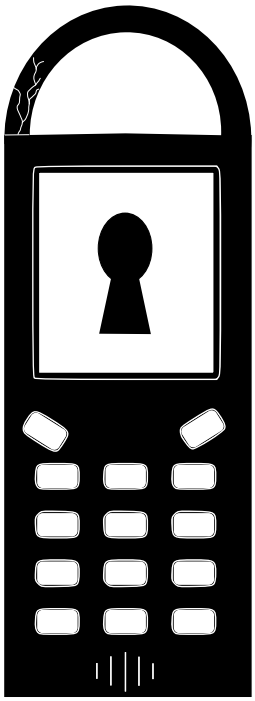
Radboud University Nijmegen

Master's Thesis Computer Science

Catching and Understanding GSM-Signals



Fabian van den Broek

Thesis Number: 628

March 22, 2010

Supervisors:

Prof. dr. Bart Jacobs

Dr. ir. Erik Poll (reader)

Abstract

4.1 billion people around the world depend on GSM for at least a part of their day to day communication. Besides communication, more and more additional services - like payment functionality - are being deployed on top of GSM. It has been over 20 years since GSM was designed, and in that time several security problems have been found, both in the protocols and in the - originally secret - cryptography. However practical exploits of these weaknesses are complicated because of all the signal processing involved and have not been seen much outside of their use by law enforcement agencies.

This could change due to recently developed open-source equipment and software that can capture and digitize signals from the GSM frequencies. This might make practical attacks against GSM much simpler.

This thesis discusses the current state-of-affairs in vulnerabilities of the GSM air-interface, using open-source signal processing.

To this end some currently available open-source hardware and software which can be used for signal capturing and some offshoot projects which specifically target GSM are described. Most importantly these include the Universal Software Radio Peripheral (USRP) together with the GnuRadio implementation for signal capturing and the AirProbe and OpenBTS project for handling GSM signals. An in depth view on the functionality of the air-interface of GSM and its security measures is presented and the feasibility of several attacks on the GSM air-interface using the open-source tools is discussed.

Acknowledgments

I would like to take this opportunity to thank some people for their advice and support. Firstly my supervisor Bart Jacobs, for all the helpful suggestions, for suggesting this topic to me in the first place, but most importantly for your motivating enthusiasm. Secondly, but not less enthusiastic, Erik Poll, for helping me structure and improve the final chapter.

There where many others who have helped me in some form or other during my work on this thesis. Whether by helpful advice, enlightening and interesting discussions or by proof reading, this work benefited in some way. In an attempt to name you all I thank Arthur, Bas, Freek, Gerhard, Michiel, Peter, Roel and Ronny and my apologies to those I may have forgotten here.

Finally, there where of course those who did not directly contribute to this thesis, but who supported me in various ways during the writing of my thesis and the duration of my study. Most notably these include my friends, my parents, and especially my girlfriend Klaartje — my thanks for all their love and support.

My sincerest gratitude to you all.

Contents

Traces

Introduction

GSM was developed in the late 1980s and deployed in most Western countries in the early 1990s. Since then GSM has seen an enormous rise both in its coverage and in the number of subscribers.

GSM is perhaps the most successful technology of the last twenty years. A survey by the International Telecommunication Union (ITU) showed that by the end of 2008 around 1.5 billion people in the world (some 23%) use the Internet. But there were around 4.1 billion people in the world (over 60%) who had a mobile subscription, while over 90% of the worlds population lived in a region that at least has access to GSM [1]. According to numbers by the GSM Association (GSMA) from 2005 the number of cell phones in circulation outnumbered that of personal computers and television sets combined [2]. The African continent has seen the fastest growth in the number of subscribers, with an increase of nearly 500% in eight years, while the number of subscriptions in Europe exceeds the population by 11%. These are staggering numbers showing a tremendous spread of GSM technology.

Internet, its protocols and equipment, have seen a lot of scrutiny over the years. A lot of people more or less understand the TCP/IP stack and tests against different equipment and configurations happen all the time. In the mean time GSM has not received the same level of scrutiny, apart from academic interest in the encryption. Its protocols are public, but mostly unknown and equipment is not tested by its users.

There are several reasons for this lack of interest in GSM security. The specifications are mostly public [3], but consist of around 2000 documents, each between a couple and several hundreds pages long. Several books exists that describe the GSM protocols, but they are pricey and often over-simplified or even incomplete. GSM research is also a lot more expensive than Internet research. For the latter all you pretty much need is a network card. But in the case of GSM, equipment capable of demodulating the traffic signals with enough precision was ,until recently, very expensive and always proprietary. In short, researching GSM requires much effort and was very costly.

However recently Software Defined Radio (SDR) solutions have appeared in signal processing, that allow a lot of the traditionally hardware operations to be handled by software, opening up this hardware oriented field to more software oriented experts. Especially the emergence of GNU Radio and the Universal Software Radio Peripheral (USRP) has made precise signal processing available to a much larger audience. It is a relatively cheap and fully open-source solution, backed by a large and very diverse community. This SDR solution can handle the modulations and frequency ranges needed for GSM. The security of GSM is paramount for the privacy of 4.1 billion subscribers and its intrinsic security on the air-interface - the connection between mobile phone and cell tower - may have just turned to an intrinsic weakness by the arrival of SDR.

The fact that GSM has weaknesses is nothing new. The fact that GSM does not use mutual authentication - a mobile phone authenticates itself to the cell tower, but not vice versa - was quickly

seen as a problem [4]. Also GSM specifically does not use point to point encryption between callers. It only encrypts the messages while on the air interface. This allows law enforcers to tap conversations in the core of the GSM network.

During the development of GSM there was a fierce debate between NATO signals agencies on whether GSM should use a strong encryption algorithm. In particular Germany, sharing a large border with the Soviet empire, was a strong proponent for using strong encryption. France and most other NATO countries were opposed [5]. Both factions proposed a design for the cipher, with the French design finally becoming the A5/1 cipher used in GSM.

Once the, originally proprietary, cipher was reverse engineered several weaknesses where found [6, 7, 8]. The first implementation of the encryption was also shown to be deliberately weakened by setting the ten least significant bits of the key to zero [9]. Because of export restrictions on the A5/1 cipher an even weaker variant, A5/2, was created to be exported to less trusted nations.

Police typically uses so called IMSI-catchers, which can request the IMSI - a number identifying the SIM card inside a mobile phone - of all phones in the vicinity. Using an IMSI-catcher the police can recover the IMSIs of the phones of suspects, who typically use anonymous pre-paid SIMs. With those IMSIs the police can then try to get a warrant for tapping those phones. Besides tapping phone conversations in the GSM back-end, government agencies can also eavesdrop on the air-interface directly. Commercial equipment for eavesdropping on GSM and the aforementioned IMSI catchers are being sold restrictively to government officials [10].

So we know that a lot of these attacks are technically possible, but the equipment needed is very pricey and sold restrictively. But again the surfacing of SDR technology might bring these attacks within reach of nearly everybody.

Meanwhile more and more services are being deployed on top of the GSM network, increasing the incentive for criminals to attack GSM. In several countries you can pay for services or products via text messaging. Several Internet banking applications use the mobile phone as an external (out-of-band) channel to verify transactions. The Dutch ING bank stated only last January that they will start to use the mobile phone to send users their password reminders, even though they already use it for transaction verification [11]. Where previously making un-billed calls was the major economic attraction in attacking GSM, increasingly real money can be made.

This thesis started as a research into the state-of-affairs in GSM vulnerabilities on its air-interface, using the new SDR capabilities. This air-interface is called the Um-interface in official GSM terminology. However, pretty soon the GSM standard itself proved a complicated matter that took a lot of time to understand. So a large part of this thesis now focuses on describing the specifications and verifying them where possible by captured signals. Hopefully this thesis can prove itself useful as a starting point for people looking into GSM.

This thesis starts with a chapter on the open-source equipment and software available today that might be used to attack GSM. Throughout this document actual GSM traces are shown to illustrate the theory or specifications with practical real-world examples. Chapter 1 therefore pays special attention to the equipment used to make these traces, during this research. Then a broad overview of the entire GSM system is sketched out in chapter 2. Allthough this thesis focuses on the Um-interface, it is unavoidable to know the basics of the setup of a GSM network. Chapters 3 to 6 then focus on the Um-interface at different levels. Chapter 3 again gives a more broad overview of the air-interface, while chapters 4 to 6 describe the three most fundamental layers of the Um-protocol in detail. Chapter 7 focuses specifically on the encryption. It looks at the authentication in GSM and at the workings of

the A5/1 encryption algorithm in detail. The security of GSM is assessed in chapter 8. First it looks at the security goals of GSM and then some attacks against GSM are described and their feasibility using the equipment described in chapter 1 is discussed. Finally some conclusions are drawn in chapter 8.2.5.

If this is your first introduction into GSM, then perhaps a word of warning is appropriate on the sheer number of acronyms that will be introduced. There is actually a specification document which consists of 10 pages filled with a listing of the acronyms used in the GSM specifications [12]. In this thesis most often the acronyms are used to refer to objects or properties, instead of their full names. This was done simply because all other material referring to GSM, whether it is the specification itself, or books and articles on the matter, do the same. So if you ever pick up another source of information on GSM you might already be used to some of the acronyms. In order to alleviate these pains, a listing of all acronyms used in this thesis can be found at the back of this document (appendix A).

Chapter 1

Hardware and software

This chapter shows some of the hardware and software that can be used today, to try and sniff GSM traffic. Because the USRP hardware and the GNU Radio software were developed as an SDR implementation, we will first look at the principles of SDR. Then the USRP hardware is discussed and we will look at some of the available open-source software. Finally the exact hardware and software that where used during this research are mentioned.

1.1 Software Defined Radio (SDR)

Traditionally radio's were a hardware matter. They are often very cheap, but also very rigid. A radio created for specific transmit and receive frequencies and modulation schemes will never divert from these, unless its hardware is modified. Please note that the word radio here is used as a generic transceiver using electro-magnetic waves for transmissions, not specifically as the device known for the reception of programmed broadcasts made by radio stations.

The main idea behind Software Defined Radio (SDR), is to create very versatile transceivers by moving a lot of the, traditionally, hardware functions into the software domain. However a radio can never be purely software, because you need a way to capture and create the radio waves. Analog radio waves can be converted to digital samples using a Analog to Digital Converter (ADC) and vice versa using a Digital to Analog Converter (DAC). The ideal SDR scheme involves an antenna connected to a computer via an ADC for receiving and via a DAC for transmitting. All the processing on the signals, like (de)modulation, are then done in software, but the actual transceiving is done in the hardware subsystem. This makes for a much more adaptable system, able to for instance receive GSM signals as well as GPS and also television broadcasts by only changing something in the software.

This ideal scheme however is not practically viable, because in practice ADCs and DACs are not fast enough to process a large portion of the spectrum and antennas are designed for specific frequency bands. This has led to the creation of more extended hardware subsystems for SDRs. Typically such a hardware subsystem consists of a wide band receiver that shifts a frequency band to a standard intermediate frequency, which can be sampled by ADCs and the resulting digital signal can be sent to a computer. Often other common equipment like amplifiers and band-pass filters are also a part of the hardware subsystem.

One of the most versatile and widely used SDR systems is GNU Radio, mostly combined with a USRP as the hardware subsystem.

1.2 USRP

The Universal Software Radio Peripheral (USRP) is designed as a general purpose hardware subsystem for software defined radio. It is an open-hardware device developed by Matt Ettus and which can be ordered through his company Ettus Research [13].

There are currently two types: the USRP1 and the USRP2. Both consist of a motherboard which contains a Field Programmable Gate Array (FPGA), Programmable Gain Amplifier (PGA), ADC(s), DAC(s) and a communication port to connect it to the computer. Daughterboards can be plugged into the USRP motherboard according to the specific frequency bands needed. These daughterboards can be hooked up to appropriate antenna's. On the receiving path (RX), a daughterboard captures the required frequency range and sends it through the PGA, possibly amplifying the signal, towards the ADC. The resulting digital signal is passed on to the FPGA, where it is transformed into 16 bit I and Q samples. These are complex samples, with the real part (Q) describing the cosine of the signal, and the imaginary part describing the sine of the signal plus 90 degrees. One sample is thus 32 bit long and can be sent to the host computer through the communication port, for further processing.

The FPGA and the host CPU both do some processing on the signal, and though the exact division of labor can be changed, standard the high speed general purpose processing, like down and up conversion, decimation, and interpolation are performed in the FPGA, while waveform-specific processing, such as modulation and demodulation, are performed at the host CPU.

The USRPs have a 64 MHz crystal oscillator internal clock. Most GSM implementations use a 13 MHz symbol clock with a much better accuracy. Of course the 64 MHz samples can be re-sampled to (a multiple of) 13 MHz, and all of the software discussed in this chapter perform these calculations on sampling rates that are not dividable by thirteen. However this brings an extra computing complexity. Also the USRPs oscillators are much less accurate and can show quite some drift, resulting in bad reception. An external clock can be attached to the USRPs. Choosing a multiple of 13 MHz external clock solves these issues.

1.2.1 USRP1

The USRP1 has four daughterboard slots. Two for receiving and two for transmitting. It contains four 12 bit ADCs (two for every receive board), that have a sampling rate of 64 Msamples per second. Nyquist's theorem states that you need a sampling rate of at least 2 times the frequency you wish to capture in order to be able to reconstruct the signal [14]. Therefore the USRP1 can capture a bandwidth of 32 MHz at once, for every receive daughterboard. There are also four 14 bit DACs with a sampling rate of 128Msamples per second. Making the maximum transmit frequency band 64 MHz wide.

At the heart of the USRP1 lies its FPGA, an Altera Cyclone EP1C12. This FPGA can be programmed using the Verilog hardware description language. The compiler for this can be downloaded for free from the Altera website [15]. The communications port is a USB 2.0 chip, with a practical maximum data throughput of 32 Mbyte/s. Since the analog signals are processed into 16 bit I and Q channels, this limits the data throughput to 8 Msamples per second.

1.2.2 USRP2

The USRP2 contains only two daughterboard slots. One transmit and one receive side. It does contain faster and higher resolution ADCs and DACs. Two 14 bit 100 Msamples per second ADCs and two 16-bits 400 Msamples per second DACs. So it can capture a bandwidth of 50 MHz and transmit

on a bandwidth of 200 MHz wide. With respect to the USRP1, the USRP2 also contains a much faster FPGA (Xilinx Spartan 3-2000) and an ethernet port instead of the USB connection. The gigabit ethernet port allows for over 3 times higher bandwidth throughput.

The USRP2 is much more costly however; double the price of an USRP1.

1.2.3 USRP Daughterboards

Different frequencies require different antennas and sometimes different signal processing, like amplifiers or filtering, to receive or transmit correctly. So in order to keep the USRPs as general as possible the actual receiving and transmissions are handled by daughterboards that can be plugged into the USRP motherboard. These daughterboards are specifically meant for certain frequency bands. Currently there are thirteen daughterboards available, of which three are interesting in relation to GSM signals:

- DBSRX, a 800 MHz to 2.4 GHz Receiver.
- RFX900, 800-1000MHz Transceiver, 200+mW output.
- RFX1800, 1.5-2.1 GHz Transceiver, 100+mW output.

The most used GSM frequencies are GSM900 (890.2-959.8 MHz) and GSM1800 (1710.2-1879.8 MHz) in Europe, and GSM850 (824.0-894.0 MHz) and GSM1900 (1850.0-1990.0 MHz) in America and Canada. The DBSRX board covers all these frequencies, but is only a receiver board. In order to actively transmit a RFX board is needed. Keep in mind that most countries require a license to transmit on these frequencies.

1.3 Hardware used for this thesis

With regard to GSM the USRPs have some restrictions. Table 3.1 shows the different GSM frequency bands. GSM900 has a bandwidth of 70MHz, two times 25MHz for the up and downlink and a 20 MHz unused guard band between them. GSM1800 uses 75MHz for its uplink and 75MHz for its downlink. This means that the USRP1 can only capture the up or the downlink of a GSM900 signal at one time. However because the USRP1 can be equipped with two receive boards an entire conversation on GSM900 can still be captured, although sampling two frequency bands at once will run into limitations at the USB data rates. The GSM1800 uses too wide a bandwidth for capturing with any USRP at this time. The USRP2's better specifications of both the communication port and the FPGA, make the USRP2 the better tool for capturing GSM.

For this thesis we used a USRP1 together with a DBSRX daughterboard. At the time this research started the USRP2 was not yet available, so there was no active choice there. The DBSRX board was chosen because it can be used for nearly all GSM frequencies and is a receive-only board, since transmissions on GSM frequencies are illegal in the Netherlands without a license.

No external clock was used, and for most reception tasks, this proved hardly a problem. Some packages may suddenly arrive garbled, which is possibly due to the inaccuracy of the USRP clock, but this does not occur often enough to be problematic. Literature suggests that these issues become much more troublesome when you are turning your USRP into a GSM cell tower [16].

1.4 GNU Radio

GNU Radio [17] is a free software toolkit licensed under the GPL for implementing software-defined radios. It was started by Eric Blossom. It works with several different types of RF hardware, like soundcards, but it is mostly used in combination with an USRP. Basically GNU Radio is a library containing lots of standard signal processing functions. These functions, usually called blocks, are often divided into three categories: source blocks (USRP drivers, but also file readers and tone generators), sink blocks (USRP drivers, graphical sinks like an oscilloscope and soundcard drivers) and processing blocks (like filters, FFT and (de)modulations). These blocks can be attached to each other to make a graph.

All the low level blocks are written in C++, while higher level blocks and GNU Radio graphs are made in Python. These two languages are glued together using SWIG. This means that, for performance reasons, the actual computations are done in C++, while on a higher level a more user friendly language is used to define a software radio. This also abstracts from implementation details for the processing functions. If I want to see a Fast Fourier Transform (FFT) of a certain frequency on screen, I only need to instantiate a source block (for instance a USRP source, with a frequency) and a graphical FFT sink and link these two together. I do not need to know or understand how the actual FFT is computed, in order to use it. And there are hundreds of implemented blocks inside GNU Radio.

GNU Radio, out-of-the-box, does not offer much in terms of GSM sniffing capabilities, although it can be used to locate the beacon frequencies of GSM masts [18]. However GNU Radio can be used by other software packages, like AirProbe in the next section, to perform the low level functions of GSM sniffing, like reception and demodulation.

1.5 AirProbe

Airprobe [19] is an open-source project trying to built an air-interface analysis tool for the GSM (and possible later 3G) mobile phone standard. This project came forth out of the GSM-sniffer project [20].

When you currently clone the git repository, you will get nearly ten projects. Some of these serve as libraries for the other projects (e.g. gsmstack), some of these have more or less the same function (e.g. gsm-receiver and T-void) and some of these don't even compile anymore (e.g. T-void).

The most interesting part of AirProbe is the gsm-receiver project. It is, at this moment, the best working capture tool for GSM. It comes with two simple shell scripts that call all the necessary functions for saving the signals on a frequency to a file and for interpreting the signals in this file. Calling

```
capture.sh <freq> [duration==10] [decim==112] [gain==52]
```

with a frequency will capture the signals on that frequency to a file. The duration, decimation and gain are optional arguments with default values. A file will be created called `capture_<freq>_<decim>.cfile`, containing the captured IQ samples. These can then be interpreted by calling:

```
go.sh <file.cfile> [decim==112]
```

The file name has to be provided, but the decimation is again optional, though you should use the same decimation value that was used during capturing. The go.sh script runs a python file that defines

a software radio, which does all the processing needed (see section 3.3) to get the information bits out of the samples. This results in a series of hex values that represent the information as sent by the GSM network. The go.sh script uses a UNIX pipe method to have these hex-codes interpreted by gsmdecode - one of the other projects in the AirProbe repository. You could also try to convert these hex codes to a .pcap file, which can be read by the wireshark program [21].

Currently the gsm-receiver project will only decode the downlink (GSM network to mobile phone). Standard it will look at the first time slot of a frequency (time slots in GSM will be discussed in section 3.2), though this can be changed in the python code. At this moment it can handle several of the control channels in GSM (control channels will be discussed in section 4.2), and speech channels. However due to encryption (chapter 7) and frequency hopping (section 3.1.2) this will not yet work in most real world situations. For a discussion on this see chapter 8.

1.6 Gammu

Another way to get traces of GSM communication is by using Gammu [22]. Gammu is an open-source project which can manage various functions on cellular phones. You will also need a Nokia DCT3 phone. Nokia used a simple remote logging facility for debugging their DCT3 firmwares remotely, but apparently forgot to remove this when going into production. So you can enable it back using Gammu.

You will also need to download a special file `nhm5_587.txt` which helps decoding trace types. You will also need a cable to connect the specific DCT3 phone to a computer. Once Gammu is installed on this computer [22] and the mobile phone is connected to the computer, you can run Gammu using the following command:

```
gammu --nokiadebug nhm5_587.txt v20-25,v18-19
```

The software will then interface with the phone and create a .xml debug log of lots of packages sent to and from the mobile phone. The .xml file that can be interpreted either by wireshark [21] or AirProbe's gsmdecode [19].

The Gammu + Nokia phone method has a much better receive quality than the USRP + AirProbe, after all the mobile phone is specifically made to receive these signals. Also when using gammu you can even see some message that where encrypted and you have no problems tuning to the correct frequencies. However you can only see the messages to or from the phone hooked up on the computer. You cannot see any message for other phones, nor is it possible to change the phone's behavior. So it is a great tool to learn more about signaling on the GSM interface, but it is not versatile enough to use in any real world attack.

1.7 OpenBTS / OpenBSC

In chapter 2, the architecture of a GSM network will be discussed in detail. For now though it is useful to know that a Base Transceiver Station (BTS) is a GSM cell tower, and a Base Station Controller (BSC) is a control center for several BTSs. Both of these systems have an open-source implementation: OpenBTS [23] and OpenBSC [24] respectively.

This does not mean that both systems work together. In fact they are different approaches to the same problem. OpenBTS, founded by David Burgess, offers a BTS implementation using the USRP and turning it into a BTS. Some of the logic normally present in a BSC is placed inside OpenBTS.

OpenBSC, developed by Harald Welte, on the other hand implements most of the BSC functions and currently includes support for two BTS types (nanoBTS and the Siemens BS-11 microBTS). It does not support an OpenBTS driven USRP.

Both systems give you the opportunity to run your own GSM network, though this requires a license in most countries. This can be very useful for testing purposes, but another great use is their implementation of the GSM protocol stack. These open source systems offer an extra way to understand the GSM protocol through their implementation, and these implementations can be used to create GSM analyzers.

1.8 A5/1 Cracking project

In the Western world most GSM traffic is encrypted using an algorithm known as A5/1, which is detailed in chapter 7. In August of 2009 a project was started to use a generic time-memory-trade-off to break A5/1, by pre-computing a large rainbow table.

The pre-computation is done distributed on the Internet. Volunteers can download the table-generating code from the project's website [25], and run it on their own computers. The code is specifically written for graphics cards (NVIDIA and ATI currently), because of their parallel computing power. Tables that are finished need to be shared, e.g. via bit torrent. When someone has access to enough tables he should have a chance of around fifty percent to find the encryption key for an encrypted conversation. More on this attack will be detailed in chapter 8.

1.9 Software used for this thesis

For this thesis all the software described in this chapter was evaluated. Because we did not have the hardware to actually transmit, as explained in section 1.3, the OpenBSC and OpenBTS projects were not actively used. However since they are both open-source projects, they both aided in understanding GSM, through their source code.

The A5/1 cracking software was not actively used, because in its current state it does not yet serve any practical purpose.

The Gammu software, together with an Nokia 3210 was used extensively, and many example traces shown throughout this thesis result from it. The AirProbe software, together with the USRP were also used extensively. Although the reception and decoding qualities are not always great, it is a very promising tool for the future.

1.10 Traces throughout this thesis

Throughout this thesis traces of actual GSM traffic will be shown that where captured as research for this thesis. Only self-caught traces where used in this thesis. These traces are either made with the USRP + AirProbe combination, or with the Nokia 3210 + Gammu combination. Both combinations are explained in this chapter.

These traces deliver information in a not very human friendly way. Therefore we use either wireshark or gsmdecode to examine the traces. The AirProbe section discusses both these tools. Figure

1.1 shows what a trace examined with wireshark looks like.

Wireshark is the more convenient tool for analyzing these traces, because it orders all the information and coveniently shows extra information like the current frame number and frequency. The results of the interpreting with Wireshark (from version 1.2.6 on) are also better than those of Gsmdecode. However throughout this thesis traces will be shown decoded by Gsmdecode. This was done because Gsmdecode immediately displays all information in plain text, which displays a lot better onto paper. An example of a Gsmdecode trace is given in trace 1.1. A list of all traces throughout this document can be found in the front.

Traces are included throughout this thesis at those places where they can illustrate what is being said in the main text. However, especially the first couple of traces will show a lot of GSM content that is explained in later chapters. Specifically chapters 4 to 6 will provide a lot of help in understanding these traces.

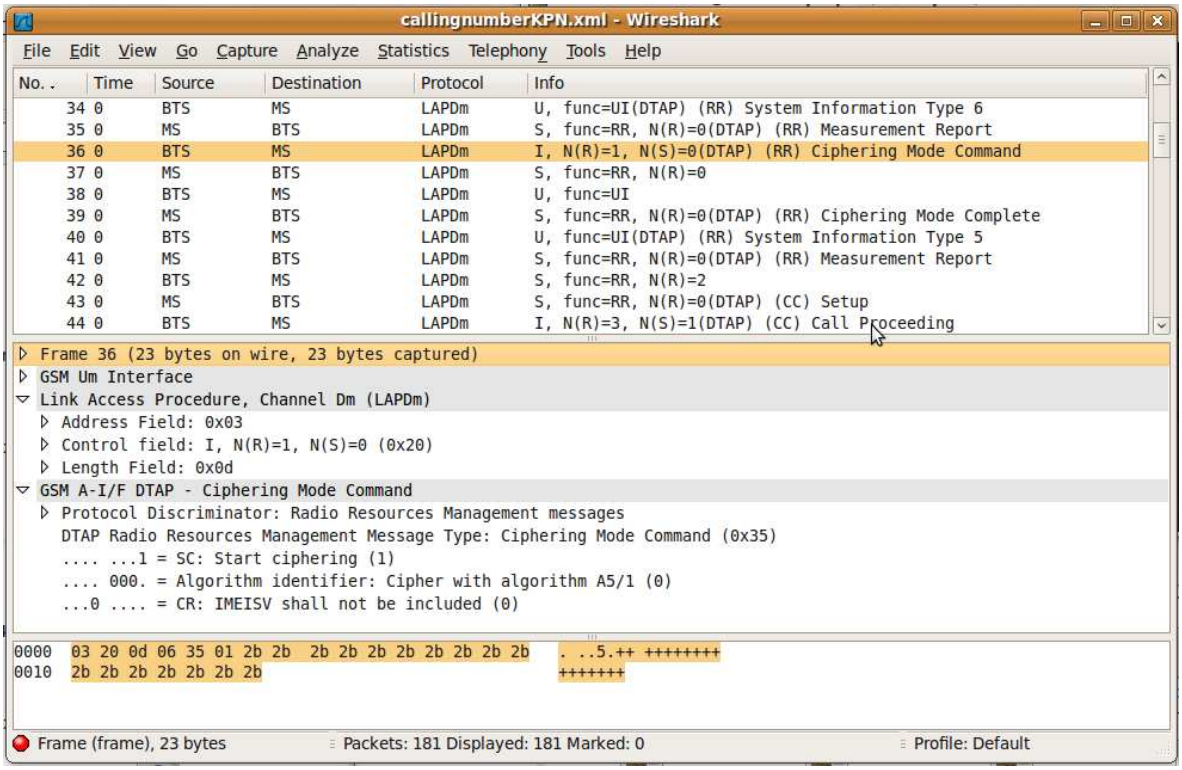


Figure 1.1: A GSM trace examined with WireShark

Trace 1.1: RR System Info 3

```
HEX l2_data_out_Bbis:462 Format Bbis DATA
000: 49 06 1b 32 22 02 f4 80  - 11 7f d8 04 28 15 65 04
001: a9 00 00 1c 13 2b 2b
     0: 49 010010-- Pseudo Length: 18
     1: 06 0------- Direction: From originating site
     1: 06 -000---- 0 TransactionID
     1: 06 ----0110 Radio Resouce Management
     2: 1b 00011011 RRsystemInfo3C
     3: 32 12834    [0x3222] Cell identity
     5: 02 204      Mobile Country Code (Netherlands)
     6: f4 08f      Mobile Network Code (KPN Telecom B.V.)
     8: 11 4479     [0x117f] Local Area Code
    10: d8 1------- Spare bit (should be 0)
    10: d8 -1------ MSs in cell shall apply IMSI attach/detach procedure
    10: d8 --011--- Number of blocks: 3
    10: d8 -----000 1 basic phys chan for CCCH, not combined with SDCCHs
    11: 04 00000--- spare bits (should be 0)
    11: 04 -----100 6 multi frames period for paging request
    12: 28 00101000 T3212 TimeOut value: 40
    13: 15 0------- spare bit (should be 0)
    13: 15 -0------ Power control indicator is not set
    13: 15 --01---- MSs shall use uplink DTX
    13: 15 ----0101 Radio Link Timeout: 24
    14: 65 011----- Cell Reselect Hyst. :  6 db RXLEV
    14: 65 ---xxxxx Max Tx power level: 5
    15: 04 0------- No additional cells in SysInfo 7-8
    15: 04 -0------ New establishm cause: not supported
    15: 04 --xxxxxx RXLEV Access Min permitted = -110 + 4dB
    16: a9 10------ Max. of retransmiss : 4
    16: a9 --1010-- slots to spread TX : 14
    16: a9 ------0- The cell is barred  : no
    16: a9 -------1 Cell reestabl.i.cell: not allowed
    17: 00 -----0-- Emergency call EC 10: allowed
    17: 00 00000--- Acc ctrl cl 11-15: 0 = permitted, 1 = forbidden
    17: 00 ------00 Acc ctrl cl  8- 9: 0 = permitted, 1 = forbidden
    17: 00 -------0 Ordinary subscribers (8)
    17: 00 ------0- Ordinary subscribers (9)
    17: 00 -----0-- Emergency call (10): Everyone
    17: 00 ----0--- Operator Specific (11)
    17: 00 ---0---- Security service (12)
    17: 00 --0----- Public service (13)
    17: 00 -0------ Emergency service (14)
    17: 00 0------- Network Operator (15)
    18: 00 00000000 Acc ctrl cl  0- 7: 0 = permitted, 1 = forbidden
    18: 00 00000000 Ordinary subscribers (0-7)
    19: 1c YYYYYYYY REST OCTETS (2)
```

This shows a System Info 3C message, made with Gammu. This specific message is broadcast by a cell tower to show its identity. In this case it is a "KPN Telecom" cell tower located in the Netherlands.

Traces are displayed with first the entire message displayed as hex values. Most signaling messages consist out of 23 octets; 46 hex values. Then the message is interpreted. First the number of the current octet is shown. Then the value of that octet in two hex values, followed by the bits from this string that are being interpreted. Finally the interpretation of those bits is placed at the end in human readable form.

Chapter 2

Network Architecture

This chapter will give an overview of all the entities in the GSM network. Then the different protocols that are used between these entities are briefly highlighted and finally some of the most important functions within GSM are shown via global scenario's showing the interactions between the entities in a GSM network.

A single GSM network is often referred to as a Public Land Mobile Network (PLMN). This is the network operated by a single provider in a single region. In most countries each provider maintains a single PLMN, but in certain large countries, like the USA, several PLMNs can be maintained by a single provider. A PLMN manages all traffic between mobile phones and all traffic between mobile phones and the other land networks. These land networks can either be the Public Switched Telephone Network (PSTN), an ISDN network or the Internet. Figure 2.1 shows an overview of all the entities in a GSM network. We will now look at each in detail.

