Intercoms Hacking: call the frontdoor to install your backdoors

Sébastien Dudek - sebastien.dudek@synacktiv.com

Abstract

To break into a building, several methods have already been discussed, such as trying to find the code paths of a digicode, clone RFID cards, use some social engineering attacks, or the use of archaic methods like lockpicking a door lock or breaking a window. New methods are now possible with recent intercoms. Indeed, these intercoms are used to call the tenants to access the building. But little study has been performed on how these boxes communicate to request and grant access to the building.

In the past, they were connected with wires directly to apartments. Now, these are more practical and allow residents to open doors not only from their classic door phone, but to forward calls to their home or mobile phone. Private houses are now equipped with these new devices and it's common to find these "connected" intercoms on recent and renovated buildings.

In this short paper we introduce the Intercoms and focus on the particular one that are installed in buildings today: the numeric/connected intercoms. Then we present our analysis on a main interesting attack vector, which already has its own history. After this analysis, we present our environment to test the intercoms, and show some practical attacks that could be performed on these devices. To finish, we talk about recent observations we have made on M2M (Machine2Machine) intercoms that are also very common in buildings today.

Acknowledgement

I would like to thank my employer Synacktiv for giving me time to study this subject and many other cool ones, as well as my teammates for their time reviewing this paper and giving me advices and feedbacks.

I hope also this short paper will be interesting to read and any other feedback would also be appreciated to complet this research subject.

Contents

1 Introduction

1.1 Context

An intercom [1], door phone, or a house intercom, is generally a voice communication device for use within a building. Independent from the public telephone network, this device allows people to call a local resident to access to a building.

The classic version of intercom consists of a device that establishes a communication between the street door and the door phone device of a house. The device of the street door is generally equipped with a loudspeaker, a microphone, and buttons to a call residents. These classic versions of intercoms generally have $4 + n$ wires where 4 wires are used for power, door system, and where n is the number of homes to call.

New generation of intercoms become installed especially in new or renewed buildings. This new generation is called "numeric" and includes a GSM and 3G/4G module, but could also includes a Wi-Fi module as well. This generation avoids complex installation and ensures a maximum capacity, and they can easily include video communication in addition to the voice system.

1.2 Wiring topology

Three different types of house intercoms exist [2]:

- conventional: which are the classic version connected with $4 + n$ wires. This system is used in medium-sized buildings;
- simplified: one pair of wire that replaces the 4 wires of the conventional system and a wire for each house. This system is also used in medium-sized buildings.
- numeric: the wire for each house is replaced by a mobile technology like GSM, 3G, or 4G. Sometimes an Internet cable can be used with a TCP/IP stack, but communication through GSM, 3G, or 4G is often chosen over a cumbersome cable for the ease of installation. This intercom is installed in new and renovated buildings but also private residents in Europe.

More outputs can also be included to control other doors and increase the number of cables.

1.2.1 Numeric intercoms

Numeric intercoms need less wires to link resident door phones to the building intercom. These intercoms offer a video call system, and also many other features to seduce the customers.

One of these practical features allows a resident to use its own telephone, or mobile phone, to open the building street door. The figure 2 represents a simple architecture of a numeric Intercom installation.

When a person is calling a resident; the intercom uses a mobile network (GSM, 3G, or 4G) to reach the phone of the resident. The resident doesn't have to move anymore but only to reply with its smartphone and then open the door.

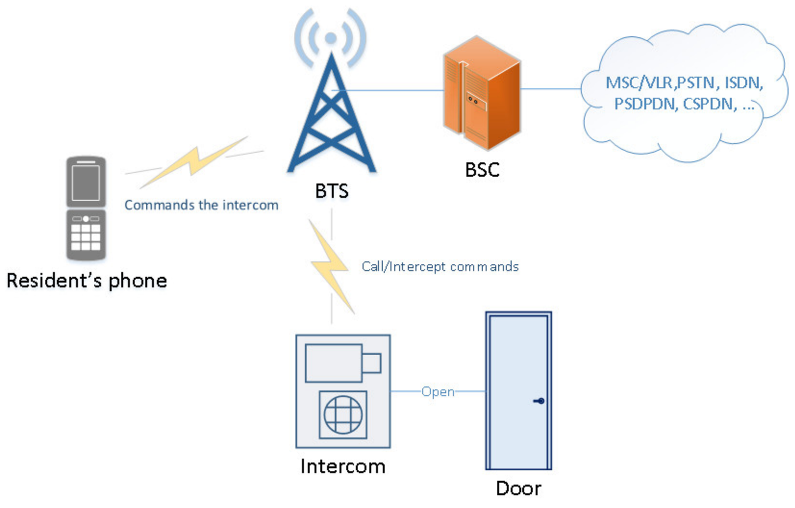


Figure 1: *Simple numeric intercom architecture*

1.3 Leaders in the French market

In this market, 4 brands are generally present:

- Intratone;
- Noralsy;
- Urmet Captiv;
- Comelit.

It's not easy to recognize a numeric intercom with a mobile module. We can try to spot them thanks to a new stainless steel case, or sometime with a LCD screen and a front camera as a first approach. But if we are lucky enought, some of these intercoms can directly be spotted thanks to the installation of a mobile network block. Indeed, as an example Intratone provides a 3G block that is connected to the intercom, as shown in figure 2. If we look at the documentation, we can also read an interesting point saying that if the 3G network is not reachable, the intercom will automatically fall back to the GSM network [25]. This allows us to think that a downgrade attack is possible on these intercoms.

These devices are pretty expensive and cost from 2k€, but cheaper devices that provide pretty the same functionalities exist.

1.4 Cheaper alternatives

For those who are not seduced by the price, expecially private residents, only few alternatives exist:

- Linkcom which is commonly used by private residents;

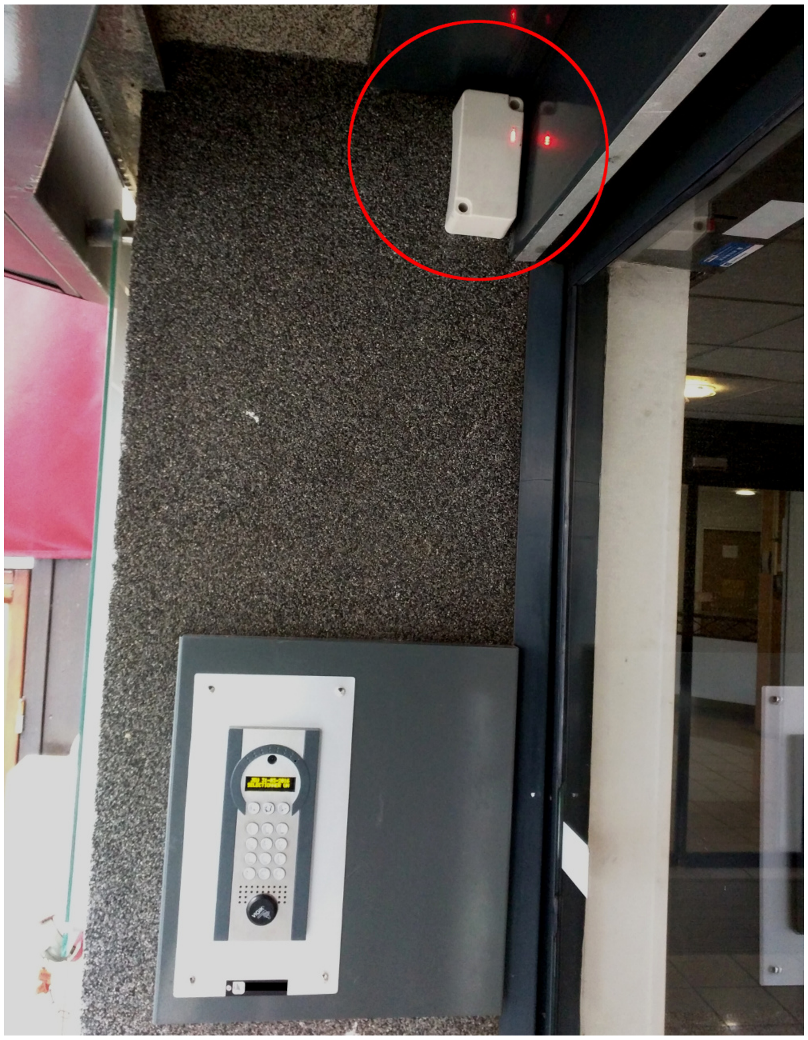


Figure 2: *Example of 3G block of an intercom*

- GSM Activate (UK company);
- and many other less famous [26].

1.5 Other variants of wireless intercoms

Other variants of numeric intercoms exist that use Wi-Fi, or DECT. These Wi-Fi or DECT intercoms are certainly very interesting to look at, and will probably inspire people in this area. In addition to Wi-Fi and DECT, other devices can be found on the market with an insecure sub-1 GHz radio communication system [27][3].

As we can see, wireless intercoms exist in every tastes and colors, but we will only focus on intercoms that are currently installed and that use the mobile network.

2 State Of the Art

2.1 Publications

Very few publications exist on the subject of numeric intercoms, and probably none about their security. But, some non-security publications are inspiring like the article of Oliver Nash, which explains how to modify a conventional intercom to open the door with a cell phone [4]. Moreover, as they use the mobile network to communicate, we can start looking at previous researches on mobile security to start our attacks.

Speaking of which, an interesting paper was published about IMSI Catchers by Daehyun Strobel, showing that GSM tracking and taping is possible [8]. At Black Hat Briefings 2008, Steve Dhulton has highlighted some methods to capture and crack the GSM signal. It should be noted that between 2007 and 2008, The Hacker Choice (THC) group also regrouped and documented a lot of materials on GSM internals and A5/1 cracking. These documentation have been deleted in 2009, but are still available when browsing the archives. At the 26c3 in 2009, Chris Paget and Karsten Nohl presented attacks on GSM with rainbow tables [5]. In 2010 at 27c3, Harald Welte and Steve Markgraf have presented their OsmocomBB project that aims to run an open source GSM stack on few Motorola models, and capture the GSM traffic [6]. Using the same OsmocomBB stack, Sylvain Munaut and Kastern Nohl also showed at 27c3 that it is possible to intercept hopping calls using a cheap phone [7].

