio channel with the frame value of interest and starting encryption to send a CLASSMARK ENQUIRY to request classmark information. The MS will respond with a CLASSMARK CHANGE message and if the attacker knows the classmark information because most phones in an operator network have the same classmark then the attacker can recover the uplink pseudorandom stream value for that frame by XORing the encrypted text and the expected plaintext. This can be repeated for other frames until sufficient uplink pseudorandom stream values are discovered.

3.5 SACCH expirations

Another potential problem may be the slow associated control channel (SACCH) frames which cause the channel to be disconnected if too many frames are in error. The SACCH channel uses one frame from the 26-frame multiframe on the traffic channel (TCH) in the full rate case. The BLOCK value which encrypts the SACCH frame needs to be somewhat known otherwise when the network impersonator sends the SACCH ciphertext frame, the MS will decrypt it with high probability of error which will be detected due to CRC. If too many frames are in error then the channel would be dropped. Collecting the BLOCK value during the collection phase is not direct since we do not know what the values of the power control messages are for example that are sent on the SAACH. However, since some of the fields are fixed or have redundancy, partial values of the BLOCK can be recovered. Note that not able to decrypt the SACCH frames on the uplink is not an issue, we are mainly concerned with the frames on the downlink that the MS is decrypting.

Another way to mitigate this is to set the frame error value that causes disconnection to a high value in the Broadcast channel. The largest value the cell can set would cause the MS to accept around 30 seconds of SACCH frame errors. A more durable solution might be to shift the frame number so that the known BLOCK values overlap the SACCH frames and the unknown BLOCK values overlap some speech frame. An error in speech frame would not cause the channel to be dropped and concealment would mitigate against speech degradation. Furthermore now we can use the SACCH frames to send handover commands to the MS.

3.6 SMS attack

Eavesdropping: SMS messages are also vulnerable to the stream replay MITM attack. After a collection phase the network impersonator can try to eavesdrop on SMS messages in the attack phase. When the MS sends a SMS message on the radio link which is encrypted by a session key Kc, the network impersonator knows how to decrypt the SMS message because the attacker knows the BLOCK pseudorandom bit stream that is associated with the frame number of the frame used to send the SMS message. Thus the network impersonator is able to listen to the SMS message. The attacker has an option to discard the message at this point or else the attacker can forward the SMS message to the intended destination. However, in the later case the source address of the SMS message would be different than the address of the MS unless the attacker has a way to spoof messages.

This last limitation (i.e. source address change) can be removed by noticing that the MS retries to deliver an unsuccessful SMS message. After successfully eavesdropping on the SMS message

sent by the MS, the attacker doesn't send an acknowledgement to the MS as the MS would expect. Instead the impersonator sends an error message (or nothing) and goes silent. The MS will reconnect with the legitimate network again (which the attacker allows this time) and try to send the SMS message which will succeed and reach the destination with proper MS source address. Thus the attacker has eavesdropped without causing any detection!

Spoofing: Another damage an attacker can do is to send bogus messages to the MS. Knowing the BLOCK values on the downlink the network impersonator can create any SMS message originating from any source address, encrypt it using the current frame's BLOCK value and send it to the MS. The MS will decrypt the message and accept the SMS message as coming from some legitimate source address.

3.7 Analysis of the attack

It might be tempting to think that if some specific and accidental feature of the GSM system was different (e.g. TFO) then may be the session replay MITM attack would not go through. We should avoid the temptation because there are good reasons for the non-security features, for example TFO is very useful in reducing the speech distortion caused by converting between GSM-coded speech and A,μ Law speech. And these decisions about configurations and upper layer applications are typically made independent of security considerations. The real reason the attack goes through is that the session key agreement is not mutual but unilateral. If the authentication and session key agreement protocol was mutually authenticated then even if an attacker discovers the BLOCK values for different frames in a session it won't be of any use because for each session a new session key would be created and the network impersonator would not be able to force the MS to reuse some previous key. Thus the simple replay of the session or its BLOCK values is not possible.