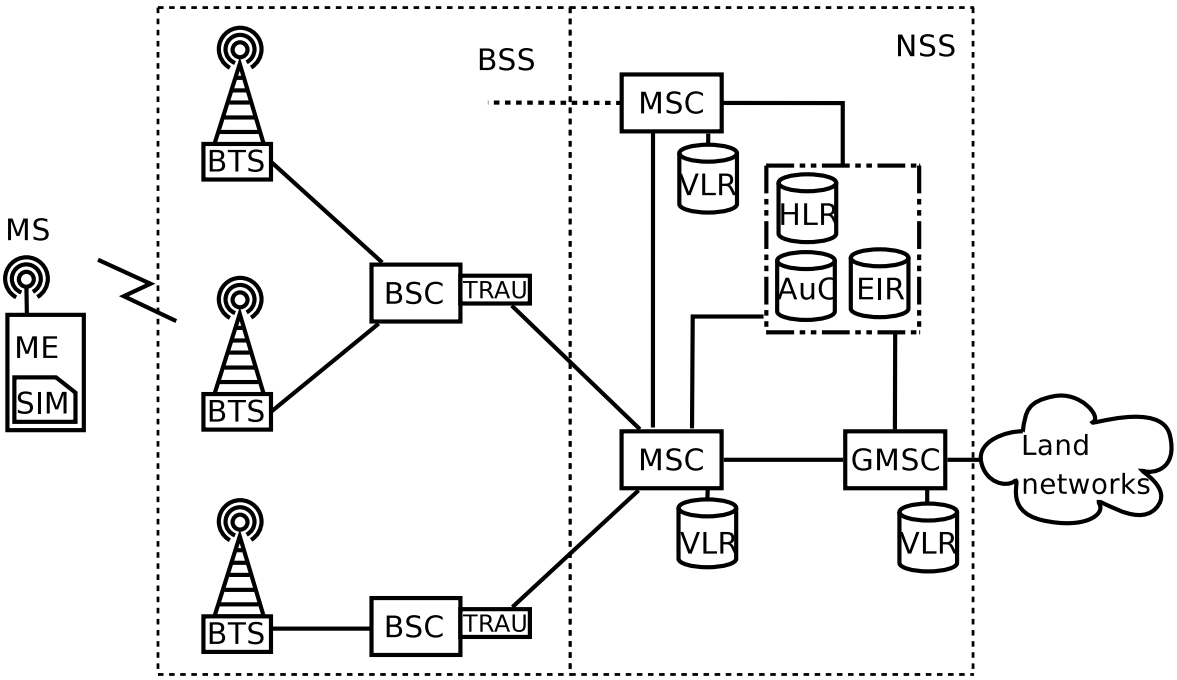


Figure 2.1: Network layout of a generic Public Land Mobile Network (PLMN).

2.1 Mobile Station (MS)

The Mobile Station (MS) is the subscribers module most people are familiar with. It consists of both some Mobile Equipment (ME) and a Subscriber Identity Module (SIM). Both are needed for the Mobile Station (MS) to function in the GSM network. Hence MS = ME + SIM.

2.1.1 Mobile Equipment (ME)

The Mobile Equipment (ME) is simply the GSM phone people use to make and receive calls in a cellular network. It is basically a transmitter-receiver unit that is independent from network providers[1].

Every ME contains an International Mobile Equipment Identity (IMEI) number, consisting of 15 digits which uniquely identify this particular ME. An ME can be asked for its IMEI by typing in the string '*♯06♯' on the mobile phone.

2.1.2 Subscriber Identity Module (SIM)

The Subscriber Identity Module (SIM) is provided to a subscriber as a smart card; a SIM card. It contains a users identity in a GSM network and is dependent on a network provider. It is uniquely identified by its International Mobile Subscriber Identity (IMSI) number. A SIM contains the following information:

- The International Mobile Subscriber Identity (IMSI), consisting of 15 digits or less with a 3 digit Mobile Country Code (MCC), a 2 digit Mobile Network Code (MNC) and an up to 10 digit Mobile Subscriber Identification Number (MSIN).
- The Temporary Mobile Subscriber Identity (TMSI), temporary identifier passed on to the MS by the network to hide the IMSI. The TMSI is only valid within a certain region, and the MS can always request a new one from the network.
- The secret key K_i
- The current encryption key, also called session key; K_c
- The Ciphering Key Sequence Number (CKSN), a 3 bit number send by the network, acting as an identifier of the current session key
- The encoding algorithms A3 and A8
- The current Location Area Identity (LAI), consisting of a 3 digit Mobile Country Code (MCC), a 2 digit Mobile Network Code (MNC) and an up to 5 digit Location Area Code (LAC). The Location Area Identity (LAI) is transmitted by the network regularly and stored in the SIM. It identifies a certain area in the PLMN.
- List of preferred Public Land Mobile Network (PLMN)s
- List of forbidden PLMNs

[1] In most countries MEs can be bought from providers in some form of packaged deal. A ME can have provider specific firmware, and can be modified to only accept the providers SIM (SIM-locking). Also some providers produce their own MEs. However their networks still accept other MEs and their MEs can, with a small modification (SIM-unlocking), function in another network.

- List of beacon frequencies of the home PLMN
- The Personal Identification Number (PIN) used to gain access to the SIMs functionality
- The Pin Unblocking Code (PUK) used to reset the Personal Identification Number (PIN) and unlock the SIM, when the wrong PIN number has been entered three times.
- Storage of Short Message Service (SMS), telephone numbers, etc.

Note that the IMSI is not equal to the Mobile Subscriber ISDN Number (MSISDN), the phone number belonging to this sim. Both numbers are created independently and linked to each other in the HLR (section 2.3.3). However the IMSI is the identifier in the GSM system for an MS and it belongs uniquely to a single SIM. while a new MSISDN can be linked to the IMSI. The beacon frequencies refer to a set of "main" frequencies on which the cell towers of the provider advertise themselves to the MS.

2.2 Base Station Subsystem (BSS)

The Base Station Subsystem (BSS) is the part of the PLMN that manages the communication between the MSs and the Network Switching Subsystem (NSS).

2.2.1 Base Transceiver Station (BTS)

A Base Transceiver Station (BTS) is another name for a cell tower, or more accurately a name for the transceivers on a cell tower. One BTS defines a single cell. In general it is simply a relay station that broadcasts to the MS the packages it receives from its BSC (next section) and vice versa. Because the BTS is the link between the air interface and the land interface, it is responsible for all the channel encoding/decoding, ciphering, Slow Frequency Hopping (SFH), Gaussian Minimum Shift Keying (GMSK) and burst formatting.

'Land interface' is a somewhat misleading term to describe the link between a BTS and the rest of the network. Although most BTSs are connected via a land line, some use a microwave directional radio link for this connection. Whether through a land line or via a directional radio link, the signal uses the same Abis interface (section 2.4).

Cellular systems, like GSM, gain a lot of extra capacity when compared with traditional (single transmitter) systems, because cellular systems divide the geographic area up into cells. These cells allow for frequency re-use. Because of interference, two neighboring cells can never use the same frequencies, so they must be geographically divided. This is schematically shown in figure 2.2, which also shows that cells can differ in size, for instance to accommodate a densely populated area. Figure 2.2 is a simplification of the practical situation in which one GSM tower often contains three BTSs. Each BTS handling 120 $^{\circ}$ around the tower. This does not change anything about the general working of a BTS, it just defines a smaller cell.

The maximum reach of a BTS is 35km. Though a transmitted signal might travel beyond this distance, the delays that occur in the transmissions become to large to still function within GSM.

A BTS can hold between one and sixteen transceivers, depending on geography and user demand in the area. Eight transceivers for the uplink frequencies (MS to BTS) and eight for the downlink frequencies (BTS to MS). Each transceiver can handle eighth different channels which MSs can use. Because some of these channels are used for sending control information, a BTS can never handle

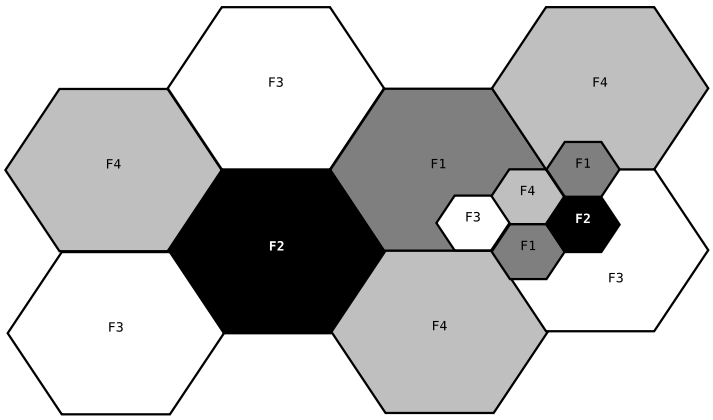


Figure 2.2: A schematic division of frequencies in a geographical area, with four different frequencies (F1 - F4).

more than about 60 ($8 \times 8-$ some control channels) conversations with mobile phones in its area. However many more phones can be connected to a cell, while not actively using it.

A BTS is identified by its Cell Global Identification (CGI). A fourteen digit number uniquely identifying this cell. It is composed of a Location Area Identity (LAI) and a Cell Identity (CI).
There exists an open-source implementation of a BTS named OpenBTS [23] which was also discussed in section 1.7. In most countries a license is required to operate a BTS.

2.2.2 Base Station Controller (BSC)

The Base Station Controller (BSC) is the center of intelligence in the Base Station Subsystem (BSS). A single BSC controls one or more BTSs and typically serves a population of around $100,000$ to $250,000$ people [28]. It manages the radio channel setup and handovers from a MS between BTSs that are connected to this BSC. It also watches the status of the BSS hardware.

There exists an open-source implementation of a BSC named OpenBSC [24], which was discussed in section 1.7. It is not specifically build to work with OpenBTS; actually both projects attempt to be usable as an entire BSS.

The BSC side of the network can also contain a Transcode Rate and Adaption Unit (TRAU). The air-interface uses a voice encoding, regular pulse excited-long term prediction (RPE-LPC), which manages a data rate of 13 kbit/s. However the voice encoding used on the land interface, Pulse Code Modulation (PCM), reaches 64 kbit/s. The transcoding needed between these signals is performed in the Transcode Rate and Adaption Unit (TRAU). Although this transcoding is defined as a responsibility of the BSC, it is often performed by a distinct subsystem. Some vendors have implemented the TRAU at the Mobile Switching Centre (MSC) side of the network, thereby compressing the signals earlier on and saving bandwidth between the BSC and the MSC (see 2.3.1).

2.3 Network Switching Subsystem (NSS)

The Network Switching Subsystem (NSS) is the central part of any PLMN. A single NSS controls multiple BSSs. The NSS houses all subscriber services. It authenticates the SIM for access to the

network and for setting up calls, and it finds the MS when a call is being made to it or routes a call through to the Public Switched Telephone Network (PSTN) or a neighboring NSS.

2.3.1 Mobile Switching Center (MSC)

The Mobile Switching Centre (MSC) is the main component of any NSS. It is a modified version of a standard ISDN-switching system and it performs several functions:

- Manage the location (which BSC/BTS) of all MSs in its service area.
- Set up and release end-to-end connection.
- Controls handovers between BSCs.
- Manages call data and sends this to the billing system.
- Collects traffic statistics for performance monitoring

Every BSS is connected to a single MSC. All the BSSs connected to a MSC comprise its service area.

2.3.2 Gateway Mobile Switching Center (GMSC)

All communication between different PLMNs or between the PLMN and the PSTN is routed via the Gateway Mobile Switching Centre (GMSC). When a MS attempts to log-on to a different network than his home network, the GMSC of the visited network asks the GMSC of the home network to authenticate the MS. When a call request arrives at a MSC it will check whether the destined MS is within this MSCs service area. If this is not the case the request is forwarded to the Gateway Mobile Switching Centre (GMSC) which will then route the call to the correct MSC, to the PSTN, or to the responsible GMSC of another provider.

The GMSC is a special form of a MSC. The practical implementation of a GMSC can vary. Some networks contain a single, high performance MSC as dedicated GMSC, but there are also PLMNs where every MSC can function as the GMSC. In the latter case the term GMSC is only valid in the context of a single call or sign-on, because its role can be carried out by a different MSC each time.

2.3.3 Home Location Register (HLR)

The Home Location Register (HLR) contains the subscriber's information for call control and location determination. Logically there is only one Home Location Register (HLR) per provider per GSM network, although this can be implemented as a distributed database.

The HLR stores the following information per IMSI:

- The subscribers MSISDN.
- The current VLR (see section 2.3.4) serving the subscriber, used to locate the MS.
- GSM services that the subscriber is allowed to access.
- Possible call divert settings.

Both the IMSI and the MSISDN fields have primary database keys over them. The HLR is where the standard phone numbers (MSISDN) are linked to their IMSIs. When a call is placed to a MSISDN the GMSC requests the corresponding IMSI and the current MSC serving it, from the HLR. The HLR receives location updates for every IMSI in its database from the current serving VLR.

2.3.4 Visitor Location Register (VLR)

Each MSC maintains one Visitor Location Register (VLR) which stores all the SIMs that are active within the MSC's service area. When a MS is successfully logged on to an allowed PLMN, the home network's HLR is queried for some subscriber information which is stored in a record in the VLR. This happens after the VLR informs the HLR of the presence of the IMSI in its VLR area. These messages between VLR and HLR are even exchanged when the subscriber is within his own home network. The VLR can be used by the MSC to route incoming calls to the correct BSS.

After some period of inactivity or when a MS has traveled to a different service area, the record for an IMSI is removed from the VLR. In the latter case the removal is commanded by the HLR. For every IMSI the VLR stores the following information.

- The subscribers current Temporary Mobile Subscriber Identity (TMSI), which is allocated by the VLR.
- The subscribers MSISDN.
- The subscribers current LAI, or a different VLR is maintained for every LAI.
- The subscribers current Cell Identity (CI). The LAI and the CI together form the Cell Global Identification (CGI) and define a unique cell in every PLMN.
- GSM services that the subscriber is allowed to access.
- The HLR address of the subscriber.
- Up to five authentication triplets, received from the Authentication Centre (AuC).

2.3.5 Authentication Centre

The AuC contains the information needed to authenticate a SIM and to set up an encrypted connection with a MS. The AuC is often co-located with, and in most implementations even integrated in, the HLR.

In the AuC the following information is stored per IMSI:

- The secret key K_i, the same as on the SIM.
- The encoding algorithms A3 and A8, the same as on the SIM.

Despite its name, the AuC does not directly authenticate a SIM. Instead it computes a random challenge and the corresponding reply and encryption key, K_c, using the A3 and A8 algorithms. These three values form so called triplets. Authentication triplets are discussed in more detail in sections 2.5.1 and 7.1. These triplets are then stored at the VLR and from there supplied to the MSC where a MS tries to authenticate itself. The real authentication takes place at the MSC, which sends the random challenge to the MS (via the BSC and BTS) and verifies the MSs response. The encryption key is sent on to the BTS, because only the ME - BTS link is encrypted with it.

The implementations for the A3 and A8 algorithms are only stored and invoked on the SIM and in the AuC. Both are under control of the provider, so the specification leaves some room here for every

provider to implement their own algorithms. However it is commonly assumed that most providers followed the advised implementations of A3 and A8 [4].

Authentication of MSs will be discussed extensively in sections 2.5.1 and 7.1.

2.3.6 Equipment Identity Register (EIR)

The Equipment Identification Register (EIR) is often co-located with the HLR. It contains lists of International Mobile Equipment Identity (IMEI)s [29]. When a MS is connected to a network, the network can always give it the identify command. In response to this command the MS will transmit its IMSI, identifying the SIM, and IMEI, identifying the physical phone (ME). The IMSI ends up at the HLR, but the IMEI is checked against the stored identifiers in the EIR.

Originally the EIR was meant to be used to blacklist all stolen phones, making it possible to track them or render them useless. However it is clear that several countries make no use of the EIR's function. Like in the Netherlands, where there is no administration of stolen IMEIs.

This built-in IMEI security also has several problems; it hinges on the difficulty to change a phone's IMEI, but in most mobile phone models today this proves rather simple. There also is no specified method to unblock a IMEI once it is registered in the EIR.

2.4 Interfaces

Within the GSM network several different interfaces are defined. These are all shown in figure 2.3.
The main interfaces, those interfaces that connect a MS to the land interfaces (Um, Abis, A and E),

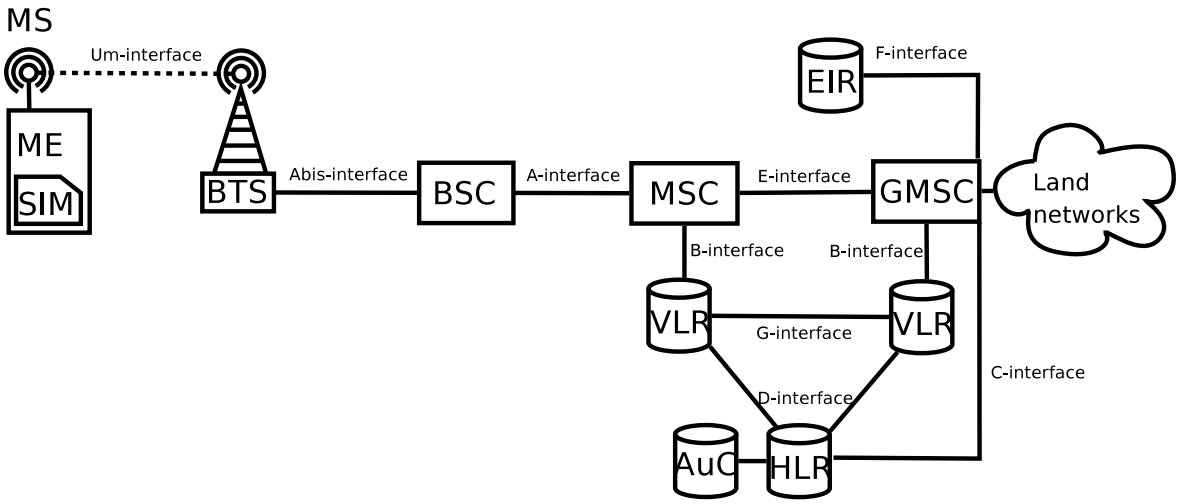


Figure 2.3: The defined interfaces within a GSM network.

are all split in traffic channels that contain the speech information and control channels on which the meta data is transmitted.

Of all the interfaces here the Um interface, or air-interface, is the main concern of this thesis, because it is this interface that can be sniffed using the USRP and GNURadio/AirProbe (chapter 1).

Geographically speaking everyone has access to this interface, making it a likely target. The Um interface will be discussed in detail in chapters 3 to 6.

The Abis interface connects the base stations (BTSs) to the base station controllers (BSCs). This interface is defined as an LAPD (standard ISDN) interface and largely coincides with the data link layer of the Um interface (see chapter 5). The Abis interface also allows control of the radio equipment and radio frequency allocations in the BTS.

The A interface connects the BSS with a NSS and the E interface is the main interface inside a NSS. All the control channels on the A and E interface are part of the Signaling System #7 (SS7), a collection of telephony signaling protocols defined by the International Telecommunication Union (ITU) [30]. The TRAU (section 2.2.2) does not interfere with any of the signaling channels. It only transcodes the voice data.

The B, C, D, F and G interfaces are defined to synchronize all the different information sources within a PLMN. The ETSI has not defined an interface between the AuC and the HLR, so every provider can make their own decision here. Most providers have the AuC located at the HLR site and often these two databases are integrated.

2.5 Scenarios

We will now discuss how some of the major functions of GSM are handled by the network. These scenario's are only discussed broadly, to show the way in which the network entities interact. This should lead to a top level understanding of the GSM network, but these examples should not be seen as the actual message interactions happening. Several of these examples will be discussed in further detail in the following chapters, where we look in detail at the communication on the Um protocol.

All of the examples discussed here will be shown in message sequence diagrams. In these diagrams a convention is used for the two arrow types, dashed and solid, when they cross other entities. A dotted arrow denotes a message that is transmitted directly from the sending to the receiving entity, without passing through the entities that the arrow crosses. When a solid arrow is used then this denotes that the message passes through all the different entities it crosses. Those messages will have to be transcoded somewhat by the passed entities, because every entity in the GSM network is connected with a different interface (figure 2.3), but the message contents will be almost identical. So a solid arrow from A to C passing entity B, actually denotes two arrows (A to B and B to C) with an almost identical message.

Notice that all the diagrams that will follow in this chapter do not show the BSC, either by showing the BSS as a single entity or by just completely omitting the BSC. This is because the BSC is only actively involved at a much lower level, like deciding on which frequencies to use. So in order to save space the BSC is not shown in these diagrams. Just remember that all the message that pass the space between the MSC and the BTS (or the BSS) will pass through the BSC.

2.5.1 Authentication

The authentication of a MS to the network is of course one of the most important security functions in GSM. After a successful authentication the MS has proven its identity to the network and at that point both the MS and the BTS will know a shared session key, K_c, which they could use for encrypted

communication.

Authentication is often used within the other functions of the GSM network. Whenever a MS is not yet known to a network, the full authentication described in figure 2.4 takes place. At the end of such an authentication both the MS and the network know the session key and a Ciphering Key Sequence Number (CKSN) identifying this session key. For every subsequent authentication the network can choose to either completely re-authenticate the MS, or accept the MS as already authenticated, and if encryption is needed (re-)use the encryption key that resulted from the previous full authentication. This choice for key re-use is fully up to the network, but if the MS does not know the key identified with the CKSN, then full authentication again needs to take place.

Figure 2.4 shows the successful full authentication of a MS, which is always initiated by the MSC. At some stage during a connection with a MS the MSC will decide to initiate an authentication procedure. Most often this will happen after a request for a service is made by the MS. The MS is known to the MSC either by its IMSI or its TMSI. The MSC requests the authentication triplets corresponding to the MS's IMSI from the Visitor Location Register (VLR). The VLR knows the IMSI $\leftrightarrow$ TMSI relation and is therefore capable to respond with authentication triplets on a supplied TMSI as well as to an IMSI. The VLR has a supply of authentication triplets and returns one of these to the MSC. If the VLR runs out of triplets it can then request up to five new ones from the AuC.

The AuC actually creates the authentication triplets. It can do so because it has stored the secret key K_i and the A3 and A8 algorithms per IMSI. Authentication triplets are defined as:

Authentication triplet	=	(RAND, SRES, K_c)
	where	
RAND	=	A randomly chosen number
SRES	=	A signed response computed as $A3(K_i, \text{RAND})$
K_c	=	The session key computed as $A8(K_i, \text{RAND})$

The K_i is a secret key uniquely linked to every IMSI / SIM.

So a triplet actually contains the challenge, the response and the session key, everything needed for the MSC to authenticate the MS and for the BTS to set up the encrypted channel.

The MSC sends the challenge (RAND) on to the MS. Because the same K_i, A3 and A8 are stored on the SIM, the MS can now compute SRES and K_c and then transmit SRES to the MSC. The MSC verifies the SRES it receives from the MS with the SRES from the authentication triplet. If they are equal then the session key K_c is sent to the BTS. From this moment on it is possible to start encrypting the Um-interface connecting the BTS with the MS.

If the responses received by the MSC are not equal, then the MSC will send a message to the MS telling it that authentication failed. Possibly the MSC can re-attempt authentication, or end the MS connections. What ever the case, every subsequent authentication should never re-use the same RAND.

The authentication in GSM is discussed in more detail in section 7.1.

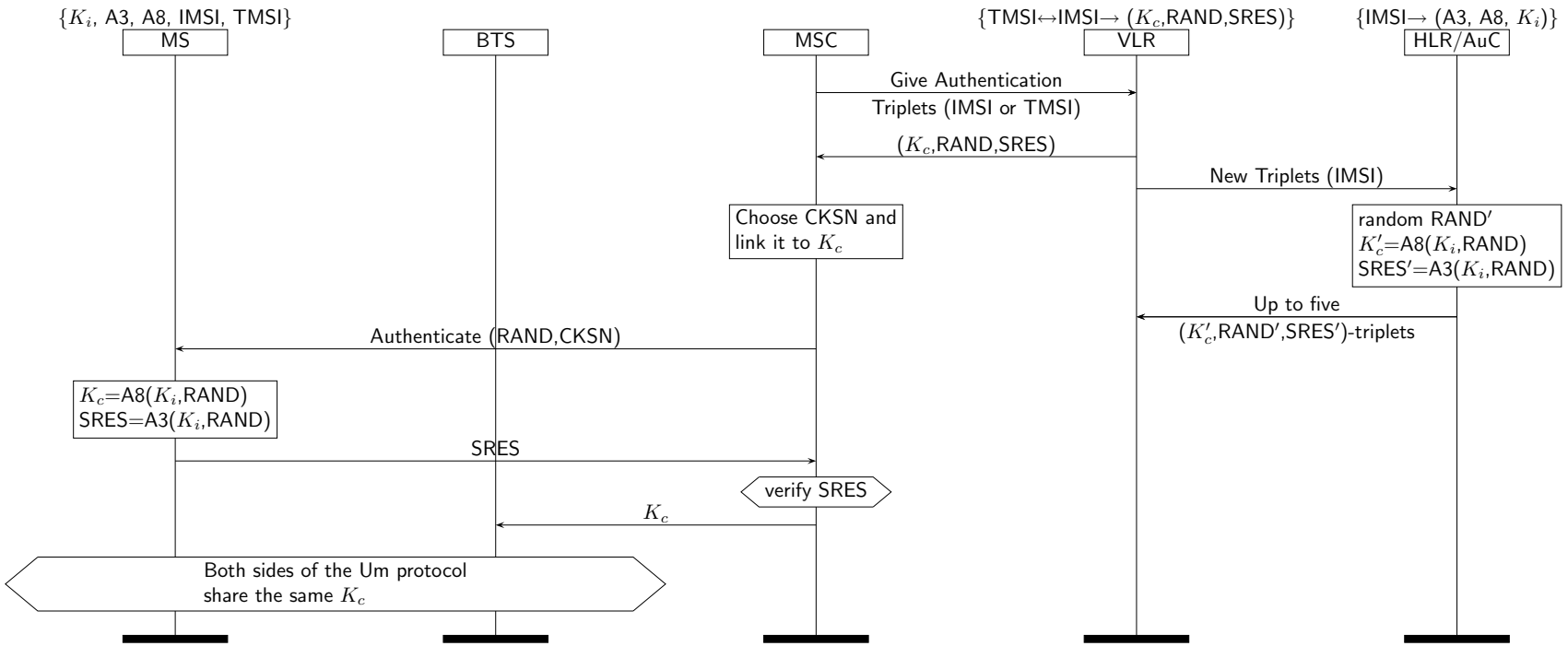


Figure 2.4: Global overview of successful authentication of an MS inside a GSM network

2.5.2 Location Updates

The network needs to know where every MS is located in order to route calls there. For this purpose the MS regularly informs the network of its location. This location is represented by the LAI, which the MS learns from its current BTS. This process is called “location update”. Location updates happen when a MS moves into the cell area covered by another LAI (figure 2.6), but location updates also happen periodically when a MS remains in the same LAI area (figure 2.5). Location updates are always initiated by the MS and always result in a new TMSI being assigned to the MS.

In both figures showing these different scenario’s the BTS, BSC and MSC are shown together as a single entity; the BSS/MSC. This saves space in the diagrams, because most of these entities do not play an important part in these scenario’s, though all communication does pass through them. The BSC however does add the CGI of the current BTS to the message, which the receiving VLR uses to update the MSs location. Please note that the ciphering that is being set-up between the MS and the BSS/MSC is actually only used between the MS and the BTS, so on the Um-interface.

Timed Location Update

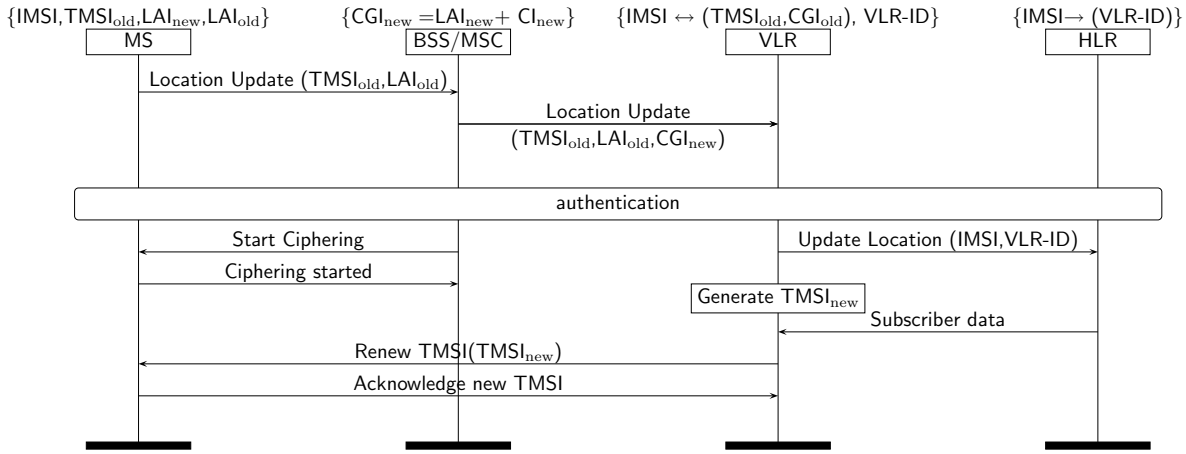


Figure 2.5: Global overview of a timed location update in the same cell area.

Figure 2.5 shows a timed location update. The MS requests to perform a location update, identifying itself with its current TMSI (TMSI$_{old}$) and its current LAI. Note that a single VLR can serve several LAIs, however when a timed location update occurs the new LAI will often be equal to the old one. The BSC then appends the CGI of the current BTS to the location update message. The CGI consists of the LAI and the CI of the current BTS.

Subsequently it is checked whether this MS is already authenticated and possibly full authentication as explained in section 2.5.1 takes place. If authentication was successful, the VLR stores the new LAI for this IMSI/TMSI and transmits its VLR-ID to the HLR together with the MS’s IMSI. In this case the VLR-ID received by the HLR will be the same as the VLR-ID the HLR already had in its records, but still this step should be preformed. The HLR responds by sending additional subscriber data to the serving VLR. This subscriber data is mostly a collection of services that this MS is entitled to use. This data was already present at the VLR in this case, but it is retransmitted between HLR and VLR nevertheless, because some of these services might have changed.

In the meantime the VLR also generates a new TMSI which is transmitted to the MS, once the HLR acknowledges the location update. If the current network provides encryption, then the new IMSI is always transmitted on an encrypted Um-interface.

Log on

If you change every occurrence of $TMSI_{old}$ in figure 2.5 by the MS's IMSI, you have the exact procedure for a log-on of a MS to a GSM network. This is often referred to as a "location registration" instead of a location update, the simple difference being the absence of an already assigned TMSI.

Roaming Location Update

A MS is always listening to all BTSs it can receive, in order to judge which one has the best reception. When another BTS gives a better reception then the current BTS, the MS will conclude that it has moved in a different cell area. It will listen to the new BTS for its LAI (LAI_{new}) and when this is different form its current LAI (LAI_{old}), the MS will initiate a location update (via the new BTS). Notice that the MS does not initiate the location update when it comes in a different CI area (the ci together with the LAI form the CGI). The CI will get updated through a timed location update. The VLR only knows the LAI of a MS for certain, at any given time, the CI might have changed.

When a MS has moved into the area serviced by a new VLR, you get the scenario detailed in figure 2.6. The major difference between the timed location update and the roaming location update is the communication between the old and new VLR. The new VLR will actually query the old VLR for the IMSI belonging to this TMSI and corresponding authentication triplets. After that the MS can be authenticated by the new MSC/VLR. The new VLR can find the old VLR through the LAI_{old}. A VLR can service several LAIs, but every LAI is serviced by exactly one VLR.

After authentication nearly the same actions are performed as with a timed location update, except that the HLR recognizes that the MS has moved to a new VLR area because the received $VLR\text{-}ID_{new}$ does not match the already stored $VLR\text{-}ID_{old}$. Inciting the HLR to command the old VLR to remove its records belonging to the MS's IMSI.

2.5.3 Call setup

Figure 2.5.3 shows all the entities involved in a call between two MSs belonging to different providers and thus to different PLMNs. There are two types of scenario's in call set-ups; the calls initiated by a MS (Mobile Originating Call (MOC)) and the calls received by an MS (Mobile Terminating Call (MTC)). Both are discussed here.

Mobile Originating Call (MOC)

In a MOC the MS naturally initiates the procedure by requesting a call. Figure 2.8 shows the message flow for an MOC. In this diagram it is assumed the MOC is directed towards another mobile phone. If the call is being made to a land line, the messages would be routed via the GMSC to the PSTN. Also note that the GMSC and MSC entity in this diagram can be the same entity, depending on the set-up of this PLMN.