At BlackHat 2011, the hacking Vodaphone Femtocell gateways was presented by Ravishankar Borgaonkar, Nico Golde, and Kevin Redon. The unauthenticated firmware update vulnerabilities allowed attackers to push their own firmware to get a shell and turn these gateways into 3G IMSI-Catcher, without having to care about the mutual authentication constraint in UMTS [9]. Some details are also available in the THC Wiki [10].

At SSTIC 2014, Benoit Michau has presented an analysis of baseband security and highlighted the existance of bugs in some baseband implementations that could lead to a mutual authentification bypass in 3G and 4G [34].

Later in October 2015, attacks on privacy and availability of 4G were presented by the researchers Altaf Shaik, Ravishankar Borgaonkar, N. Asokan, Valtteri Niemi and Jean-Pierre Seifert [11]. The paper describes some attacks on the RRC (Radio Resource Control) and the EMM (EPS Mobility Management) protocols, that can lead to some leaks and downgrade the 4G UE to 3G or GSM.

Then at SSTIC 2016, Christophe Devine and Benoit Michau published their researches on the LTE crypto, highlighting also the issues in 4G baseband implementations [35].

2.2 Tools

In addition to the publications, the following tools will be useful to analyze and to attack intercoms:

- OpenBTS [12] and YateBTS [13]: software to run a GSM and GPRS Base Station;

- USRP [14], bladeRF [15], and so on: hardware to run a Base Station and capture mobile traffic;
- HackRF [16]: software-defined radio hardware that can be used to monitor mobile traffic, and to perform downgrade attacks for our case;
- GNU Radio [17] software-defined radio development toolkit;
- OpenLTE [18]: Open implementation of the 3GPP LTE specifications;
- OsmocomBB [31]: Open source GSM baseband;
- Airprobe [32]: GSM sniffer.

3 Short basics on GSM, GPRS, 3G, and 4G

Before attacking numeric intercoms, a better understanding of mobile security mechanisms and weaknesses is required.

3.1 Brief overview of GSM and GPRS authentication mechanisms

GSM (Global System for Mobile communications) and GPRS use an authentication mechanism A3/A8 called COMP128-3. The figure 3 shows the authentication process using the COMP128-3 mechanism. Only the SIM card and the AuC (Authentication Center) know the value of the subscriber key Ki (128 bits) that will be used to generate the RES (32 bits) resulting from the $RAND$ value in A3 and processed on the BSC/MSC side. Then RES will be compared to $SRES$ (32 bits) processed in the UE side. If $RES = SRES$ then the user will be authenticated to the network.

The ciphering in GSM is initially made with the A5/1 algorithm [20] at Layer 1 on the TCH (Traffic CHannel) and DCCH (Dedicated Control CHannel) [19]. To encrypt and decrypt the conversation, the key Ki generates a Kc key with the A8 mechanism as shown in figure 3. In GPRS, the user equipment authenticates to a SGSN and the ciphering is done at Layer 2 LLC (Logical Link Control) [21] initially with the GEA1 algorithm. So one of the main difference between GSM and GPRS is their way to maintain confidentiality. In GSM the communication is confidential unlike GPRS where the data is confidential including user traffic and the signalization.

On GSM and GPRS an attacker is able to perform different known attacks:

- attract a victim to its rogue Base station, and intercept the communication with some forwarding tricks;
- reuse the authentication triplet $RAND, RES, Kc$ many times;
- attack the signaling channel which is not encrypted at all in GSM;
- attack the A5/1 algorithm [24];
- and son on.

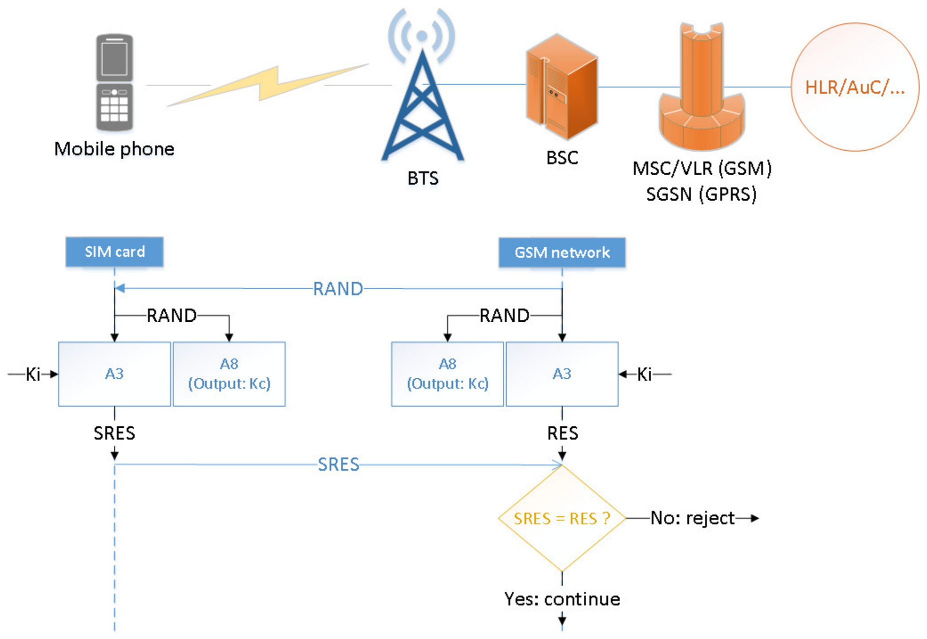
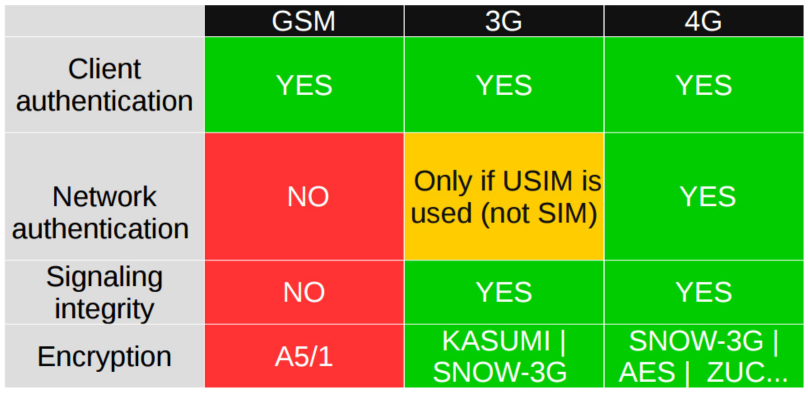


Figure 3: *GSM authentication process*

3.2 The advantages of 3G/4G networks compared to GSM/GPRS

The security of mobile evolved with UMTS (3G) and LTE (4G) networks. Indeed, other algorithms for integrity and ciphering of radio access have been adopted. It started with KASUMI [23] deployed for UMTS initially. Then SNOW-3G [22] appeared as a second algorithm for 3G but also for 4G. Additionally to SNOW-3G, 4G uses the AES CBC with 128-bit keys to cipher the communication. Then in 2011, the ZUC algorithm was release to be used in 4G communications [39].

The figure 4 shows the main differences between 2G, 3G and 4G in security:



	GSM	3G	4G
Client authentication	YES	YES	YES
Network authentication	NO	Only if USIM is used (not SIM)	YES
Signaling integrity	NO	YES	YES
Encryption	A5/1	KASUMI | SNOW-3G	SNOW-3G | AES | ZUC...

Figure 4: *Security differences between 2G, 3G and 4G*

Thanks to the USIM (Universal Subscriber Identity Module), 3G and 4G networks use mutual authentication. The access to a UMTS network is possible with the previous SIM card, while in LTE the use of USIM is mandatory. So some attacks applied on GSM are possible on 3G with a targeted subscriber that uses a SIM card instead of a USIM card. An other attack is to jam the radio signal to let the user equipment select a GSM netowrk cell instead.

3.3 Signal attraction

Wireless technologies use handover techniques, especially when users are generally mobile like in Wi-Fi, GSM, 3G and 4G. Devices always look for the best reception, so a user can move to one place and its UE (User Equipment) will try to connect to the closer station. Attackers in this area know if there is no mutual authentication, it is easy to attract a UE in a rogue base station with stronger signal than a legit station. Others method like jamming, or precise attacks on the targeted protocol can be used to downgrade the UE to GSM and bypass the mutual authentication. But protocol attacks are material dependend. Indeed, to trigger a bug with these attack an attacker will have to fingerprint his victim's mobile phone. That scenario is possible to play, but a little hard and time consuming when looking on downgrade vulnerabilities for one specific baseband or more equipments.

At the other side, jamming is pretty basic and only requires a transmitter to send random data over-the-air in a specific frequency bandwidth. The example figure 5 below shows a GSM station channel at 925.4MHz before getting jammed by a transmitter as shown in figure 6.

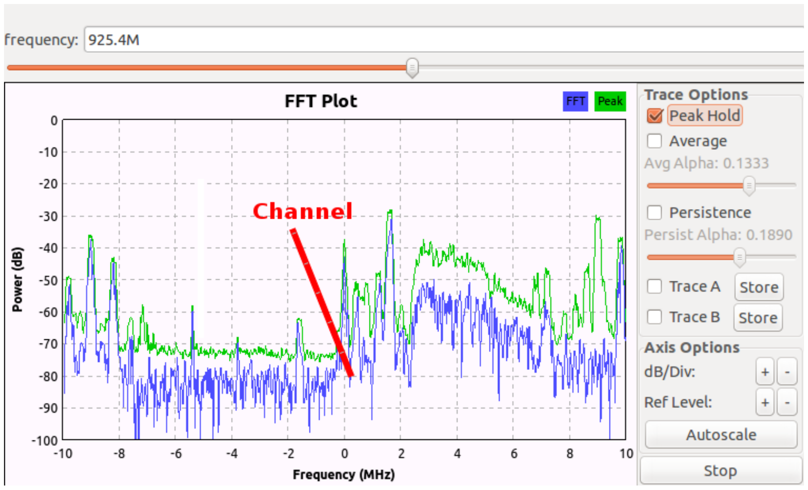


Figure 5: *FFT sink displaying the beginning of a channel*

As a consequence, devices close to the jammer will not be able to see the targeted channel.

