GSM Sniffing with OsmocomBB

Joshua Pereyda

Introduction

- In November 2011, Karsten Nohl and Sylvain Munaut presented a passive sniffing attack on modern cell phone systems
- My goal was to reproduce this attack
- Secondary goals:
  - Recycle as much software as possible
  - Stay under $100

Outline

- Introduction to GSM
- The Attack
- My attempt
  - Hardware
  - Software

GSM Sniffing with OsmocomBB

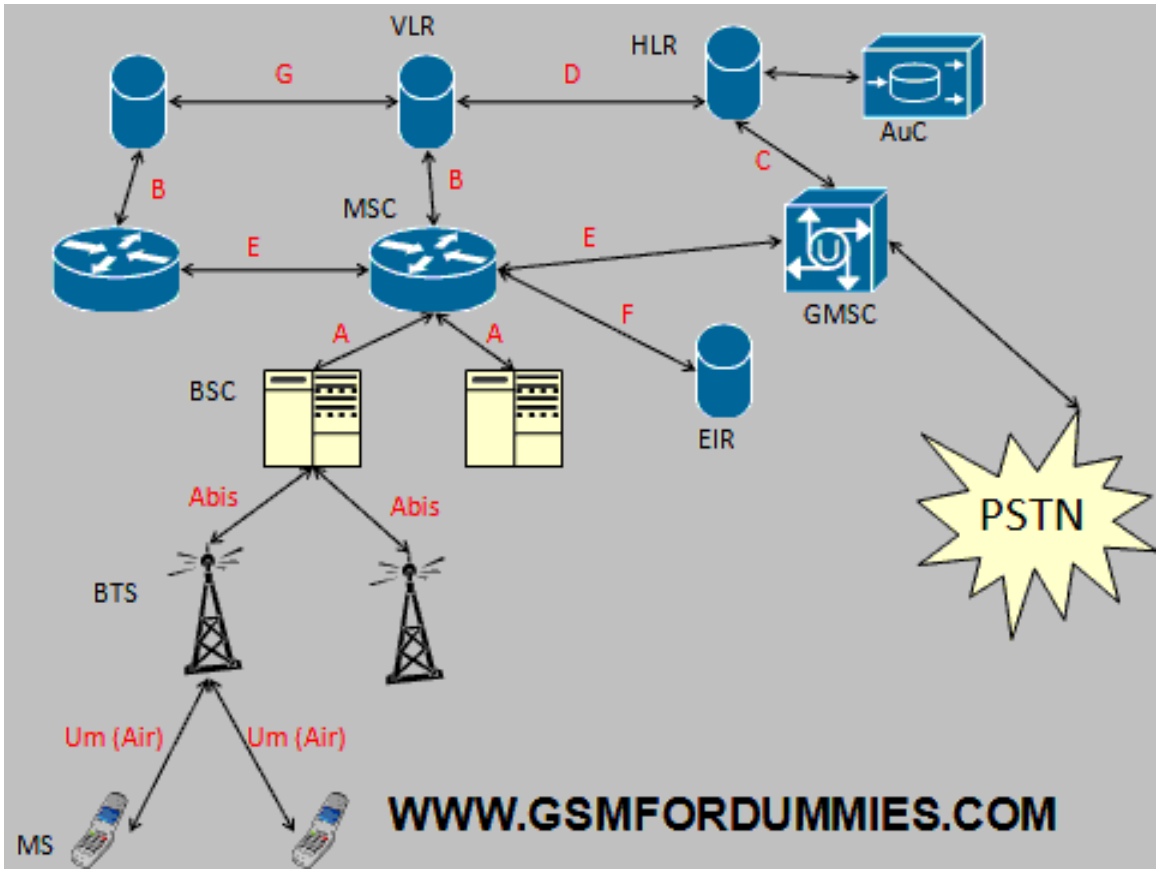
INTRODUCTION TO GSM

Brief GSM Intro

- Global System for Mobile Communications (GSM)
  - Published in 1990
  - Includes: Mobile Station (MS) to Base Transfer Station (BTS) communications and backend infrastructure details
- Credit: http://gsmfordummies.com/