After authentication and subsequent encryption of the communication, the MS supplies the number (MSISDN) it wishes to call. The first digits of a MSISDN identify the country and the provider of the callee. The HLR maintaining the MSISDN (the HLR of PLMN identified by the country and

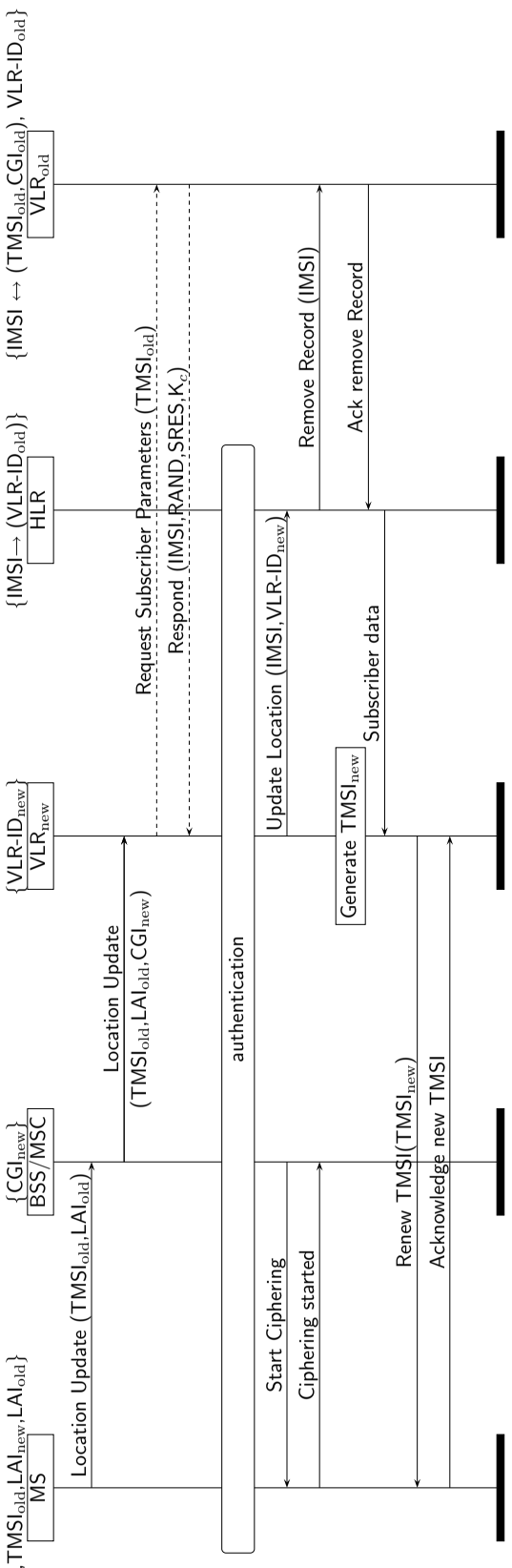


Figure 2.6: Global overview of roaming location update when a MS moves into a different VLR area.

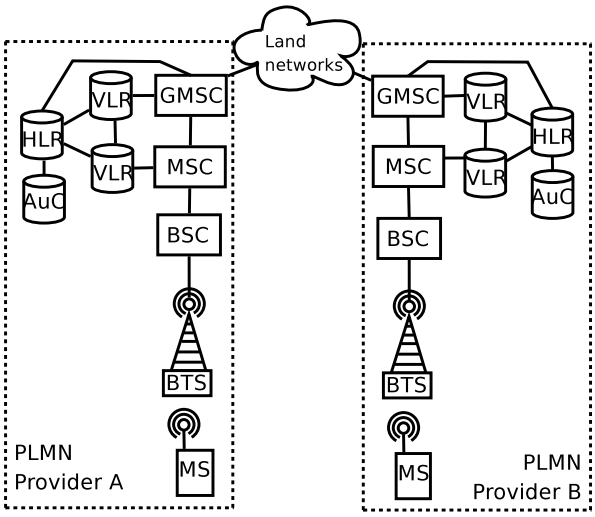


Figure 2.7: Agents in call setup between two MSs serviced by different providers

provider) is queried for the corresponding IMSI and current VLR location. With that information the GMSC can set-up a connection to the MSC serving the subscriber being called. Meanwhile a call connection is being established between the MSC and MS that initiated the call. If all goes well there will be a call connection between both mobile phones and a conversation can take place.

Mobile Terminating Call (MTC)

The diagram in figure 2.9 shows a MTC and can be seen as the follow-up of the diagram in figure 2.8.

A connection request for an IMSI arrives. The corresponding TMSI is found by the VLR and a page command is sent to the MSC (trace 2.1). The MSC commands the correct BSC to page the TMSI and in the meanwhile the MSC sets-up a call connection to the calling entity, possibly a GMSC. Note that the calling MSC can be the same as the called MSC, in which case, naturally, no connection needs to be set up between them. However all other signaling messages are still necessary, because the MSC/VLR combination does not know the link between the called number (MSISDN) and the IMSI/TMSI.

Authentication of the MS can be initiated by the MSC at this point. After encryption on the Um-interface is started the call connection between this MS and MSC is set up and the entire conversation can begin.

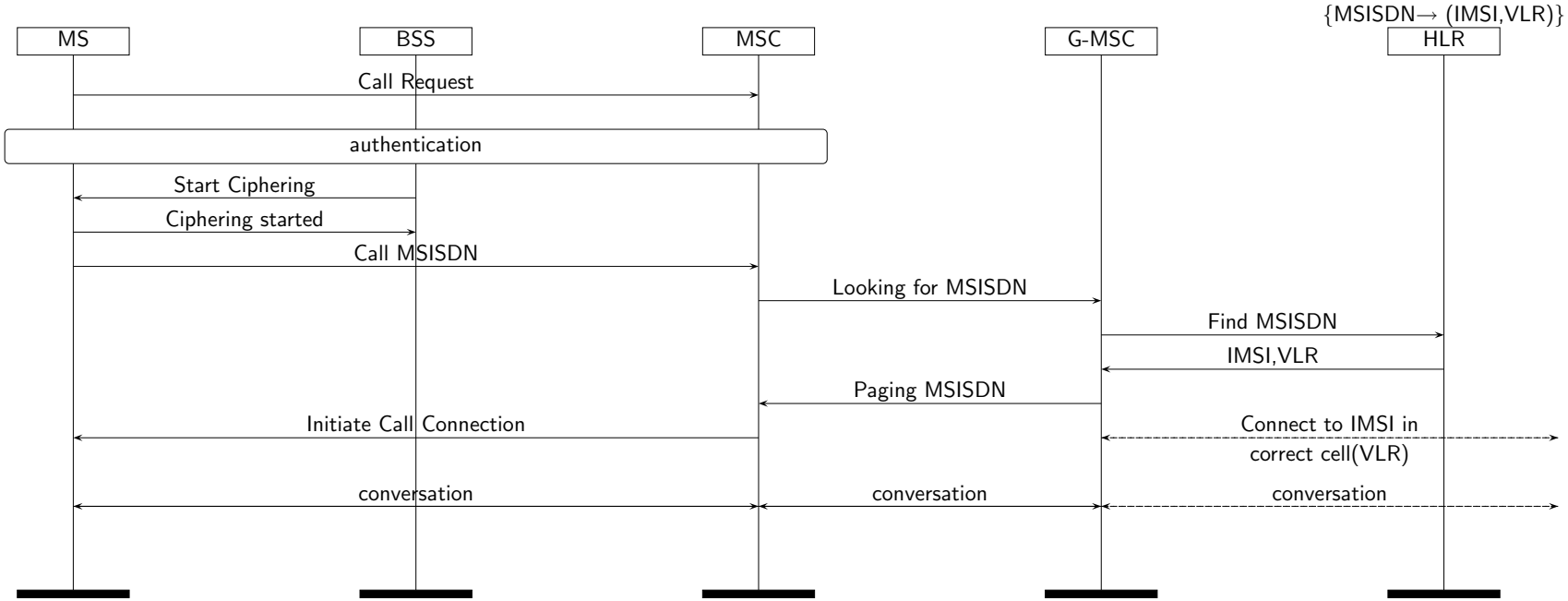


Figure 2.8: Global overview of mobile initiated call setup.

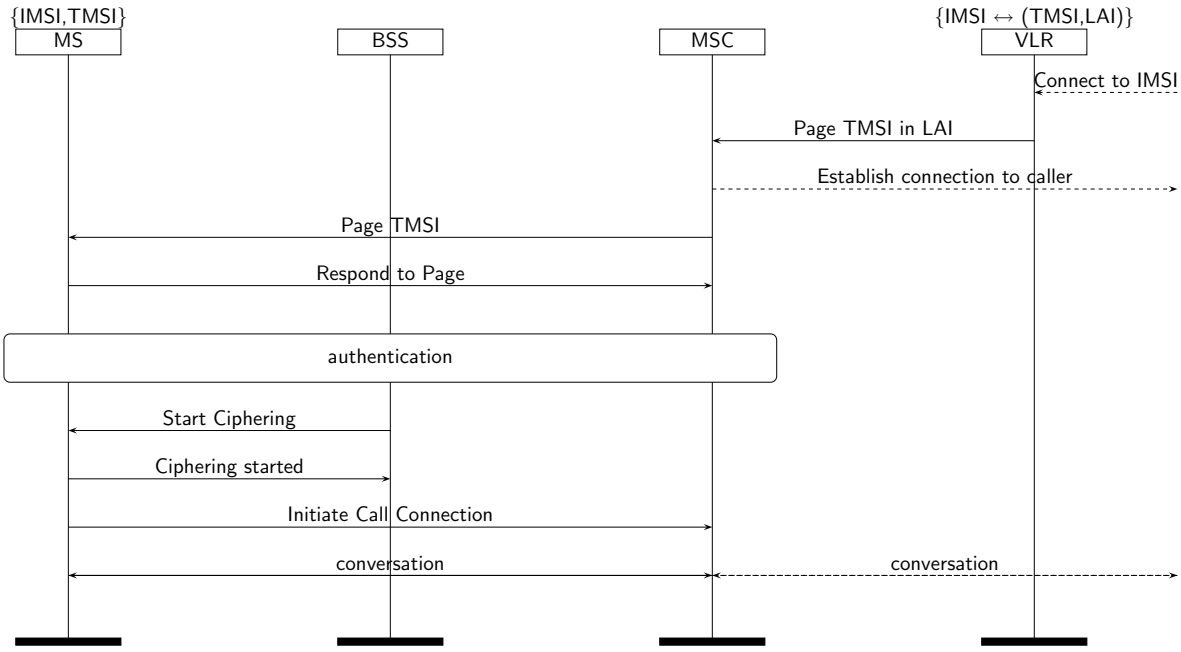


Figure 2.9: Global overview of mobile terminated call setup.

Trace 2.1: RR Paging Request

```
HEX l2_data_out_Bbis:462 Format Bbis DATA
000: 25 06 21 00 05 f4 c1 8c  - 45 ce 2b 2b 2b 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
     0: 25 001001-- Pseudo Length: 9
     1: 06 0------- Direction: From originating site
     1: 06 -000---- 0 TransactionID
     1: 06 ----0110 Radio Resouce Management
     2: 21 00100001 Paging Request Type 1
     3: 00 ------00 Page Mode: Normal paging
     5: f4 -----100 Type of identity: TMSI/P-TMSI
     6: c1 -------- ID(4/even): C18C45CE
```

A standard paging message transmitted on a special paging channel (PCH). This is a Paging Request Type 1, there are two other types, but they only differ in the number of MSs that can be paged in one message. The paging request contains the reason for the paging - here “Normal paging”, which is the standard reason for most pagings. Most importantly is the identity of the MS for which the paging is intended. In this case it is intended for the TMSI C18C45CE.

The type of channel that should be requested in a reply to this paging is encoded in the fourth octet. Gsmdecode, for some does not decode it correctly, but this particular page is for a SDCCH channel.

Chapter 3

The air-interface

The interface between a mobile phone and a base station is officially called the Um-interface. It was so named because it is a mobile equivalent to the U interface in ISDN. The Um interface is defined as a full duplex interface with a separate frequency range for the uplink (cell phone to tower) and downlink (tower to cell phone). These frequency bands are at a certain minimum distance from each other to prevent interference.

Although the Um interface is defined as duplex, most cell phones are unable to send and receive at the same time. They use a switch to toggle the antenna quickly between the transmitter and the receiver.

Usable frequencies are in short supply in our world. A lot of frequencies are already in use and in every geographical location a frequency can only be used once. In order to service the many cell phones that populate the world today, the GSM frequencies had to be used economically. For this end GSM uses both Frequency Division Multiple Access (FDMA) and Time Division Multiple Access (TDMA). The available frequency bands are divided in smaller frequency channels, FDMA, and each frequency channel is divided among users to use at a designated time-slot, TDMA.

This chapter will elaborate further on the workings of the Um interface.

3.1 On Frequencies

GSM started out with standard frequency bands, 890 - 915 MHz (uplink) and 935 - 956 MHz (downlink). This system is now called GSM-900. However the popularity of GSM caused for a frequency shortage. This first led to the definition of an Extended GSM band (E-GSM) and later to the definition of several other frequency bands (table 3.1). Of these bands the GSM-900 is defined as the preferred band and together with GSM-1800 the most commonly used in most parts of the world. Europe, the Middle East, Africa, Oceania, and the most of Asia use these bands. The United States and Canada use the GSM-850 and GSM-1900 bands. The GSM-450 and GSM-480 frequencies where once defined to make use of the radio spectrum reserved for the first cellular technologies. At the time of writing this thesis no provider seems to have a license to operate on these frequencies [31].

3.1.1 FDMA

In a GSM network many cell phones can be transmitting at the same time. In order to prevent interference the GSM bands are split in different frequency channels of 200kHz wide. These channels are called carrier frequencies or carrier channels, and can be assigned to different functions. This division

Name	Uplink (MHz)	Downlink (MHZ)	Offset (MHz)	Channel Numbers (ARFCN)
GSM-450	450.4-457.6	460.4-467.6	10	259-293
GSM-480	478.8-486.0	488.8-496.0	10	306-340
GSM-850	824.0-249.0	869-894.0	45	128-251
GSM-900	890.0-915.0	935.0-960.0	45	1-124
EGSM-900	880.0-915.0	925.0-960.0	45	975-1023, 0-124
GSM-1800	1710.0-1785.0	1805.0-1880.0	95	512-885
GSM-1900	1850.0-1910.0	1930.0-1990.0	80	512-810

Table 3.1: The GSM frequency bands

is referred to as Frequency Division Multiple Access (FDMA).

Each uplink carrier channel is linked to a single corresponding downlink carrier channel by a standard spectral difference: the offset, see table 3.1.

In order to communicate which carrier frequency will be used, the Absolute Radio Frequency Channel Number (ARFCN) is communicated. Given a frequency band and an ARFCN the carrier frequency can be computed. For instance for the GSM-900 this is:

$$\begin{aligned} F(up) &= 890.0 + 0.2 \times \text{ARFCN} \\ F(down) &= F(up) + 45 \end{aligned}$$

Where $F(up)$ calculates the uplink and $F(down)$ the downlink channel. As you can see both channels always differ their offset (45 MHz). Similar equations are defined for every GSM band. The last column in table 3.1 shows the available ARFCNs per GSM band. Notice how there are only 124 channels in the GSM-900 band. With a spectrum of 25 MHz and a channel width of 200 kHz, there is room for 125 channels. However in GSM-900 the lowest channel is used as a guard band to prevent interference from other services. In practice this did not prove to be much of a problem, so in EGSM-900 this lowest channel is again used as a carrier channel (ARFCN 0) [31].

A single cell is defined by a BTS with up to 16 transceivers (section 2.2.1). Each transceiver manages one ARFCN.

3.1.2 Frequency Hopping

Each carrier channel is influenced differently by local propagation conditions. Atmospheric noise, interference, and multipath wave propagation amongst other things can cause distortions in signals of specific, arbitrary frequencies; influencing only some of the carrier channels. To even out the unpredictable differences in signal quality between carriers, GSM uses Slow Frequency Hopping (SFH). Some of the signals between the MS and the BTS, like the speech data, can 'hop' between different carriers. The 'slow' part of GSM's frequency hopping is relative. Signals can hop to another frequency at around every 4.615 ms. This is considered slow compared to other frequency hopping algorithms. The main reason to opt for a slower variant was to cut down on the price of MEs.

A BTS does not need to use frequency hopping to correctly implement GSM, but if a BTS chooses to use frequency hopping, a receiving ME needs to be able to follow the hopping signal.

GSM defines a hopping algorithm that produces the next carrier frequency given the current Frame Number (FN) (more on frames will be explained in section 4.1), a list of frequencies to hop between - the Mobile Allocation (MA)-, an offset - the Mobile Allocation Index Offset (MAIO) for different MSs inside the same sequence - and the Hopping Sequence Number (HSN), which works as a seed for the algorithm. The algorithm makes a permutation of the MA into a sequence of a specific order based on the HSN and then uses the MAIO to shift the sequence accordingly and uses the frame number to decide the current index in this sequence [32].

Frequency hopping is initiated by the BTS which sends the MS all the required parameters (trace 3.1). Both sides then send their messages through the sequence of carriers produced by the hopping algorithm.

Frequency hopping was designed purely as a means to get a better overall quality on the signals. Though frequency hopping has recently proved to be a big problem for eavesdroppers. This problem is further discussed in chapter 8.

3.2 Time Division Multiple Access

In GSM each of the carrier frequencies is divided into eight time slots, effectively creating eight new logical channels out of one 'physical'. The time slots are labeled 0 to 7 and each is assigned to a single user. This division of frequencies in the time domain is called Time Division Multiple Access (TDMA). In TDMA each user can only be assigned a single time slot in a frequency and so up to eight users can make use of a single frequency. This leads to MEs ignoring 7 out of the eight time slots and forming a channel from the time slots they do use (see figure 3.1).

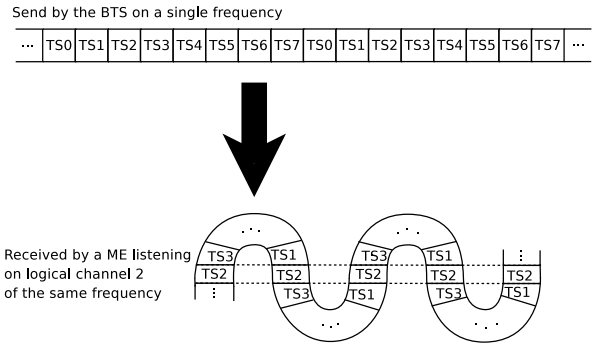


Figure 3.1: Logical channel in TDMA for time slot 2

These time slots are also called bursts. Each burst lasts approximately 576.9μs and they are the de facto unit of time in GSM. The name 'burst' is derived from the bursty nature of transmission you get because of the TDMA; when a BTS transmits only to a single user on a certain frequency (ARFCN see 3.1.1), transmissions on this frequency will occur only $\frac{1}{8}$ th of the time.

Each ARFCN is divided into eight logical channels. There are several types of channels, that can be divided into two groups: Traffic Channels (TCH), used for speech and data, and Control Channels (CCH), used for network management messages and channel maintenance tasks. These channels will be discussed in more detail in section 4.2.

Trace 3.1: Excerpt of an Immediate Assignment

```
000: 31 06 3f 00 52 f0 ab 85  - ad e0 01 01 0f 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
     0: 31 001100-- Pseudo Length: 12
     1: 06 0------- Direction: From originating site
     2: 3f 0-111111 RRimmediateAssignment
     3: 00 ---0---- This messages assigns a dedicated mode resource
     4: 52 -----010 Timeslot number: 2
     5: f0 111----- Training seq. code : 7
     5: f0 ---1---- HoppingChannel
     6: ab ........ Mobile Allocation Index Offset (MAIO) 2
     6: ab --101011 Hopping Seq. Number: 43
     7: 85 100----- Establishing Cause: Answer to paging
     8: ad xxxxxxxx T1/T2/T3
     9: e0 xxxxxxxx T1/T2/T3
    11: 01 00000001 Length of Mobile Allocation: 1
    12: 0f ----1--- Mobile Allocation ARFCN #4
    12: 0f -----1-- Mobile Allocation ARFCN #3
    12: 0f ------1- Mobile Allocation ARFCN #2
    12: 0f -------1 Mobile Allocation ARFCN #1
```

This is a part of an immediate assignment message. Some parts that are not interesting at this point have been removed. The full trace can be found in section 6.1

This message is used to assign a channel to the MS. It gives the MS a time-slot, a list of ARFCN numbers, the hopping sequence number and the MAIO. The T1/T2/T3 of octets 8 and 9 encode the frame number, though this is not correctly decoded by gsmdecode.

There is also another *offset* in GSM between the downlink and uplink, besides them being on different frequencies. This offset is three bursts. What this means is that when a MS is assigned a time slot (and each MS can only be assigned a single time slot), it will broadcast on the uplink frequency and receive on the downlink frequency at that exact time slot. But the time slots between uplink and downlink differ exactly 3 bursts; an offset. This leads to the MS broadcasting and receiving on the same time slot, but not at the same time. This is exemplified in figure 3.2. This offset leads to simpler and cheaper MEs.

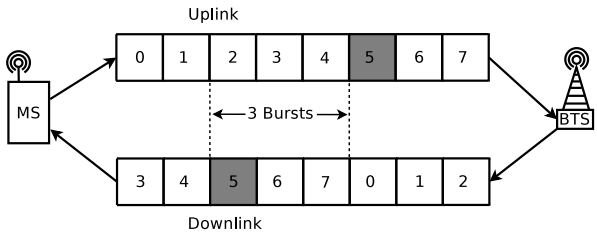


Figure 3.2: The offset difference between downlink and uplink, where the MS is assigned to TS 5

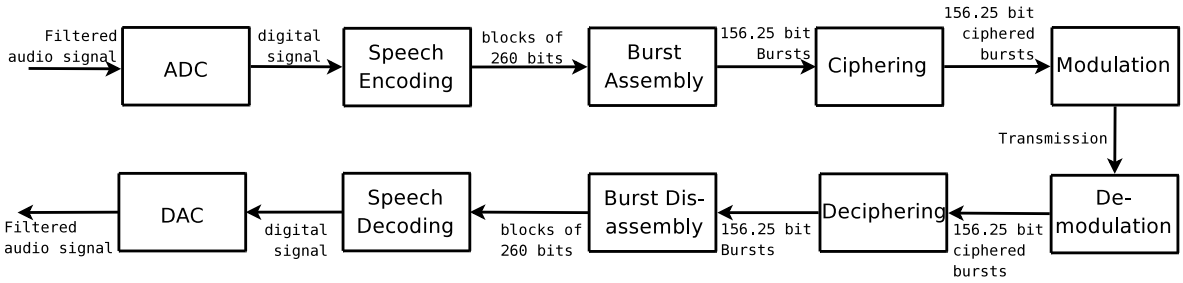


Figure 3.3: Schematic flow from audio to transmission and back

3.3 From speech to signal

Naturally, the main function of the GSM network is transmitting voice data from phone to phone. When someone speaks this generates an analog sound wave. Because GSM is a digital system this sound wave has to be transformed to a digital signal via an ADC. This signal then goes through several processing stages before it is ready to be transmitted over the air to the nearest BTS using radio waves. In order to send the signal, it ironically has to again be converted to an analog signal, this time using a DAC.

The entire process from speech to signal and back is explained schematically in figure 3.3. A microphone inside an ME receives sounds. These sounds are first passed through a filter which removes the frequencies not used in human speech. This filtered analog signal is then digitized by an ADC leaving a digital representation of the analog signal. This digital signal is then compressed with a method known as regular pulse excited-long term prediction (RPE-LPC). This compression is specifically useful for human speech. The ADC generates a digital signal of 104 kbps and the RPE/LPC compresses this to 13 kbps, creating 260 bit blocks representing 20ms of audio [33].

These 260 bit blocks are then divided among bursts. First some redundancy bits are added to the data blocks in order to detect, and if possible correct, transmission errors. The data blocks are then divided into smaller blocks and these blocks are rearranged in a specific order, for protection against multiple package losses. Then bursts are created from these blocks. The burst assembly step is explained in more detail in section 4.4.

The ciphering step encrypts data packets with the current session key, and is discussed in more detail in chapter 7. In the final step the packets are modulated to radio waves in a process called Gaussian Minimum Shift Keying (GMSK) [34].

When a MS sends pure data and no speech then the same process takes place except for the ADC and speech encoding steps. The data is then provided in 260 bit blocks en fed directly into the burst assembly block.

In figure 3.3 it looks as if the transmission step is an atomic step. However, it is important to note that this is not the case; the burst packets are first picked up by the BTS. There they are decrypted, disassembled and channel-decoded until the original digital signal is left. This is sent up in the network via the BSC to the serving MSC. The MSC routes the data to the serving MSC of the receiving MS, which sends the data on to the correct BTS. This BTS then recreates the bursts, encrypts them with the session key of the receiving MS and then broadcasts the bursts to the receiving MS. MSs never communicate directly with each other, but only via the GSM network.

Chapter 4

Um layer 1

This chapter will look further into the first layer of the Um-interface. This layer corresponds with the first layer of the OSI model and is responsible for the direct communication of bits between a mobile phone and a cell tower. It defines logical channels and bursts; the basic units used to send information. This layer is called the physical layer, which may be a misleading name since we are talking about wireless communication, but this does correspond with the OSI model. This chapter ends with some scenarios detailing communication between BTS and MS on the lowest layer.

This section only looks at the bits that are received or send by MSs and BTSs in GSM. We will not look further into the modulation and demodulation steps. Any experimental results shown here use the GNURadio implementation of GMSK demodulation together with the USRP or the Gammu implementation (both described in chapter 1) to transform the radio waves back into bits. Anyone interested in the GMSK (de)modulation is referred to [34] or the GNURadio implementation [17].

4.1 Frames

Eight bursts compose a single *TDMA frame*, which lasts 4.615 ms ($8 \times 576.9\mu$s). These TDMA frames are grouped in multiframes. There are *traffic channel multiframes* and *control channel multiframes*. Each traffic channel multiframe consists of 26 TDMA frames (120ms) and each control channel multiframe consists of 51 TDMA frames (235.4ms). These multiframes are again combined to form *superframes*. These superframes again come in two types, one for control channels and one for traffic channels, conveniently named control channel superframes and traffic channel superframes, respectively. Control channel superframes contain 26 control channel multiframes and traffic superframes contain 51 traffic multiframes. This means that both types of superframes last exactly the same time: $51 \times 26 \times 4.615$ms = 6119.49ms. Finally there is a *hyperframe* which consists of 2048 superframes, lasting 12,533.76s; nearly three and a half hours. One hyperframe consists of 2,715,648 ($26 \times 51 \times 2048$) TDMA frames. These TDMA frames are numbered with a Frame Number (FN), ranging from 0 to 2,715,647, according to their occurrence within the hyperframe sequence [35]. These frame numbers have several uses, e.g. they are one of the parameters needed in the encryption algorithm. Figure 4.1 illustrates the relationship between all the different frames discussed here.

Since the control and traffic multiframes consist of different numbers of TDMA frames, their durations are different, causing them to drift with respect to each other. They will synchronize every superframe. This is a feature of GSM. Every 26th TDMA frame in a traffic multiframe is left idle so a MS can scan the control frequencies. Because the numbers 26 and 51 are coprime, this idle TDMA

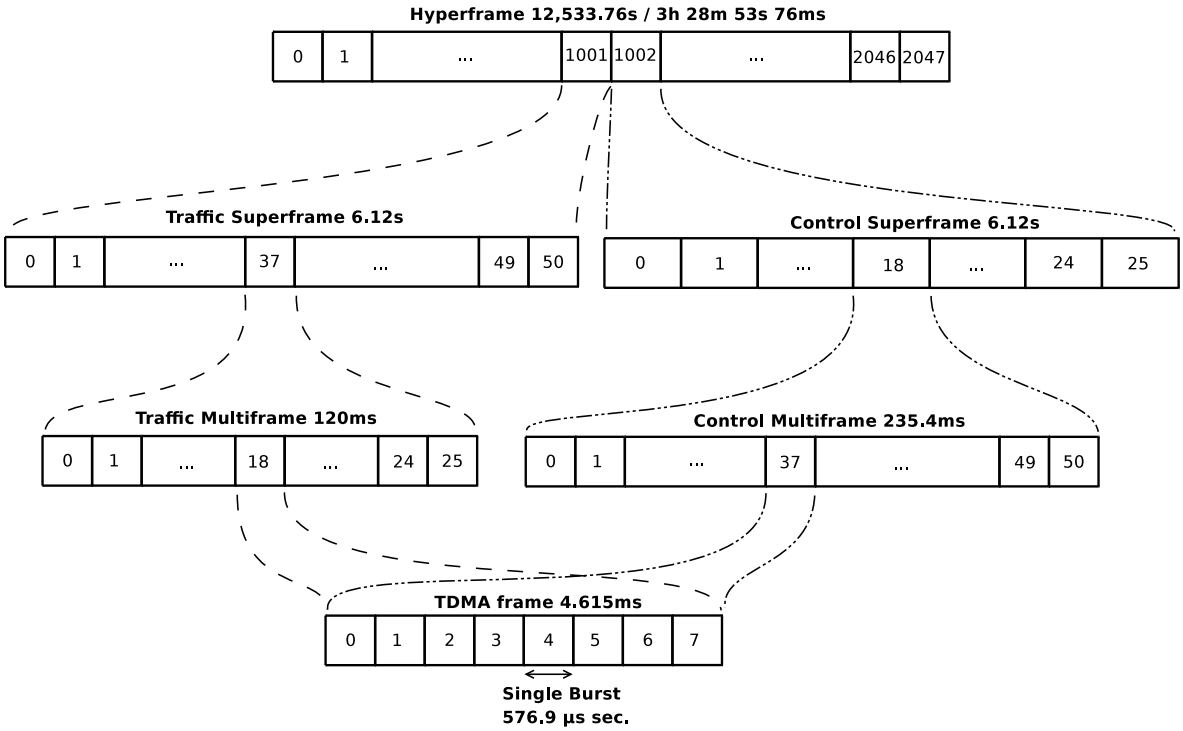


Figure 4.1: Overview of all the frames used in GSM

frame will coincide with every other TDMA frame in the control multiframe once before the entire sequence synchronizes. So even during a conversation the MS has the opportunity to listen in on all control signals, to e.g. measure the signal quality of a nearby BTS to see whether a hand-over should occur.

4.2 Channels

As we discussed in section 3.2, the frequency carriers available to GSM are divided into logical channels. There are several types of defined channels that can be assigned to one of these logical channels, each with a specific function. These types are usually divided between Control Channels (CCH) or signaling channels, and Traffic Channels (TCH). The traffic channels contain almost all the user payload data (speech, data), while the control channels transmit all signaling needed to let the network function. An overview of all the channels can also be found in table 4.1.

Traffic Channels (TCH) Traffic channels are used to send speech and data to and from the user. Speech is encoded to bits and send over a traffic channel. A TCH may either be fully used -Full Rate TCH (TCH/F)- or split into two half-rate channels -Half Rate TCH (TCH/H). When split into two half-rate channels, each of these can be assigned to a different user.
TCH channels are combined in the 26 traffic multiframes, in which 24 frames are actually used for TCH frames.

Control Channels (CCH) The control, or signaling, channels refer to all channels that carry information needed for network management and channel maintenance. There are ten different channels

Trace 4.1: Excerpts showing BTS identities

```
3: 32 12834     [0x3222] Cell identity
5: 02 204       Mobile Country Code (Netherlands)
6: f4 08f       Mobile Network Code (KPN Telecom B.V.)
8: 11 4479      [0x117f] Local Area Code
```

```
3: 0f 3881      [0x0f29] Cell identity
5: 02 204       Mobile Country Code (Netherlands)
6: f4 04f       Mobile Network Code (Vodafone Libertel N.V.)
8: 00 25        [0x0019] Local Area Code
```

This shows two parts of two different so called "system information type 3C" messages. An entire system information type 3C trace can be seen on page 16.

This is an example of information that is transmitted on the BCCH channel of a BTS. Both parts show the identity parameters from BTSs in Nijmegen. This information is the only information an attacker would need to act as a false base station, since the BTS is never authenticated to the MS.

The Local Area Codes, are the Location Area Code (LAC)s named throughout this thesis and define an area within the MCC and Mobile Network Code (MNC). The provider can choose how to number its location areas. As you can see in this example both providers chose different area codes in the same geographical area. It is unknown if a table linking location codes to actual locations is publicly available.

in this category, which are usually divided into three subcategories. In contrast to the names of the actual channels, the names of the three subcategories are hardly ever used in the GSM specification and are added here only for completeness sake [36, 37].

Most control channels, except for the SACH and FACH channels, are usually only found in the 51 control multiframes.

Broadcast Channels (BCH) Used by a BTS to publish system parameters and synchronization information to MSs. These channels are continuously broadcasting from a cell-site, so all these channels are downlink only.