3.4 Our observations with 3G jamming

Depending on the operator, an autority like ARCEP in France (Autorité de régulation des communications électroniques) regulates the frequency allocations for many purposes including

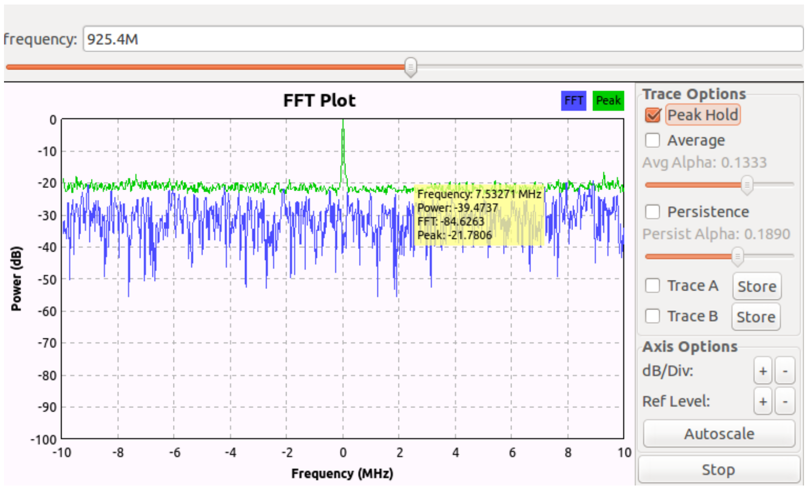


Figure 6: *FFT sink showing a station channel jammed by our transmitter*

GSM, UMTS, LTE and so on. The details about the different allocations are public, and can be found in the different documentations of the ARCEP. In UMTS 900 MHz bands are allocated as shown in figure 7.

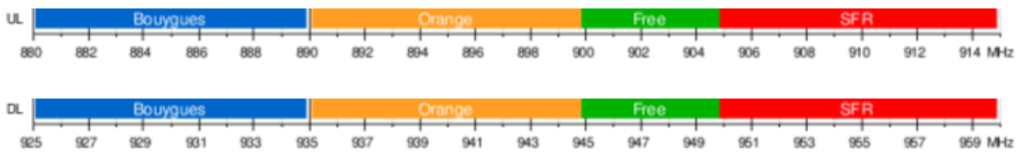


Figure 7: *UMTS bands in 900MHz (source: lowcostmobile.com)*

And in the 1900-2100MHz frequencies, the allocation looks like in figure 8.

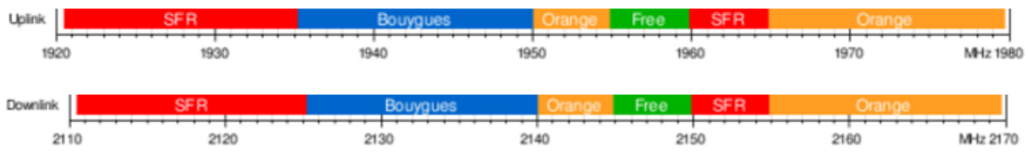


Figure 8: *UMTS bands in 1900MHz (source: lowcostmobile.com)*

Knowing the operator, it is possible to focus on precise ranges, and have a better idea of frequencies used by a User Equipment in a location by messing orthogonal codes of each channel. As WCDMA systems have 5 MHz bandwidth, we are able jam a bit more than 3 channels with a simple HackRF (20 MHz bandwith) considering the space between each channel.

Sending gaussian noise over the used frequencies while a target is calling someone is enough to force this target to use the 2G network as shown in figure 9. Moreover, it was also observed that mobile can also stay in 2G for hours without being jammed anymore, even if the mobile phone is rebooted. This last behavior was observed on Nexus 5 and Nexus 6 devices.

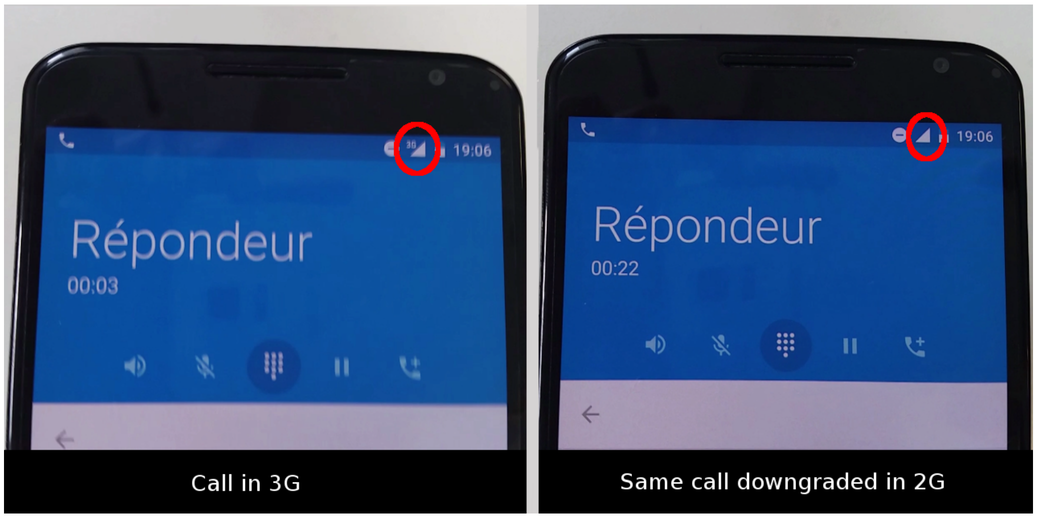


Figure 9: *3G conversation downgraded in 2G*

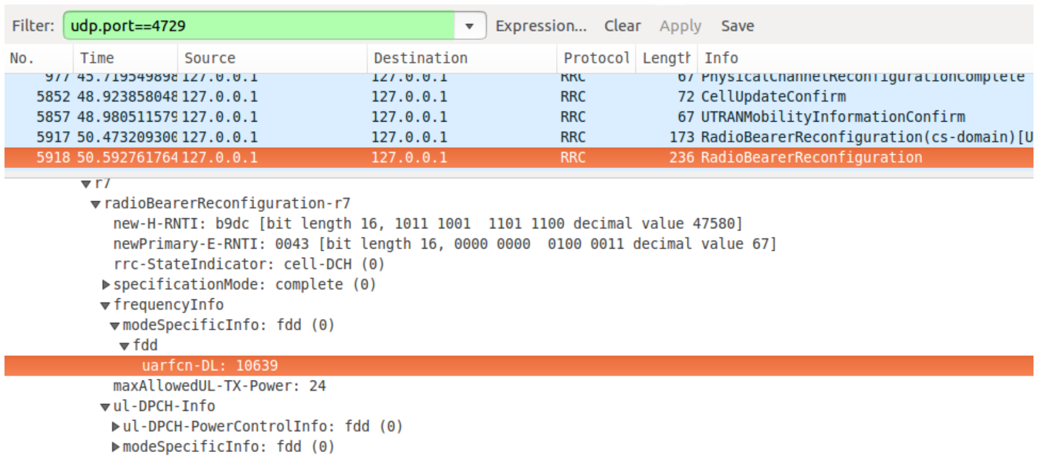
So we are able to downgrade the communication from 3G to 2G using a cheap SDR (Softwre-Defined Radio) device like ithe HackRF. But jamming all channel with random gaussian noise is unefficient, because we need to flood the channels that are only visible for the cellphone. To fix this issue, we have found another cheap alternative that we use to enumerate the 3G and 4G channels semi-automatically.

3.5 Focus on right frequencies

Unfortunatly, OsmocomBB cannot help to enumerate 3G UARFCN (UTRA Absolute Radio Frequency Channel Number) as for ARFCN in 2G, but some alternative exist. Indeed, devices equipped with a X-Gold baseband can expose a *diag mode* interface */dev/ttyACM0*. Then we can make use of *xgoldmon* [38] to parse data from this interface, and encapsulate radio messages in a GSMTAP layer to analyze the 3G traffic with Wireshark or other tools.

X-Gold basebands are not the only option to capture the 3G traffic. Indeed, some mobile phones equipped with the Qualcomm baseband have a QXDM `/dev/diag` interface that can be exposed. For these Qualcomm basebands, the library of the *SnoopSnitch* [40] tool can be used to make captures and analyze the traffic with the `diag_import` call.

While analyzing the traffic we have observed that UMTS RRC (Radio Resource Control) messages can be extracted from GSMTAP to get Downlink UARFCN as we can see in figure 10.



```
No.   Time         Source     Destination   Protocol Length Info
 977  45.71954989  127.0.0.1  127.0.0.1     RRC        67 PhysicalChannelReconfigurationComplete
5852  48.92385804  127.0.0.1  127.0.0.1     RRC        72 CellUpdateConfirm
5857  48.98051157  127.0.0.1  127.0.0.1     RRC        67 UTRANMobilityInformationConfirm
5917  50.47320930  127.0.0.1  127.0.0.1     RRC       173 RadioBearerReconfiguration(cs-domain)[U
5918  50.59276176  127.0.0.1  127.0.0.1     RRC       236 RadioBearerReconfiguration

▼r7
 ▼radioBearerReconfiguration-r7
    new-H-RNTI: b9dc [bit length 16, 1011 1001  1101 1100 decimal value 47580]
    newPrimary-E-RNTI: 0043 [bit length 16, 0000 0000  0100 0011 decimal value 67]
    rrc-StateIndicator: cell-DCH (0)
   ▶specificationMode: complete (0)
   ▼frequencyInfo
    ▼modeSpecificInfo: fdd (0)
     ▼fdd
        uarfcn-DL: 10639
    maxAllowedUL-TX-Power: 24
   ▼ul-DPCH-Info
    ▶ul-DPCH-PowerControlInfo: fdd (0)
    ▶modeSpecificInfo: fdd (0)
```

Figure 10: *Downlink UARFCN index display in a RRC RadioBearerReconfiguration message*

However, we were only able to get the Downlink UARFCN for channels the UE could register to. Some further work should be done on diag data, maybe looking at other types not processed by xgoldmon's `logparse` module. But again another method exists to get the list of UARFCN around us very easily.