The attack became much more powerful because as described in section 3.3 a small amount of plaintext or equivalently the BLOCK values are enough to carry the attack by resetting the frame numbers. This means a partial replay of the session is also possible. Even if the session key is mutually authenticated an attacker who discovers part of the plaintext and the associated BLOCK values can keep resetting the frame numbers and using the BLOCK values to receive or send messages encrypted by those BLOCK values. To make sure this does not happen the sequence numbers used to create BLOCK values within a session should be protected in some way against replay. One way is to make sure that they are monotonically increasing and have an anti-replay detection method at the MS and at the BSS. Furthermore, at handover the sequence number used to create BLOCK values should be handed off to the new cell to make sure they are not repeated.

4. Weak encryption MITM attacks

Although our stream replay MITM attack does not depend on the MS using a weak encryption algorithm we explore weak encryption MITM attacks in this section because we want our solution to the stream replay MITM attack to also work against the weak encryption MITM attacks. We also look at proposed solutions to the weak encryption MITM attacks and see if they are successful against the stream replay MITM attacks.

[BBK] presented a weak encryption MITM attack which proceeds as such. Suppose the network expects the mobile to use a strong encryption algorithm, perhaps A5/3, however, the MS can perform either the strong encryption algorithm or the weak encryption algorithm like A5/2. A MITM attacker pretends to be a mobile to the network and receives the RAND challenge. The MITM attacker pretends to be a network to the MS and sends the RAND challenge and asks the MS to use the weak encryption algorithm A5/2. The attacker breaks the encryption and recovers the Kc key. The attack can now use Kc to encrypt and decrypt, using A5/3, the conversation between the true network and itself. Thus the attacker is able to eavesdrop on the entire conversation between the MS and the network while relaying the messages.

4.1 Proposed solutions

Special RAND: One proposal is to partition the RAND space in such a way that a part of the RAND space is used when weak encryption algorithm is used and another part of the RAND space is used when the strong encryption algorithm is used. The idea is that even if a MITM attacker is able to break the weak encryption algorithm, he will not be able to use the recovered key to encrypt/decrypt using the stronger encryption algorithm because the session key used with the stronger encryption is independent of the session key used for the weaker encryption algorithm due to the RAND values being different in the two cases. An advantage of his solution is that only the handset needs to be upgraded and the AuC has to change when it creates the triplets and forwards them to the VLR, but the flip side is that the AuC needs to know whether the strong or the weak encryption algorithm is in use in the visited network.

Authenticate cipher mode command: This solution is proposed in [Brookson] where it is observed that the [BBK] attack is successful because the cipher mode command message is not cryptographically protected when the BSS tells the MS to start encryption using a particular encryption algorithm. If we denote the choice of A5 by the parameter algflag then the unused bits following the cipher mode command message can be filled with hash(Kc, SRES, algflag). The MS will proceed with encryption only if it is able to verify the hash value. More abstractly, [Brookson] has made the observation that the algflag should be protected from tampering via a MAC (e.g. HMAC). A disadvantage of this solution is that along with the upgrade of handsets, it requires the BSS to be updated everywhere to attach the MAC value to the cipher mode command message. On the positive side, it does not require any upgrade in the SIM or any other NE in the network.

4.2 Two steps MITM attack

Unfortunately, there is another MITM attack that works with a weak encryption algorithm and is not protected by the previous two solutions. This attack also works in two phases. In the first phase the attacker tries to discover a RAND and Kc pair. Suppose the MS roams in to a visited network that uses a weak encryption algorithm then the attacker can gather the ciphertexts from the session and break the encryption algorithm to recover Kc.

In phase 2 this attack is similar to the phase 2 of the stream replay MITM attack where the attacker impersonates the network and tries to intercept a call origination attempt by the MS. The network impersonator challenges the MS with the RAND associated with the weak encryption and makes the MS use the Kc to encrypt the conversation. The attacker will then place the call to the called party intended by the MS via another phone and patch the two phone calls together.