- **Broadcast Control Channel (BCCH)** This channel contains system parameters needed to identify the network and gain access. These parameters include the LAC, MNC, the frequencies of neighboring cells, frequencies and time slots of other important logical channels, and access parameters. A MS also uses the BCCH broadcasts to determine which BTS to connect to, by measuring signal strengths and error rates from different BCCHs. BTSs also broadcast a recommended transmit power level on this channel, which can help MSs to select the most optimal BTS.
- **Frequency Correction Channel (FCCH)** This channel is used by the MS to find BTSs. A MS is always scanning its known beacon frequencies for FCCH channels. The FCCH generates a tone on the radio channel that is used by the mobile station to adjust its local oscillator.
- **Synchronization Channel (SCH)** On the SCH numbers are transmitted that allow a MS to compute the current TDMA frame number (FN) for frame synchronization. The SCH is always located after the FCCH on a beacon frequency. This channel also transmits a so called Base Station Identity Code (BSIC). This code's name should not be taken to

literally, it consists of 6 bits that can be used by the MS to differentiate between nearby BTSs broadcasting on the same frequency. The code does not state the actual identity of a cell.

Cell Broadcast Channel (CBCH) This is not really its own type of logical channel. It is used to broadcast specific information to network subscribers; such as weather, traffic, sports, stocks, etc. The CBCH uses the same physical channel as the SDCCH (further down in this list).

Common Control Channels (CCCH) The CCCH defines a group of channels used for signaling between the BTS and the MS and to request and grant access to a traffic or control channel. Of these, only the RACH is an uplink channel, the PCH and AGCH are both downlink channels.

Paging Channel (PCH) This channel is used to inform the MS of incoming traffic. The traffic could be a voice call, SMS, or some signaling messages. Typically an MS will continuously listen on the PCH for incoming transmissions addressed to its IMSI or TMSI. A message on the PCH contains the reason for the paging. If this reason is an incoming call, then this will usually cause a MS to warn its user of an incoming call; the mobile phone will 'ring'. Because the PCH is often allocated on a single time slot in the downlink beacon frequency, MEs need only listen for incoming traffic a fraction of the time, saving battery power.

Random Access Channel (RACH) This channel is used by a MS to request a channel on which to send or receive traffic or signaling information. When the MS initiates a communication, a message is transmitted on the RACH. When a MS receives a page, this RACH message will follow a message on the PCH. The RACH is a shared uplink channel.

Access Grant Channel (AGCH) Typically a BTS will respond to a message on the RACH with a message on the AGCH, granting a MS a certain channel (ARFCN). This message both acknowledges the reception of a RACH message and answers it with a channel to use.

The RACH is usually allotted time slot zero of the uplink beacon frequency and the PCH and AGCH usually use time slot zero of the downlink beacon frequency at certain intervals (remember that due to the offset between the downlink and uplink, the time slots zero do not coincide). However more PCHs can be allocated in other frequencies if heavy traffic is to be expected. The exact frequencies (ARFCNs) and time slots are communicated via the BCCH messages.

Dedicated Control Channels (DCCH) These are the point-to-point signaling channels used for call-setup, handovers, location updates and other management tasks.

Standalone Dedicated Control Channel (SDCCH) The SDCCH channels are mostly used for call-setup, location updates and SMS. A channel grant message on the AGCH will usually assign a SDCCH channel to a MS. On the SDCCH, messages are exchanged between MS and BTS that take care of identification and authentication of the MS, instantiating of the cryptography and assignment of a traffic channel, when handling a call-setup. Short Message Service (SMS) messages are send directly via an appointed SDCCH. When the business to perform is over, a channel release message can be send over the SDCCH to free it for a new assignment.

Slow Associated Control Channel (SACCH) A SACCH channel is always assigned and used with a TCH or SDCCH channel. The SACCH carries information for optimal radio operations like synchronization commands and channel measurements. The SACCH is usually transmitted on frame 12 of the 26 traffic channel multiframe, and is therefore considered 'slow.' Data must be transmitted continuously on the defined SACCH time slots, because reception of the SACCH messages is taken as proof of the continuing existence of the 'physical' radio connection. Because this channel does not use the traffic channels for its communication - it uses the same frequency, but does not use TCH bursts - the SACCH is also called "out band signaling".

Fast Associated Control Channel (FACCH) The FACCH is always paired with a traffic channel (TCH). The FACCH information can actually 'steal' TCH bursts (or half TCH bursts) to transmit information. To this end TCH bursts actually have 'stealing bits' indicating whether this package contains data or signaling information (see section 4.3). The FACCH is used for urgent (unscheduled) signaling like call disconnects and handovers. When a new call is established, the first communication on the traffic channel (TCH) is actually FACCH traffic. The FACCH packages actually replace TCH data, so the use of FACCH signaling degrades the conversation quality. Therefore there is a limit of up to one out of six speech bursts that may be stolen by the FACCH. The FACCH manages a much higher data rate than the SACCH and is therefore considered fast. Because the FACCH uses bursts reserved for a TCH channel it is also called "in band signaling".

The SDCCH is called stand-alone, because it is not linked to any other channel unlike the other two dedicated control channels, which are sometimes collectively referred to as Associated Control Channels (ACCH).

4.2.1 Channel combinations

Every logical channel described here can be allocated to its own ARFCN. However that would be highly uneconomical since a lot of these channels are not used often enough to warrant them hogging $\frac{1}{8}$th of a frequency continuously. In fact most of these logical channels share an ARFCN. This practice is standardized into several often occurring sequences.

As was explained earlier in section 2.2.1, each BTS has a certain beacon frequency. Time slot 0 of this beacon frequency is always reserved for control channels. In principle the layout of this logical channel can be decided by the operator, as long as at least the FCCH is present on this channel, directly followed by an SCH. However in practice nearly all operators seem to keep to the advised layout shown in figure 4.2. Notice that the use of CCCH in this figure points to the category of the PCH, RACH and AGCH channels, and is used here to mean either PCH or AGCH (since RACH is only present on the uplink frequency).

When auxiliary channels are used for control channels, their layout is often different. They contain a subset of the BCCH, CCCH, SDCCH and SACCH channels.

Figure 4.2 shows a possible layout of all time slots within one ARFCN. It contains two control channel 51-multiframes on time slots 0 and 1. Time slots 2 to 7 contain the traffic channels and some SACH messages in the 26-multiframes. Remember that these channels are alternated sequentially on the same frequency. The numbered SACH messages in time slot 1 correspond with the numbered

TDMA frame	TS0	TS1	TS2-TS7	TDMA frame	TS0	TS1	TS2-TS7
0	FCCH	SDCCH 0	TCH	26	CCCH	SDCCH 6	TCH
1	SCH		TCH	27			TCH
2	BCCH		TCH	28		SDCCH 7	TCH
3			TCH	29			TCH
4		SDCCH 1	TCH	30	FCCH		TCH
5			TCH	31	SCH		TCH
6	CCCH		TCH	32	CCCH	SACCH 0	TCH
7			TCH	33			TCH
8		SDCCH 2	TCH	34			TCH
9			TCH	35			TCH
10	FCCH		TCH	36	CCCH	SACCH 1	TCH
11	SCH		TCH	37			TCH
12	CCCH	SDCCH 3	SACCH	38			SACCH
13			TCH	39			TCH
14			TCH	40	FCCH	SACCH 2	TCH
15			TCH	41	SCH		TCH
16	CCCH	SDCCH 4	TCH	42	CCCH		TCH
17			TCH	43			TCH
18			TCH	44		SACCH 3	TCH
19			TCH	45			TCH
20	FCCH	SDCCH 5	TCH	46	CCCH		TCH
21	SCH		TCH	47			TCH
22	CCCH		TCH	48		IDLE	TCH
23			TCH	49		IDLE	TCH
24		SDCCH 6	TCH	50	IDLE	IDLE	TCH
25			IDLE	0	FCCH	SDCCH 8	TCH

Figure 4.2: Possible channel layout for a single frequency.

SDCCH messages. Since only one SACCH message is needed for every two SDCCH messages the shown multi frame only transports SACCH 0 to 3, the next multiframe would contain SACH 4 to 7.

The unnumbered SACH messages in time slots 2 to 7 correspond to the traffic 26-multiframe they are transported on.

4.3 Burst types

GSM uses Gaussian Minimum Shift Keying (GMSK) as its modulation method. GMSK provides a modulation rate of 270.833 kb/s. The duration of a single burst is 576.9μs. So the amount of bits that can be transmitted in one burst equals nearly 156.25 bits.

There are five different kinds of bursts that are send over the channels [36, 38]. The normal burst is the main burst type used for most communications. These five bursts have different structures which allows for some differentiation, but the different types of bursts are mostly recognized because most bursts are only used on a single, specific logical channel. All five of the bursts will be explored in more detail here and are shown in figure 4.3 and table 4.1.

Normal Burst. The normal burst carries speech or data information. When the burst is sent on

a TCH it contains speech or FACCH data, when it is transmitted on a control channel it contains signaling information. The structure of the normal burst is shown in figure 4.3(a).
MSs can be at different distances from a BTS. The further away a MS is from a BTS the more transmission delays will cause their bursts to move out of their time slots. To prevent overlapping and interference from other transmissions in other time slots every burst has a *guard period* at the end. The normal burst has 8.25 bits of guard period. The quarter bit in here might raise eyebrows, but remember that we are talking about periods in which to send these bits, so from every normal burst transmission the period it takes to transmit 8.25 bits are ignored.

All bursts also have *tail bits* at the beginning and end of every transmission. These tail bits are also ignored. The starting tail bits are used to compensate for the time it takes a transmitter to reach the peak of its power, and the ending tail bits for the time it takes the transmitter to power down.

The normal burst contains two data payloads of 57 bits each, the *data bits*. In a fullrate TCH both payloads are used for the same conversation,on a halfrate TCH each payload belongs to a different conversation or a FACCH message.

The *stealing bits* indicate whether the corresponding data bits contain voice data (when set to '0'), or if the FACCH channel has 'stolen' it for signaling information (when set to '1').

Finally the *training sequence* is a predefined, known, sequence of bits which can be used by an equalizer to reduce interference between symbols caused by multipath propagation. There are eight defined training sequences in GSM. The MS is informed which training sequence will be used by a Training Sequence Code (TSC), a code consisting of three bits that are a part of the Base Station Identity Code (BSIC) which is sent on the SCH.

Frequency Correction Burst The frequency correction burst (figure 4.3(b)) is used for frequency synchronization of an MS. The sending of frequency correction bursts makes up the FCCH channel. A frequency correction burst contains the standard guard time and tail bits. The 142 *fixed bits* contain a standard modulated signal of only '0's. This allows a MS to find the beacon frequency of the BTS and adjust its oscillator to maximize reception.

Dummy Burst The dummy burst looks exactly like a frequency correction burst. It is only used when a burst is expected but no information needs to be transmitted, like on a SACCH. The fixed bits of a dummy burst can be all zeros, or the middle 26 bits can contain a training sequence like in the normal burst. In a dummy burst the fixed bits are often referred to as *mixed bits*.

Synchronization Burst A synchronization burst again has the same tail bits and guard period as the previous bursts. It has two data payloads of 39 bits which contain the TDMA frame number (FN) and the Base Station Identity Code (BSIC). A synchronization burst also contains an *extended training sequence*, which has the same function as the training sequence in the normal burst, only some bits longer (totaling to 64 bits) and there exists only one defined extended training sequence. Synchronization bursts are send on the SCH and are used to synchronize the MS's time with that of the BTS.

Access Burst Access bursts can only be sent by the MS and are only sent on the Random Access Channel (RACH). The access burst is also the only burst type that has a different guard period and starting tail bits. Both are longer than in the other bursts.

The modulated signal from a MS needs to cross the distance between MS and BTS, which takes time. This might cause bursts to arrive outside of their burst period; a timing delay. GSM compensates for this by having the Base Station Subsystem (BSS) compute a timing advance and assigning it to

an MS. The MS then sends its bursts delayed according to the timing advance causing them to arrive on time at the BTS. The first time a MS tries to transmit to a new BTS, it doesn't know any timing advance, so the chances of this message arriving outside of its burst period are much larger. This first message is an access burst sent on the RACH. To prevent interference from different MSs using the RACH channel, the bursts on this channel have the much larger guard period (68.25 bits) and starting tail bits (8 bits). The *synchronization sequence* is a known sequence of bits used by the BTS to compute the timing advance. The data payload of 36 bits contains the request the MS makes, and will usually result in the assignment of a Standalone Dedicated Control Channel (SDCCH) on the Access Grant Channel (AGCH) by a normal burst.

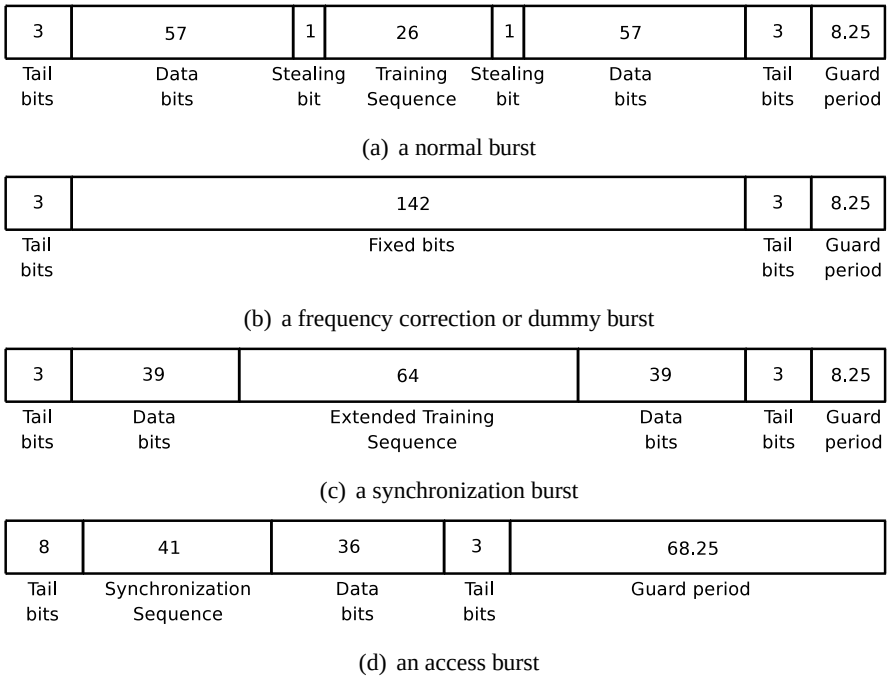


Figure 4.3: Overview of the layout of different burst structures

4.4 Burst assembly and channel encoding

In section 3.3 we saw an overview of the steps that data in a phone goes through before it is transmitted as a burst. Some details were left out in that section that we will look into here.

Control frames are always 184 bits in size, speech frames are 260 bits, before they are channel encoded. After all encoding steps both frames end up being transmitted as 456 bits, including all the redundancy bits. This means they are transmitted in four (using both 57 bit payloads) or eighth (using one payload) bursts.

The burst assembly step of figure 3.3 is decomposed in figure 4.4. The first two steps together are often called the channel encoding step, and they add redundancy bits to the data.

When speech data is concerned the block encoding step divides the data into three classes according to function and importance. As you may imagine in the encoding of sound waves some bits can be more important to recreate a somewhat identical sound wave then others. Of the 260 bits of data

Acronym	Full name	Up / down	burst types
TCH	Traffic Channels		
TCH/FS	Full Rate TCH	unicast up/downlink	normal
TCH/HS	Half Rate TCH	unicast up/downlink	normal
BCH	Broadcast Channels		
BCCH	Broadcast Control Channel	multicast downlink	normal
FCCH	Frequency Correction Channel	multicast downlink	frequency correction
SCH	Synchronization Channel	multicast downlink	synchronization
CBCH	Cell Broadcast Channel	narrowcast downlink	normal
CCCH	Common Control Channels		
PCH	Paging Channel	unicast downlink	normal
RACH	Random Access Channel	shared uplink	access
AGCH	Access Grant Channel	unicast downlink	normal
DCCH	Dedicated Control Channels		
SDCCH	Standalone Dedicated Control Channel	unicast up/downlink	normal
SACCH	Slow Associated Control Channel	unicast up/downlink	normal
FACCH	Fast Associated Control Channel	unicast up/downlink	normal

Table 4.1: Summary of all channels with the burst types that are send over them

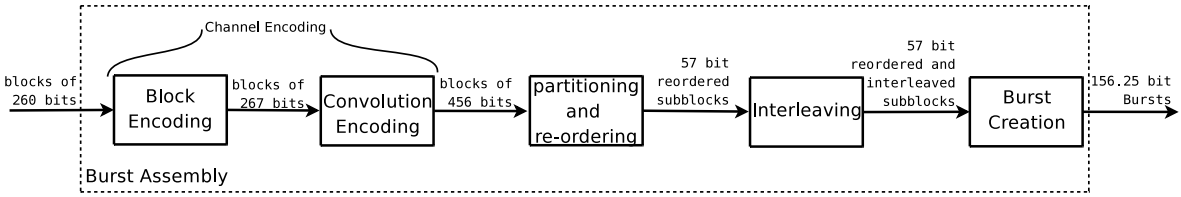


Figure 4.4: Decomposition of the burst assembly block shown in figure 3.3

50 bits are assigned to class Ia, 132 bits to class Ib and the remaining 78 bits to class II. The class Ia bits are then protected by a cyclic code resulting in a 3 parity bits that are added at the end of the class Ia bits. Four ‘0’ bits are added after the class Ib bits en the resulting 267 bits (50 class Ia + 3 parity + 132 class Ib + 4 zeros + 78 class II) are send to the convolution encoder. There only the class I bits are protected with a convolution encoder that encodes every bit with two bits, doubling the number of class I bits from 189 to 378 bits. The convolution encoder computes every redundancy bit out of five consecutive bits, which is the reason for adding the four zeros at the end of the block encoding step. The resulting data is now a packet of 456 bits (378 class I and 78 class II) which still encodes 20 ms of audio.

Signaling information that will be transmitted via normal bursts can not be divided into the three importance classes and it arrives in blocks of 184 bits. In the block encoding a 40 bits fire code is computed over the input and added to them together with four zero bits. The resulting 288 bits (184 + 40 + 4) are then doubled through the same convolution encoding as speech data goes through resulting in 456 bits.

For the RACH, FCCH and SCH data this entire burst assembly step does not apply (they are transmitted via different bursts). The exact channel encoding steps for every conceivable bit of data is

defined in [38].

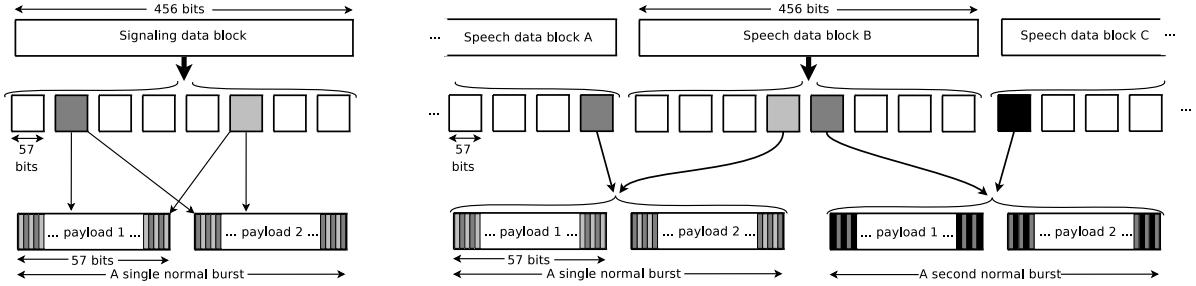
The 456 bits resulting from the channel encoding are divided into sub blocks of 57 bits by the following formula:

$$\text{bit}(i, j) = 64i + 57j \pmod{456}$$

This formula shows with which bit in the original 456 bit block the ith bit of the jth sub block corresponds. So bits 0-5 of the sixth sub block are 342, 406, 14, 78 and 142 of the original sequence. The resulting sub blocks are then interleaved.

If this is signaling information then the data will end up spread out over four consecutive normal bursts by putting the first four sub blocks in the even numbered bits and the last four sub blocks in the odd numbered bits of the two 57 bit payloads per burst.

If this is speech data then the first four sub blocks are put in the even numbered bits of the payloads of four consecutive bursts and the last four sub blocks are put in the odd numbered bits of the *next* four consecutive bursts. So each speech block is smeared out over eight bursts and each burst contains two times 57 alternating data bits from two different speech blocks.



(a) Interleaving for signaling data

(b) Interleaving for speech

Figure 4.5: Interleaving

All this reordering and interleaving is done to strengthen the error correction. Transmission errors become less concentrated and this significantly increases the chances of recovering the original data when errors occur. Even when entire bursts disappear. The burst creation step then adds the tail bits, stealing bits and training sequence to every two 57 bit blocks to create the normal bursts we know from figure 4.3(a)

4.5 Scenarios

In this section the usage of channels is illustrated by some scenarios. First the initial steps during a sign-on are discussed. Then we will look at the requests and assignments for channels, which is necessary every time a MS and BTS want to communicate. Finally the three possible channel assignments for a Mobile Originating Call (MOC) - the steps between BTS and MS when a MS initiates a call - are discussed.

These scenarios are often illustrated using message sequence charts, where the arrows are labeled with message, preceded by the logical channel they travel on in capitals.

4.5.1 Sign on

The first basic steps of a sign on procedure occur only at the time a mobile is powered on, during hand-overs, or when a mobile leaves an area that has no GSM coverage, and thus enters the reach of a BTS. These basic steps are the following:

1. Scan the known list of beacon frequencies for the occurrence of a frequency correction burst.
2. When found, capture the next burst on this time slot in this frequency, which will always be a synchronization burst.
3. Use the training sequence in the synchronization burst to fine tune to the beacon frequency, and set the TDMA frame number according to the one in the synchronization burst.
4. Start listening on the Broadcast Control Channel (BCCH) channel on this frequency to gather system information.
5. Use the system information to tune to the RACH and AGCH, and send an access burst.
6. Listen to the AGCH for a response. Repeat steps 5 and 6 until a control channel is assigned.

A MS can encounter multiple BTSs transmitting on the beacon frequencies of its SIMs provider. It will then order these by reception quality and connect to the one with the best reception. The eventual response on the AGCH will contain a ARFCN number and time-slot, to which the MS can tune. This ARFCN + time-slot will be a reserved SDCCH for this MS, on which authentication (as discussed in sections 2.5.1 and 7.1) can take place.

4.5.2 Channel setup

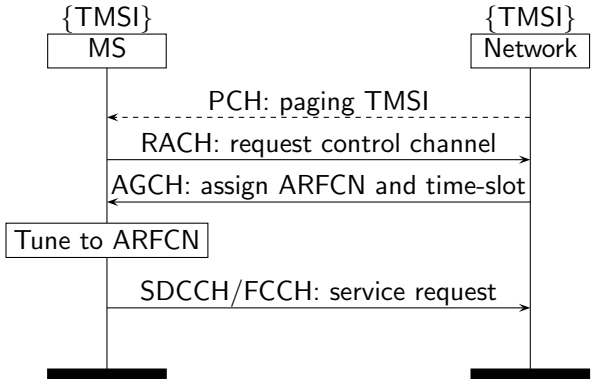


Figure 4.6: Generic channel request/assignment.

Figure 4.6 shows the generic way for a MS to get a control or voice channel connection from the BTS. In this channel setup a MS is already linked to a BTS as was discussed above. Channel setup precedes most scenarios, because it assigns a control or voice channel to the MS. These assigned channels are needed for dedicated signaling between the MS and BTS, e.g. for authentication or call establishment.

The first dashed arrow in this diagram shows a possible message on the paging channel. If this message occurs, then the channel setup is initiated by the network, e.g. because there is an incoming voice call for the subscriber (MTC). If the dashed arrow is ignored, then the diagram shows a channel setup initiated by the MS, e.g. to perform a location update. Who ever initiates the channel setup, all the messages besides the "paging message" remain the same.

The "request control channel" message contains the reason for the request (e.g. answering page) and the type of channel requested. The RACH channel is a free for all channel where all MSs in the neighborhood can transmit an access burst on. This can cause different access bursts to collide. Therefore every MS will wait a random time until it retransmits the access burst.

At some point the access burst will be received correctly and a response will be transmitted on the AGCH channel. Because the AGCH is also a general broadcast channel, this response contains an identifier linking this response to the original request. The MS recognizes that this response is meant for him and tunes to the ARFCN and time-slot mentioned in the response. On this channel further communication can happen to achieve the goal for which the channel setup was initiated.

4.5.3 Mobile Originated Call (MOC)

When making a call the MS and BTS first go through the standard channel setup as discussed above. This section shows the signaling to setup a MOC. The signaling for a mobile terminated call (MTC) is so similar, that we will not discuss it here.

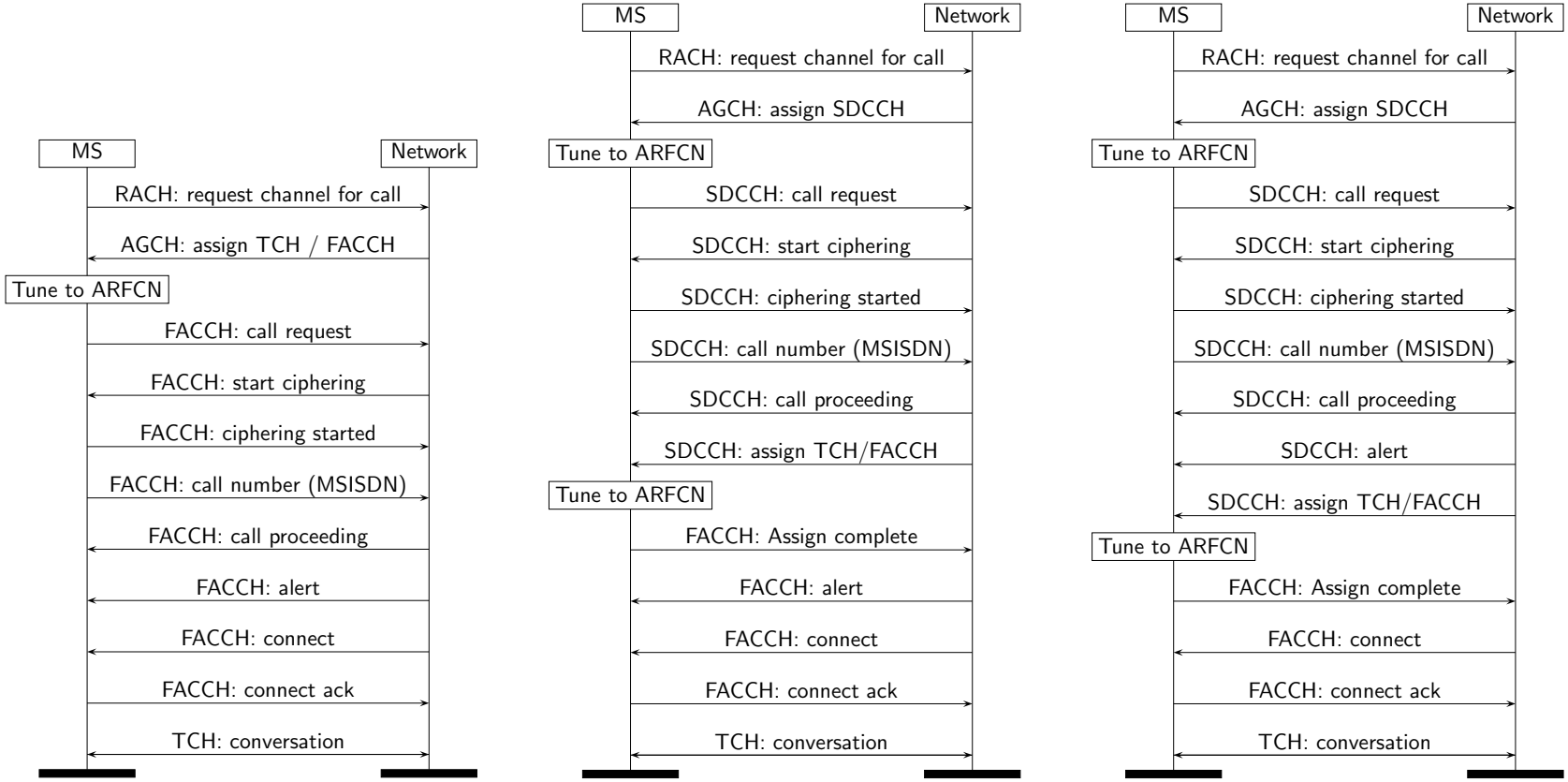
Figure 4.7 shows three different possibilities for a MOC call setup. The differences between these possibilities lie in the moment at which a traffic channel is assigned to the MS. All three figures incorporate the general channel assignment described above at the beginning. They omit the paging message, because they describe an MOC. An MTC would start with a paging message from the BTS.

These scenarios also omit authentication. Authentication will often happen directly after sign-on and can be skipped during MOC. Although a network can always choose to have the MS authenticate itself. In these scenarios this would occur between the "call request" and "start ciphering" messages. Authentication is discussed in more detail in sections 2.5.1 and 7.1.

The first is called "Very early assignment" and is detailed in figure 4.7(a). Here the network responds by immediately assigning a traffic channel (Traffic Channels (TCH)) to the MS. Remember that the traffic channel is also used as a Fast Associated Control Channel (FACCH) channel, by replacing the traffic payload with signaling information.

After assignment the traffic channel is first used as FACCH signaling channel, to exchange the signaling necessary for call initiation. The "call request" message from the MS signals the network that call establishment should begin. If the network supports ciphering, then ciphering is now negotiated, possibly preceded by authentication. Then the MS signals the number it wishes to call (MSISDN) to the network. The "call proceeding" message from the network acts as an acknowledgment that a call connection is being initiated. At this point the display on the ME should show that calling is being initiated. The "alert" message from the network carries the ringing tone, indicating the waiting for the other side to pick up (trace 4.3). When the other side does so, the connection is finalized via the "connect" and corresponding "connect ack" message which will cause the MS to consider the FACCH channel as a pure TCH channel on which the conversation can take place.

Very early assignment is not very efficient with precious traffic channel resources. A call could be aborted after the initial assignment, e.g. because the MS fails authentication, or the receiving MS can not be found. Then for some time the traffic channel has been unnecessarily in use. Typically traffic



(a) Very early assignment (b) Early assignment (c) Late assignment

Figure 4.7: Channel assignment for Mobile Originated Call (MOC)

channels are more precious in a network than stand-alone signaling channels.

Early assignment differs by first assigning a SDCCH channel for the initial steps in the call set-up. It does require the MS to tune to an ARFCN twice, but if the call is aborted in the initial phase, then no traffic channel was reserved needlessly. The extra channel assignment takes place between the "call proceeding" message and the "alert" message. Besides the extra channel assignment and the initial steps taking place on the SDCCH the early assignment is identical to the very early assignment.

Early assignment is more efficient with traffic channels than very early assignment. However if the callee takes a long time to answer the call, or if he doesn't answer the call at all, then early assignment isn't much more efficient.

Late assignment is in many ways identical to early assignment, accept that the traffic channel is only established when the callee picks up the call.

This is of course most resource efficient, but it does have a few troublesome side effects. Firstly the ringing tone of the alert message has to be generated by the MS, because there is not yet a traffic channel to transmit this on. Most MSs generate a tone of continental European taste, which will sound confusing to subscribers outside of continental Europe, like Britain. But secondly and most importantly the traffic channel assignment can cause a post-pick-up delay, potentially swallowing the first few syllables of the conversation [39].

During this research only early assignment was encountered as call setup procedure. It is commonly assumed that early assignment is the preferred method for most providers [39, 27].

Trace 4.2: CC Call Proceeding

```
HEX l2_data_out_B:194 Format B DATA (down)
000: 03 62 09 83 02 2b 2b 2b  - 2b 2b 2b 2b 2b 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
     0: 03 -------1 Extended Address: 1 octet long
     0: 03 ------1- C/R: Command
     0: 03 ---000-- SAPI: RR, MM and CC
     0: 03 -00----- Link Protocol Disciminator: GSM (not Cell Broadcasting)
     1: 62 -------0 Information Frame
     1: 62 ----001- N(S), Sequence counter: 1
     1: 62 ---0---- P
     1: 62 011----- N(R), Retransmission counter: 3
     2: 09 -------1 EL, Extended Length: y
     2: 09 ------0- M, segmentation: N
     2: 09 000010-- Length: 2
     3: 83 1------- Direction: To originating site
     3: 83 -000---- 0 TransactionID
     3: 83 ----0011 Call control. call related SS messages
     4: 02 00------ Send Sequence Number: 0
     4: 02 --000010 Call Proceesing
```

The CC Call Proceeding message lets the MS know that the requested call has been accepted by the network. Which means the MS is authenticated and allowed to make the call. The provided MSISDN number was also a correct number. In the mean time the network is also trying to establish a connection to the other phone.