Indeed, while looking at diag messages and using the ServiceMode application triggered with the `*#0011#` code on Samsung mobiles for example, we were able to get the UARFCN on diag logs as shown in figure 11. This also includes UARFCN of other operators the UE tried to register to, even if it failed to authentication to this stranger operator.

```
[...]
LOG:>>[HIGH]oemtestmode.c,403,Idle: dl_uarfcn 10688 ul_uarfcn 9738<<
[...]
```

Figure 11: *Log parsed from diag interface with `xgoldmon`*

Thanks to this ServiceMode application, it is possible get the different frequencies exposed of the different MCC/MNC the UE tried to register almost automatically. Devices which do not expose a diag mode easily on the host require manual interaction to force the mobile to connect to another network. Moreover, it is not required to have an access on the diag interface to get these UARFCN, because the ServiceMode exposes the exact same information in logcat logs too. So using ServiceMode and logcat, it is possible to retrieve the complet list of UARFCN around us, even with a Qualcomm baseband, and that also includes LTE EARFCN with the same method. Then with that list, we can focuse on precise UARFCN for our jamming attacks on 3G intercoms.

We have now all the tools to start our attacks on GSM and 3G intercom, but a mobile lab is required to perform all the interception part.

4 Intercoms analyses

4.1 Environment

4.1.1 Lab setup

To analyze the intercoms we use a bladeRF x115 [28] (that is cheaper than a USRP) powered through USB 3.0 by a computer, 2 antennas with 9 dBi for transmission (TX) and reception (RX) , and YateBTS as a radio access network software, like OpenBTS, as shown in figure 12.

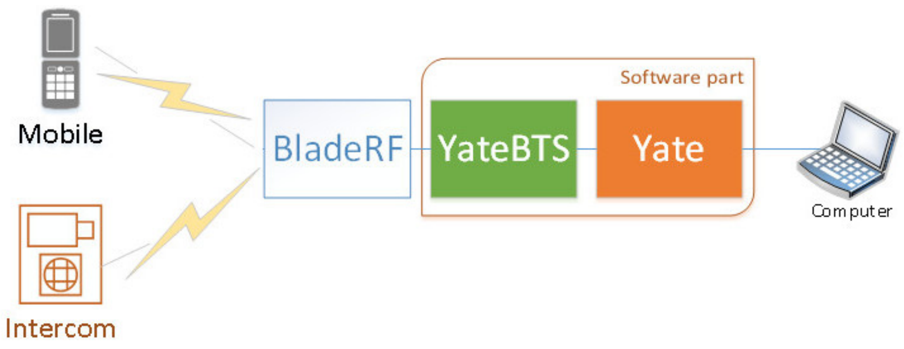


Figure 12: *Lab setup*

As a first sample, we used a Link GSM iDP [29] intercom with a USIM card that belongs to Bouygues Telecom provider.

4.1.2 Intercom configuration

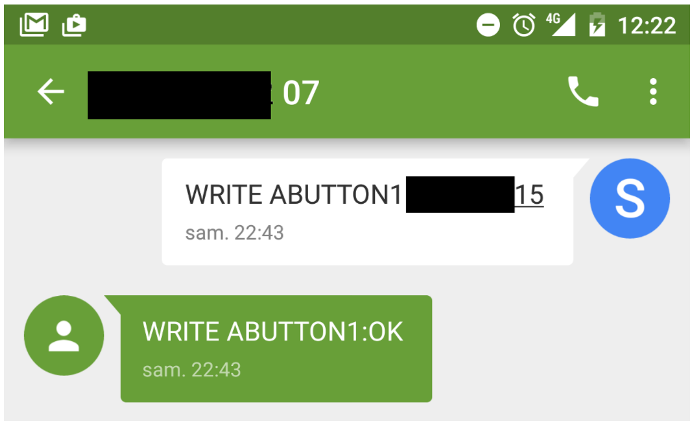
Following the Link iDP GSM manual [30] there are 3 ways to configure the intercom:

- programming the SIM manually thanks to a mobile phone, or a SIM reader/programmer;
- via SMS messages;
- or via the Link iDP manager software;

For security reasons, a first administrator "ADMIN1" number is required to command the intercom via SMS messages. So we have added a contact "ADMIN1" number to the SIM card with a mobile phone that is supposed to belong to the manager of this intercom. As a first impression, our goal as an attacker will be to impersonate the administration number, or find another way to bypass the number verification remotely to send commands to the intercom.

After setup the ADMIN number, this valid ADMIN number can send commands to the intercom to configure it or update resident numbers. For example, this subscriber can send a command update to change "ABUTTON1" number associated to a resident, as shown in figure 13.

The ADMIN user who sent the text gets an acknowledgement message.


12:22

[redacted] 07

WRITE ABUTTON1[redacted]15
sam. 22:43

WRITE ABUTTON1:OK
sam. 22:43

Figure 13: *ABUTTON1 updated through a SMS message*

4.1.3 Assumptions

Before attacking the intercom, we have to put ourself to an attacker's place to keep things real:

- the attacker don't know the operator used by the intercom;
- the attacker don't know the number associated to the SIM of the intercom;
- the targeted intercom cannot be opened;
- and commands can be retrieved with public or leaked documentations, or retrieved with a firmware analysis of the same product.

4.2 Monitoring: passive attack

In our case, we know the intercom uses GSM to communicate, but the operator and the mobile number are unknown. To get this information, we will listen to CCCHs (Common Control Channels) and try to locate the intercom.

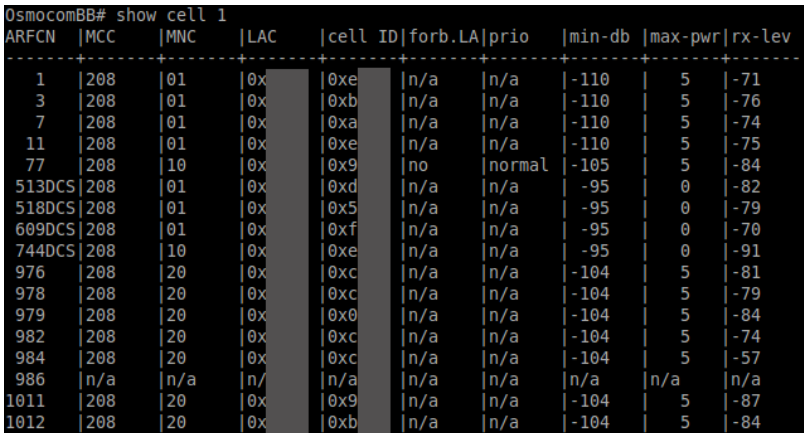
4.2.1 Looking for paging messages

To establish a call, or to receive a SMS, the MSC/VLC (Mobile Switching Center/Visitor Location Center) needs to locate the subscriber in the network. To locate this subscriber, or more precisely the subscriber, the stations send paging messages to the suscriber. If the subscriber is connected a mobile network cell, it will reply to this cell with a paging response to update its location.

To analyze these paging messages, two relevant tools exist:

- Airprobe (supported by BladeRF, RTL-SDR, USRP, and so on);
- OsmocomBB (only supported by some Motorola equipped with a Calypso baseband).

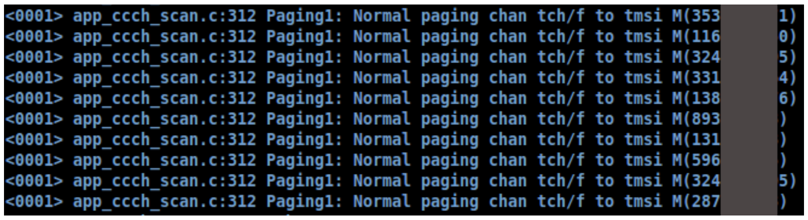
We have chosen the OsmocomBB and used the `mobile` command to walk automatically through the different ARFCN (Absolute Radio Frequency Channel Number) indexes, and list operators that surround us as shown in figure 14.



```
OsmocomBB# show cell 1
ARFCN  |MCC    |MNC    |LAC    |cell ID|forb.LA|prio   |min-db |max-pwr|rx-lev
-------+-------+-------+-------+-------+-------+-------+-------+-------+-------
   1   |208    |01     |0x[redacted] |0xe[redacted] |n/a    |n/a    |-110   |   5   |-71
   3   |208    |01     |0x[redacted] |0xb[redacted] |n/a    |n/a    |-110   |   5   |-76
   7   |208    |01     |0x[redacted] |0xa[redacted] |n/a    |n/a    |-110   |   5   |-74
  11   |208    |01     |0x[redacted] |0xe[redacted] |n/a    |n/a    |-110   |   5   |-75
  77   |208    |10     |0x[redacted] |0x9[redacted] |no     |normal |-105   |   5   |-84
 513DCS|208    |01     |0x[redacted] |0xd[redacted] |n/a    |n/a    | -95   |   0   |-82
 518DCS|208    |01     |0x[redacted] |0x5[redacted] |n/a    |n/a    | -95   |   0   |-79
 609DCS|208    |01     |0x[redacted] |0xf[redacted] |n/a    |n/a    | -95   |   0   |-70
 744DCS|208    |10     |0x[redacted] |0xe[redacted] |n/a    |n/a    | -95   |   0   |-91
 976   |208    |20     |0x[redacted] |0xc[redacted] |n/a    |n/a    |-104   |   5   |-81
 978   |208    |20     |0x[redacted] |0xc[redacted] |n/a    |n/a    |-104   |   5   |-79
 979   |208    |20     |0x[redacted] |0x0[redacted] |n/a    |n/a    |-104   |   5   |-84
 982   |208    |20     |0x[redacted] |0xc[redacted] |n/a    |n/a    |-104   |   5   |-74
 984   |208    |20     |0x[redacted] |0xc[redacted] |n/a    |n/a    |-104   |   5   |-57
 986   |n/a    |n/a    |n/[redacted] |n/a[redacted] |n/a    |n/a    |n/a    |n/a    |n/a
1011   |208    |20     |0x[redacted] |0x9[redacted] |n/a    |n/a    |-104   |   5   |-87
1012   |208    |20     |0x[redacted] |0xb[redacted] |n/a    |n/a    |-104   |   5   |-84
```

Figure 14: *Cell information with OsmocomBB*

Then we used the `ccch_scan` command of the osmocomBB tool and jumped on different ARFCN to capture messages on the CCCHs. As we can see in figure 15, many TMSIs can be collected.