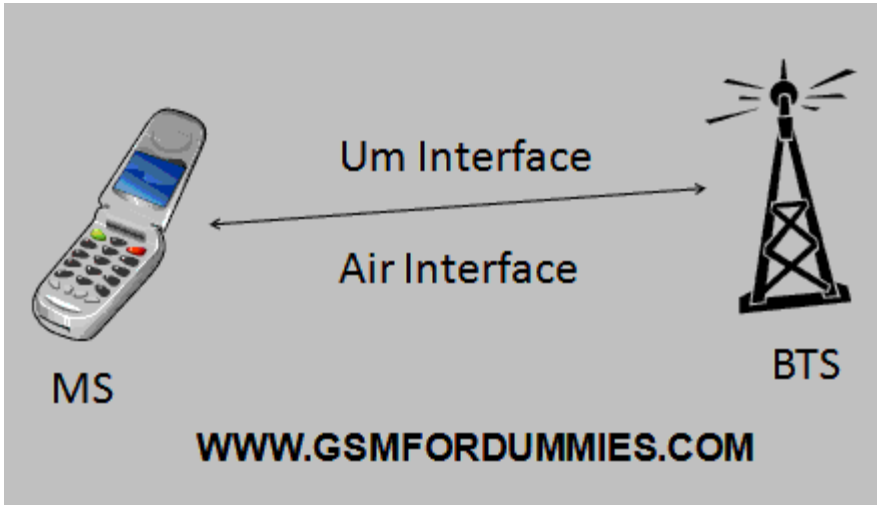
GSM

High-level View of GSM Architecture (for context):



GSM

- While all elements of the GSM system could be a point-of-attack, the Um (or Air) interface between phone and tower presents the most accessible target:



GSM Security

- Authentication is strong, encipherment weak
- A5 Algorithm: Used for Um encryption
  - A5/0 – No encryption
  - A5/1 – Most common cipher (say Nohl and Munaut)
  - A5/2 – Deliberately weakened algorithm
    - Supposedly for export to untrusted nations
  - A5/3 – Newer algorithm, “academically” broken
- The GSM protocol is also vulnerable to active MITM attacks (not covered here).

GSM Security

- Frequency hopping is utilized for phone calls
  - Reduces Noise
  - Hops are unpredictable to enhance security

GSM Terminology

- GSM Uses a *lot* of acronyms.
  - MSISDN – Mobile Subscriber ISDN Number
    - ISDN: Integrated Services Digital Network
    - AKA “Phone number”
    - “This abbreviation has several interpretations” depending on the standardization body. (http://en.wikipedia.org/wiki/MSISDN)
  - International Mobile Equipment Identity (IMEI)
    - Identifies phone (NOT the SIM card or subscriber)
    - More recently IMEISV (includes software version)

GSM Terminology

- MCC – Mobile Country Code
- MNC – Mobile Network Code
  - Identifies the carrier (e.g., AT&T)
- IMSI – International Mobile Subscriber Identity
    - Associated with SIM card
    - Composed MCC, MNC, and MSIN
    - Uniquely identifies an individual caller

GSM Terminology

- TMSI – Temporary Mobile Subscriber Identity
  - A sort of session ID; typically reused several times
  - Forms ID in Um interface communications
  - Used to obfuscate a caller's identity (IMSI is still transmitted in initialization)
    - Only used in cipher mode (A5/1, A5/2, or A5/3)