Trace 4.3: CC Alerting

```
HEX l2_data_out_B:296 Format Bbis (RR, MM or CC)
000: 83 01 1c 10 a1 0e 02 01  - 00 02 01 10 30 06 81 01
001: 28 84 01 07 1e 02 ea 88
     0: 83 1------- Direction: To originating site
     0: 83 -000---- 0 TransactionID
     0: 83 ----0011 Call control. call related SS messages
     1: 01 00------ Send Sequence Number: 0
     1: 01 --000001 Call Alerting
     2: 1c XXXXXXXX UNKNOWN DATA (22 bytes)
     2: 1c YYYYYYYY REST OCTETS (22)
```

This message shows the MS that the receiving phone is being contacted. It contains the ringing tone the subscriber hears while the called party has not yet picked up their phone.

Chapter 5

Um layer 2

This chapter will look further into the second layer of the Um protocol. Layer 2 is also called the data link layer and more or less coincides with the data link layer of the OSI model. The signaling protocol used on the data link layer is called LAPDm. It is a slightly modified version of the Link Access Protocol on the D channel (LAPD) protocol, which is used within ISDN systems and on the Abis interface (connecting the BTS with a BSC, section 2.4). Both protocols are derived from, and strongly resemble, the High level Data Link Control (HDLC) protocol [40].

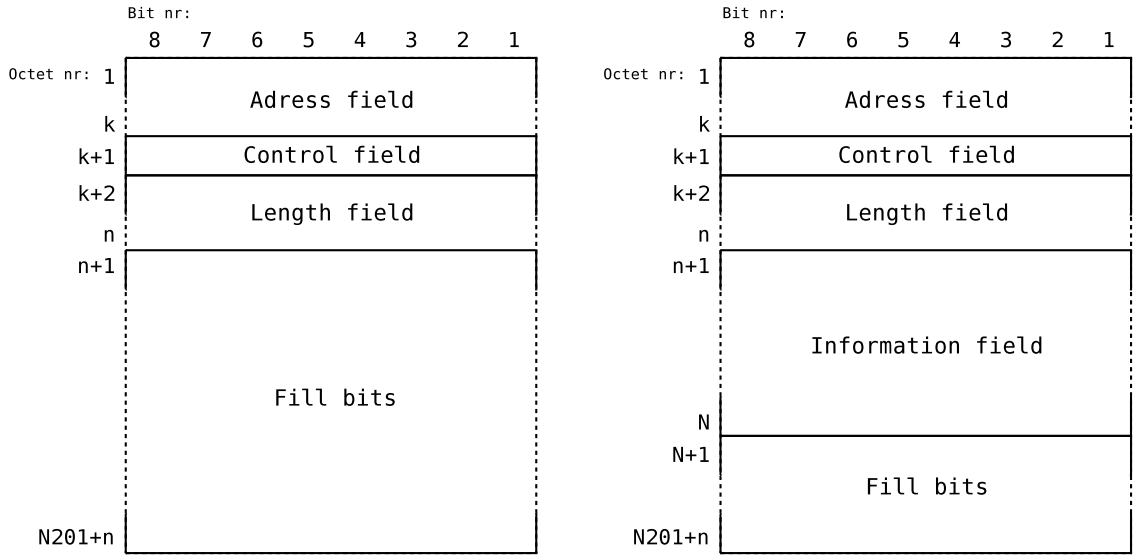
As we saw in chapter 4 both control signals and speech data are sent over the Um interface in bursts. The data link layer is only defined for the signaling channels, not for the speech channels (TCHs). In the case of speech packets, eight consecutive bursts contain one speech frame (and two halves) of 20ms of audio. These bursts contain no further headers or other meta information, only speech data. During a phone conversation, the traffic on the dedicated TCH channel needs no meta information in order to be reconstructed correctly at the receiving end. The combination of the TDMA conventions and the redundancy encoded into the bursts are enough to ensure this under most circumstances.

5.1 Layer 2 control frames

All frames on the second layer consist of 23 bytes, usually named octets. We saw in section 4.4 that control frames are processed for transmission in blocks of 184 bits, which are eventually transmitted using 4 bursts. So one frame consists of $184/8 = 23$ octets.

Signaling frames can be sent through the data link layer in two modes; *acknowledged* and *unacknowledged*. In acknowledged mode, data is sent in Information (I) frames, which provide positive acknowledgment, error protection through retransmission and flow control. Acknowledged mode can only be used on the Dedicated Control Channels (DCCH) channels. In unacknowledged mode, data is sent in Unnumbered Information (UI) frames. There is no flow control nor are there acknowledgments or layer 2 error protections. Unacknowledged mode is always used on the BCCH, Paging Channel (PCH) and AGCH. The RACH channel uses neither modes. Because it is accessed by multiple MSs it can not be protected through reservation of channel or timeslot.

Figure 5.1 shows generic layer 2 frames for acknowledged operations called the A and B type. The, not shown, Bbis type consists of purely 23 octets of information bits, which is only used in



(a) A-type frame

(b) B-type frame

Figure 5.1: General layer 2 frames, used in acknowledged mode.

unacknowledged mode. There are more frame types, but the A, B and Bbis types are most used and most interesting in the context of decoding GSM signals. All frame definitions can be found in [41, 42]. The A-type frame is sent in acknowledged mode as a fill frame when no payload is available on an active connection, the B-type frame transmits actual signaling data.

The numbering and naming in figure 5.1 are consistent with the ETSI documentation [42]. The first bit received is the least significant bit of the first octet; the bit numbered 1 of octet 1 in the figure. The N201 variable represents the size of the information field in octets.

The I-frames, used for acknowledged operation, distinguish themselves from UI-frames by their header. This header contains an address field, which may consist of one or several octets. However for the current GSM standard this field never surpasses the length of one octet. The control field always consists of a single octet and is used for the acknowledged operations. The length field resembles the address field in that it is defined as spanning one or several octets, yet in the current GSM standard it never uses more than a single octet. These three header fields are shown in more detail in figures 5.2 to 5.4.

After the header zero or more information octets follow, which can be passed on to the third layer of the Um protocol. These are often called Information Element (IE)s. The name IE will return when we are talking about the third layer of the Um protocol, there they also designate information bits. However in the context of the third layer IEs define a variable number of information bits, while in layer two they always point to octets of information bits. The maximum number of IEs is named N201 in the ETSI documentation [42]. The N201 value is coded in the length field. The maximum number of IEs present in a layer 2 frame differs per logical channel and are shown in table 5.1. A Bbis frame consists only of information octets, and the presence of information octets make the difference between the A and B type frames.

Finally if a layer 2 frame has not yet reached a length of 23 octets it is filled with appropriately named fill bits. An octet of fill bits is always coded as ‘00101011’ when sent by the network. The

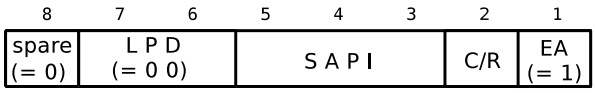


Figure 5.2: Layer 2 address field format

same coding, together with '11111111' are used as fill bits when sent by the MS. The '00101011' encodes the string "2B".

Table 5.1: Logical channels and the maximum length of the LAPDm information field.

Logical channel	N201
SACCH	18 octets
SDCCH, FACCH	20 octets
BCCH, AGCH, PCH	23 octets

5.2 The I-frame header

The A and B type frames shown in figure 5.1 are used as I frames. Their headers, the address, control and length fields, are the interesting parts, setting them apart from UI frames. The possible information field is sent up to the third layer of the Um-protocol.

5.2.1 The Address field

The address field is so named, because the combination of SAPI and the channel on which a frame is received identifies the 'address' of the third layer service primitive to which the information bits need to be forwarded. It should not be confused with the addressing of packages to the correct agent, which is entirely decided by the geographical location, used frequency and time-slot. The address field is shown in figure 5.2.
The address field consists of:

- the address field extension (EA) bit;
- the Command / Response (C/R) bit;
- the Service Access Point Identifier (SAPI);
- the Link Protocol Discriminator (LPD);
- and a spare bit.

For the Link Protocol Discriminator (LPD) field only the combination "0 0" should occur within control frames on the Um interface.

The Service Access Point Identifier (SAPI), consists of three bits, so it can take the values 0 to 7. Currently only two values are defined; 0 ("000") for all control signals and 3 ("011") for sms

messages. the other six values are reserved for future standardization. Table 5.2 shows the possible modes for different SAPIs on different channels. It shows, amongst other things, that SMS messages are always sent in acknowledged mode. The channels that only support unacknowledged mode, do not use frames with an address field, but they do not need to know the SAPI.

The Command / Response (C/R) bit identifies a frame as either a command or a response. Table 5.3 shows the possible meanings of the C/R bit given the direction within the Um interface, of the frame.

The address field extension (EA) bit can be used to extend the size of the address field with another octet. When the EA bit is set to '0' it indicates that the address field extends to the following octet, when it is set to '1' it indicates that this is the final octet of the address field. Currently in the GSM system no layer 2 frames are sent with an address field spanning more than one octet, so for all intent and purposes this bit will always be set to '1'.

The spare bit is, as the name suggests, currently unused in the GSM protocol. Standard this bit is set to '0', however if a receiving entity finds this bit set to '1' it should simply disregard it without yielding an error.

Table 5.2: Mode of operation and allowed SAPIs

Type of channel	SAPI=0	SAPI=3
BCCH	Unacknowledged	Not supported
CCCH	Unacknowledged	Not supported
SDCCH	Unacknowledged and Acknowledged	Acknowledged
SACCH associated with SDCCH	Unacknowledged	Not supported
SACCH associated with TCH	Unacknowledged	Acknowledged
FACCH	Unacknowledged and Acknowledged	Not supported

Table 5.3: C/R field bit usage

Type	Direction	C/R value
Command	BS → MS	1
	MS → BS	0
Respond	BS → MS	0
	MS → BS	1

5.2.2 The control field

The control field, which is detailed in figure 5.3, can have several compositions. For further understanding of this control field another set of, regrettably confusing, identifiers (and thus more acronyms) need to be introduced. The control field can be transmitted in three formats:

- the Information transfer format (I format); not to be confused with the Information (I) definition,

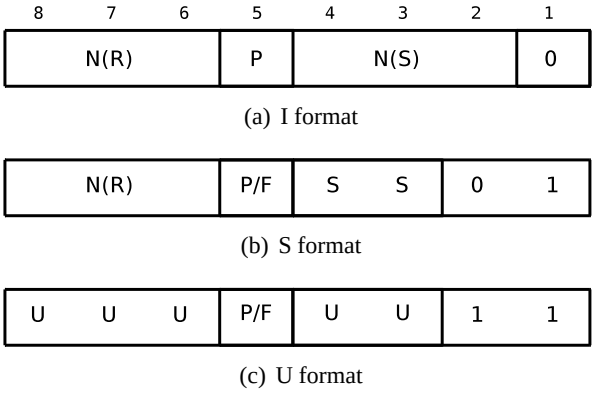


Figure 5.3: Layer 2 control field format

is used to perform an information transfer between layer 3 entities, with positive acknowledgment.

- The Supervisory format (S format); is used for layer 2 supervisory control functions, such as: acknowledge I frames, request retransmission of I frames and request a temporary suspension of I frame transmissions.
- The Unnumbered format (U format); not to be confused with the Unnumbered Information (UI) definition, is used to provide additional data link control functions and unacknowledged information transfers.

As figure 5.3 shows, these different formats contain several fields:

- the Send sequence Number (N(S)) field; this is only send by I format frames and encodes the send sequence number of the current frame modulo 8.
- The Receive sequence Number (N(R)) field; this is send by I and S format frames and encodes the *expected* send sequence number of the next received frame modulo 8. It positively acknowledges receiving all I format frames up to and including N(R)-1.
- The Poll/Final bit (P/F) bit; is send by all frames and functions either as a Poll bit when send in a command frame, or as a Final bit when send in a response frame. A poll bit set to '1' triggers the creation of a response frame (either S or U formatted) in the receiving entity with the final bit set to '1'.
- The Supervisory function (S) and Unnumbered function (U) bits encode certain command messages like *receive ready command (RR)* or *disconnect command (DISC)*. The definition of these commands can be found in [42].

Because the N(S) and N(R) fields are encoded modulo 8, the maximum number of outstanding unacknowledged frames is seven. However the GSM specification limits this maximum number to one in almost all cases. This increases the need for the A-type frames, who do not transport any layer 3 information, but can serve to acknowledge the reception of frames.

Trace 5.1: CC Call Connect

```
HEX l2_data_out_B:194 Format B DATA (down)
000: 03 24 09 83 07 2b 2b 2b  - 2b 2b 2b 2b 2b 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
     0: 03 -------1 Extended Address: 1 octet long
     0: 03 ------1- C/R: Command
     0: 03 ---000-- SAPI: RR, MM and CC
     0: 03 -00----- Link Protocol Disciminator: GSM (not Cell Broadcasting)
     1: 24 -------0 Information Frame
     1: 24 ----010- N(S), Sequence counter: 2
     1: 24 ---0---- P
     1: 24 001----- N(R), Retransmission counter: 1
     2: 09 -------1 EL, Extended Length: y
     2: 09 ------0- M, segmentation: N
     2: 09 000010-- Length: 2
     3: 83 1------- Direction: To originating site
     3: 83 -000---- 0 TransactionID
     3: 83 ----0011 Call control. call related SS messages
     4: 07 00------ Send Sequence Number: 0
     4: 07 --000111 Call Connect
```

The Call Connect message tells the MS that the called party has picked up its phone. When the MS responds with a Connect Acknowledge message (trace 5.2), the current channel will be used to transmit voice data back and forth between the callers. The message is transmitted in an I frame, asking for acknowledgment, Poll bit.

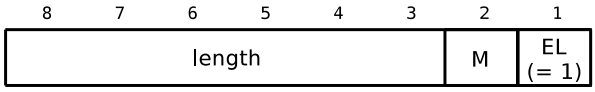


Figure 5.4: Layer 2 length field format

5.2.3 The length field

The length field encodes the length of the Information Elements (IEs); N201. It contains three fields:

- the Extension bit (EL); works like the EA in the address field. It is used to allow the control field to span more octets. When this bit is set to ‘0’ it indicates the control field continuing in the following octet. When set to ‘1’ it indicates the last octet of the control field. In the current GSM standard there exist no layer two messages with a control field larger than a single octet.
- The More data bit (M); this bit is used to indicate segmentation of a layer 3 message on the data link layer. If the M bit is set to ‘1’ this indicates that the IEs of this message contain only a part of a layer 3 message. If the M bit is set to ‘0’, this either indicates that this message contains an entire layer 3 message, if the previous frame also had a control message with the M bit set to ‘0’, or it indicates that this message contains the last part of a layer 3 message, if the previous frame had a control field with the M bit set to ‘1’. Segmented layer 3 messages always use I frames for transport. In all other frames the M bit is set to ‘0’.

- The length; this is a six bit field, that actually encodes the length of the information elements.

Remember that the total length of every frame is 23 octets or 184 bits. The length field just shows the number of octets that contain useful information to deliver to the third layer.

Trace 5.2: CC Call Connect Acknowledge

```
HEX l2_data_out_B:194 Format B DATA (up)
000: 01 01 08 03 0f 2b 2b 2b  - 2b 2b 2b 2b 2b 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
     0: 01 -------1 Extended Address: 1 octet long
     0: 01 ------0- C/R: Response
     0: 01 ---000-- SAPI: RR, MM and CC
     0: 01 -00----- Link Protocol Disciminator: GSM (not Cell Broadcasting)
     1: 01 ------01 Supvervisory Frame
     1: 01 ----00-- RR Frame (Receive ready)
     1: 01 ---0---- Poll/Final bit (P/F)
     1: 01 000----- N(R), Retransmission counter: 0
     2: 08 -------0 EL, Extended Length: n
     2: 08 ------0- M, segmentation: N
     2: 08 000010-- Length: 2
     3: 03 0------- Direction: From originating site
     3: 03 -000---- 0 TransactionID
     3: 03 ----0011 Call control. call related SS messages
     4: 0f 00------ Send Sequence Number: 0
     4: 0f --001111 Connect Acknowledge
```

The acknowledge message belonging to the earlier CC Call Connect message (5.1). Transmitted in a supervisory frame. The N(R) counter is almost always zero, because the GSM specification sets the maximum outstanding message window to 1, instead of the possible 7.

As you can see the connect message and this acknowledgment are very similar in contents.

Chapter 6

Um layer 3

The third layer of the Um-protocol is specified by the ETSI in [43, 44]. The third layer is often divided into three sublayers, of which the last is again divided into three sublayers:

1. Radio Resource management (RR); this concerns the configuration of the logical and physical channels on the air-interface.
2. Mobility Management (MM); for subscriber authentication and maintaining the geographical location of subscribers (mobility management).
3. Connection management (CM); consists of three sublayers itself:
   (a) Supplementary Services (SS); manages all kinds of extra services that are not connected to the core functionality of GSM.
   (b) Short Message Service (SMS); the handling of the SMS messages.
   (c) Call Control (CC); creating and ending telephone calls.

Figure 6.1 gives an overview of the GSM protocol stack. Most of the layers in the BSS and NSS will not been discussed in this thesis, but it does show how all the layers of the Um interface are built upon each other, and how the several layers are defined between different entities. The first two layers of the Um-protocol as well as some of the RR functionality end at the BTS. The rest of the RR sublayer ends at the BSC. The RR sublayer is responsible for the channel management of radio channels, so it does not need to go beyond the BSC. When a “physical” connection between MS and BTS has been established through RR functionality, a MM connection can be opened on top of that. A CM connection on its turn can be created on top of a MM layer (e.g. via the CM Service Request, trace 6.2).

On figure 6.1 the MM layer is defined between the MS and MSC but since it passes transparently on to the VLR the MM sublayer is often said to be defined between MS and VLR.

The RR, MM and CC sublayers are the most interesting sublayers of the third layer. The SMS sublayer is very similar to the CC sublayer (with the CC sublayer being more extended) and the SS sublayer is not involved in any core functionality. Therefore the RR, MM and CC sublayers will be discussed in detail further along in this chapter, but first we will look into the frame structure of layer 3 messages.

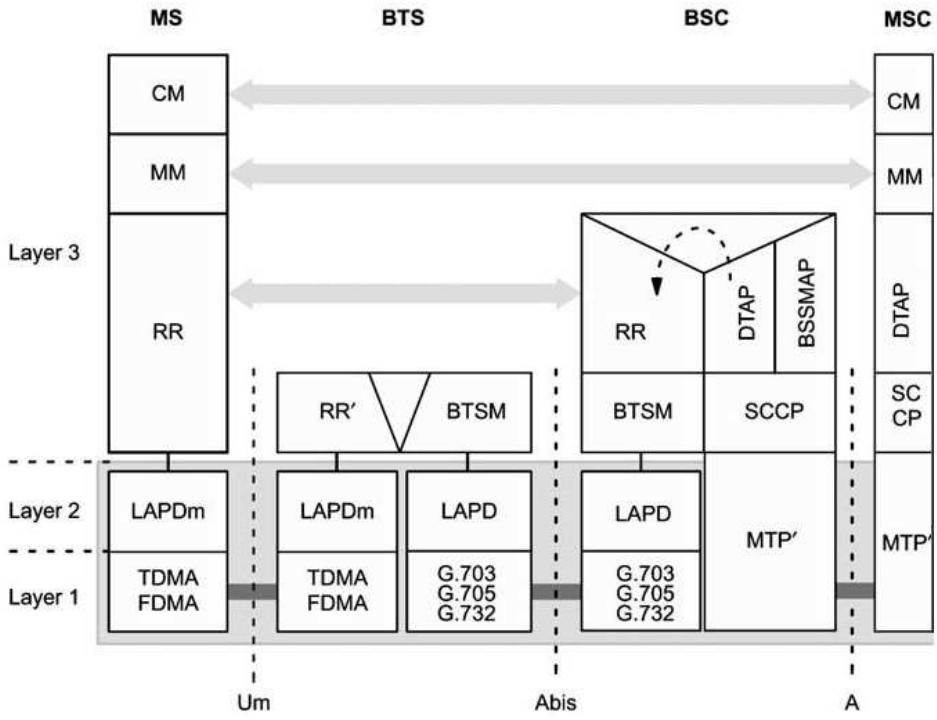


Figure 6.1: Overview of the GSM protocol stack. Figure adapted from [45]

6.1 Layer 3 frames

The frames used on the third layer of the Um protocol do not have a fixed size, as opposed to layer 2 frames. They do however consist of a two byte header, possibly followed by data bits. The structure of a layer 3 frame is shown in figure 6.1.

6.1.1 Layer 3 frame header

The header of a layer 3 frame consists of two byte-sized, separate fields; the Type ID octet and the Message Type octet. These are shown in figures 6.3 and 6.4 respectively.

The layout of the Type ID octet is dependent on the sublayer that uses the frame. However the four least significant bits always contain the Protocol Discriminator (PD). The PD identifies the layer three protocol to which the entire frame belongs. The possible values of the PD and their meaning can be found in table 6.1. The PD coded as “1110” is currently unused withing the Um protocol. It is a reserved sequence for possible future use, indicating the other four bits of this octet also belong to the PD field. This case, at the moment unused, is the only way for the other four bits of the Type ID octet to deviate from the way shown in figure 6.3.

The four most significant bits of the Type ID octet differ whether the message belongs to the CM (figure 6.3(a)) or the RR or MM sublayer (figure 6.3(b)). When a message belongs to the RR or MM sublayer these four bits are coded as “0000”, called the skip indicator, and serve no function. When the message belongs to the CM sublayer these four bits encode the Transaction Identifier (TI). The Transaction Identifier (TI) allows for distinguishing several simultaneous transactions for a single MS. It is divided into the TI flag (the most significant bit) and the TI Value (the other three bits). The TI

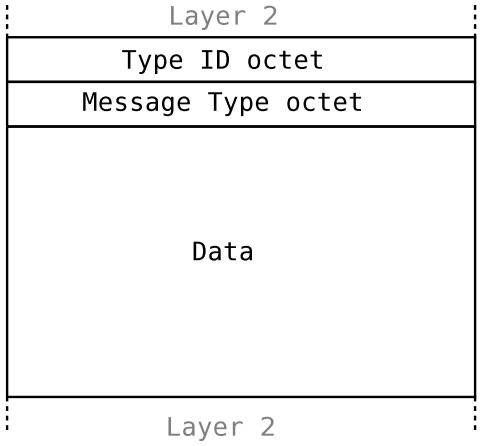
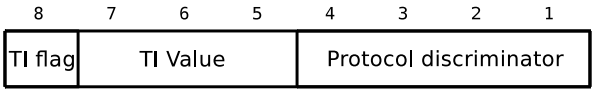


Figure 6.2: A general Um layer 3 frame



(a) CM format

(b) RR and MM format

Figure 6.3: Layer 3 Type ID octet format

flag distinguishes between the initiating (coded ‘0’) and responding side (coded ‘1’) of a transaction. The TI value is a unique number between 0 and 6 (“111” is again reserved for possible future use) for this transaction, within this SAPI and PD. When a new transaction is started the initiating side chooses a number between 0 and 6, that is not yet used within the current Um connection, with the same SAPI and PD, and sets the TI Value to this number. The responding side will use the same number when responding to this message, with the TI flag set to zero. After the transaction ends, the TI value is released. So every ME can have seven self-initiated transactions open on the Um protocol within every PD category at any given time, and respond to another seven initiated by the network.

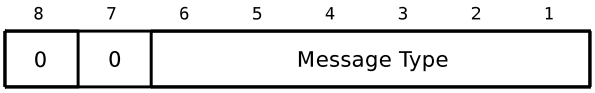
The second byte of a standard layer 3 message is called the Message Type (MT) octet. The Message Type is detailed in figure 6.4. Regrettably the MT octet contains a similarly named Message Type field, which may cause some confusion. When we speak of the MT field, it should always mean the 6 bit field inside the MT octet. Where the Type ID octet shows to which sublayer and connection a message belongs, the MT octet encodes the content of the message.

The most significant bit is always coded ‘0’ and if this is a RR message then the next bit is also coded ‘0’. For CM and MM messages this bit is coded with a Send Sequence Number (SSN) modulo 2 (figure 6.4(b)). So this bit flips every CM or MM message. The only real usage for this bit lies in protocol testing.

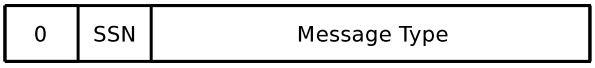
The interesting part of the MT octet lies in the Message Type field, which identifies the function

Table 6.1: Protocol Discriminators (PDs) on layer 3 of the Um interface

bits 4 3 2 1	Sub layer
0 0 1 1	Call Control (CC), call related Supplementary Services (SS)
0 1 0 1	Mobility Management (MM)
0 1 1 0	Radio Resource management (RR)
1 0 0 1	SMS messages
1 0 1 1	non call related Supplementary Services (SS)
1 1 1 0	reserved for extension of the PD to one octet length
1 1 1 1	reserved for test procedures



(a) RR format



(b) CM and MM format

Figure 6.4: Layer 3 Message Type octet format

of the message. All defined messages can be found in the specification [44]. Every possible Um-message has a MT number associated with it, which is unique within the Um-protocol (the same MT number can have different meanings on the other interfaces), the direction (the same MT number can have different meanings depending on whether it is sent by the MS or by the BTS), and the channel it was transmitted on (the same MT number can have different meanings depending on on which logical channel it was sent, e.g. on a SACCH or on a DCCH). For every message the number of possible arguments is also specified. These arguments are divided between the mandatory and the optional arguments and are included in the data bits.

6.1.2 Layer 3 frame data

The frame data is built up out of Information Elements (IEs). The mandatory arguments are always the first IEs, followed by the optional arguments. In contrast with the IEs on the second layer, the IEs in the third layer have a variable size between one and several octets. These IEs can be built up out of a Information Element Identifier (IEI), a Length Indicator (LI), and a value part. Every IE contains at least one and possibly all of these elements. The Information Element Identifier (IEI) can be seen as a type declaration. Therefore these IEs are so called Type-Length-Value encoded. All the possible combinations of the type, length and value inside an IE can be found in table 6.2.

The different types are introduced to save bits that need not be used. For instance if a mandatory IE is always of the same type, then no IEI is necessary. The same goes for the length and value.

Table 6.2: Possible formats of information elements.

Format	IEI present	LI present	Value part present
Type only (T)	yes	no	no
Value only (V)	no	no	yes
Type and Value (TV)	yes	no	yes
Length and Value (LV)	no	yes	yes
Type, Length and Value (TLV)	yes	yes	yes

The IEI field is always present in the non-mandatory IEs. In the mandatory IEs it can be left out. When the IEI is present it can either have a size of a half or one octet. When the IEI has the size of a half octet, the IE is always of the TV-type, with the other half of the octet reserved for the value.

When a Length Indicator (LI) field is present in a IE, it always consists of a single octet. It encodes the number of octets in the Value part of the IE. When a Length Indicator part is present, the IE should also contain a value part (it is either of type LV or TLV), unless the LI encodes a zero length Value part. If the Value part is present in an IE, then it can either consist of a half octet (when of type TV) or one or more octets (for types V, LV and TLV).

6.2 Radio Resource (RR)

The Radio Resource management is in fact the lowest sublayer within the third layer of the Um-protocol, managing the allocation and teardown of dedicated signaling and traffic channels that can be used by the higher layers. The RR functionality is also used on the BCCH to transmit the system information and for the paging, access requests and access grants on the Common Control Channels (CCCH). Furthermore the RR layer also handles the handover management and controls the ciphering of channels, though the actual ciphering happens on the first layer of the Um protocol.

The RR sublayer is defined between the MS and the BSC, though when concerning handovers, some messages will reach the MSC.

Some of the often used RR messages are defined in table 6.3. The names of these messages have a couple of letters in upper case which make up the commonly used abbreviations for the messages. All the RR messages can be found in [44].

6.3 Mobility Management(MM)

The Mobility Management sublayer uses the control channels provided by the RR sublayer to manage the mobility of subscribers. Or, more accurately, manage the mobility information; for every ME/IMSI the code of the BTS it is currently connected to is stored. This code, the CGI represents each cell within a GSM network, and this information is stored inside the serving VLR.

The MM sublayer is used for location updates and paging control. It also handles the authentication of MSs. Basically the MM sublayer handles most traffic whenever a TMSI / IMSI is involved, or a cell identity number.

Table 6.3: Example RR message definitions

MT field	Name	Direction	Description
-	CHANnel REQuest	MS → BTS	This message requests a channel for further communication in channel setup (section 4.5.2). It has no MT field code, because this message is only transmitted via access bursts (section 4.3). In only a single octet this message encodes both the reason for the channel request - e.g. “emergency call” or “answer to paging” - and a random reference which the BTS uses in its reply to this message.
110101	CIPHer MODe CoMmanD	BTS → MS	Command to start encryption on the Um interface. This command contains the algorithm (A5/X) that should be used and whether the CIPHer MODe COMplete reply should contain the International Mobile Equipment Identity and Software Version (IMEISV).
111111	IMMediate ASSignment CoMmanD	BTS → MS	This message is always transmitted on the AGCH. This message can use a lot of parameters to assign a channel to the MS. These parameters include: the random reference taken from a channel request message, the ARFCN and time-slot of the dedicated channel, the final variables needed for frequency hopping (HSN, MA and MAIO, section 3.1.2) and the timing advance - a number representing the time delay between transmission and reception, so the MS can adjust its transmissions accordingly.

To this end the MM sublayer is defined between the MS and the VLR. Some of the often used MM messages are shown in table 6.4. The letters of the message names that are written in upper case compose the commonly used abbreviations for these messages. All the MM messages can be found in [44].

6.4 Call Control (CC)

The Call Control sublayer manages the actual calls made on the GSM network. It uses an MM connection to a MS to manage call establishments and teardowns to it. This sublayer is very similar to the ITU-T Recommendation Q.931, which is ISDN’s connection control protocol.

The CC sublayer is defined between the MS and the MSC.

Some of the often used CC messages are shown in table 6.5. The letters of the message names that are written in upper case compose the commonly used abbreviations for these messages. All the CC messages can be found in [44].

Table 6.4: Example MM message definitions

ID	Name	Direction	Description
001000	LOCation UPDate REQuest	MS → BTS	Request for a location update service (sections 2.5.2 and 6.5.1). This request contains a parameter which identifies whether this is a periodic or normal location update or if it is a sign-on procedure. This request also contains the old LAI and MS parameters like the IMSI/TMSI and the supported ciphers.
010010	AUTHentication REQuest	BTS → MS	The authentication command contains the random challenge (RAND) and the CKSN - the identifier for the resulting session key. Authentication is described in sections 2.5.1 and 7.1.
011010	TMSI REALlocation CoMmanD	BTS → MS	At the end of a location update the MS should be assigned a new TMSI. This command assigns the new value, although the specification does allow for the TMSI reallocation command to contain the IMSI, instead of a new TMSI.

6.5 Scenarios

We will now look at two scenarios in detail: the location registration - which is essentially the same as a location update - and a Mobile Originating Call (MOC) setup. Both scenarios are illustrated with an message sequence chart (figure 6.5). Above each message arrow the channel name and the layer 3 sublayer where this message occurs are named. In some rare cases “ack” also appears above the message arrow, which denotes that this message purely acts as an acknowledgment on the second layer of the Um interface. In the general case however a message is automatically acknowledged by the reception of the next message shown in the chart. Below the message arrows the actual content of the message is placed. These contents are shown by their ETSI defined abbreviations and when relevant they are followed by the most important parameters between brackets. When parameters is denoted as “X” then this means the actual value X, as opposed to, for instance IMSI denoting the IMSI value of this specific MS.

These scenarios show possible ways in which these message exchanges could run. A practical situation might differ, though the occurrence of messages shown here is a likely course. Both scenarios start with a channel setup phase. The Location registration scenario also shows an authentication and identification phase, which are both optional. A network can decide to perform either phase when it wishes.

6.5.1 Location registration

The location registration is a location update in which the MS uses its IMSI to identify itself with instead of its TMSI. So effectually this scenario shows a sign-on procedure. Changing every occurrence of the IMSI for TMSI will make this a general location update scenario. The use and general idea of the location update has been discussed earlier in section 2.5.2.