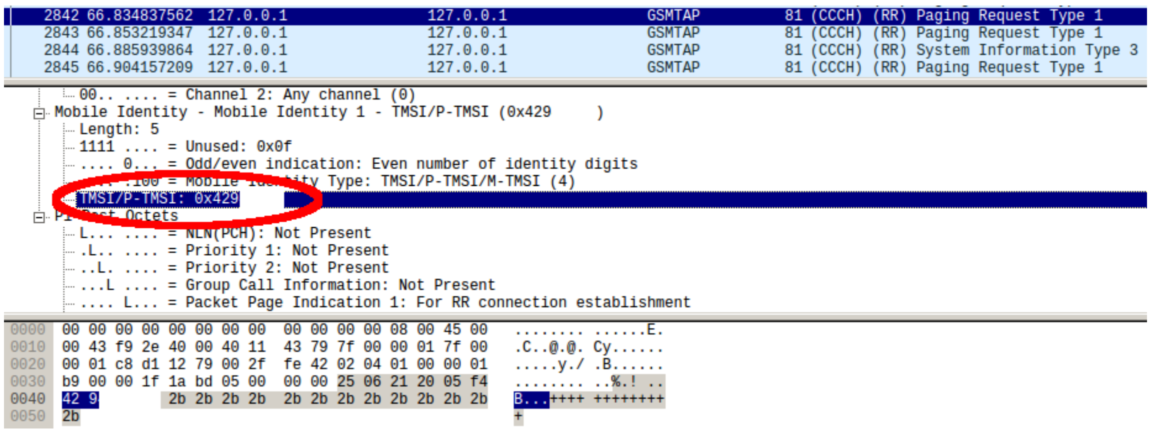
```
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(353[redacted]1)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(116[redacted]0)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(324[redacted]5)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(331[redacted]4)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(138[redacted]6)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(893[redacted])
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(131[redacted])
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(596[redacted])
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(324[redacted]5)
<0001> app_ccch_scan.c:312 Paging1: Normal paging chan tch/f to tmsi M(287[redacted])
```

Figure 15: *TMSIs leaked in paging message*

With `ccch_scan` it is also possible to perform a GSMTAP to script a frequency analyzer based on the use of the TMSI. This GSMTAP can be observed also in Wireshark as shown in figure 16.

Based on the fact that a subscriber will be paged each time someone wants to call or text him, the main idea is to send a lot of paging requests to highlight the TMSI of our target. This type of attack inspired a lot of attackers who also were looking for a way to silentely send paging requests sending SMS Class 0 messages [33] (known as Flash SMS or Silent SMS).

Nevertheless, paging requests are difficult to send without knowing the number, and paging responses are also rare for intercoms because when the intercom is installed, the resident rarely update it. Morever, to succeed in this path, a lot of sniffers have to be setup to monitor has many mobile cells as possible like in figure 17.



```
 2842 66.834837562  127.0.0.1            127.0.0.1            GSMTAP         81 (CCCH) (RR) Paging Request Type 1
 2843 66.853219347  127.0.0.1            127.0.0.1            GSMTAP         81 (CCCH) (RR) Paging Request Type 1
 2844 66.885939864  127.0.0.1            127.0.0.1            GSMTAP         81 (CCCH) (RR) System Information Type 3
 2845 66.904157209  127.0.0.1            127.0.0.1            GSMTAP         81 (CCCH) (RR) Paging Request Type 1

      00.. .... = Channel 2: Any channel (0)
   Mobile Identity - Mobile Identity 1 - TMSI/P-TMSI (0x429      )
      Length: 5
      1111 .... = Unused: 0x0f
      .... 0... = Odd/even indication: Even number of identity digits
      .... .100 = Mobile Identity Type: TMSI/P-TMSI/M-TMSI (4)
      TMSI/P-TMSI: 0x429
   P1 Rest Octets
      L... .... = NLN(PCH): Not Present
      .L.. .... = Priority 1: Not Present
      ..L. .... = Priority 2: Not Present
      ...L .... = Group Call Information: Not Present
      .... L... = Packet Page Indication 1: For RR connection establishment

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ........ ......E.
0010  00 43 f9 2e 40 00 40 11  43 79 7f 00 00 01 7f 00   .C..@.@. Cy......
0020  00 01 c8 d1 12 79 00 2f  fe 42 02 04 01 00 00 01   .....y./ .B......
0030  b9 00 00 1f 1a bd 05 00  00 00 25 06 21 20 05 f4   ........ ..%.! ..
0040  42 9        2b 2b 2b 2b  2b 2b 2b 2b 2b 2b 2b 2b   B...++++ ++++++++
0050  2b                                                 +
```

Figure 16: *GSMTAP with OsmocomBB and Wireshark to leak TMSIs*

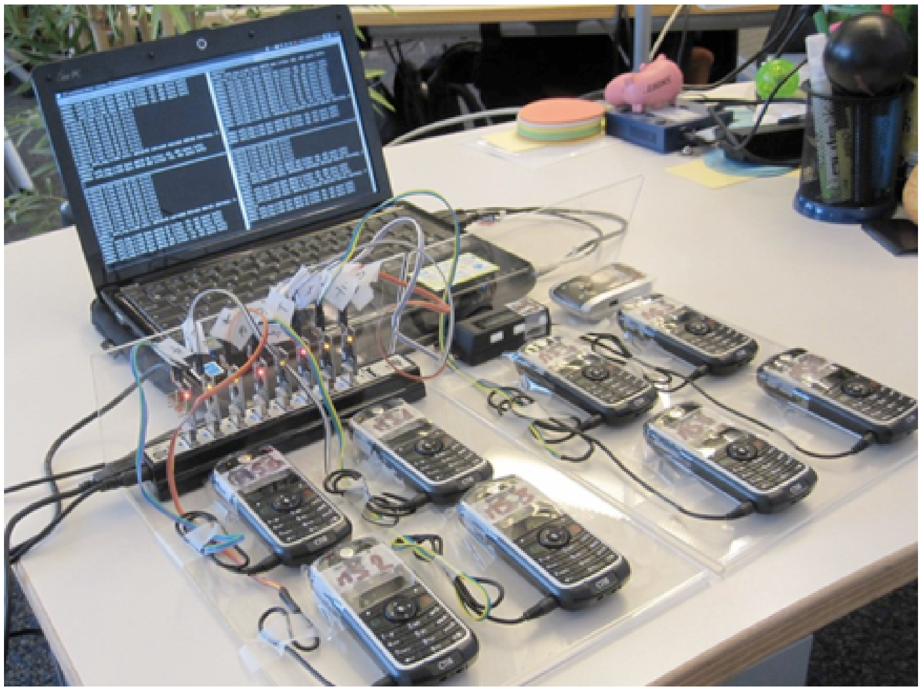


Figure 17: *GSMTAP with more OsmocomBB phones (source: malanris.ru)*

An other way would be to use some Social Engineering tricks to ask to a resident the number displayed by its intercom. But in our case, we will make use of active attacks to attract this intercom and retrieve the operator MCC/MNC first.

4.3 Trapping the intercom: active attack

Baseband behaviors are sometimes unpredictable when it comes to handover, even if specification make this clear. As far as we would know, a mobile phone is always looking for better signal. But with a certain experience, researchers observed also that a baseband can decide to handover if:

- it can register to any MCC/MNC BTS close to it;
- it can register to a test network close to it. This behavior has been observed in some cases with a foreign SIM/USIM card, a 4 years old Qualcomm baseband in the HTC Desire S/Z, and cellphones that use the GSM stack only;
- the current used network isn't reachable anymore;
- the signal is strong and the mutual authentication succeeded (not the problem in GSM/GPRS for an attacker).

To attract the Link GSM iDP we used different MCC/MNC codes, and wait 15 minutes to let a chance to our rogue station to trap the intercom. After few minutes with a MCC/MNC that belongs to the operator used by the user equipment, the Link intercom connects to our rogue station. The in figure 18 shows the GSM lab while attacking the intercom.