GSM Terminology

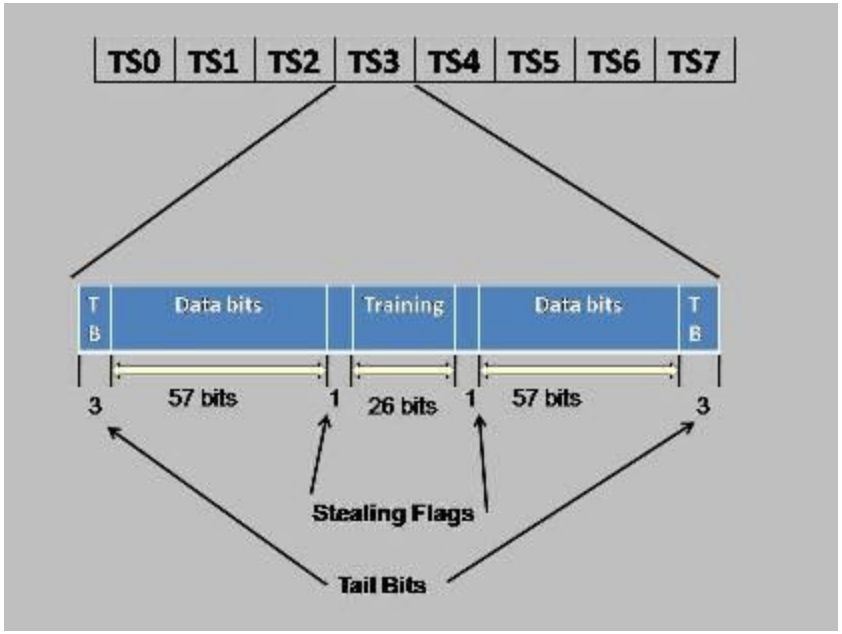
- Ki – Master key used for authentication
  - Conjecture: ‘i’ stands for initialization
- Kc – Session key used for ciphering
  - Derived from Kc and cryptographic exchange
- ARFCN – Absolute Radio Frequency Number
  - Uniquely identifies a specific radio frequency which can be calculated from the number

GSM Terminology

- CCCH – Common Control Channels
- Relevant Common Control Channels:
  - PCH – Paging Channel
    - Used to notify MS of an incoming transmission.
  - RACH – Random Access Channel
    - Like PCH, but used by MS to notify BTS of an outgoing transmission.

GSM Terminology (not acronyms)

- Burst – A sort of low-level packet; transmitted in one Time Slot (TS) of one logical channel
  - Normal bursts have 114 bits of payload data



Bear Holding a Shark

THE ATTACK

The Attack

- Nohl and Munaut presented the first (cheap) public, passive, real-world eavesdropping attack.
- The attack only requires knowledge of the phone number ahead of time.

The Attack

1. Find a phone's location area using the internet.
   - Not necessary in this experiment.
2. Get TMSI from phone number.
3. Record encrypted data.
4. Use Kraken to get key.
5. Decrypt communications and follow frequency hops (for voice calls).

Acquiring TMSI

- The TMSI is necessarily transmitted in plaintext at least once per communication.
- Send text messages to target number, listen to Paging Channel (PCH) for TMSIs.
- Repeat to narrow down until a unique TMSI is found.

Harvest Encrypted Bursts

- Following the initial MS/BTS exchange will allow us to grab several encrypted packets.
- These packets contain a significant amount of known plaintext.
- We will feed this data to Kraken…

Kraken

- Open source software used to crack A5/1 cipher.
- Setup is tricky, but runs in minutes once up.
- Requires about 2 TB of spare disk space.
  - Time/memory tradeoff attack

Receiving Software

- Must decrypt communications, acquire new channel number, and switch channels in time.
  - Not publicly released.

Well that all sounds pretty easy…

MY ATTEMPT

My Attempt – Hardware

- Hardware consists of one old-school phone and USB to UART bridge with 2.5mm audio adapter.
- The phone uses its 2.5mm audio jack as a serial port.
- The USB/UART bridge is used to connect to my computer.
  - Note, normal computer serial ports may fry the phone due to voltage levels!