The location registration is shown in figure 6.5(a). The first two message show the channel setup phase, in which the MS signals the BTS on the RACH a “CHANnel_REQuest”. This request contains the reason for the request (a location update) and a reference number, which the network uses in its

Table 6.5: Example CC message definitions

ID	Name	Direction	Description
000001	ALERTing	MS ↔ BTS	The alerting message is transmitted by the MS in a MTC to indicate that the ME has started signaling the user of the incoming call (ringing). The alerting message is also transmitted by the network to the MS in a MOC to indicate that the called user alerting has been initiated (dial tone). The MTC and MOC procedures are discussed in sections 2.5, 4.5.3 and 6.5.2.
000101	SETUP	MS ↔ BTS	In a MOC the setup message is transmitted from the MS to the BTS. In a MTC the setup message is transmitted from the BTS to the MS. It contains many parameters, but most importantly it contains the number of the caller / callee (MSISDN) and the kind of TCH connection requested/used.
000111	CONnect	MS ↔ BTS	This message is transmitted by the MS in a MTC and it indicates that the user accepted the call. In a MOC the message is transmitted by the BTS to indicate that the connection was successfully established.

"IMMediate_ASSignment_CoMmanD", so the MS can see this assignment is meant for him. The immediate assignment command assigns a SDCCH channel to the MS by giving it the ARFCN and time-slot of the reserved channel. The channel setup happens at the Radio Resource management (RR) sublayer.

The MS then tunes to the SDCCH channel and transmits a request. The request on this channel is seen by the BTS as the acknowledgment of the immediate assignment. The request in this case is a "LOCation_UPDate_REQuest" containing the IMSI and old LAI of the MS. This request is a Mobility Management (MM) request. It is acknowledged by the network through a layer 2 acknowledge frame, which completes the MM sublayer hand shake.

Then an authentication phase starts, initiated by the network. The network can always choose to perform an authentication, though if the MS is already authenticated, this is not obliged. Authentication is a MM sublayer task.

In the original location update request to the network the MS already gave a list of the encryption algorithms it supports. The network decides that this communication needs to continue encrypted, so it issues the "CIPHer_MODe_CoMmand" with the encryption algorithm to use. Upon correct reception of this command the MS starts ciphering all its transmissions and deciphering all the messages it receives. The BTS in the meantime has started deciphering all messages it receives on this channel, but the BTS only starts to encipher its transmission when it receives the "CIPHer_MODe_COMplete" message. From this moment on all communication between this MS and BTS on this channel are encrypted. Remember that although the cipher command and response are RR sublayer messages, the encryption happens on the first layer of the Um protocol.

Once encryption has started the network can choose to request identity numbers of the MS. The "IDENTity_REQuest" asks for specific identifiers, in this case the IMEI, which the MS provides in its response.

The network then assigns a new TMSI to the MS with the "TMSI_REALlocation_CoMmanD". The MS signals it has correctly received the new TMSI via the "TMSI_REALlocation_COMplete"

message. After which the network ends the location update procedure by transmitting the “LOCation_UPDate_ACCept” message.

6.5.2 Mobile Originating Call (MOC)

Figure 6.5(b) shows a Mobile Originating Call (MOC) setup using early assignment. MOCs and early assignment where discussed earlier in sections 2.5.3 and 4.5.3.

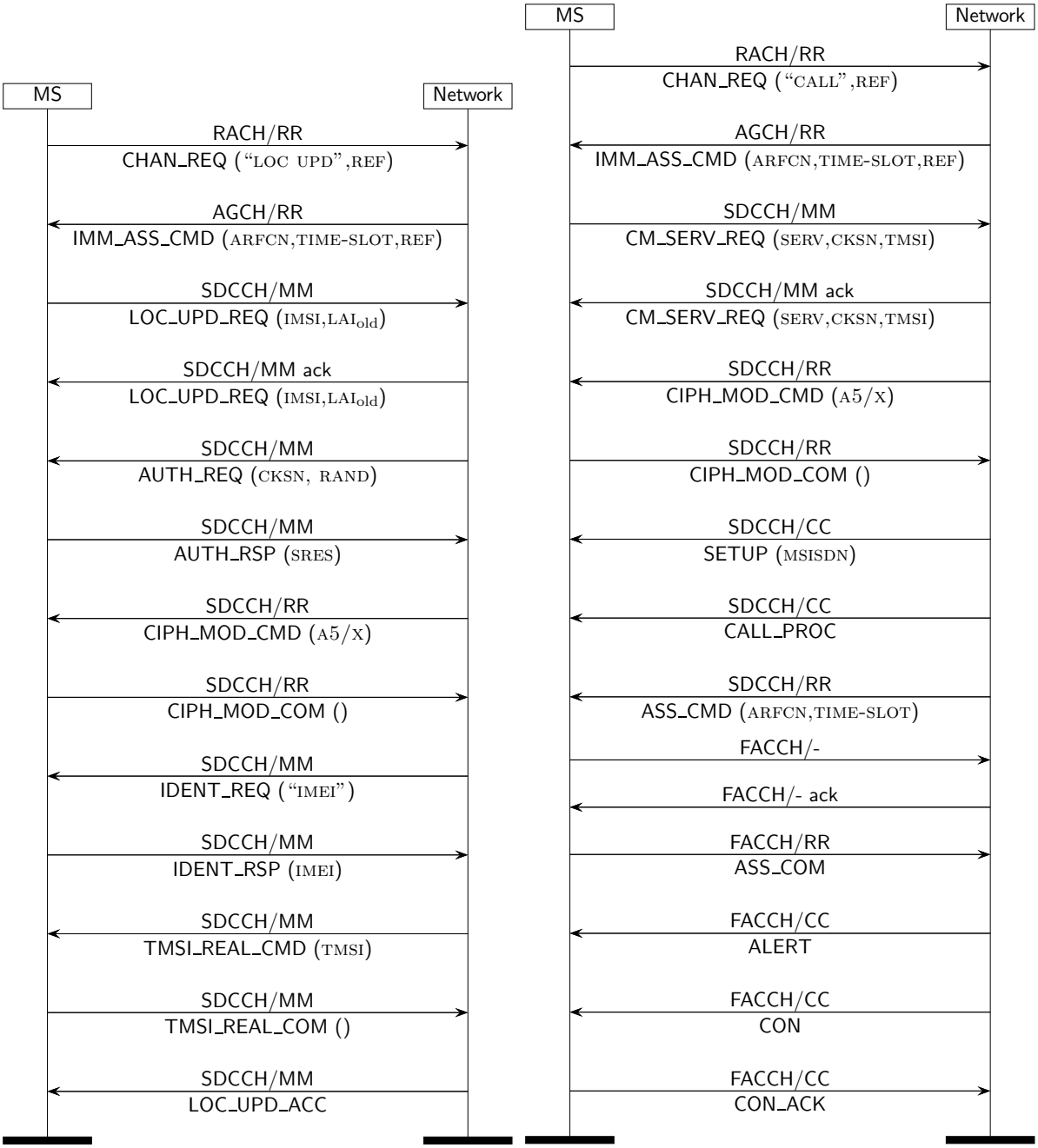
The subscriber wants to make a call and thus the MS signals the network for a channel, in the channel setup phase. This is identical to the channel setup in the previous scenario, except for the reason the MS gives in its request.

After the channel setup the MS transmits a “CM_SERVice_REQuest” with the service that is requested - a MOC in this case, but it could also be e.g. SMS - and its current TMSI and Ciphering Key Sequence Number (CKSN). The network recognizes that the MS is already authenticated and the network also has a session key stored for this TMSI with the same CKSN, prompting the network to skip authentication and directly start ciphering. The ciphering procedure is the exact same as in the previous scenario.

After the layer two acknowledgment of the “CM_SERVice_REQuest”, there is Call Control (CC) sublayer connection. After ciphering has started on this channel the MS begins the “SETUP” by transmitting the directory number - MSISDN - the subscriber wishes to call. If the call is accepted the network replies with a “CALL_PROCeed” message.

The MS is now assigned a Traffic Channels (TCH), which first functions as a Fast Associated Control Channel (FACCH). The MS transmits an empty message once it is tuned to the correct channel. The network responds with a layer 2 acknowledge frame, .. the MS to transmit an “ASSignment_COMplete” message which fully establishes the channel. Note that although the MS and BTS have switched to a new channel the encryption settings are taken along. So all succeeding messages are still encrypted.

The network now keeps sending “ALERT” messages transmitting the ringing tone until the called phone is picked up. When that happens the network will transmit a “CONnect” message, which the MS acknowledges. From this moment on channel is used as a TCH channel and voice data can be exchanged.



(a) Location registration (b) Mobile Originating Call setup with early assignment

Figure 6.5: Scenarios showing actual message exchanges between MS and BTS

Trace 6.1: RR Immediate Assignment Command

```
HEX l2_data_out_Bbis:462 Format Bbis DATA
000: 31 06 3f 00 52 f0 ab 85  - ad e0 01 01 0f 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
     0: 31 001100-- Pseudo Length: 12
     1: 06 0------- Direction: From originating site
     1: 06 -000---- 0 TransactionID
     1: 06 ----0110 Radio Resouce Management
     2: 3f 0-111111 RRimmediateAssignment
     2: 3f -x------ Send sequence number: 0
     3: 00 ------00 Page Mode: Normal paging
     3: 00 -0------ No meaning
     3: 00 --0----- Downlink assign to MS: No meaning
     3: 00 ---0---- This messages assigns a dedicated mode resource
     4: 52 -----010 Timeslot number: 2
     4: 52 01010--- Chan. Descript.: SDCCH/8 + SACCH/C8 or CBCH (SDCCH/8)
     5: f0 111----- Training seq. code : 7
     5: f0 ---1---- HoppingChannel
     6: ab ........ Mobile Allocation Index Offset (MAIO) 2
     6: ab --101011 Hopping Seq. Number: 43
     7: 85 100----- Establishing Cause: Answer to paging
     7: 85 ---xxxxx Random Reference : 5
     8: ad xxxxxxxx T1/T2/T3
     9: e0 xxxxxxxx T1/T2/T3
    10: 01 --xxxxxx Timing advance value: 1
    11: 01 00000001 Length of Mobile Allocation: 1
    12: 0f ----1--- Mobile Allocation ARFCN #4
    12: 0f -----1-- Mobile Allocation ARFCN #3
    12: 0f ------1- Mobile Allocation ARFCN #2
    12: 0f -------1 Mobile Allocation ARFCN #1
```

This message is used to assign a control channel to the MS. It is a response to a Channel Request message from the MS on the AGCH. This command refers to that message via the Random Reference (octet 7, value 5). The BTS also computed the transmission delay from the Channel Request and sends a Timing advance value along that the MS can use to make sure its transmissions arrive in the correct time window.

This command provides a channel with a certain layout detailed in the Channel Description. It gives the MS a time-slot, a list of ARFCN numbers, the hopping sequence number and the MAIO. The T1/T2/T3 of octets 8 and 9 encode the frame number, though this is not correctly decoded by gsmdecode.

Trace 6.2: MM CM Service Request

```
HEX l2_data_out_B:194 Format B DATA (up)
000: 01 01 34 05 24 21 03 33  - 19 81 05 f4 6c 3a 01 8e
001: 2b 2b 2b 2b 2b 2b 2b
     0: 01 -------1 Extended Address: 1 octet long
     0: 01 ------0- C/R: Response
     0: 01 ---000-- SAPI: RR, MM and CC
     0: 01 -00----- Link Protocol Disciminator: GSM (not Cell Broadcasting)
     1: 01 ------01 Supvervisory Frame
     1: 01 ----00-- RR Frame (Receive ready)
     1: 01 ---0---- Poll/Final bit (P/F)
     1: 01 000----- N(R), Retransmission counter: 0
     2: 34 -------0 EL, Extended Length: n
     2: 34 ------0- M, segmentation: N
     2: 34 001101-- Length: 13
     3: 05 0------- Direction: From originating site
     3: 05 -000---- 0 TransactionID
     3: 05 ----0101 Mobile Management Message (non GPRS)
     4: 24 00------ SendSequenceNumber: 0
     4: 24 --100100 MMcmServiceRequest
     5: 21 -010---- Ciphering key sequence: 2
     5: 21 ----0001 Request Service Type: MS originated call
     6: 03 00000011 MS Classmark 2 length: 3
     7: 33 -01----- Revision Level: Phase 2
     7: 33 ---1---- Controlled early classmark sending: Implemented
     7: 33 ----0--- A5/1 available
     7: 33 -----011 RF power class capability: Class 4
     8: 19 -1------ Pseudo Sync Capability: not present
     8: 19 --01---- SS Screening: Phase 2 error handling
     8: 19 ----1--- Mobile Terminated Point to Point SMS: supported
     8: 19 -----0-- VoiceBroadcastService: not supported
     8: 19 ------0- VoiceGroupCallService: not supported
     8: 19 -------1 MS supports E-GSM or R-GSM: supported
     9: 81 1------- CM3 option: supported
     9: 81 --0----- LocationServiceValueAdded Capability: not supported
     9: 81 ----0--- SoLSA Capability: not supported
     9: 81 ------0- A5/3 not available
     9: 81 -------1 A5/2: available
    11: f4 -----100 Type of identity: TMSI/P-TMSI
    12: 6c -------- ID(4/even): 6C3A018E
```

This is an up link message by the MS indicating that it wants to make a call. This message also acts as an acknowledgment of the immediate assignment message (trace 6.1) and is the first message to pass on the dedicated SDCCH.

Among a large number of parameters are the CKSN, the encryption capabilities of the ME - in this case A5/1 and A5/2 but not A5/3 - and the MSs identity. The message is acknowledged by an nearly identical message in the down link, only the second octet shows it to be an Unnumbered Acknowledge frame.

Trace 6.3: CC Call Setup

```
HEX l2_data_out_B:194 Format B DATA (up)
000: 01 01 50 03 45 04 04 60  - 02 00 81 5e 06 81 60 **
001: ** ** ** a1 15 01 01
     0: 01 -------1 Extended Address: 1 octet long
     0: 01 ------0- C/R: Response
     0: 01 ---000-- SAPI: RR, MM and CC
     0: 01 -00----- Link Protocol Disciminator: GSM (not Cell Broadcasting)
     1: 01 ------01 Supvervisory Frame
     1: 01 ----00-- RR Frame (Receive ready)
     1: 01 ---0---- Poll/Final bit (P/F)
     1: 01 000----- N(R), Retransmission counter: 0
     2: 50 -------0 EL, Extended Length: n
     2: 50 ------0- M, segmentation: N
     2: 50 010100-- Length: 20
     3: 03 0------- Direction: From originating site
     3: 03 -000---- 0 TransactionID
     3: 03 ----0011 Call control. call related SS messages
     4: 45 01------ Send Sequence Number: 1
     4: 45 --000101 Call Setup
     5: 04 00000100 Bearer Capability
     6: 04 00000100 Length: 4
     7: 60 0------- Extension: no
     7: 60 -11----- Radio Channel: dual rate MS/full rate preferred
     7: 60 ---0---- Coding Standard: GSM
     7: 60 ----0--- Transfer Mode: Circuit
     7: 60 -----000 Transfer Capability: speech
     8: 02 0------- Extension: no
     8: 02 -0------ Compression: no
     8: 02 ----0--- Duplex Mode: half
     8: 02 ------1- Rate Req.: Data 4.8 kb/s, full rate, n. transp. 6kb req
     9: 00 0------- Extension: no
     9: 00 ---00--- Rate Adaptation: no rate adaption
     9: 00 -----000 Signalling Access Protocol: UNKNOWN
    10: 81 -------1 Asynchronous
    11: 5e FIXME: some data might be in extentions
    11: 5e 01011110 Called Party BCD Number
    12: 06 00000110 Length: 6
    13: 81 -000---- Type of number: unknown
    14: 60 -------- Number(10): 06********
    19: a1 10100001 CLIR supression
    20: 15 00010101 FIXME
    21: 01 XXXXXXXX UNKNOWN DATA (2 bytes)
    21: 01 YYYYYYYY REST OCTETS (2)
```

This CC Call Setup message is transmitted by the MS to the network and contains the number the user wishes to call. The '*' tokens in the trace where placed to hide the telephone number that was called. As you can see in the final octets, the gsmdecode still has a few bugs.

Trace 6.4: RR Assign Command

```
HEX l2_data_out_B:296 Format Bbis (RR, MM or CC)
000: 06 2e 0f f0 eb 05 62 00  - 00 00 00 20 00 10 10 00
001: 00 00 00 00 01 00 00 63  - 21 72 01 0f
     0: 06 0------- Direction: From originating site
     0: 06 -000---- 0 TransactionID
     0: 06 ----0110 Radio Resouce Management
     1: 2e 00101110 RR Assign Command
     2: 0f -----111 Timeslot number: 7
     2: 0f 00001--- Channel Description: TCH/F + ACCHs
     3: f0 111----- Training seq. code: 7
     3: f0 ---1---- HoppingChannel
     4: eb ........ Mobile Allocation Index Offset (MAIO) 3
     4: eb --101011 Hopping Seq. Number: 43
     5: 05 ---00101 Power Level: 5
     6: 62 UNKNOWN. FIXME
     6: 62 XXXXXXXX UNKNOWN DATA (22 bytes)
     6: 62 YYYYYYYY REST OCTETS (22)
```

The RR Assign Command which switches the call setup from the current SDCCH to a TCH. The GSM network used early assignment for call setup (Section 4.5.3) so this message was transmitted after a call proceeding message (trace 4.2), but before the alert messages (trace 4.3).

This message looks a lot like the immediate assignment (trace 6.1) and functions similarly. However immediate assignment messages are only transmitted on the AGCH to assign a dedicated control or traffic channel, while assign messages occur only on dedicated channels.

Again you can see the not always perfect decoding abilities of Gsmdecode.

Trace 6.5: RR Assign Complete

```
HEX l2_data_out_B:194 Format B DATA (up)
000: 01 01 0c 06 29 00 2b 2b  - 2b 2b 2b 2b 2b 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
     0: 01 -------1 Extended Address: 1 octet long
     0: 01 ------0- C/R: Response
     0: 01 ---000-- SAPI: RR, MM and CC
     0: 01 -00----- Link Protocol Disciminator: GSM (not Cell Broadcasting)
     1: 01 ------01 Supvervisory Frame
     1: 01 ----00-- RR Frame (Receive ready)
     1: 01 ---0---- Poll/Final bit (P/F)
     1: 01 000----- N(R), Retransmission counter: 0
     2: 0c -------0 EL, Extended Length: n
     2: 0c ------0- M, segmentation: N
     2: 0c 000011-- Length: 3
     3: 06 0------- Direction: From originating site
     3: 06 -000---- 0 TransactionID
     3: 06 ----0110 Radio Resouce Management
     4: 29 0-101001 RR Assign Complete
     5: 00 00000000 RR-Cause (reason of event) = Normal event
```

The acknowledgment of the RR Assign Command (trace 6.4). The MS has been able to tune to the new channel and this message is the first message to pass on the new channel. In this case it is a traffic channel (TCH), but until the conversation begins it is used as a Fast Associated Control Channel (FACCH).

Keep in mind that the messages remain encrypted although the connection was moved to an other channel.

Chapter 7

Encryption

This chapter will explain the uses of encryption in the GSM air interface and will look in detail into some of the encryption algorithms used. Some of the weaknesses of these algorithms will be discussed in chapter 8.

GSM networks can offer several security features through encryption. The most important of these are *authorized network usage* and *call confidentiality*. For these goals three generic algorithms are defined:

- A3, used to generate a response (SRES) from the secret key K_i and the challenge (RAND).
- A8, used to generate the session key (K_c) from the secret key K_i and the challenge (RAND).
- A5, used to generate keystream from the session key (K_c) and the current TDMA frame number (FN).

These names refer to generic algorithms that can be instantiated with practical algorithms that conform to the specifications set for these algorithms [46].

7.1 Authentication

Authentication of a MS to the network is mostly achieved by a challenge/response (for which the A3 algorithm is needed) and because both parties need to be able to compute the same session key for encryption (the A8 algorithm). The exact message exchange was already discussed in section 6.5.1; as a brief, and simplified recap see figure 7.1. The implementations of the A3 and A8 algorithms for every user exist only in two places; the SIM of the user and the providers' Authentication Centre (AuC). Both of these are controlled by the provider. So providers are free to choose the actual implementations of the A3 and A8 algorithm, as long as they follow the contracts of A3 and A8. For A3 this means that given two arguments, a 128 bit K_i and a 128 bit random (RAND) yields a 32 bit Signed Response (SRES). For A8 this means that given two arguments, a 128 bit K_i and a 128 bit random (RAND), result in a 64 bit session key (K_c). Both A3 and A8 should be implemented by cryptographic hash functions. Even though providers are free to choose their own implementation nearly all providers opted to use the COMP128 algorithm, suggested by the ETSI for both A3 and A8.

COMP128 was an example design by the ETSI given in a memorandum of understanding. It is a proprietary hash function that was kept confidential. In 1998 it was reverse engineered by Briceno,

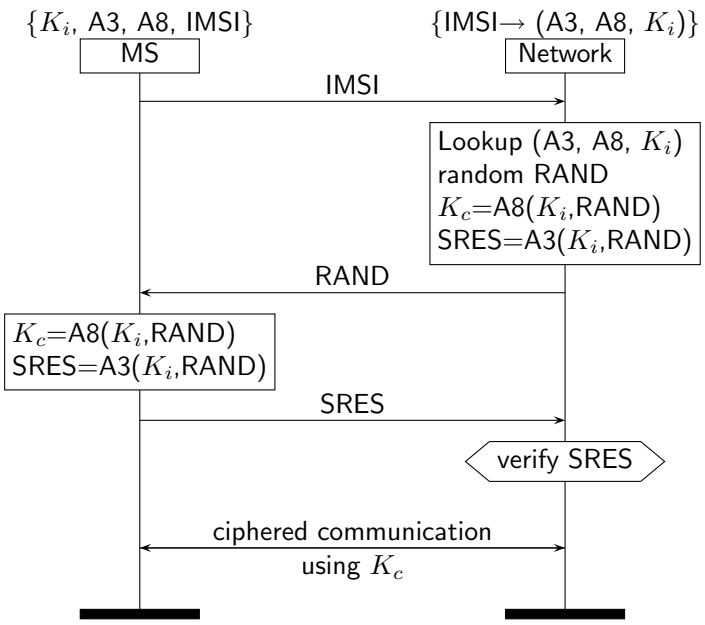


Figure 7.1: Simplified authentication of a MS to the network.

Goldberg, and Wagner [9], using a leaked document [5] and an actual GSM phone. Some improvements were introduced into a newer version of COMP128: COMP128v2, retrospectively renaming the original algorithm to COMP128v1. The second version is also kept secret, as are possible other A3/A8 algorithms. Currently the designs of the A3/A8 algorithms used in newer SIM cards is unknown.

7.1.1 COMP128v1

COMP128 performs both the A3 (response calculation) and the A8 (session key calculation) in a single algorithm. It is a cryptographic hash function that receives two 128 bit arguments (the secret key K_i and the challenge RAND), which results in the response (SRES) concatenated with the pseudo session key (K_c'):

$$\text{COMP128}(\text{RAND}, K_i) = (\text{SRES} \| K_c')$$

where || denotes concatenation.
However the SRES and K_c' are not concatenated directly. COMP128v1 results in a 128 bit binary number. Of this result the first 32 bits are the signed response (SRES) and only the final 54 bits are used as K_c'. We call this a pseudo session key, because the actual session key needed in the A5 algorithms is 64 bits long. Ten zeros are added to the end of K_c' to make the actual session key K_c. It is unknown if this also happens in COMP128v1's successors.

COMP128v1 is a hash function with 40 rounds, where each round consists of a table look up followed by mixing. There are five different tables, one used for each consecutive round, which is repeated eight times. Source code of the entire COMP128v1 function can be found at [9].

7.2 Confidentiality

To achieve call confidentiality phone calls are encrypted on the Um interface (chapters 3 and 4), using an A5 encryption algorithm. Since the encryption only happens on layer 1 of the Um interface, only the air interface between the BTS and the MS is protected. Unlike the A3 and A8 algorithms, the A5 encryption is programmed into the phone hardware (ME) and the cell towers (BTSs), and therefore there is only a choice between three pre-defined algorithms; A5/1, A5/2 and A5/3. These algorithms encrypt the payload in the bursts (section 4.3) transmitted on the first layer of the air-interface, under the session key (K_c) and the current frame number. It is worth noticing that there is a fourth option, namely using no encryption. A BTS decides whether to use encryption, and if so which algorithm to use, and sends this to the MS over a DCCH.

A5/1 was the original encryption algorithm used in GSM. It was introduced in 1987, but at that time is was an export restricted encryption algorithm. So when GSM grew beyond Europe, a modified (and actually weakened) version was created: A5/2.

Originally the internal designs of A5/1 and A5/2 was kept secret. It was only disclosed to GSM manufacturers under an NDA. However in 1999 Marc Briceno reverse engineered the design of both A5/1 and A5/2 from a GSM phone [7]. Both algorithms are stream ciphers, generating keystream from the current frame number and the session key (K_c) which is XOR-ed with the plain text.

In 2002 an additional A5 algorithm was introduced: A5/3. Unlike with its predecessors, the internal designs of A5/3 where immediately published [47]. It was based on the block-cipher KASUMI, which was already used in third generation networks, and which in turn was based on the block-cipher MISTY (KASUMI is the Japanese word for “mist”). A5/3 is currently considered unbroken and the best cryptographic alternative in GSM. A5/3 differs from its predecessors in yet another way; it uses a 128 bit session key. This is the standard in third generation mobile communication networks, but in GSM the session key provided by the A8 algorithm is defined as 64 bits. How these 64 bits lead to the 128 bit session key is currently unknown, but it seams unlikely that the session key in A5/3 will have more than 64 bits of entropy.

In January of 2010, Dunkelman, Keller and Shamir published a new attack on KASUMI reducing the time-complexity down to 2^{32} (the previous best attack had a time-complexity of 2^{76}) [48]. This new attack is a related key attack and needs chosen plain text messages, making it impractical as an attack against GSM or the third generation networks. However this theoretical break is still worrying considering that the designers of KASUMI payed specific attention to related key attacks. Also this new attack does not break MISTY in any way, indicating that the changes made for KASUMI actually weakened the cipher significantly.

7.2.1 A5/1

The GSM encryption is used to encrypt bursts over the Um interface. These bursts contain 114 bits of plain text (section 4.3). For each burst A5/1 generates 228 bits of keystream. The last 114 bits of this keystream are XOR-ed with the plain text, in effect encrypting it into cipher text. The other 114 bits are used to decrypt an incoming burst, by XORing the keystream with the cipher text, yielding the plain text layer 1 burst.

The internal design of A5/1 contains three Linear Feedback Shift Register (LFSR)s of different size, named R1, R2 and R3, as is shown in figure 7.2. These registers achieve irregular clocking through their clock bits. At the end of every step the three clock bits (bit 8 for R1 and bit 10 for R2

and R3) are compared to find the value of the majority of these three bits. Those registers with a clock bit that is equal to this majority will clock. So at each step two or all three of the registers will clock and each register will clock with a chance of $\frac{3}{4}$.

When a register clocks its tap bits (for R1 numbers 13, 16, 17, and 18, for R2 numbers 20 and 21 and for R3 numbers 7, 20, 21 and 22) are XOR-ed and the resulting bit value is input at index 0 of the specific register. All bits inside the register move on to the next index. The highest number bits come out the registers and are XOR-ed together to form a keystream bit.

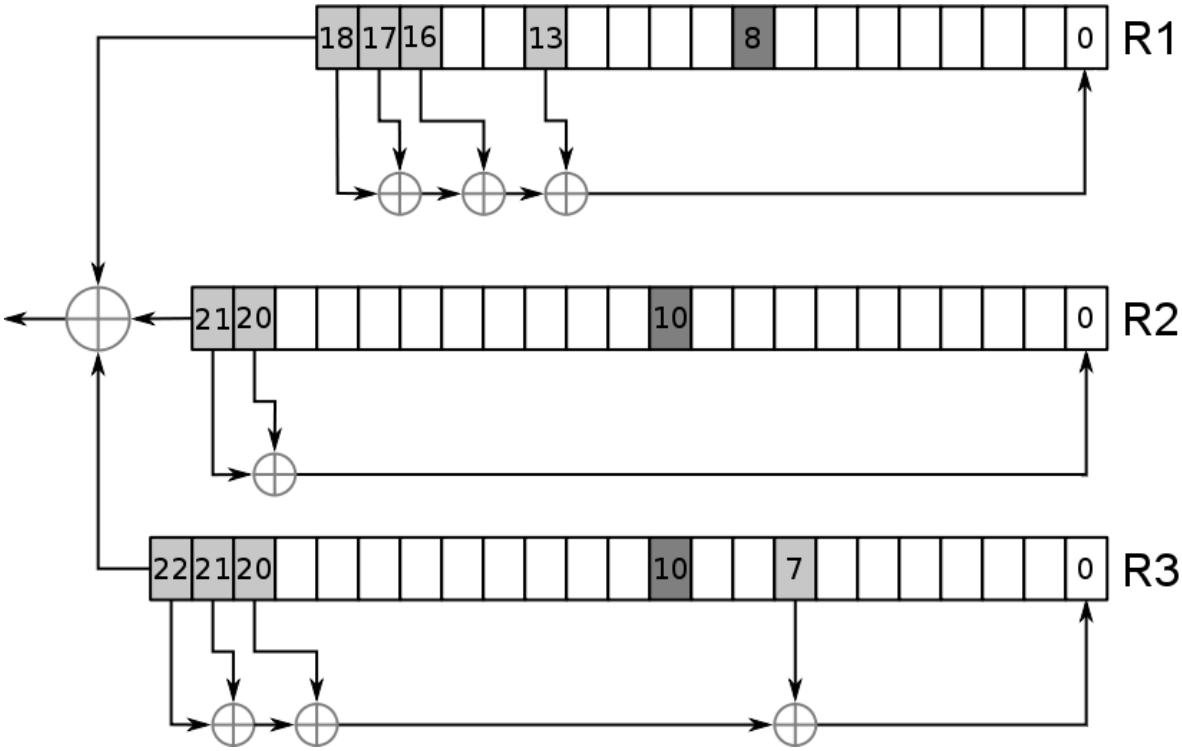


Figure 7.2: Diagram of the internal design of the A5/1 stream cipher.

The internal state of the A5/1 cipher needs to first be instantiated with two variables; the session key (K_c) and the current frame number. The session key is a private key that both MS and network can compute (section 7.1.1) and the Frame Number (FN) is a public key that denotes the current TDMA frame within the Hyperframe (section 4.1). The frame number used in the A5/1 setup is encoded by 22 bits number we will call f. f consists of three fields T1, T2 and T3. T1 is encoded by 11 bits, T2 by 5 and T3 by 6. The definitions are given by the following equations:

$$\begin{aligned} f \quad (22 \text{ bits}) &= \quad T1||T3||T2 \\ T1 \quad (11 \text{ bits}) &= \quad \text{FN div } (51 \times 26) \\ T2 \quad (6 \text{ bits}) &= \quad \text{FN mod } 26 \\ T3 \quad (5 \text{ bits}) &= \quad \text{FN mod } 51 \end{aligned}$$

Where || denotes a concatenation.
T1, T2 and T3 denote the frame number of the current TDMA frame modulo the size of control

and traffic multiframes and superframes respectively. Note that if the encryption happens on a traffic channel, then T2 denotes the position of the TDMA frame inside the multiframe, if it happens on a signaling channel than this index is stored in T3. Frame structures where discussed in section 4.1.

The A5/1 algorithm runs as follows:

1. Set all the registers to '0'
2. For $i = 0$ to 63:
   - clock all three registers
   - R1[0] $\leftarrow$ R1[0] $\oplus$ $K_c[i]$; R2[0] $\leftarrow$ R2[0] $\oplus$ $K_c[i]$; R3[0] $\leftarrow$ R3[0] $\oplus$ $K_c[i]$.
3. For $i = 0$ to 21:
   - clock all three registers
   - R1[0] $\leftarrow$ R1[0] $\oplus$ $f[i]$; R2[0] $\leftarrow$ R2[0] $\oplus$ $f[i]$; R3[0] $\leftarrow$ R3[0] $\oplus$ $f[i]$.
4. Run for 100 rounds using majority clocking.
5. Generate 114 bits of keystream used to decipher / encipher the next *downlink* package.
6. Generate 114 bits of keystream used to decipher / encipher the next *uplink* package.
7. Start again at step 1 with the same K_c and a new (current) f.