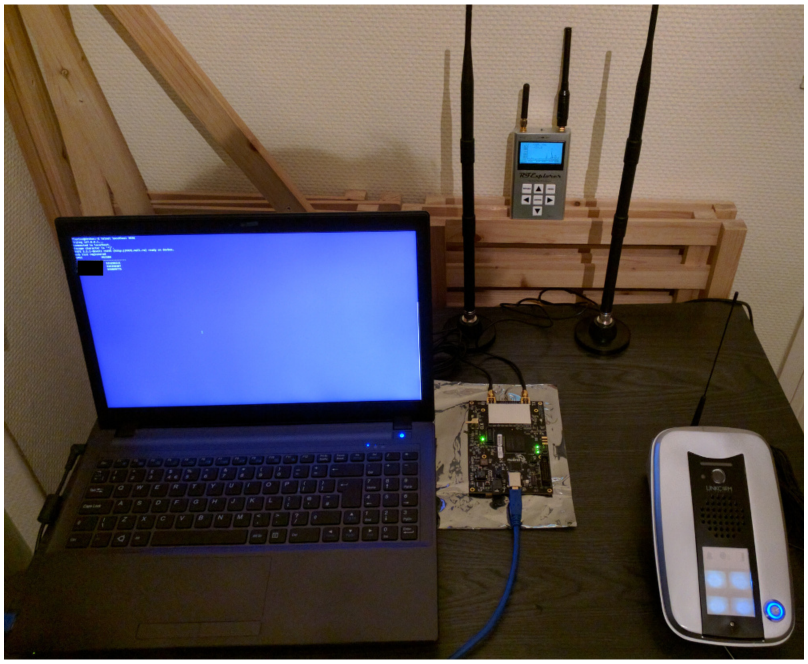


Figure 18: *The Link iDP GSM intercom attacked by the rogue BTS*

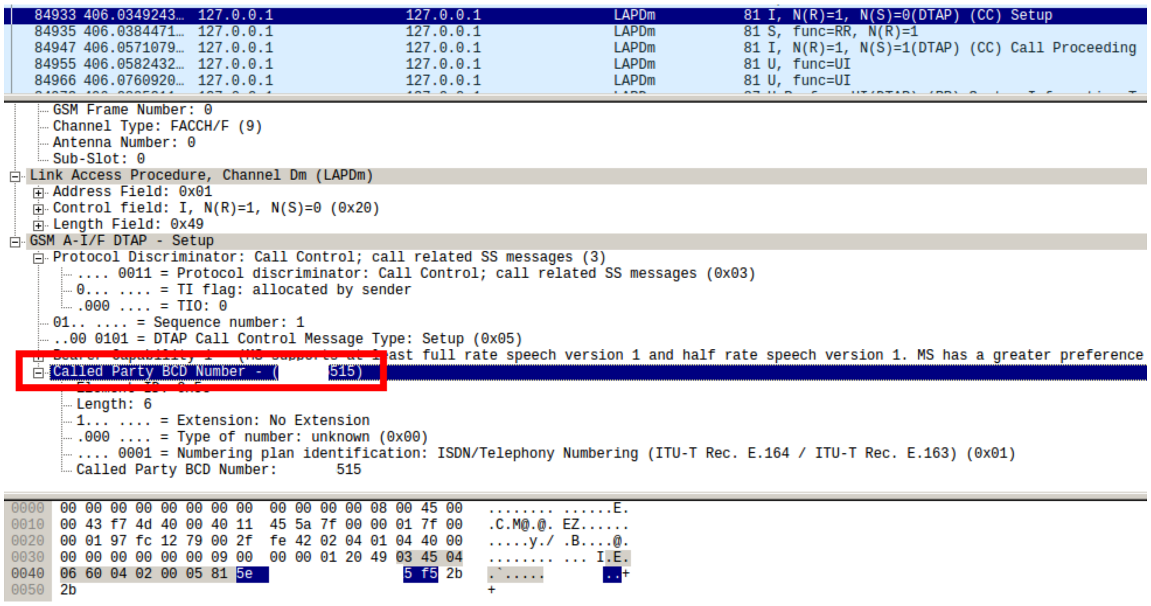
4.3.1 Leaking numbers

When the intercom is trapped in the rogue station, we have now the full control of the routing, and we are able to leak the numbers saved in the intercom just by pressing on calling buttons. Like OpenBTS, the YateBTS software is capable of opening a GSMTAP UDP socket when enabling the feature in the `ybts.conf` like in figure 21.

```
[tapping]
; GSM: boolean: Captures GSM signaling at the L1/L2 interface via GSMTAP.
; Do not leave tapping enabled after finishing troubleshooting.
; Defaults to no.
GSM=yes
```

Figure 19: *Enabling GSMTAP in ybts.conf*

The figure 20 shows the leaked number associated with the "ABUTTON1" decoded by Wireshark.



```
 84933 406.0349243…  127.0.0.1             127.0.0.1             LAPDm           81 I, N(R)=1, N(S)=0(DTAP) (CC) Setup
 84935 406.0384471…  127.0.0.1             127.0.0.1             LAPDm           81 S, func=RR, N(R)=1
 84947 406.0571079…  127.0.0.1             127.0.0.1             LAPDm           81 I, N(R)=1, N(S)=1(DTAP) (CC) Call Proceeding
 84955 406.0582432…  127.0.0.1             127.0.0.1             LAPDm           81 U, func=UI
 84966 406.0760920…  127.0.0.1             127.0.0.1             LAPDm           81 U, func=UI

    GSM Frame Number: 0
    Channel Type: FACCH/F (9)
    Antenna Number: 0
    Sub-Slot: 0
 Link Access Procedure, Channel Dm (LAPDm)
    Address Field: 0x01
    Control field: I, N(R)=1, N(S)=0 (0x20)
    Length Field: 0x49
 GSM A-I/F DTAP - Setup
    Protocol Discriminator: Call Control; call related SS messages (3)
       .... 0011 = Protocol discriminator: Call Control; call related SS messages (0x03)
       0... .... = TI flag: allocated by sender
       .000 .... = TIO: 0
    01.. .... = Sequence number: 1
    ..00 0101 = DTAP Call Control Message Type: Setup (0x05)
                                      ast full rate speech version 1 and half rate speech version 1. MS has a greater preference
    Called Party BCD Number - ([redacted]515)
       Length: 6
       1... .... = Extension: No Extension
       .000 .... = Type of number: unknown (0x00)
       .... 0001 = Numbering plan identification: ISDN/Telephony Numbering (ITU-T Rec. E.164 / ITU-T Rec. E.163) (0x01)
       Called Party BCD Number: [redacted]515

0000  00 00 00 00 00 00 00 00  00 00 00 00 08 00 45 00   ........ ......E.
0010  00 43 f7 4d 40 00 40 11  45 5a 7f 00 00 01 7f 00   .C.M@.@. EZ......
0020  00 01 97 fc 12 79 00 2f  fe 42 02 04 01 04 40 00   .....y./ .B....@.
0030  00 00 00 00 00 00 09 00  00 00 01 20 49 03 45 04   ........ ... I.E.
0040  06 60 04 02 00 05 81 5e [redacted] 5 f5 2b   .`..... [redacted]..+
0050  2b                                                 +
```

Figure 20: *Leaked number from rogue station GSMTAP after pushing the button 1*

Refering to the documentations of the Link iDP GSM intercom[30], it is possible to register `ALARMON` and `ALARMOFF` numbers. To trigger these alarms, some intercoms are sensible to button bruteforcing, or issues with the door system, but other behaviors could also trigger them. So if we are able to trigger these alarms, we will be probably be able to capture one of the administrator number in the case the intercoms doesn't have any button associated to this adminstrator.

4.3.2 Door opening

Thanks to leaked numbers, we can register ourselves as a resident just by modifying the `tmsidata.conf` configuration file displayed in figure 21.

```
[tmsi]
last=007b0005

[ues]
20820XXXXXXXXXX=007b0003,35547XXXXXXXXXX,XXXXXX515,1460XXXXXX,ybts/TMSI007b0003
# associating attacker IMSI with a resident number
[...]
```

Figure 21: *Affecting a resident number to an arbitrary IMSI in* ***tmsidata.conf***

When this file is reloaded to YateBTS, we are able to capture the traffic with GSMTAP. The tiny Python script in figure 22 allows us to reload the NIB after editing the TMSI database of YateBTS through Telnet.

```
import telnetlib
import time

tn = telnetlib.Telnet("localhost", 5038)
time.sleep(1)
print tn.read_until("\n")
time.sleep(1)
print tn.read_until("")
time.sleep(1)
tn.write("javascript reload nib\n")
time.sleep(1)
print tn.read_until("")
```

Figure 22: *script to reload the NIB with Python*

Then, when pressing the targeted resident's buttons the intercom call our mobile phone that is connected to our rogue GSM network, and we are able to open the door to penetrate into the building. It should be noted that, with an administrator number, this method could have dangerous impacts because as we already know the attacker will be able to control the intercom.

4.3.3 Backdooring

After leaking the administrator number with `ALARMON`, `ALARMOFF`, social engineering, or other methods, we can use the same tricks explained in section 4.3.2 to impersonate an administrator and send commands to the intercom. The new difficulty here is to find the commands accepted by the targeted intercom.

To find these commands, two main ways exist:

- look for public or leaked documentations of the targeted intercom;
- or buy the model in sites classified ads, like "Leboncoin.fr" (in France), dump the firmware and reverse it.

In our case, Link iDP GSM manual is public [30] and describes also commands that can be sent through SMS messages.

So reading the manual we can highlight some commands that interest us to read and write paramaters:

Command	Desciption
READ <NAME>	Read the number of a button, or an admin (ADMIN[1-9]).
WRITE <NAME> <number>	Add or update a number associated to a name.
CAL AT<command suffix>	Send an AT command to the baseband through SMS!

Note that AT commands can be sent through SMS, so an attacker impersonating a adminstrator would be able to:

- retrieve SMS messages sent by managers or residents with the command `AT+CMGL="ALL"`;
- spying building door conversations, when setting the Auto-answer parameter with the command `ATS0=1` (0: no auto-answer, 1: GSM module goes off-hook after the first ringing signal);
- and so on.

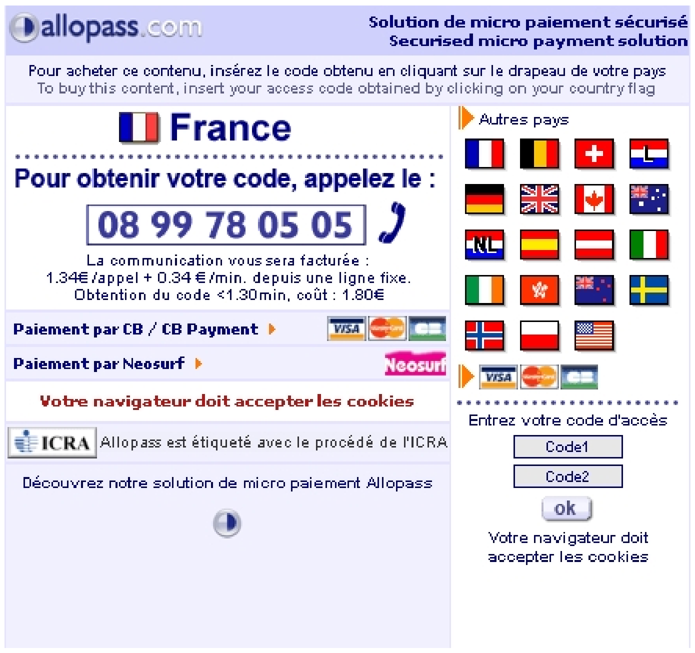
4.3.4 Call premium rate numbers and make money!