My Attempt – Hardware

- Osmocom (phone software) only supports certain phones:
  - Primary target: Motorola C123 – Not easy to find
  - Secondary target: C155 – Available on Amazon
  - A handful of others are supported.
- C155
  - $10 each
  - One for attacker, one for target.
  - $28.38 total with shipping

My Attempt – Hardware

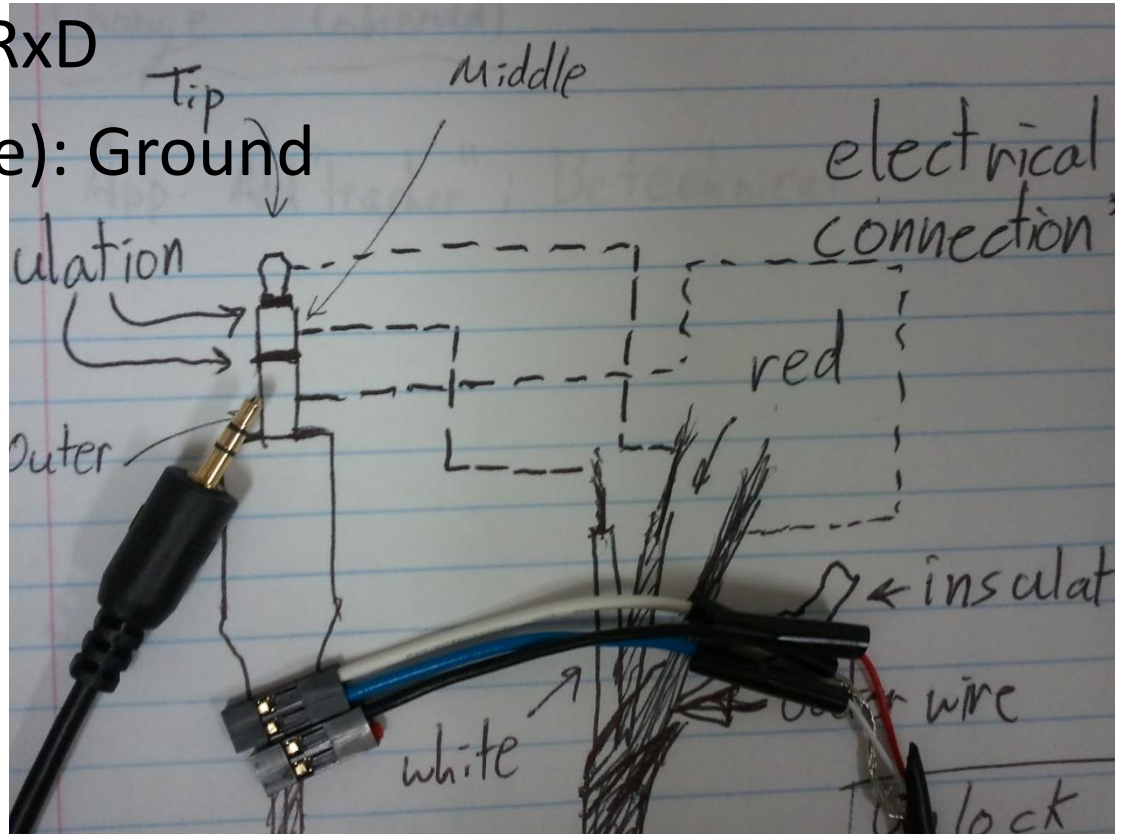
- USB/UART Bridge
  - Need FDTI or CP210x for non-standard baud rates
  - Baud – symbols/second
    - in our case, 1 symbol = 1 bit
  - $5.25 on Amazon
    - $10.98 with shipping

My Attempt – Hardware

- 2.5mm to CP210x bridge cables are not commonly available
  - 2.5mm cables: $9.29 total
  - Dr. Rinker donated some wires and soldering tools/skills

My Attempt – Hardware

- Wiring
  - Tip (red wire): TxD (Transmit PC->Device)
  - Middle (white): RxD
  - Outer (outer wire): Ground



My Attempt – Hardware

- Programming CP2012:
  - cp210x-program – OSS for programming CP210x
  - See http://bb.osmocom.org/trac/wiki/Hardware/CP210xTutorial for full instructions
  - End up with non-standard baudrates

My Attempt – Hardware

- That *was* easy!

Now for the fun part!