Steps 1 to 4 show the initialization steps of the A5/1 algorithm. First the session key is clocked in, then the frame number follows. These initialization parameters are clocked in 'normally'. That is to say all three will clock simultaneously for every bit of the frame number and session key. After that the initialization phase is ended by clocking one hundred times using the majority clocking explained above.

The internal state is then ready to produce the keystream. In steps 5 and 6 the registers clock irregularly for 228 bits. This results in 114 bits of keystream used for encryption and 114 bits used for decryption. This handles the encryption of the next burst to transmit and the decryption of the first burst to receive. After that the entire process repeats itself for the next bursts in the next TDMA frame.

Trace 7.1: RR Cipher mode command

```
HEX l2_data_out_B:194 Format B DATA (down)
000: 03 20 0d 06 35 01 2b 2b  - 2b 2b 2b 2b 2b 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
     0: 03 -------1 Extended Address: 1 octet long
     0: 03 ------1- C/R: Command
     0: 03 ---000-- SAPI: RR, MM and CC
     0: 03 -00----- Link Protocol Disciminator: GSM (no CBCH)
     1: 20 -------0 Information Frame
     1: 20 ----000- N(S), Sequence counter: 0
     1: 20 ---0---- P
     1: 20 001----- N(R), Retransmission counter: 1
     2: 0d -------1 EL, Extended Length: y
     2: 0d ------0- M, segmentation: N
     2: 0d 000011-- Length: 3
     3: 06 0------- Direction: From originating site
     3: 06 -000---- 0 TransactionID
     3: 06 ----0110 Radio Resouce Management
     4: 35 00110101 RR Cipher Mode Command
     5: 01 ----000- Cipher: A5/1
     5: 01 -------1 Start ciphering
     5: 01 ---0---- Cipher Response: IMEISV shall not be included
```

The Cipher mode command tells the MS to begin ciphering on the Um-interface. It is a Radio Resource message and it contains two parameters - the algorithm to use for ciphering, in this case A5/1, and whether the response (the Cipher mode complete message, trace 7.2) should contain the IMEISV.

Trace 7.2: RR Cipher Mode Complete

```
HEX l2_data_out_B:194 Format B DATA (up)
000: 01 01 08 06 32 2b 2b 2b  - 2b 2b 2b 2b 2b 2b 2b 2b
001: 2b 2b 2b 2b 2b 2b 2b
     0: 01 -------1 Extended Address: 1 octet long
     0: 01 ------0- C/R: Response
     0: 01 ---000-- SAPI: RR, MM and CC
     0: 01 -00----- Link Protocol Disciminator: GSM (not Cell Broadcasting)
     1: 01 ------01 Supvervisory Frame
     1: 01 ----00-- RR Frame (Receive ready)
     1: 01 ---0---- Poll/Final bit (P/F)
     1: 01 000----- N(R), Retransmission counter: 0
     2: 08 -------0 EL, Extended Length: n
     2: 08 ------0- M, segmentation: N
     2: 08 000010-- Length: 2
     3: 06 0------- Direction: From originating site
     3: 06 -000---- 0 TransactionID
     3: 06 ----0110 Radio Resouce Management
     4: 32 00110010 RR Cipher Mode Complete
l2_RRciphModCompl:2378 TRUNKATED (0x0x7fff561d4ab9 - 0x0x7fff561d4ab9)
```

With the Cipher Mode Complete message the MS acknowledges the receipt of the Cipher Mode Command (trace 7.1). This message also indicate that the MS has started ciphering its communication. In fact this message is already transmitted under encryption. Because the Gammu software gets the packages before they are encrypted, or after they are decrypted, we can still read this message.

The message is for the largest part filled with fill bits - the “2b” encoding. This message is actually the largest source of known plain text. The only thing that can vary is the possible transmission of the MEs IMEISV in the Cipher Mode Complete message. Which would be encoded in the fifth octet of this message.

Chapter 8

GSM Security

This chapter discusses the security of the Um-interface of GSM. First we will look at the security goals for the GSM system as stated by the ETSI. Each goal will be discussed specifically, looking at its definition and the methods employed to achieve those goals. Then we will look at some of the weaknesses of the GSM system, by discussing some specific attacks. These attacks are all directed at the Um-interface of GSM, and for every attack the feasibility using the affordable and open-source equipment discussed in chapter 1 will be discussed.

This discussion therefore focuses on attacks using a modest budget. These attacks will often suffer from practical problems, which with enough resources to spend, could all be overcome. E.g. trying to break the A5/1 encryption would be possible real-time using a FPGA array, costing over a hundred thousand dollars [49]. These attack scenarios also assume a trusted GSM network (other than the Um-interface) and a trusted GSM provider and assume that A5/1 encryption is being used on the Um-interface. World wide this is of course not always the case, but in many Western countries this seems a reasonable assumption. If this encryption is not present, or a weaker one is used, e.g. A5/2, then attacking the GSM system becomes much easier. This also excludes attacks by law enforcers. In many countries the police have means for lawful interception, which is accommodated by the GSM system. Finally, this discussion will also assume a trusted ME and SIM module.

8.1 Security goals

The security of GSM is at an obvious disadvantage when compared to traditional, wired telecommunication networks, because of the availability of the radio transmissions. Therefore the ETSI stated that GSM security measures, where present, should provide the same level of security at the Um interface as corresponding items enjoy in fixed networks [50].

8.1.1 ETSI defined security goals

The ETSI considers five security goals in the GSM system:

- subscriber identity (IMSI) confidentiality
- subscriber identity (IMSI) authentication
- user data confidentiality on physical connections

- connectionless user data confidentiality
- signaling information element confidentiality

We will now look at each of these properties in more detail.

Subscriber identity confidentiality

This property states that the IMSI should not be made available or disclosed to unauthorized individuals, entities or processes. This feature should provide for identity privacy and location privacy of the subscriber and enhance other security features, like user data confidentiality.

Subscriber identity confidentiality is achieved by allocating a TMSI to a MS and using the TMSI for all further communications. The ETSI recognizes that in some cases, e.g. when a MS does not yet have a TMSI allocated (location registration, section 6.5.1), the IMSI is still transmitted in the clear on the Um interface. This is what is exploited by IMSI catchers.

Subscriber identity authentication

This is basically the authentication of the subscriber identity (IMSI/TMSI) by the network. This property both protects the providers from unauthorized use of their network, and protects the subscribers from communicating with an attacker impersonating an actual subscriber.

This feature is achieved by the authentication process, as described in sections 2.5.1 and 7.1, which can be initiated by the network at several times; most notably in location registration and when a MS requests a service. If an authentication fails the specific MS is denied access to the PLMN.

User data confidentiality on physical connections

The reference to 'physical' connections here can be confusing, but this property refers to the privacy of all communication made on traffic channels (TCHs). A physical connection in this sense is a point-to-point connection between two MSs (via the other entities in the network).

This is quite obviously achieved through the encryption discussed in chapter 7. When no encryption is used by the network then this feature is of course not maintained. A MS can signal the network with a list of encryption's it supports, though the networks should deny any MS that do not support A5/1 and A5/2. So MSs that do not support encryption should not be allowed access to services of a PLMN. A5/3 is not named as mandatory probably because there are still a lot of ME in circulation that do not support A5/3. The ME should check whether one of the encryption algorithms is being used (but not which), and signal the user if no encryption is used [51]. Interestingly the specifications offer the SIM module the ability to overwrite this behavior, keeping a ME silent when no encryption is used [52].

Connectionless user data confidentiality

"Connectionless user data" refers to data transmitted between MSs while no voice connection exists between them (which is covered by the previous goal). It does not refer to signaling messages transmitted on the Um-interface, which is the last security feature. In essence this comes down to privacy of SMS messages.

The ETSI states no functional requirements for this particular goal. However seeing how the specification allows for signaling channels to be encrypted just like the traffic channels, it would seem that that is the way to achieve this goal.

Signaling information element confidentiality

Formally this is defined as: a given piece of signaling information which is exchanged between MSs and base stations is not made available or disclosed to unauthorized individuals, entities or processes. This should achieve privacy of user related signaling information, like the phone number that is being called. This security goal is only defined on the following selected information fields in the signaling messages:

- the International Mobile Subscriber Identity (IMSI)
- the International Mobile Equipment Identity (IMEI)
- the Mobile Subscriber ISDN Number (MSISDN) of the callee during a mobile originated call
- the Mobile Subscriber ISDN Number (MSISDN) of the caller during a mobile terminated call

Whenever a connection already exists, and these values are transmitted, they should be protected. The means of protection are not defined, but since there is no other tool available then encryption of a bursts' payload, this informally means that whenever these values are transmitted, the carrying channel should be encrypted. Note that a connection must already exist, thus again excluding the transmission of the IMSI during sign on.

8.1.2 Other security goals

Next to the goals explicitly stated by the ETSI several other security goals can be thought of. *Location privacy* is one, but this is mainly covered by Subscriber identity confidentiality. *Authenticity* and *non-repudiation* are covered by the Subscriber identity authentication.

A goal that the ETSI does not mention is *Availability*. When we just look at the Um-interface, both the availability of the network, and availability of the MSs is important. Naturally it will be very hard to defend against an attacker who just jams the GSM frequencies in a certain region, but there are certainly other availability attacks imaginable. Some of those are described in Section 8.2.4. Even though GSM was not designed to cope with these attacks, they could have serious consequences.

8.2 Attacks

The security goals described above are achieved by the following security measures:

- secrecy of the SIMs secret key, K_i
- one-way authentication of the MS via challenge-response
- encryption both of the signaling channels and the traffic channels
- secrecy of IMSI/TMSI relation
- authentication of the user to the SIM, by the PIN code

We will now look at some attacks against the security goals of the GSM system performed on the air-interface. Because of the focus on the air-interface, attacks against the PIN, which unlocks the SIM, are not discussed. Also the provider and network are assumed trusted, since an evil provider could easily break all security goals, using just the standard operation of the GSM system.

An important distinction between these attacks is whether they are active or passive. Active attacks described here always require transmitting by the attacker. This means that the active attacks are much more noticeable. Also in most countries these transmissions in themselves are already illegal.

The feasibility of these attacks using the equipment mentioned in chapter 1 is discussed with every attack. Because we have neither a license for operating a BTS nor transmission equipment, the active attacks were all untested.

The attacks that are going to be discussed are:

Attacking confidentiality According to the security goals of the ETSI, when encryption is supported all conversations, SMS traffic and the signaling of the IMSI, IMEI and MSISDN should not be disclosed to unauthorized parties. Here we will discuss three eavesdropping attacks that could break this confidentiality:

- Passive eavesdropping: an attack that attempts to capture, decrypt and interpret GSM signals passively.
- Active eavesdropping: this is basically a man-in-the-middle attack, that is easier to perform than a passive attack, but the required transmissions can be detected.
- Semi-Active eavesdropping: this attack passively captures GSM signals and then actively finds the key through a fake base station attack.

Location privacy and identity privacy attacks The GSM system needs to know the geographic location of MSs, in order to route calls to the correct BTSs. These attacks attempt to find out the location of a subscriber. Also a MS transmits a unique identifier - the IMSI - which could break the subscribers identity privacy. Two attacks are discussed here:

- IMSI catchers: devices that attempt to recover the IMSI of the MS under attack. They can break both location and identity privacy.
- Radio Resource Location Protocol (RRLP): software running on certain smart phones that will reveal a MSs location. This only breaks location privacy.

Authenticity attacks Authentication of the MS should prevent attackers from impersonating authentic subscribers. Two attacks are discussed that might break this:

- SIM cloning: if an attacker can clone a SIM, then he can impersonate that SIM.
- Fake base station attack: an attack mainly targeting SMS messages, attempting to change the content, receiver, or both.

Availability attacks The availability of the GSM system is not an ETSI security goal, so the GSM system was not designed to prevent attacks on its availability. Two attacks are discussed:

- Network availability: an attack that attempts to make a BTS unreachable for MSs.
- MS availability: an attack that attempts to make MSs stop working.

Attacking implementations These attacks are not pointed against the GSM protocols, but against the implementations running them. A short discussion is included here on a variety of attacks that in themselves fall under one of the previous categories, but they all use implementation errors for the attack.

8.2.1 Confidentiality attacks

Attacks against confidentiality attempt to retrieve the information transmitted encrypted, at least when encryption is available. This refers to the final three security goals of the ETSI stated in the previous section. The confidentiality of phone conversations, SMSs, and the IMSI, IMEI, and MSISDN. Though the plain-text transmission of the IMSI during sign-on is excluded from these goals. The three attacks described here each attack all these confidentiality issues. It does not matter whether they intercept voice calls, SMS or encrypted signaling information. The confidentiality of the IMSI and IMEI are also vital for location privacy, so these attacks could also be noted there.

There is commercial equipment on the market that can eavesdrop on GSM [10]. However this equipment is very expensive and sold exclusively to governments. This does however show that an eavesdropping attack is possible.

Passive eavesdropping

The process of trying to passively intercept and decrypt a GSM phone conversation logically separates into three steps.

I. Capturing the GSM signals and demodulating them.

II. Decrypting the resulting bits.

III. Interpreting the resulting payloads.

First the GSM bursts themselves need to be captured. In this thesis the possibility of doing this using the USRP has been researched. The second step is dependent on what, if any, encryption is used. Also even when encryption is used some bursts are always transmitted unencrypted. The final step, interpreting the bursts, is not going to be a limiting factor. As several example traces throughout this thesis show, several bursts can already be interpreted without any trouble using the AirProbe project. Each of these steps we will now look at in further detail.

I. Acquiring Signals Capturing the signals between MSs and BTSs is still the hardest part of eavesdropping on conversations. It is not impossible, but it is hard when using relatively cheap hardware like the USRP. This is because of the frequency hopping employed on most BTSs. Using the AirProbe software out of the box, will help you receive a single carrier on the Um downlink. To capture an entire conversation when frequency hopping is used, you will need a way to gather all the bursts on all the different frequencies. There are two general approaches to achieve this:

1. Let the USRP follow the hopping sequence.

2. Capture all possible frequencies and attempt to follow the sequence afterwards.

Approach 1 requires a lot of processing inside the USRP's FPGA. All the parameters for the hopping sequence need to be retrieved from control bursts, then the hopping sequence needs to be calculated and followed for every TDMA frame. However there is a high chance that some of these parameters are transmitted when encryption is already enabled, and the network can always command a new hopping sequence under encryption. This makes it a necessity to break the encryption really fast, which is currently not possible without serious investments. Currently the FPGA contains all the Gnu Radio specific functions to sample a certain frequency at a certain rate and send these samples to a computer. So a lot of implementation is still needed here. It is questionable whether even the USRP2's much faster FPGA will be able to decrypt messages and then compute the hopping sequence in time. This approach might need additional FPGAs, and thus additional costs, to pull off. However it is an approach that should work for every BTS and regardless of the amount of traffic.

The second approach requires the capture of large amounts of data. The problem here lies in reducing the data the USRP sends on to the computer. A hopping sequence can maximally hop between 64 carriers. These carriers are all 200 KHz wide, and can be spread out evenly on the entire GSM spectrum. So worst case the attacker has to capture the entire GSM band (for GSM900 this is 25MHz for the up or downlink). The problem here is data throughput to the PC. Remember that each sample of an USRP is represented by two 16 bit numbers (section 1.2). For the USRP1's USB2.0 connection this means that the maximum bandwidth that can be sent to the computer is around 8MHz (8MHz $\times 2 \times 16$ = 256Mbit/s). The USRP2's GBE connection can manage around 30MHz, which is enough for one sided capture of GSM900. However this would require the host computer to be able to process 100 MByte/s of data (25MHz $\times 2 \times 16$ = 800Mbit/s) and even 200 MByte/s for both up and down link, which is too much for most PCs.

Of course some optimizations are possible. As said a BTS can never serve the entire GSM frequency band. This means that you can already discard all carriers above the top frequency and all carriers below the lowest frequency of a specific BTS. However that approach will not work for every BTS, since the maximal number of carriers (64) a BTS can serve, is still too much for this approach. Also if the top and bottom frequencies are too far apart, this approach will also not work.

Another optimization would be to have the FPGA discard all channels that have no traffic on them. However if too many phones are active at the same time, then this will not help much. It would be even better to have the FPGA interpret enough of the bursts, so it can already drop some that are not a part on the conversation the attacker tries to capture. However this optimization sees the same problems as those with the first approach because it requires a lot FPGA computation.

The objections stated above however, do not form a problem if the BTS being attacked does not employ frequency hopping, or only transmits on a few frequencies in a tight spectrum. On such BTSs eavesdropping using an USRP seems a genuine possibility. Regrettably there seem to be no numbers on how many, if any, of the BTSs match one of these conditions. This makes it hard to estimate the risk in the current situation. During this research only a handful of BTSs were observed, but none of those fulfilled these conditions.

Currently the second approach seems to be the one that most people in the AirProbe community believe is the correct one to follow. It does indeed seem the easier way out, with a higher chance of success (on at least some BTSs), though the FPGA programming needed here is by no means a simple task. At least no one has communicated a way to tackle this. This approach will probably not work for every BTS, but a working implementation for some BTSs is enough for the AirProbe community to show that they can listen in on GSM.

II. Breaking the Encryption For most providers in most Western countries breaking the encryption will currently equal breaking the A5/1 cipher. The A5/1 cipher was explained in detail in section 1.8.

Several weaknesses have been spotted in A5/1 since its internal structure was published [53, 8, 54] (and some even before that [6]). However in the past half year a practical project has emerged to actively break A5/1 [25, 49]. This A5/1 cracking project mainly consists of creating large tables in a generic time-memory trade off. It is pretty much the same tactic that was shown two years earlier by Steve Muller and David Hulton [55], who also computed large tables, but never released them. The distinguishing factor of this new project is that instead of computing the tables at a single point everybody on the Internet can join in and compute a table and then share them via bit torrent [56]. The code to compute these tables can be downloaded and it runs on certain types of NVIDIA and ATA graphics cards.

Further discussion on the A5/1 cracking project The idea behind these tables is as follows. The contents of several bursts that are sent after encryption is enabled on the Um interface can for the most part be guessed. This gives known plain text samples. XORing those plain text samples with the actual cipher text reveals keystream samples. The tables now function as a code book with 64 bits of keystream that are mapped to internal A5/1 states producing that exact piece of keystream.

A swift look at the internal design of A5/1 (figure 7.2) learns that the internal state has 64 bit positions, leading to 2^{64} possible internal states. That number is too large to be able to map all internal states. So instead of just storing an internal state and its 64 bits of keystream, they actually compute large chains where for each link the 64 output bits are again used as the internal state. They only store the begin and end points of these chains. Now when a piece of keystream is recovered, the attacker starts to make a chain out of this keystream, but for every link he checks whether the resulting value is stored in its table. If the attacker finds a hit, then the attacker can recover the internal state by computing the original chain from its stored begin point. This shrinks the size of the tables, but the attack time is increased and the tables no longer guarantee that an internal state can be found. But a much bigger problem that surfaces with this approach is chain mergers. Several different internal states will compute the same 64 keystream bits causing different chains to 'merge' and cover the same part of the key space. This makes the tables much less effective. To counter some of these problems a combination of two techniques, one to decrease the attack time and one to decrease the number of chain mergers, is used. Those techniques are distinguished points and rainbow tables respectively.

Distinguished points decrease the attack time because you only store values that fulfill a certain criteria. This way when attacking, an attacker does not need to query the table for every value, just those values that also satisfy the same criteria. Distinguished points do still suffer from chain mergers, which is exactly what rainbow tables help to decrease. By performing a slightly different computation (different rainbow color) for every link, chain mergers are drastically decreased. Though the attack time will increase exponentially with the number of links. In this A5/1 cracking project a table is created by computing chains until a distinguished point is reached, after which a specific transformation is performed on the current value (adding a new rainbow color). The computation then continues with the resulting value in the same way, for 31 distinguished points. When the thirty-second distinguished point is reached then the actual end value that is stored.

The reason for using both distinguished points and rainbow tables was to combine the positive sides of both optimization techniques in one. However, intuitively it seems that this also combines the downsides of using both distinguished points and rainbow tables. So there can still be chain mergers

inside a single rainbow color and the attack time increases because the attacker needs to compute the attack for all the different links / colors. The projects documentation is vague on why this approach or these parameters where chosen.

Because of the chain mergers, a certain number of different tables need to be produced. Those tables also need to cover a certain amount of the entire key space. According to the projects documentation around 380 tables with height $2^{28.5}$ are needed to have a 50 % chance of finding the key with the current amount of known plain text in mobile calls. In the first few months of 2010 several tables have shown up via bit torrent. These tables on themselves are only a means. The end, decrypting conversations, requires a search tool to search within these tables given some keystream samples. This search tool has not yet been released. □

On the other two encryption algorithms we can be brief. A5/2 has been shown weak or broken several times [57, 58]. Barkan et. al. describe a cipher-text-only attack on A5/2 that only requires a few dozen milliseconds of cipher text and computes in less than a second on a personal computer [53]. A5/3 very recently saw a theoretical break [48]. This attacks requires 2^{26} chosen plain text messages encrypted under related keys. For now this does not lead to a practical attack on A5/3. At the moment of writing no attack on A5/3 is feasible, the future will have to prove whether this remains so.

III. Interpreting the bursts After the demodulation and decryption steps the bursts need to be interpreted. AirProbe is currently the best open-source tool for this, but it can not yet interpret all bursts. For one, deciding which type of burst is received is decided by the standard, most likely, division of the broadcast channel, instead of making this decision based on the burst. This means that the results are worse when BTSs use non-standard division of the broadcast channels. Also currently the AirProbe sniffers can not decode any of the CCCH messages. Furthermore AirProbe is only able to listen to the downlink (BTS $\rightarrow$ MS) of conversations. Currently a lot of development to Airprobe is necessary in order to be able to interpret the uplink bursts.

Finally, it is worth noticing that development on AirProbe seems to be very slow lately. The last couple of months have seen no new functionality updates, and a lot of the software projects inside AirProbe do not even compile at the moment. This mostly seems due to the limited number of active developers with intimate knowledge of the workings of GSM.

Active eavesdropping

Due to the one way nature of the authentication (the MS authenticates to the BTS, but not the other way around) the GSM protocol is vulnerable to a man in the middle attack. An attacker could set up a false base station towards the MS and a false MS towards the BTS. This is detailed schematically in figure 8.1, where the authentication shown in 7.1 is being attacked by a man-in-the-middle.

The interesting part of this figure is in the setup of the ciphering steps. The difficulty for the attacker with this approach is getting the K_c from the MS through the “RR CIPHERING MODE COMPLETE” message, before the communication with the network, waiting for a response SRES, times-out. The maximum time the network should wait, according to specification, is 12 seconds. The only cipher text that the attacker has to work with is the 456 bits “RR CIPHERING MODE COMPLETE” message and its possible retransmissions. Until the attacker knows the session key he can not communicate to the MS after he has sent the “RR CIPHERING MODE COMMAND” message, because after that the MS only expects encrypted messages. According to Barkan et. al. [53] the time and cipher text limitations should not form a problem when performing this attack. It is unknown whether such a man-in-the-middle attack has ever been tested practically.

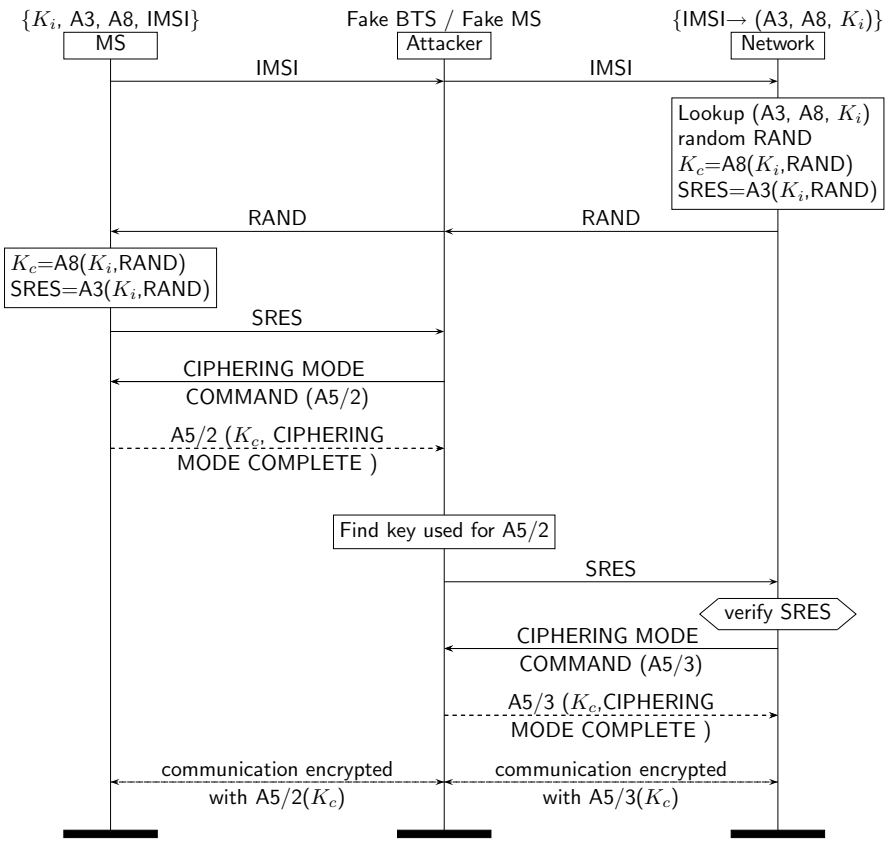


Figure 8.1: A Man-In-The-Middle attack on the GSM authentication.

An alternative for this approach would be for the attacker to set up an A5/0 (no encryption) connection to both sides. However this would be a more noticeable attack. The ME might show the user that no encryption is being used. Also the Network has to accept that the MS does not support any encryption algorithm and switch to A5/0, which might raise suspicion on the Network's side.

Naturally these man-in-the-middle attacks are already noticeable. It requires transmissions by the attacker, which can be spotted.

The success of such an attack hinges on the successful emulation of a BTS (towards the MS's side) and a MS (towards the Network's side). An attacker is able to act as an BTS using an USRP and the OpenBTS software. Currently however there does not seem to be any software allowing an attacker to act as an MS. The attack itself seems feasible, but it would require a lot of work to get general purpose hardware, like the USRP, to act as a MS. It might be possible for an attacker to use an actual ME and simulate the SIM card. This would make the implementation job of an attacker a lot easier, because low level actions, like synchronizing with the BTS and following the hopping sequence, would be handled by the phone. This suggestion has been made several times in the AirProbe community, but no account seems to exist of someone actually trying this.

Semi-active eavesdropping

If an attacker solves steps I and III of the passive attack - so if he is able to capture GSM communication, and interpret the bursts after decryption - then there is an alternative to breaking the encryption, which is finding the session key. Finding the session key is an active attack, but the active part of this attack can take only a couple of seconds, while the rest of this attack can be done passively, minimizing the time window in which the attack is detectable.

The signals transmitted during authentication are sent in the clear. If an attacker is able to capture those signals, then he knows the challenge (RAND) as shown in section 7.1 that was send to the MS. Assuming the attacker also stored some encrypted communication after that together with the frame numbers in which this communication took place, then he is ready to actively gather the session key.

If at any point later in time the attacker performs a fake base station attack on the same MS, having it sign on to his fake base station and perform an authentication procedure with the same RAND that was used for the tapped signals, then the MS will compute the same session key. After all its secret key and the A3 algorithm will not have changed and the RAND is the same, so the resulting K_c will also be the same. Now in order to find out this key the attacker can make use of another weakness in the GSM system, namely that a MS will use the same key for every encryption algorithm. So if the attacker sets up a new encrypted communication between the corrupted BTS and the target MS, he can have them communicate through an A5/2 encrypted channel. Using known attacks on A5/2, the attacker can recover the session key in a matter of seconds, like in the man-in-the-middle attack of section 8.2.1. After that the attacker can stop the active attack and start using the recovered session key and correct frame number on the recorded signals and decrypt them.

This attack has the benefit of working for every encryption (so even A5/3), as long as MSs support weaker encryption's (if the original encryption is too strong) and use the same key for all encryption algorithms.

It is an active attack, but the active attack time is much shorter than during a fully active attack. This is useful for attackers because generally active attacks can be detected. Naturally this attack suffers from the same practical implementation problems detailed in sections 8.2.1, 8.2.1 and 8.2.1, when trying to perform it using the USRP.

8.2.2 Location privacy and identity privacy attacks

The IMSI is the main identifier in the GSM system, and a lot depends on the secrecy of the IMSI. The GSMA's security goals clearly state that the IMSI should be protected at any point except during the sign on procedure, or after a failed TMSI identification. In order to protect the IMSI it is immediately replaced with a TMSI, which is transmitted in encrypted form to the MS and subsequently used as the identifier for this specific MS while inside the VLR area.

Because the IMSI uniquely identifies a MS, and because MSs are often linked to persons, the leaking of the IMSI can break the anonymity of the subscriber. If an attacker can link the IMSI to a location, then he can break location privacy. Depending on the situation he might find out either where the victim is, or who are all present in a current location. Because leaking the IMSI can endanger both location and identity privacy, these two are combined here.

Ideally the IMSI $\leftrightarrow$ TMSI relation should remain a secret. Due to some form of traffic analysis this relation could possibly be revealed, but traffic analysis will probably be limited to just a couple of cells at the maximum, because of the extreme effort needed to place receivers in many cells. However

IMSI catchers could reveal this connection. IMSI catchers are the first attack discussed here and they can break both identity and location privacy. The second attack discussed here, an attack on the RRLP protocol, only break location privacy.

IMSI catchers

IMSI catchers are already employed by law enforcement agencies to find out the IMSIs of the MSs that a suspect is carrying. With the IMSIs they can e.g. order taps on these MSs.

As we have seen in sections 2.5.2 and 4.5.1 the IMSI is transmitted in the open by MSs and the BTS during sign on / location registration. So at this point the IMSI can be intercepted. This is not a violation of the ETSI security goals, since at this point no secure connection has been established.

Intercepting these IMSIs can be done in two ways, passive and active:

- In passive interception an attacker simply listens on the AGCH channel of a BTS. After a MS has sent a message, which includes its IMSI, on the RACH that it wants to sign on, the BTS will respond by granting the MS a signaling channel. This response comes on the AGCH and will be addressed to the IMSI. So listening to that channel will result in some IMSIs, though an attacker will probably not know whose MSs these IMSIs belong to.

- A much easier way would be an active fake base station attack. If an attacker starts a fake base station, seemingly from the correct provider, in the neighborhood of his victim MS, then this MS will try to register to the fake base station. Probably the MS will start sending its TMSI, which the fake base station will reject, resulting in the MS transmitting its IMSI. Naturally the attacker would have to make sure that the reception from his fake base station is better than the reception of the current serving BTS. This attack, being an active attack, can be detected. This is typically how industrial IMSI catchers work.

The active attack will expose the IMSI $\leftrightarrow$ TMSI relation that existed in the current VLR area. So it is possible to reveal this connection, though the MS will not use this TMSI again after it was rejected by the fake base station. Once it returns to an authentic BTS, the MS will think its TMSI invalid and identify itself with its IMSI, like in a sign on, and receive a new TMSI. So for all previously intercepted communication in the VLR area, the IMSI $\leftrightarrow$ TMSI relation can be revealed. To reveal this relation for all subsequent communications, until the next TMSI reallocation, a man-in-the-middle attack like described in section 8.2.1 could be used.

In principle this attack breaks location privacy. It reveals IMSIs present in a geographical area (the area of the fake base station). However this is the area in which the attacker, or at least his equipment, is already present. Informally this attack reveals “who is here?” and not “where is he?” Note that this does not make this attack irrelevant just because it is infeasible to use it for following a victim within a large area. Because of the small geographical scale it is for instance easy to link a MSs IMSI to a subscriber. An attacker could then scan for this IMSI at certain areas. Or this attack could be used to judge how crowded a certain area is.

Implementing an IMSI catcher, using the USRP seems fully possible. Using OpenBTS will already make a BTS out of an USRP. Changing an OpenBTS implementation to an IMSI catcher seems a straight forward adaption. Though we did not experiment with this, due to the legal restrictions on operating a BTS. Alternatively an attacker could use OpenBSC, but an extra BTS would be needed.

Radio Resource Location Protocol (RRLP)

During a large scale test of OpenBSC at the HAR conference in 2009, the OpenBSC community noticed that several smart phones supported the Radio Resource Location Protocol (RRLP) [59]. The RRLP can be used to request the MSs location by the network. This location can either be determined roughly through measurements of nearby BTSs and delays of transmitted bursts, or if the ME supports it more precisely through GPS.