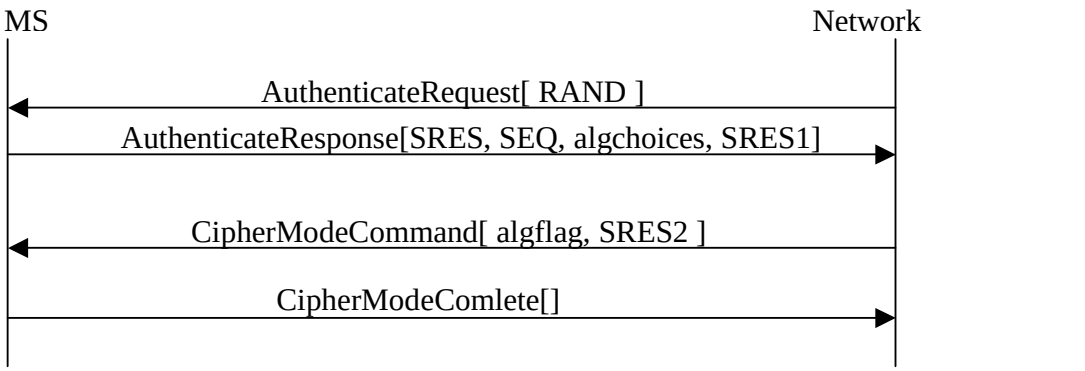
The MITM then forwards the messages from one call to another while properly encrypting and decrypting the messages. Note that the same limitation as in section 3.2 applies here.

We shall see that the previous solutions do not protect against this two steps attack. First the RAND variation attack is not by itself sufficient because the attacker will use the weaker RAND value and cause the MS to use the weaker encryption with the known Kc value. Similarly, the proposal in [Brookson] will not be sufficient because the attacker possesses the key Kc and can use it to create the MAC of the cipher mode command message with any algorithm it chooses. It should be noted that the attacker can happily request that a stronger algorithm be used as long as Kc is the session key to be used.

It should also become clear now that neither the 'special RAND' proposal nor the 'authenticate cipher mode command message' proposal will protect against the stream replay attack.

5. Limited mutually authenticated session key generation

In this section we describe a limited mutually authenticated session key generation protocol which protects against both the stream replay MITM attack and the various weak encryption MITM attacks. It is limited in the sense that its security depends upon the VLR or other entities in the network not to reveal any RAND and Kc pair value. A fully mutually authenticated protocol would only require that the root key Ki not be revealed, hence, we qualify this protocol as a limited mutually authenticated protocol.



SEQ: MS challenge using a sequence number.
Algchoices: the MS sent list of A5 algorithms supported.
Algflag: the A5 algorithm chosen by BSS.
SRES1: $f1_{Kc}$(RAND,SEQ,algchoices)
SRES2: $f2_{Kc}$(SEQ, algflag)
SK: session key.

Fig 7: A limited mutual authentication and session key agreement protocol.

The network challenges the MS with a RAND value as usual (Fig 7) in the authentication request message; the RAND value has the key Kc associated with it via the A8 algorithm. Instead of simply responding with a SRES in the authentication response message, the MS sends its own challenge SEQ and another MAC value SRES1 which is based on challenges from both sides, the MS and the network. Also SRES1's computation involves algchoices, the list of algorithms supported by the MS; the algorithms supported by the MS has been either sent to the network previously or will be sent to the network before the cipher mode command is issued to start encryption. Also we have assumed that the algorithms supported by the MS do not change between the time SRES1 is received and when cipher mode command is issued.

The SEQ is a counter value kept at the MS and is incremented every time the SRES1 is sent by the MS. The MS does not synchronize the SEQ value with the network nor does the network keep track of the current SEQ value in the handset. SEQ is a local value stored in non-volatile memory at the handset. We assume that the SEQ value is large enough (e.g. 32 bit) that it will never wrap around during the useful life of the handset. Upon receiving the authentication response, the BSS may already possess the SRES and Kc values or it may forward the SRES to the MSC/VLR for verification and later receive the Kc value. The BSS will use the Kc value to verify SRES1. If its not successful then the session is aborted or else the BSS can send the cipher mode command. Using algchoices in calculation protects the negotiation part of the protocol. As was described by [Sempel] not protecting negotiation leads to attacks.