MY ATTEMPT: SOFTWARE

My Attempt – Software

- Software components:
  - OmsocomBB – Custom phone software
  - Unavailable: Program to get TMSI from phone number (using Osmocom)
  - Unavailable: Program to record encrypted data
  - Kraken – Cracking program
  - Unavailable: Program to listen, decrypt messages, and follow frequency hops.
  - Unavailable: Program to decode voice data.

My Attempt – Software

- My goals:
  - OmsocomBB – Custom phone software
  - **Create program to get TMSI from phone number (using Osmocom)**
  - **Create program to record encrypted data**
  - Kraken – Cracking program
    - Need interfacing program
  - **Create program to record text message**
  - Must get Osmocom running before anything else.

My Attempt – Software

- OsmocomBB – Baseband software
  - "OsmocomBB is an Free Software / Open Source GSM Baseband software implementation."
    - Can even be used to make calls and send messages.
  - Sylvain Munaut created Sylvain/burst_ind branch for sniffing; burst_ind gets raw low-level burst data.
  - git clone git://git.osmocom.org/osmocom-bb.git -b sylvain/burst_ind

My Attempt – Software

- OsmocomBB difficulties:
  - Sparse documentation
    - Some out of date wiki pages
  - Not designed for end users
  - A general cloud of unknowing on my part (“noob”)
  - Result: A somewhat frustrating experience.

My Attempt – Software

- OsmocomBB – Compiling Issues
  - Largest compiler hold-up: Bad cross-compiler
    - A cross-compiler is used to compile programs for other machines.
  - Resulted in mysterious error messages during Osmocom compiling process.
  - Tried on two different systems before finally switching the cross-compiler – then it worked!

http://bb.osmocom.org/trac/wiki/GettingStarted
http://bb.osmocom.org/trac/wiki/toolchain

My Attempt – Software

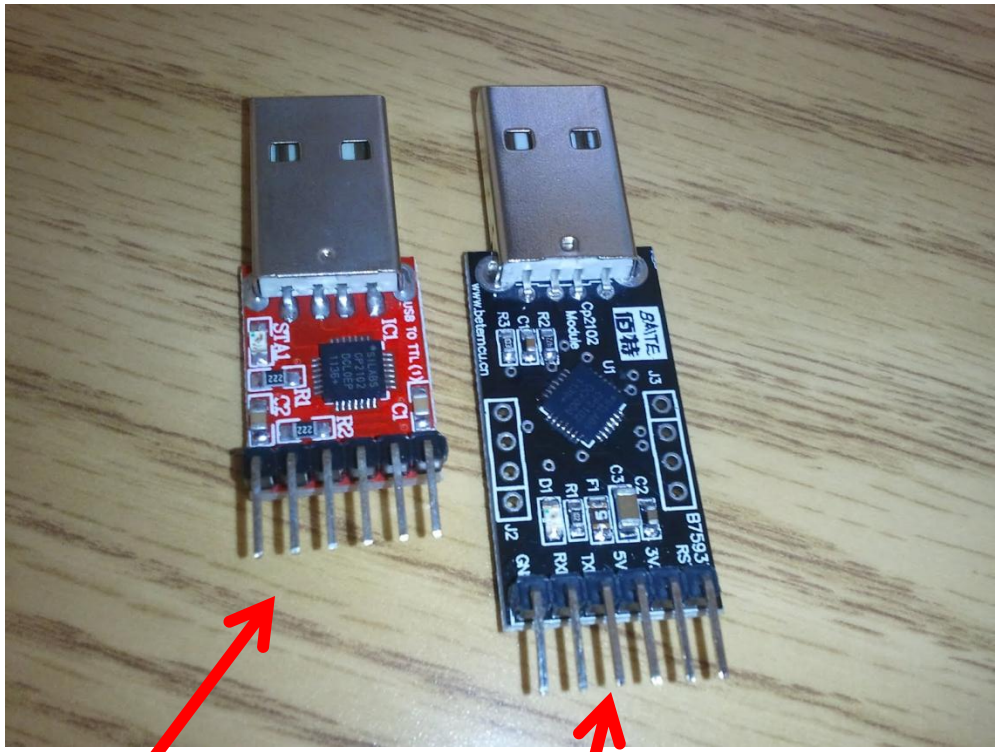
- OsmocomBB: Compiled
- Hello world test:
- ./osmocon -p /dev/ttyUSB0 -m c155 ../../target/firmware/board/compal_e99/hello_world.compalram.bin