If the ME has an active RRLP client then the MS will transmit its location to the network upon request, without any form of authentication. Naturally this breaks location privacy.

This might not seem that terrible, because in order to transmit the RRLP command and receive its answer, the attacker already need to be in the neighborhood of his victim, thus already roughly knowing the MSs location. However, what is most worrying about this is that subscribers do not know whether their ME supports the RRLP and have no way to shut it off. This again stresses the closedness of the GSM world.

8.2.3 Authenticity attacks

In a GSM context authenticity attacks typically entail an attacker either trying to impersonate a subscriber, or trying to alter the content of the communications between two subscribers.

In order to impersonate a subscriber an attacker would have to identify himself with the victims IMSI and pass the authentication by the network. Because the network should choose a different challenge for every authentication, the attacker needs to be able to compute the corresponding response. For this the attacker would need the secret key K_i. This key is only stored in secure memory of the SIM card and at the providers authentication centre (AuC). The secret key is never transmitted and there should be no way the secret key can be derived out of the IMSI, other then querying the AuC. Assuming the providers protect there AuC adequately, recovering of the K_i directly should be very hard. However the secret key is used in the A3 and A8 algorithms, which might provide an attack window. This is what the SIM cloning attack described below attempts.

If an attacker wants to impersonate a subscriber towards another subscriber, then he could also use a fake base station attack. In this attack the attacker acts as the GSM network and can make fake calls or SMSs to a connected MS. This is not a man-in-the-middle attack as was described in section 8.2.1, because the attacker does not act as a MS towards an authentic network. The IMSI catcher described above is also a fake base station, but the difference here is that in this attack the fake base station also communicates to the user of the MS.

An attacker trying to alter the content of communications is actually an attack against integrity. The only way to attack integrity in a communication between two subscribers, assuming they are not both within the reach of the attacker, would be to act as a man-in-the-middle. This attack was described in the confidentiality context as the active eavesdropping attack, section 8.2.1. This man-in-the-middle would have to be placed between one of the subscribers under attack and its base station. Attacks on integrity do not seem very problematic in voice calls, for obvious reasons, but for SMS messages this attack could very well work.

SIM cloning

An attacker can try to clone the SIM of a subscriber. The attacker can then identify himself as the victim, either just towards the network allowing him to make calls that will be billed to the victim,

or also to another subscriber. The attacker would probably get the best results if he uses his cloned SIM at the time his victim is not signed on to the GSM network. The GSM specifications assume the IMSI to be unique, it is not directly clear what would happen if two identical IMSIs are signed in on the network at the same time. This is of course detectable in the back-end of the GSM system. It is unknown if providers try to detect this.

Cloning the SIM means basically finding its secret key. The secret key is used as input to the A3 and A8 algorithm. Once the original A3/A8 implementation (Comp128v1), discussed in section 7.1, was reverse engineered, a weakness was soon discovered [9]. The response (SRES), computed by the A3 algorithm on the challenge (RAND) and the secret key K_i, can actually leak information of the secret key. With about 150.000 chosen challenges the entire secret key can be revealed.

The easiest way to perform this attack is with physical access to the SIM card. An attacker can construct a SIM reader which can reveal the secret key in a short time [60].

More interestingly is of course the feasibility of this attack via the Um-interface. Using a fake base station an attacker could get a MS to connect to his fake base station. This fake base station would then need to keep sending authentication requests to the MS. This attack seems at all possible [61], but it would take a long time, in which the fake base station can be spotted. Also during this time the victims MS would not be able to make or receive calls and the ME would probably experience some battery drain. Nevertheless such an attack might be possible during the night. The fake base station attack is discussed further in the following section.

Since Comp128v1 several newer versions have been introduced. It is unknown what type of A3/A8 algorithm is currently used by providers. Most A3/A8 algorithms are proprietary, so they have never seen any public scrutiny.

Fake base station attack

If an MS is connected to a fake base station an attacker could intercept all communications made by the MS, and act as the person being contacted. The attacker could also initiate communications to the MS, again impersonating who ever he wants.

Since the base stations do not authenticate themselves to the MSs, all that is needed here for the attacker is to operate a BTS in the vicinity of his victim. The attacker needs to make sure that his BTS transmits on a beacon frequency of the victim's provider, and that his BTS transmits the MCC + MNC of the same provider.

Implementing such a fake base station using e.g. OpenBTS seems fully possible, though again we did not experiment with this, due to the legal restrictions on operating a BTS. The subscriber might notice that he is under an attack if the fake base station attack is continued long enough. Because he is no longer connected to his provider no call can be made to this MS, no calls by the MS will end up anywhere accept at the attacker, and calls made by the MS will not appear on bills. A short lasting fake base station attack however, e.g. only to transmit one fake SMS message to the MS, seems entirely likely.

8.2.4 Availability attacks

Here we discuss some attacks that were coincidentally all demonstrated at the 2009 CCC conference in Berlin. One against the availability of the network and a few against the availability of MSs. Both

attacks are active attacks.

Network availability

Dieter Spaar demonstrated a Denial of Service (DoS) attack against a BTS [62]. For this attack he used the only known phone that has a reprogrammable base-band chip, the TSM30 with a TI Calypso GSM chipset. The baseband chip is responsible for all low level actions, like the lowest layers of the Um-protocol. All other baseband chips are closed source devices that are hard to reprogram. The source code together with a compiler for the TI Calypso leaked at some point, making the TSM30 an easier reprogrammable phone.

This attack is targeted at the sign on procedure. As a reminder, the sign on procedure is initiated by the MS sending an access burst on the RACH. The BTS will then reserve a signaling channel for the MS and assign it to the MS by responding with a "immediate assignment" command on the AGCH. The MS can then tune to this signaling channel and start communicating with the BTS. This procedure was detailed in section 4.5.1.

In this attack the TSM30 is reprogrammed to transmit access bursts continuously on the RACH. Without ever looking at the responses on the AGCH. The effects of this attack are twofold, first the continuous transmissions on the RACH can disturb the transmissions of genuine MSs. And secondly all the correctly received access bursts will prompt the reservation of a signaling channel. These signaling channels are released after some time, if no traffic is received on them. However the reservation takes long enough to exhaust the supply of channels.

At this point the MS has not yet needed to authenticate itself. No information of the MS has been transmitted yet, making it an anonymous attack. Though the source of the transmissions might be located.

This attack effectively renders one BTS useless for all MSs that might want to connect to it. This attack does not influence open connections. If a MS is already in a conversation via a BTS, this conversation will not be effected. But when the conversation is ended it will be very hard to start a new one via this BTS.

This attack requires a TSM30 phone. These phones were mainly sold in Spain, Mexico and Chile around 2003. They are very rare now. However it seems entirely feasible to implement such an attack using a USRP. It would simply entail the USRP continuously transmitting access bursts on the correct frequency. With the current state of the software some implementation will be needed. Although the implementation would be quite simple.

MS availability

Attacks on the availability of MSs are of course also possible. In the presentation on the A5/1 tables Chris Paget noted that OpenBTS was able to send a "you have been stolen" message to MSs [49]. The effect of this message differs per phone models, but it can range from requiring a hard reboot before functioning again to completely shutting down for good. This of course is a very easy DoS attack. According to the presentation this attack was stumbled upon simply using a USRP and OpenBTS.

In another presentation, also at the 2009 CCC conference, Collin Mulliner showed how to disable MSs through malicious SMS messages [63]. This has to do with the so called Over-The-Air (OTA)

functionality in GSM. The OTA functions allow the installation of software on a mobile phone through signaling messages [52, 64, 65]. He showed he had successfully attacked iPhones, and smart phones running Android and Windows Mobile. These attacks only work on smart phones because ordinary MEs will not be able to run the malicious programs that are being installed.

8.2.5 Software errors

Besides all the attacks discussed in this chapter it is interesting to note that increasingly more attacks are being found in GSM software [63, 49, 66]. Only in the last year or so, with the rise of initiatives like OpenBTS and OpenBSC, it has become possible for the general public to test their mobile phones. At the moment it seems like only a very select number of people are working on this, and yet they have already found an alarming amount of bugs. This begs to question just how thorough the GSM equipment has been tested. Even though the specifications to GSM are for the most part public, the implementations are not. Only a few corporations make the actual low level GSM equipment like the baseband processors in MEs. Any software mistake found in a single ME has a major chance of occurring in many more different MEs. It seems likely that we will see a rise in attacks on GSM implementations.

Future Work

A lot of practical work is still needed into the AirProbe project before it becomes a useful GSM protocol analyzer. Firstly it should be improved to actually be able to decode all bursts correctly, but most importantly a solution needs to be found for the hopping problem, which was described in section 8.2.1.

AirProbe limits itself to the downlink side of the air interface - cell tower to mobile phone. Capturing the uplink side would be the logical next step, when the down link capturing works.

With tools like OpenBTS and OpenBSC, implementing the network side of the GSM system, it is a lot easier to test GSM equipment in a test environment. Though you need a license for this. Similarly it could be very useful to have an implementation of a mobile phone. In the final weeks of writing this thesis the OsmocomBB project was introduced, which does exactly that [67]. It is still in early stages, but the use of this project might prove a serious help in GSM research.

The flexibility of the USRP and Gnu Radio allow them to be used for research into many different systems. With this relatively cheap equipment it is possible to capture signals from all kinds of wireless protocols, like WiFi and GPS. So other protocols that, like GSM, have seen little practical research because of the difficulties in signal capturing and processing can become open for further inspection.

A good example of this would be UMTS. This system is being deployed world wide and might replace GSM in the long run. The practical problems with capturing UMTS signals are greater than with the capturing of GSM signals. UMTS's frequency bands are wider for instance, and it employs a more complicated multiplexing technique where different phones communicate at the same time on the same frequency, but their signals are modulated using a different code. So capturing these signals will be even more challenging than capturing the GSM signals. Even so, it would be interesting to judge the feasibility of capturing UMTS signals with the open-source hardware and software.

When just looking at the GSM system, it might be useful to research possibilities to increase the GSM security. Naturally GSM's successor, UMTS, is already being deployed and UMTS's security is superior to GSM's - e.g. UMTS has mutual authentication between BTS and MS. However it is unlikely that UMTS will replace GSM completely within the next decade and in the mean time attacks against GSM will only become more feasible.

It is of course not easy to adapt the GSM standard without invalidating large quantities of GSM equipment. An approach that could be researched, is to replace the fill bits in GSM bursts with a random sequence, instead of the current "2b" encoding. Doing so would remove a lot of the known plain text which is essential for the A5/1 cracking project. A lot of bursts encode the length of their information elements in the second layer header, seemingly removing the need for a standard fill bit pattern. Naturally the success of this adaptation would depend on the implementation of the GSM stack on most mobile phones.

Another approach that is already seeing interest from several commercial parties is the deployment of an extra secure channel on top of GSM. Often this requires the purchase of a new mobile phone, which effectually creates a secure tunnel on top of GSM, analogously to a SSH tunnel on top of TCP/IP.

Finally, a subject which mostly falls outside of the scope of this thesis, but which will probably become very important, is the security of the mobile phone itself. Modern mobile phones, especially smart phones, are now serious computational platforms. They combine several communication channels; next to GSM and UMTS, phones can also support Blue tooth, WiFi, GPS, a direct link to a PC and an RFID reader. All this increased functionality also makes it easier for an attacker to try and run malicious code on a victim's mobile phone. It is very likely that for most attackers it will be much more efficient to attack a series of mobile phones through its software than to attempt one of the attacks detailed in this thesis.

Conclusion

The GSM system proved hard to understand. Even though most of the specifications are publicly available, the sheer amount of documents and information can be overwhelming. There are many helpful sources to get a broad overview of GSM, but when looking for a more detailed perspective very few sources remain and those that do often contradict each other.

In this thesis the overview of GSM is by no means complete, but hopefully it will help as a sort of stepping stone for those who want to learn the details of the air-interface of GSM.

The publicly available specifications of GSM are in contrast to the extremely closed GSM industry. Only a couple of companies in the world create the core GSM equipment, like the base band processors in mobile phones. All of these implementations are closed-source and often sold exclusively to GSM providers. The immediate effect of this closed nature of the GSM industry is that billions of people walk around with a device, of which hardly anyone knows, or has verified, what it does.

Until recently there has only been theoretical research into GSM security. All the strengths and weaknesses of the GSM protocols are more or less known. Now, with affordable and adjustable tools like the USRP and Gnu Radio, more practical research becomes possible so we can actually test the implementations we rely on.

Several new attacks have been demonstrated against specific implementations of GSM, like the use of the RRLP protocol to get the location of a mobile phone and disabling mobile phones over the air interface (sections 8.2.2 and 8.2.4 respectively).

Implementations of simple theoretical attacks, like building an IMSI catcher are also within reach in the current situation.

The more complicated theoretical attacks against the GSM protocol however, are still problematic to implement. Eavesdropping for instance, whether passive or via a man-in-the-middle, is a long way from realization. Though it is feasible, considering the commercial equipment sold for this purpose, there exist a lot of practical problems when trying to implement this using just the open-source hardware and software. Most of these problems result from the use of frequency hopping in GSM, which was originally designed solely to improve transmission quality.

This is not a plea for releasing a turn-key attack tool, but to give subscribers the ability to verify the workings of GSM, e.g. to check whether, and what kind of, encryption is being used to protect their conversations.

The practical problems that at the moment prevent a general attack tool can vary with the specific practical situation. Frequency hopping might not be employed by a specific cell tower, or the cell tower transmits on only a few frequencies that lie close together. In fact a cell tower might not even use encryption. In those cases many attacks become much easier, but we do not know if and how

many cell towers have such a configuration. During this research all observed cell towers used both frequency hopping and encryption.

It is clear that in the current state-of-affairs the open-source GSM projects have made some attacks more feasible. However the impact of these attacks seems small for now.

The open-source GSM projects have not yet directly worsened the confidentiality of conversations over GSM. Despite many recent claims to the contrary no actual conversation has been captured and decrypted, and it will take a lot of effort before the current problems preventing these attacks are solved.

It is hard to predict how long it will take the current community behind these open-source projects to solve these practical problems. Though reactions from the community seem eager, the recent rate of development in for instance AirProbe do not show much progress.

Appendix A

Identifiers

Overview of commonly used identifiers in GSM.

Acronym	Full name	Consists of	Size	Identifies
ARFCN	Absolute Radio Frequency Channel Number	ranges: 1-124, 512-885, 975-1023	max.4 digits	Identifies a specific uplink and downlink frequency channel.
BCC	Base station Color Code	value range: 0-7	3 bits	Identifies one of eigth training sequences a BTS can use on the CCCH and differentiates between neighbouring cells.
BSIC	Base Station Identity Code	NCC+BCC	6 bits	Two neighbouring cells can never have the same BSIC.
CCN	Country Code Number	-	1-3 digits	Uniquely identifies a country or region.
CGI	Cell Global Identification	LAI+CI	14 digits	Uniquely identifies a single cell.
CI	Cell Identity	-	2 bytes	Uniquely identifies a cell inside a LAI.
CKSN	Ciphering Key Sequence Number	-	3 bit	Identifies a session key that was established during authentication.
HSN	Hopping Sequence Number	value range: 0-63	6 bit	Identifies the hopping algorithm used.
IMEI	International Mobile Equipment Identity	-	15 digits	Uniquely identifies a ME. since 01-Apr-2004 it contains model and revision information, prior to that date it contained a manufacturer code.

Continued on Next Page...

Overview of commonly used identifiers in GSM – Continued

Acronym	Full name	Consists of	Size	Identifies
IMEISV	International Mobile Equipment Identity and Software Version	IMEI+SV	16 digits	Uniquely identifies a ME plus the version number of its current software. The SV digits of the IMEISV are allowed to change over time.
IMSI	International Mobile Subscriber Identity	MCC+MNC+MSIN	15 digits	Uniquely identifies one subscription worldwide.
LAC	Location Area Code	-	4 digits	Uniquely identifies a location area within a PLMN.
LAI	Location Area Identity	MCC+MNC+LAC	10 digits	Uniquely identifies a location area.
MCC	Mobile Country Code	-	3 digits	Uniquely identifies a country.
MNC	Mobile Network Code	-	2 digits	Uniquely identifies a PLMN inside a country.
MSIN	Mobile Subscriber Identification Number	-	10 digits	Uniquely identifies a subscriber inside a PLMN.
MSISDN	Mobile Subscriber ISDN Number	CCN+NDC+SN	max.15 digits not counting prefixes	The directory number of a mobile subscriber.
MSRN	Mobile Station Roaming Number	CCN+NDC+TSN	max.15 digits not counting prefixes	Temporary number assigned by the serving VLR. Identifies a subscriber within a VLR area. Used to route incoming calls to the subscriber.
NCC	Network Color Code	-	3 bits	Differentiates between neigbouring PLMNs.
NDC	National Destination Code	-	2-3 digits	Uniquely identifies an area within a CCN, but it can also address a particular service (like 0800 numbers) or PLMN within a CCN.
SN	Subscriber Number	-	max.10 digits	Uniquely identifies the subscriber inside a PLMN.
TMSI	Temporary Mobile Subscriber Identity	-	4 bytes	Uniquely identifies a subscriber inside a VLR area.
TSN	Temporary Subscriber Number	-	max.10 digits	Uniquely identifies a subscriber within the serving VLR area. Assigned by the VLR.

Acronyms

ACCH	Associated Control Channels	42
ADC	Analog to Digital Converter	9, 10, 37
AGCH	Access Grant Channel	41, 45, 46, 48, 49, 53, 65, 90, 93
ARFCN	Absolute Radio Frequency Channel Number	34, 35, 41, 42, 48, 51, 66, 99
AuC	Authentication Centre	22, 25, 74, 91
BCC	Base station Color Code	99
BCCH	Broadcast Control Channel	40, 46, 48, 53, 64
BCH	Broadcast Channels	40, 46
BSC	Base Station Controller	13, 19–21, 24, 27, 30, 64
BSIC	Base Station Identity Code	40, 44, 99
BSS	Base Station Subsystem	19, 20, 24, 44
BTS	Base Transceiver Station	13, 19–21, 27, 28, 34, 37, 40, 64, 76, 83, 90, 93

C/R	Command / Response	55, 56
CBCH	Cell Broadcast Channel	41, 46
CC	Call Control	60, 63, 68
CCCH	Common Control Channels	41, 46, 64
CCH	Control Channels	35, 39
CCN	Country Code Number	99
CGI	Cell Global Identification	20, 22, 27, 28, 64, 99
CI	Cell Identity	20, 22, 27, 28, 99
CKSN	Ciphering Key Sequence Number	18, 25, 66, 68, 99
CM	Connection management	60
DAC	Digital to Analog Converter	9, 37
DCCH	Dedicated Control Channels	41, 46, 53, 76
DISC	disconnect command	57
DoS	Denial of Service	93
EA	address field extension	55, 56, 58
EIR	Equipment Identification Register	23
EL	Extension bit	58
FACCH	Fast Associated Control Channel	42, 46, 49, 68
FCCH	Frequency Correction Channel	40, 46
FDMA	Frequency Division Multiple Access	33, 34
FN	Frame Number	35, 38, 40, 77
FPGA	Field Programmable Gate Array	10
GMSC	Gateway Mobile Switching Centre	21
GMSK	Gaussian Minimum Shift Keying	19, 37, 38, 43
GSMA	GSM Association	6
HDLC	High level Data Link Control	53
HLR	Home Location Register	21, 22, 27, 28

HSN	Hopping Sequence Number	35, 65, 99
I	Information	53, 56
I format	Information transfer format	56
IE	Information Element	54, 58, 63
IEI	Information Element Identifier	63, 64
IMEI	International Mobile Equipment Identity	18, 23, 67, 82, 99
IMEISV	International Mobile Equipment Identity and Software Version	65, 100
IMSI	International Mobile Subscriber Identity	18, 19, 21–23, 25, 27, 28, 30, 41, 64, 81, 82, 89, 90, 100
ITU	International Telecommunication Union	6, 24
LAC	Location Area Code	18, 40, 100
LAI	Location Area Identity	18, 20, 22, 27, 66, 100
LAPD	Link Access Protocol on the D channel	53
LFSR	Linear Feedback Shift Register	76
LI	Length Indicator	63, 64
LPD	Link Protocol Discriminator	55
M	More data bit	58
MA	Mobile Allocation	35, 65
MAIO	Mobile Allocation Index Offset	35, 65
MCC	Mobile Country Code	18, 40, 92, 100
ME	Mobile Equipment	18, 23, 34, 35, 64, 76, 80, 81, 88, 91, 94
MM	Mobility Management	60, 63, 67

MNC	Mobile Network Code	18, 40, 92, 100
MOC	Mobile Originating Call	28, 49, 66–68
MS	Mobile Station	18, 19, 21, 22, 25, 27, 37, 40, 49, 51, 64, 65, 76, 81, 90, 92, 93
MSC	Mobile Switching Centre	20–22, 25, 30, 37, 64, 65
MSIN	Mobile Subscriber Identification Number	18, 100
MSISDN	Mobile Subscriber ISDN Number	19, 21, 22, 28, 67, 68, 82, 100
MSRN	Mobile Station Roaming Number	100
MT	Message Type	62
MTC	Mobile Terminating Call	28, 30, 49, 67
N(R)	Receive sequence Number	57
N(S)	Send sequence Number	57
NCC	Network Color Code	100
NDC	National Destination Code	100
NSS	Network Switching Subsystem	19–21
OTA	Over-The-Air	93
P/F	Poll/Final bit	57
PCH	Paging Channel	41, 46, 53
PCM	Pulse Code Modulation	20
PD	Protocol Discriminator	61
PGA	Programmable Gain Amplifier	10
PIN	Personal Identification Number	19
PLMN	Public Land Mobile Network	17–20, 28
PSTN	Public Switched Telephone Network	17, 21
PUK	Pin Unblocking Code	19

RACH	Random Access Channel	41, 44, 46, 48, 49, 53, 90, 93
RPE-LPC	regular pulse excited-long term prediction	20, 37
RR	receive ready command	57
RR	Radio Resource management	60, 63, 67
RRLP	Radio Resource Location Protocol	91
S	Supervisory function	57
S format	Supervisory format	57
SACCH	Slow Associated Control Channel	42, 46
SAPI	Service Access Point Identifier	55
SCH	Synchronization Channel	40, 46
SDCCH	Standalone Dedicated Control Channel	41, 45, 46, 48, 51, 66, 67
SDR	Software Defined Radio	6, 9
SFH	Slow Frequency Hopping	19, 34
SIM	Subscriber Identity Module	18, 20, 22, 23, 80, 81, 88, 92
SMS	Short Message Service	19, 41, 60
SN	Subscriber Number	100
SS	Supplementary Services	60, 63
SS7	Signaling System #7	24
SSN	Send Sequence Number	62
TCH	Traffic Channels	35, 39, 46, 49, 53, 67, 68, 81
TCH/F	Full Rate TCH	39, 46
TCH/H	Half Rate TCH	39, 46
TDMA	Time Division Multiple Access	33, 35
TI	Transaction Identifier	61
TMSI	Temporary Mobile Subscriber Identity	18, 22, 27, 28, 30, 41, 64, 67, 68, 81, 89, 90, 100

TRAU	Transcode Rate and Adaption Unit	20
TSC	Training Sequence Code	44
TSN	Temporary Subscriber Number	100
U	Unnumbered function	57
U format	Unnumbered format	57
UI	Unnumbered Information	53, 57
Um interface	Air interface between ME and BTS	23, 33, 38, 53, 60
USRP	Universal Software Radio Peripheral	6, 9, 10
VLR	Visitor Location Register	22, 25, 27, 28, 30, 64, 65

Bibliography

[1] Chris Tryhorn. Nice talking to you ... mobile phone use passes milestone. *The Guardian*, 2009. Tuesday 3 March http://www.guardian.co.uk/technology/2009/mar/03/mobile-phones1.

[2] James Moran. Gsma security group update. In *2nd ETSI Security Workshop: Future Security*, 2007.

[3] March 2010. http://pda.etsi.org/pda/queryform.asp.

[4] Ross J. Anderson. *Security Engineering: A Guide to Building Dependable Distributed Systems*, chapter 17. Wiley Computer Publishing, 2001. ISBN: 0471389226.

[5] Technical information gsm system security study. http://cryptome.org/jya/gsm061088.htm.

[6] Jovan Golic. *Cryptanalysis of Alleged A5 Stream Cipher*, page 23955. 1997. http://jya.com/a5-hack.htm.

[7] Marc Briceno, Ian Goldberg, and David Wagner. A pedagogical implementation of the gsm a5/1 and a5/2 "voice privacy" encryption algorithms, 1999. http://cryptome.org/gsm-a512.htm (originally on www.scard.org).

[8] Alex Biryukov, Adi Shamir, and David Wagner. *Real Time Cryptanalysis of A5/1 on a PC*, page 118. 2000.

[9] Marc Briceno, Ian Goldberg, and David Wagner. An implementation of the gsm a3a8 algorithm. (specifically, comp128.), 1998. http://www.scard.org/gsm/a3a8.txt.

[10] January 2010. http://www2.rohde-schwarz.com/en/service_and_support/Downloads/news_from_rohde_and_schwarz?issue=152&type=27&downfileid=1395.

[11] February 2010. http://www.ing.nl/particulier/internetbankieren/internetbankieren/wijzigingen-in-voorwaarden-mijn-ing/.

[12] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Abbreviations and acronyms*, 1996. ETR 350 / GSM 01.04.

[13] January 2010. http://www.ettus.com/.

[14] G.W. Gardiner. *Handbook of Stochastic Methods*, chapter 1.4.4. Springer, second edition.

[15] January 2010. http://www.altera.com/.

[16] David Burgess and Harvind Samra. *The OpenBTS Project*. Kestrel Signal Processing, Inc., Fairfield, California, October 2008.

[17] September 2009. http://gnuradio.org/trac.

[18] Robert Fitzsimons, January 2009. http://273k.net/gsm/find-a-gsm-base-station-manually-using-a-usrp/.

[19] January 2010. https://svn.berlin.ccc.de/projects/airprobe/wiki.

[20] September 2009. http://wiki.thc.org/gsm/.

[21] January 2010. http://www.wireshark.org/.

[22] January 2010. https://svn.berlin.ccc.de/projects/airprobe/wiki/tracelog and http://www.gammu.org/.

[23] September 2009. http://openbts.sourceforge.net/.

[24] September 2009. http://bs11-abis.gnumonks.org/trac/wiki/OpenBSC.

[25] January 2010. http://www.reflextor.com/trac/a51.

[26] John G. van Bosse and Fabrizio U. Devetak. *Signaling in Telecommunication Networks*, chapter 12. Wiley Series in Telecommunications and Signal Processing. Wiley-Interscience, 2 edition, 2006. ISBN: 0471662887.

[27] Joerg Eberspaecher, Hans-Joerg Voegel, and Christian Bettstetter. *GSM Switching, Services, and Protocols*. Wiley, 2 edition, 2001. ISBN: 047149903X.

[28] March 2009. http://openbts.blogspot.com/.

[29] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2); International Mobile station Equipment Identities (IMEI)*, 2000. ETS 300 508 / GSM 02.16.

[30] International Telecommunication Union. *ITU-T Q.700 : Introduction to CCITT Signalling System No. 7*, 1994. (03/93).

[31] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Radio transmission and reception*, 2009. TS 145 005 v8.5.0.

[32] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Multiplexing and multiple access on the radio path*, 2010. TS 145 002.

[33] Raymond Steele (editor). *Mobile Radio Communications*, chapter 3 and 8. IEEE press, 1992. ISBN: 047197806X.

[34] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Modulation*, 2001. EN 300 959 / GSM 05.04.

[35] Lawrence Harte, Richard Levine, and Geoff Livingston. *GSM Superphones: Technologies and Services*, chapter 3: GSM Technology. McGraw-Hill Professional, 1998. ISBN: 0070381771.

[36] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2); Multiplexing and multiple access on the radio path*, 1999. EN 300 908 / GSM 05.02.

[37] Hans-Joerg Vogel, Christian Hartmann, and Jorg Eberspacher. *GSM*, chapter 4: Air interface - physical layer. John Wiley and Sons Ltd., 2008. ISBN: 0470030704.

[38] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Channel coding*, 1999. EN 300 909 / GSM 05.03.

[39] Hamid Aghvami. *Multiple Access Protocols For Mobile Communications*. John Wiley & Sons, 2002. ISBN: 0471498777.

[40] International Organization for Standardization (ISO), ISO copyright office, Geneva. *Information technology Telecommunications and information exchange between systems High-level data link control (HDLC) procedures*, 2002. Reference number ISO/IEC 13239:2002(E).

[41] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Data Link (DL) layer; General aspects*, 1999. EN 300 937 / GSM 04.05.

[42] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Mobile Station - Base Station System (MS - BSS) interface; Data Link (DL) layer specification*, 1999. EN 300 938 / GSM 04.06.

[43] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Mobile radio interface signalling layer 3; General aspects*, 1997. ETS 300 939 / GSM 04.07.

[44] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Mobile radio interface layer 3 specification*, 1998. EN 300 940 / GSM 04.08.

[45] Jörg Eberspächer, Hans-Jörg Vögel, Christian Bettstetter, and Christian Hartmann. *GSM Architecture, Protocols and Services*. John Wiley & Sons, Ltd., third edition, 2009. ISBN: 978-0-470-03070-7.

[46] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Security related network functions*, 1998. ETS 300 929 / GSM 03.20.

[47] 2002. `http://cryptome.org/a53-gea3/a53-gea3.htm`.

[48] Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time attack on the a5/3 cryptosystem used in third generation gsm telephony. 2010. `http://eprint.iacr.org/`.

[49] Karsten Nohl and Chris Paget. Gsm - srsly? presented at 26C3 in Berlin, `http://events.ccc.de/congress/2009/Fahrplan/attachments/1519_26C3.Karsten.Nohl.GSM.pdf`, December 2009.

[50] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Security aspects*, 1998. EN 300 920 / GSM 02.09.

[51] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Mobile Stations (MS) features*, 1998. ETS 300 906 / GSM 02.07.

[52] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface*, 1995. GSM 11.11.

[53] Elad Barkan, Eli Biham, and Nathan Keller. Instant ciphertext-only cryptanalysis of gsm encrypted communication. In *Advances in Cryptology - CRYPTO 2003*, volume 2729/2003, pages 600–616. Springer Berlin / Heidelberg, 2003.

[54] Elad Barkan and Eli Biham. *Conditional Estimators: An Effective Attack on A5/1*, page 119. 2005.

[55] Steve Muller and David Hulton. The a5 cracking project. In *Chaos Communication Camp 2007*, 2007. http://video.google.com/videoplay?docid=8955054591690672567&hl=en#.

[56] January 2010. http://reflextor.com/torrents/.

[57] Ian Goldberg, David Wagner, and Lucky Green. The (real-time) cryptanalysis of a5/2. 1999.

[58] Slobodan Petrović and Amparo Fúster-Sabater. Cryptanalysis of the a5/2 algorithm. In *IACR ePrint Report*, volume 2000/052, 2000. http://eprint.iacr.org.

[59] Harald Welte. Report of openbsc gsm field test august 2009, har2009 vierhouten, the netherlands. http://laforge.gnumonks.org/weblog/2009/10/13/#20091013-har2009_report, 2009.

[60] February 2010. https://svn.berlin.ccc.de/projects/airprobe/wiki/simreader.

[61] Ian Goldberg and Marc Briceno. Gsm cloning. http://www.isaac.cs.berkeley.edu/isaac/gsm.html.

[62] Dieter Spaar. Playing with the gsm rf interface. presented at 26C3 in Berlin, http://events.ccc.de/congress/2009/Fahrplan/attachments/1507_Playing_with_the_GSM_RF_Interface.pdf, December 2009.

[63] Collin Mulliner. Fuzzing the phone in your phone. presented at 26C3 in Berlin, http://mulliner.org/security/sms/feed/smsfuzz_26c3.pdf, December 2009.

[64] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Specification of the SIM application toolkit for the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface*, 1998. GSM 11.14.

[65] European Telecommunications Standards Institute, France. *Digital cellular telecommunications system (Phase 2+); Security mechanisms for SIM application toolkit; Stage 2*, 1999. TS 101 181 / GSM 03.48.

[66] Harald Welte. Fuzzing your gsm phone using openbsc and scapy. presented at 26C3 in Berlin, http://events.ccc.de/congress/2009/Fahrplan/attachments/1503_openbsc_gsm_fuzzing.pdf, December 2009.

[67] March 2010. http://bb.osmocom.org/.