The cipher mode command from the BSS includes additional parameters algflag and SRES2. The algflag as defined in [Brookson] is the A5 algorithm chosen by the BSS. SRES2 is also a MAC computed over the SEQ and algflag. Upon receiving the cipher mode command, the MS also calculates SRES2 and verifies before proceeding with the cipher mode complete message. At this point both the MS and BSS calculate the session key SK using a pseudorandom function (PRF) keyed with Kc and its inputs are the RAND, SEQ, algchoices, and algflag. Keyed SHA-1 functions (e.g. HMAC) have been used as PRF, for example, in the past. The f1 and f2 functions need only be a MAC whereas the f3 function needs to be a PRF. However, then its prudent not to key both the MAC and the PRF with the same key. The key Kc can be expanded to other keys via a PRG, but a simpler solution might be to use the PRF for all three functions and use a 'Type' parameter to distinguish the three functions. So $f1_{Kc}() = PRF_{Kc}(1,\ldots)$, $f2_{Kc}() = PRF_{Kc}(2,\ldots)$, and $f3_{Kc}() = PRF_{Kc}(3,\ldots)$.

This is a mutually authenticated session key agreement protocol even though its based on a counter based challenge on the MS side. Typically, using counter based protocols require synchronization by both sides or else they are susceptible to attacks due to the predictable nature of the sequence number. However, in this particular case where the initiator (network) always presents a random challenge and the SEQ is used by the responder, the protocol is not susceptible to such attacks. Since a network impersonator would not be able to create SRES2 without the key Kc, this is a mutual authentication protocol.

The limited mutual authentication and session key agreement protocol protects against weak encryption MITM attacks because even if weak encryption is used and a specific session key, SK, is revealed it would not matter for the next session because the new session key will be dependent on challenges from both sides which cannot be controlled and hence the new session key will be independent. The key Kc is never revealed hence it is of no value to a network impersonator to

simply repeat the RAND value because the attacker cannot create SRES2 response. Nor can he use previously seen SRES2 responses because the SEQ value the MS chooses will always be a previously unused SEQ value thus the current expected SRES2 is fully independent of previously seen SRES2. Since the algflag is part of the MAC calculation an attacker cannot tamper with the cipher mode command message to use a different algorithm or no encryption without detection as in [Brookson]. Thus it will protect against the original [BBK] attack and against the two steps MITM attack.

The session replay MITM attack is also protected against because for each session a new session key SK is created. Thus the BLOCK values collected for one session would be of no use because for a new session a new session key would be created and hence all the BLOCK values would be independent of the previously collected values. As mentioned in [Brookson] we can also use unused bits in authentication request/response and cipher mode command messages to include the additional parameters that we want to transfer as shown in Fig 7. The complexity of this protocol is similar to [Brookson] and the impact on the network is similar. Typically a mutually interlocking protocol uses a fixed root key to answer challenges and derive a session key; the difference here is that instead of using a fixed key directly, this protocol uses a changing RAND value which has an associated key value Kc.