My Attempt – Software

- My phone still looks like this:
  - :(

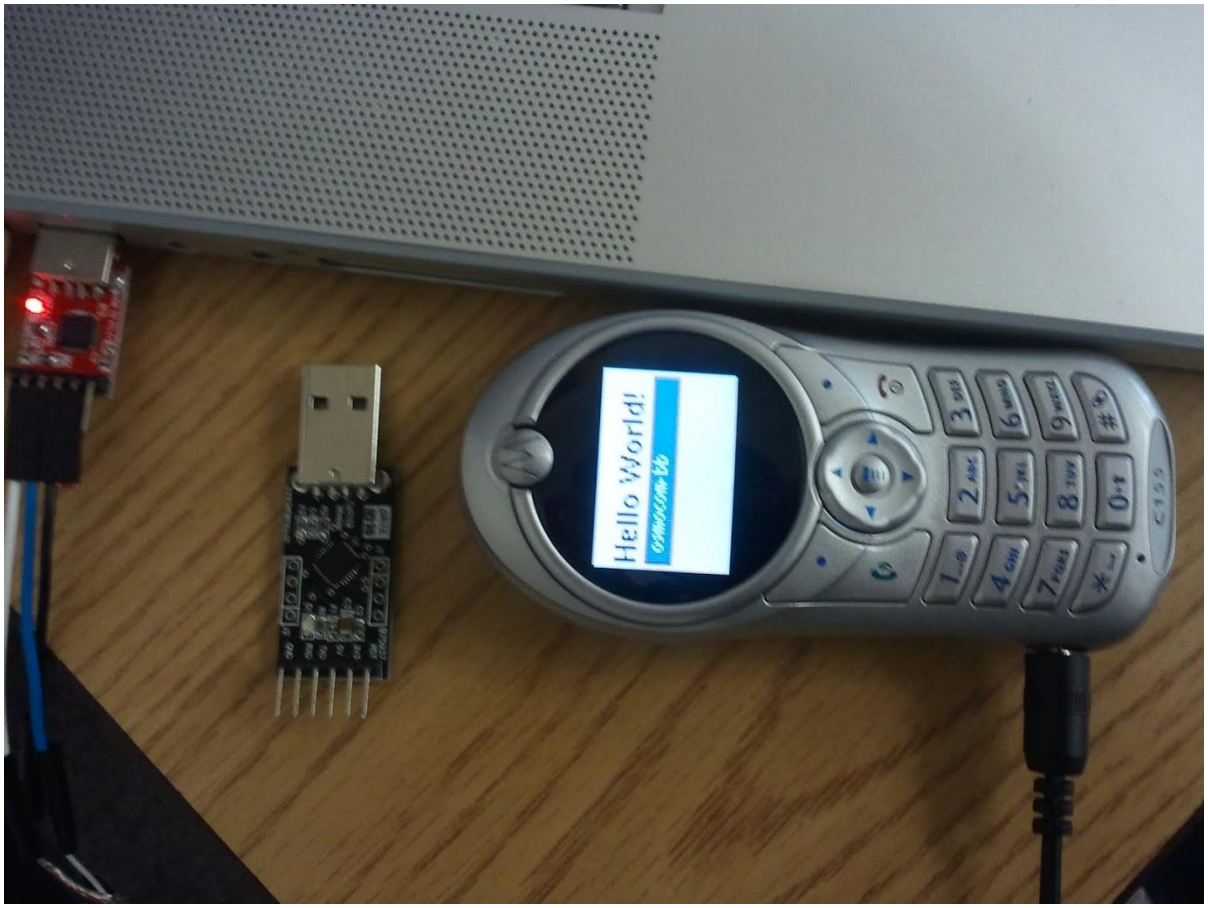
My Attempt – Back to Hardware

- Are these the same?
  - No.
  - After testing, the green board was found dysfunctional.
  - Was either the wrong part or a defective instance.
  - New part: $13.00