As we are now able to write any number we want, why we couldn't make money out of this hack? All we need is to add or update a resident number with the following premium rated numbers:

- Allopass;
- Optelo;
- Hipay;
- and so on.

As an example, code given after calling the Allopass service can be used to fill a personal account. Then, these valid codes just have to be entered in our Allopass form (figure 23).

Note that the quality of the speaker as to be good enough to understand the code given when calling the Allopass with the intercom. But every problem has its solution. Indeed, services like Allopass provide also dedicated numbers, so the attacker will be able to earn money without being forced to stay near the compromized intercom.


allopass.com

Solution de micro paiement sécurisé
Securised micro payment solution

Pour acheter ce contenu, insérez le code obtenu en cliquant sur le drapeau de votre pays
To buy this content, insert your access code obtained by clicking on your country flag

France

Pour obtenir votre code, appelez le :

08 99 78 05 05

La communication vous sera facturée :
1.34€ /appel + 0.34 € /min. depuis une ligne fixe.
Obtention du code <1.30min, coût : 1.80€

Paiement par CB / CB Payment

Paiement par Neosurf

Votre navigateur doit accepter les cookies

ICRA Allopass est étiqueté avec le procédé de l'ICRA

Découvrez notre solution de micro paiement Allopass

Autres pays

Entrez votre code d'accès

Code1

Code2

ok

Votre navigateur doit accepter les cookies

Figure 23: *The standart Allopass form*

5 Work in progress

5.1 Evolution of the lab

To perform tests on intercoms outside, the lab has been reduced replacing the laptop by a Raspberry pi 3 device that is powered with a 12000mAh battery 5V/2.4A output. That lab costs around 540 euros.

Note that antennas used in figure 24 could be replaced by smaller ones with less gain. If it do not hold a charge, the Raspberry pi 3 would be replaced by an Odroid device to support the rogue GSM basestation and the 3G jamming modules.

Actually, the device exposes a Wi-Fi hostpot to control the device even from a cellphone, or a tablet. Using the scapy [36] and libmich [37] Python modules, resident's numbers are can be easily and automatically extracted from GSMTAP UDP frames as shown in figure 25:

5.2 Intercoms with M2M

We have recently bought a 3G intercom that use a M2M network as shown in figure 26.

The main difference with the architecture presented previously is that the administration is performed from a website linked to the Machine2Machine (M2M) architecture. Nevertheless, this architecture is very interesting because it introduces a new vector of attack. Indeed, in a short time while configuring the intercom, we have found a vulnerability on the website that to control

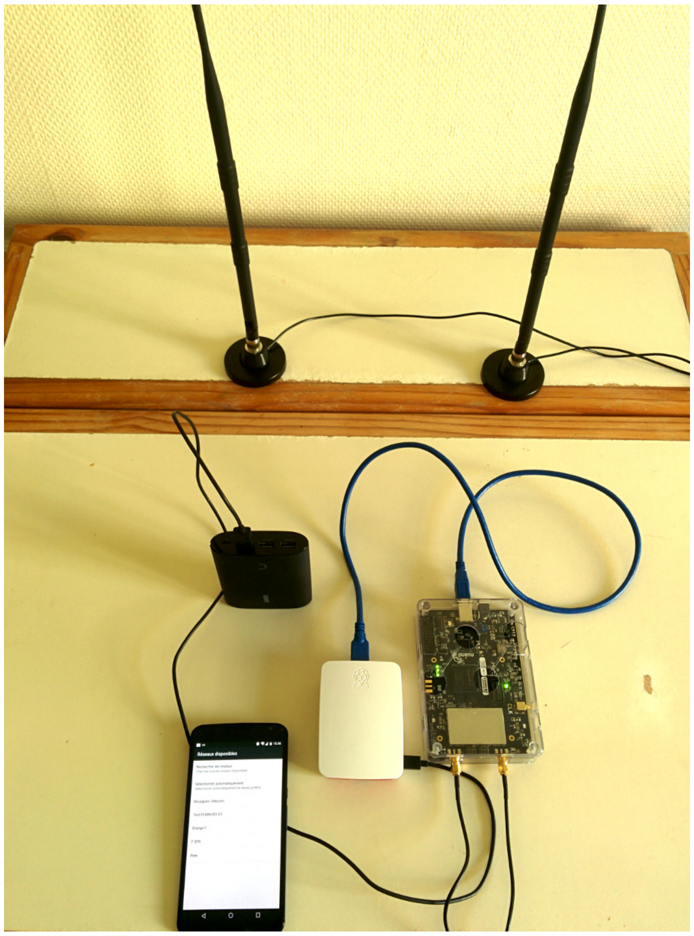


Figure 24: *New lab with a Raspberry pi 3*

```
In [24]: import binascii
In [25]: from scapy.all import Ether
In [26]: from libmich.formats.L3Mobile import *
In [27]: pkt = "0000000000000000000000000800450000437fa240004011bd057f0000017f0000019e941279002ffe"
         pkt += "4202040104400000000000000009000000012253034504066004020005815e068133236015f51502"
In [28]: t = binascii.unhexlify(pkt)
In [29]: p = Ether(t)
In [30]: parse_L3(p["UDP"].load[19:]).CalledBCD.V.Num
Out[30]: 333206515 # leaked resident number
```

Figure 25: *Extracting numbers with scapy+libmich*

no just one, but a range of intercoms in the same time and we were able to locate them thanks to public directories when the home number was registed to a button.

We talk earlier in this paper about 3G downgrade attacks, but attack of M2M websites represent a new threat because it could be performed just with a simple internet connection, without the need of any Software-Defined Radio device.

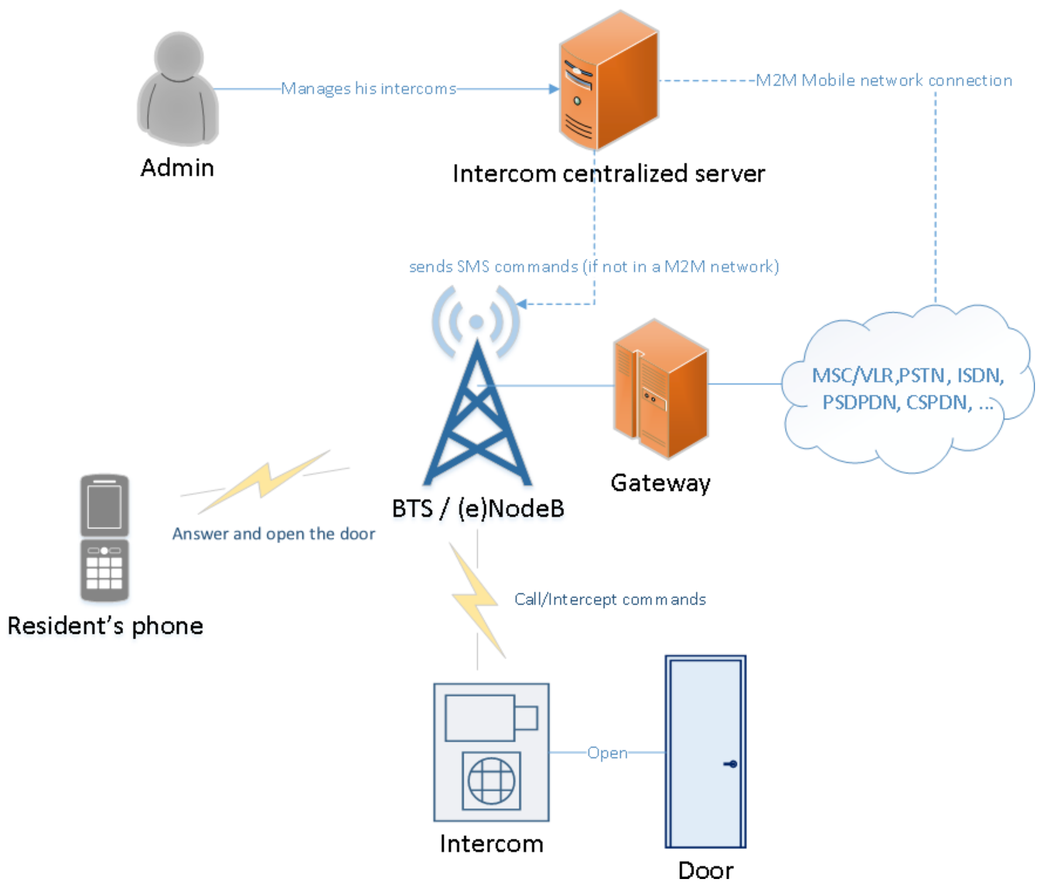

Admin
Manages his intercoms
Intercom centralized server
M2M Mobile network connection
sends SMS commands (if not in a M2M network)
MSC/VLR,PSTN, ISDN, PSDPDN, CSPDN, ...
Gateway
BTS / (e)NodeB
Answer and open the door
Resident's phone
Call/Intercept commands
Open
Intercom
Door

Figure 26: *Simple architecture of a M2M intercom connection*

Moreover, M2M intercoms are connected to a virtual network provided by the used operator. The use of the provided USIM card could be very interesting to study. Indeed, as observed, constructors of intercom provide a M2M USIM card with more than 10 years life operator subscription, but these USIM cards are often protected by a PIN.