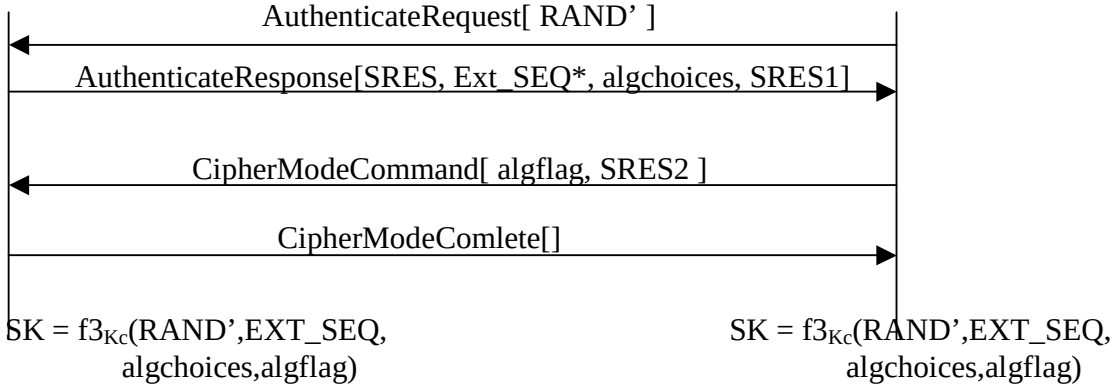
Remarks

1. **SIM moving:** We have assumed above that the SIM is not modified, hence, the SEQ number is kept in non-volatile memory in the handset. This causes a problem when the SIM is moved from one handset to another. It may be the case that the SEQ number in the new handset is lower than the number in the old handset and hence the new handset will repeat SEQ values already seen by the world. The attacker can exploit this by repeating the RAND values used with previously seen SEQ value which will cause the same session key to be created as a past session and hence the same pseudo random BLOCK stream. This means the protocol is no longer secure. A solution is to use EXT_SEQ = IMEI | SEQ instead of SEQ which is simply the IMEI concatenated with the SEQ. Now even if the SIM moves to a different handset we can guarantee that the EXT_SEQ would never repeat since the IMEI is unique to a handset. The IMEI is a 56 bit number, but it can be compressed into a 47 bit number since only 77 trillion values are possible. We assume this compressed representation is used to save bits.
2. **Anonymity:** An issue with using a counter is that it can reveal identity. Independent of using IMEI, a counter value can be tracked and can help identify the handset. We can borrow a trick from the AKA protocol to mask the EXT_SEQ value and create EXT_SEQ* = EXT_SEQ $\oplus$ $f4_{Kc}$(RAND). Thus an eavesdropper would not be able to track the counter value. At the other side, the network is able to recover EXT_SEQ. The identity is not protected against active attackers, but that is also the case with TMSI since the network can ask the MS to reveal the IMSI.
3. **Reusing Triplets:** The network may reuse a triplet in particular if its unable to communicate to receive fresh triplets. In that case the assumptions needed to keep the protocol secure are no longer met. For example, if an attacker discovered the session key for a previous session and the network repeats the RAND value then the attacker can make a call by repeating the EXT_SEQ* value used for the same session key. We can easily secure the protocol even if the network repeats the RAND value by having the network create another optional random challenge $RAND_N$. Everywhere in the above protocol (i.e. in calculating SRES1, SRES2, SK) RAND should be replaced by

RAND'=RAND|[$RAND_N$]. That is an optional random challenge may be concatenated to the RAND value. Now the protocol is secure even if the same RAND value is used multiple times. The additional random challenge introduces variability.

4. **Increasing key strength:** Instead of using Kc as the key in the cryptographic functions to derive the session key and to create responses to challenges, we can use a larger key Kc'=Kc|SRES. This could give us a larger key with up to 96 bits of entropy. All the Kc in the cryptographic functions would be replaced by Kc'. The SRES cannot be sent over the air by the MS otherwise the size of the secret in Kc' would not be increased. There is an issue here if the SRES is not available at the BSS and the verification happens at the MSC/VLR. In that case using a strengthened key would require changes to MSC/VLR also which we do not want. If so then its better not to use the strengthened key and explicitly send SRES in the authentication response message as shown in Figure 7 which does not require changes to MSC/VLR. We only mention this possibility but we do not show this in the fuller protocol described below.
5. **Protecting CKSN**. The ciphering sequence number may need to be protected, if the same key is reused for the new service request. CKSN can be used in the calculation of SRES2. However, for this limited mutually authenticated key agreement protocol to be effective, the MS needs it to be invoked after every new service request. Otherwise, an attacker who has compromised the current (weak) session key will always have the handset to continue using the current session key even when many new service requests (e.g. call origination) are made; the attacker can do a two steps MITM attack. It may be that the network decides to use the same session key, but the MS should be given the opportunity to challenge the network to prove this by responding with SRES2 that involves the CKSN in its computation.

We describe the more complete protocol with the above enhancement and options below in Figure 8. A mutually authenticated key agreement protocol is necessary for GSM security but is not sufficient. Other protocols and parts of the GSM system have to be secured also. We hinted at some of them previously, for example, an anti-replay mechanism to make sure that counter inputs (or cryptosync) to encryption and message authentication algorithms are not repeated. Also messages to change security modes should be appropriately secured.



RAND': RAND|[$RAND_N$]
EXT_SEQ: IMEI|SEQ
EXT_SEQ*: EXT_SEQ $\oplus$ $f4_{Kc}$(RAND')
Algchoices: the MS sent list of A5 algorithms supported.
Algflag: the A5 algorithm chosen by BSS.
SRES1: $f1_{Kc}$(RAND',EXT_SEQ,algchoices)
SRES2: f2 (EXT_SEQ, algflag, CKSN)

Fig 8: A limited mutual authentication and session key agreement protocol (more complete form).

5.2 Location independent MITM attack

There is another weak encryption attack that no protocol is safe from unless the MS is prohibited from using weak encryption algorithm or the user is involved in the decision to use a weak encryption algorithm. The attack works by having two attackers work in tandem. The first attacker is near the MS whereas the other attacker is in a geographically distant region that uses weak encryption. Instead of waiting for the MS to roam in to a region which uses weak encryption, the local attacker will relay responses from the MS to the remote attacker who will answer its network challenges.

The remote attacker pretends to the remote network that the MS has just visited the network. This causes the remote visited network to contact the AuC and get triplets. The remote visited network will then send a RAND challenge which the remote attacker relays to the local attacker who in turn sends the challenge to the MS. Any responses from the MS are sent to the remote attacker and through him to the remote visited network. It is as if the MS is currently visiting the remote network and is thus using a weak encryption. The local attacker will break the weak encryption and recover the session key. This will allow the attacker to do a MITM attack.

Note that this attack is not possible to protect against as long as the MS is open to use the weak algorithm and some visited network can use the weak algorithm with the MS. Even a limited mutual authentication protocol or a full mutual authenticated protocol is incapable of protecting against this attack. The one way to protect against this attack is to forbid the MS from using a weak encryption algorithm unless some user action is performed. The action could be the user pre-requesting to allow his MS to work in a region where weak encryption is being used. Or when a weak encryption is being requested by the visited network, the MS displays the request to the user and expects a response from the user whether to proceed with the weak encryption or to abandon the call. To allow better judgement by the user, the service operator and location of the visited network could be displayed to the user where a change of the encryption algorithm is occurring, but then these values should be “authenticated”.

With this provision the limited mutual authentication and session agreement protocol will protect against all the MITM attacks discussed in this paper. Note that the method of [Brookson] would not work with even this restriction because if at any time the MS actually allowed the use of a weak encryption algorithm the key Kc can be discovered and later on the attacker can do a MITM attack even with a strong algorithm and appropriately create the MAC in the cipher mode command message using the Kc key. The reason that limited mutual authentication and key session protocol is effective is that even if a key was revealed once, its of no use in the future when other independent session keys will be created and used.

6. Conclusion

We described an attack that allows an attacker to eavesdrop on conservations made by GSM phones without breaking the GSM encryption algorithm. Previous attacks exploited the handset's openness to commands to never turn on encryption or to use weak or breakable encryption algorithms. We assume that the handset meets more stringent requirements mandating that encryption is turned on all the time and that only unbroken encryption algorithms (e.g. A5/3) are used. Yet we were able to show how a man-in-the-middle type (MITM) attacker can eavesdrop on GSM handsets by exploiting weaknesses in the GSM protocol without breaking any algorithm.

We then wanted our solutions to this attack to also work against other weak encryption MITM attacks like the attacks recently described in [BBK]. Similar to the stream replay attack, we describe a two steps MITM attack, different from the [BBK] attack, when we have weak encryption. Unfortunately, the proposed solutions against the [BBK] attack are not effective against the two steps MITM attack. Finally, we build on the promising approach of [Brookson] to have the network cryptographically authenticate the cipher mode command. Our solution protects against all the MITM attacks described here, including the stream replay MITM attack and the two steps MITM attack, by augmenting the existing GSM challenge/response protocol to become a limited mutual authentication and session key agreement protocol.

Acknowledgements

I would like to thank Alain Ohana, Sudeep Palat and Mike Marcovici for helpful discussions.

References

[Brookson] Charles Brookson, "Authentication: A mechanism for preventing man-in-the-middle attacks," 3GPP TSG SA WG3 Security – S3-040036, Feb 9 2004.

[BBK] Elad Barkan, Eli Biham and Nathan Keller, "Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication," Crypto 2003.

[Mitchell] Chris Mitchell, "The security of the GSM air interface protocol," Royal Holloway technical report RHUL-MA-2001-3, 2001.

[Patel] Sarvar Patel, "Weaknesses of North American wireless authentication protocol," IEEE Personal Comm., vol 4, june 1997.

[Sempel] James Sempel, communication to S3.