Nevertheless, the PIN can quickly be recovered with a SIMtrace device [41], used as a proxy between the SIM/USIM card and the user equipment, to capture APDU messages as shown in figure 27:

As the intercom automatically types the PIN code of the USIM card, the APDU associated to the VERIFY_PIN command is captured. After decoding the APDU, we have the PIN code to use with the USIM card. With more time it could be interesting to use that USIM card with a spoofed IMEI (International Mobile Equipment Identity) to get more informations about the virtual network for example.

For the moment we are focusing on a the complet chain to downgrade the 3G level of the intercom to perform the interception on this device to present the complet Proof Of Concept

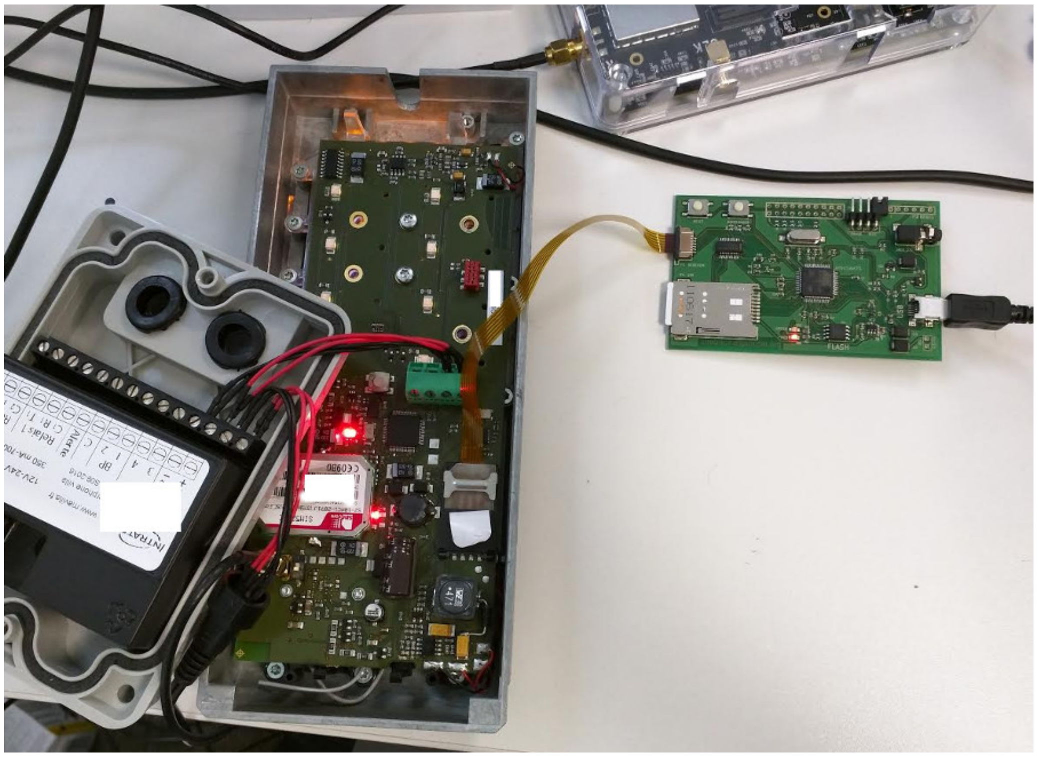


Figure 27: *Capturing PIN code typed by the intercom*

soon. Few details about the vulnerabilities of M2M intercom will also be presented to compare the difficulty, and the impact of the two approaches:

- compromizing a 3G intercom remotely with Soft-Defined Radio;
- and compromizing a 3G intercom with a M2M binded website vulnerability.

5.3 Firmware analysis and bug hunting

Other intercoms in 3G will be analysed to extract the commands sent over the air, but also other interesting researches would be made on the mobile module, that we suppose, is rarely updated. This work will be part of our next studies on the subject.

6 Summary

Combining different researches in the mobile security field, we are able to attract an intercom to our rogue base station, impersonate any legit subscriber user and administrator, backdoor the intercom and update a resident number to a premium rated number to make money. Recently were are also able to show the impact when compromizing M2M intercoms.

We are currently working on tools to automate the interception attacks on GSM and 3G. We are also looking on other samples of 3G intercoms, but possible 4G intercoms too, to complet this paper with practical downgrade attacks using cheap equipments. Moreover, a firmware analysis on targeted basebands would be interesting to find bugs in protocols stacks, but also to find maybe other ways to backdoor these devices.

More results are to come, and we hoppe we will be able to present them at 33c3.

To finish, it should be highlighted that the IoT ecosystem uses mobile networks a lot, and intercoms are not the only devices affected by these attacks.

References

[1] Intercom, Wikipedia definition - https://en.wikipedia.org/wiki/Intercom

[2] Doophone, Wikipedia definition - https://en.wikipedia.org/wiki/Door_phone

[3] RFCat tool - https://bitbucket.org/atlas0fd00m/rfcat

[4] A GSM based remote control for intercoms by Oliver Nash, http://olivernash.org/2009/10/31/locked-out-at-2am/

[5] GSM - SRSLY?, at 26c3 by Kastern Nohl and Chris Paget - https://events.ccc.de/congress/2009/Fahrplan/attachments/1519_26C3.Karsten.Nohl.GSM.pdf

[6] Running your own GSM stack on a phone, at 27c3 by Harald Welte and Steve Markgraf - https://events.ccc.de/congress/2010/Fahrplan/attachments/1771_osmocombb-27c3.pdf

[7] GSM Sniffing, at 27c3 by Sylvain Munaut and Karsten Nohl - https://events.ccc.de/congress/2010/Fahrplan/attachments/1783_101228.27C3.GSM-Sniffing.Nohl_Munaut.pdf

[8] IMSI Catcher, by Daehyun Strobel - https://www.emsec.rub.de/media/crypto/attachments/files/2011/04/imsi_

[9] Femtocells: a Poisonous Needle in the Operator's Hay Stack, Back Hat 2011 by Ravishankar Borgaonkar, Nico Golde, Kévin Redon - https://www.isti.tu-berlin.de/fileadmin/fg214/bh2011.pdf

[10] The Vodafone Access Gateway / UMTS Femto cell / Vodafone Sure Signal, by THC - https://wiki.thc.org/vodafone

[11] Practical attacks against privacy and availability in 4G/LTE mobile communication systems, by Altaf Shaik, Ravishankar Borgaonkar, N. Asokan, Valtteri Niemi and Jean-Pierre Seifert - http://arxiv.org/pdf/1510.07563v1.pdf

[12] OpenBTS project website - http://openbts.org/

[13] YateBTS project website - http://yatebts.com/

[14] USRP ETTUS website - https://www.ettus.com/

[15] bladeRF website - http://nuand.com/

[16] HackRF One - https://greatscottgadgets.com/hackrf/

[17] GNU Radio website - http://gnuradio.org/redmine/projects/gnuradio/wiki

[18] openLTE project - http://openlte.sourceforge.net/

[19] ETSI TS 145 002 - http://www.etsi.org/deliver/etsi_ts/145000_145099/145002/09.03.00_60/ts_145002v090300p.p

[20] A5/1 algorithm - http://www.scard.org/gsm/a51.html

[21] ETSI TS 141 061 - http://www.etsi.org/deliver/etsi_ts/141000_141099/141061/04.00.00_60/ts_141061v040000p.p

[22] LTE Security, 2nd Edition by Dan Forsberg, Gunther Horn, Wolf-Dietrich Moeller, Valtteri Niemi

[23] Specification of 3GPP Confidentiality and Integrity Algorithms - http://www.garykessler.net/library/crypto/3G_KASUMI.pdf

[24] Decrypting GSM phone calls - https://srlabs.de/decrypting_gsm/

[25] Installation et raccordement du bloc 3G (Intratone) - http://www.intratone.fr/media/bloc_3g__067364900_1850_20082014.pdf

[26] (Noname) INTERPHONE AUDIO GSM - http://www.laboutiqueduportail.com/audio/419-interphone-audio-gsm-.html?gclid=CNL5zrTg3ssCFUmeGwodvI8KnQ

[27] You can ring my bell! Adventures in sub-GHz RF land, by Adam Laurie - http://adamsblog.aperturelabs.com/2013/03/you-can-ring-my-bell-adventures-in-sub.html

[28] bladeRF x115 from Nuand website - https://www.nuand.com/blog/product/bladerf-x115/

[29] Link GSM iDP - http://www.linkcom.fr/en/product-detail/link-gsm-idp/

[30] LinkCom iDP GSM manual - http://downloads.linkcom.fr/DoorPhone/Link_GSM_iDP/Manuel/Link_GSM_iD

[31] OsmocomBB - http://osmocom.org/projects/baseband

[32] Airprobe - https://svn.berlin.ccc.de/projects/airprobe/

[33] 3G TS 23.038, Technical Specification Group Terminals; Alphabets and language-specific information - ftp://www.3gpp.org/tsg_t/TSG_T/TSGT_04/Docs/PDFs/TP-99127.pdf

[34] Analyse sécurité des modems des terminaux mobiles by Benoit Michau at SSTIC 2014 - https://www.sstic.org/2014/presentation/Analyse_securite_modems_mobiles/

[35] How to not break LTE crypto by Benoit Michau and Christophe Devine - https://www.sstic.org/media/SSTIC2016/SSTIC-actes/how_to_not_break_lte_crypto/SSTIC2016-Article-how_to_not_break_lte_crypto-michau_devine.pdf

[36] Scapy Python module - http://www.secdev.org/projects/scapy/

[37] Easy developpement of formats [de]coding for network protocols - https://github.com/mitshell/libmich

[38] Convert log messages by phones with XGold baseband processor back to GSM/UMTS radio messages - https://github.com/2b-as/xgoldmon

[39] Discussion of ZUC algorithm designed by DACAS - http://zucalg.forumotion.net/

[40] SnoopSnitch: Android app that collects and analyzes mobile radio data - https://opensource.srlabs.de/projects/snoopsnitch

[41] SIMtrace to capture SIM/USIM APDUs - https://osmocom.org/projects/simtrace/wiki/SIMtrace