My Attempt – Software

- Now my phone looks like this:
  - :)



My Attempt – Software

- Now to run burst_ind:
- Issues:
  - Old wiki page described the use of a program in OmsomcomBB called layer23.
  - That program was since removed and replaced with ccch_scan and bcch_scan
    - Recall that CCCH stands for Common Control Channels, which contains the Paging Channel (PCH).

My Attempt – Software

- ./osmocon -p /dev/ttyUSB0 -m c155 ../../target/firmware/board/compal_e99/layer1.compalram.bin
- While that is running, change directories, run:
- ./ccch_scan -i 127.0.0.1
  - The interface is used to send packets to for Wireshark analysis

My Attempt – Software

- ccch_scan:
  - Persistent error:
  - <000c> l1ctl.c:114 FBSB RESP: result=255
    - ???
  - Code search proved fruitless….

My Attempt – Software

```
107  dl = (struct l1ctl_info_dl *) msg->l1h;
108  sb = (struct l1ctl_fbsb_conf *) dl->payload;
109
110  LOGP(DL1C, LOGL_INFO, "snr=%04x, arfcn=%u
result=%u\n", dl->snr,
111    ntohs(dl->band_arfcn), sb->result);
112
113  if (sb->result != 0) {
114    LOGP(DL1C, LOGL_ERROR, "FBSB RESP: result=%u\n", sb-
>result);
115    fr.ms = ms;
116    fr.band_arfcn = ntohs(dl->band_arfcn);
117    osmo_signal_dispatch(SS_L1CTL, S_L1CTL_FBSB_ERR,
&fr);
118    return 0;
119  }
```

My Attempt – Software

- Web searching also proved fruitless, so I asked on the mailing list
  - "What does FBSB RESP: result=255 mean?"
- The author's response:
  - "It just means failure to sync ...

    Most likely the ARFCN you gave doesn't carry a valid C0."
    - A C0 is a beacon channel
    - http://en.wikipedia.org/wiki/Um_interface

My Attempt – Software

- He went on to say…
- “Note that it's only tested on 900/1800. US band support is not tested and probably not functional especially in burst_ind. Fixing it is left as an exercise to the reader ...”
- US bands are 850 and 1900
  - http://en.wikipedia.org/wiki/GSM_frequency_bands
  - 850: ARFCNs 128 – 251
  - 1900: ARFCNs 512 – 810

My Attempt – Software

- Tested ccch_scan with ARFCNs 128 – 251
  - ./ccch_scan -a 128 -i 127.0.0.1
  - Used bash script to loop through ARFCNs
  - Found interesting channels at 176, 178, 180, 238
- 176, 178, 238:
  - `<0001> app_ccch_scan.c:105 SI1 received.`

My Attempt – Software

- 180:
- <0001> app_ccch_scan.c:360 Paging1: Normal paging chan any to tmsi M(3022466821)
- <0001> app_ccch_scan.c:400 Paging1: Normal paging chan any to TMSI M(0xbe1413b4)
- <0001> app_ccch_scan.c:403 Paging2: Normal paging chan any to TMSI M(0xc5ac16b4)
- <0001> app_ccch_scan.c:426 Paging3: Normal paging chan n/a to tmsi M(3808509207)
- <0001> app_ccch_scan.c:360 Paging1: Normal paging chan any to tmsi M(3019388107) [...]

My Attempt – Software

- It appears that burst_ind does work on the 850 band.
  - Unfortunately, I ran out of time at this point.
- TMSI-finding code should be fairly straightforward.
- Rest of software may not be so trivial.

Conclusions

- This project appears to be viable, but was not completed due to time constraints.
  - Budget goal was successful: Spent $61.65
  - 2 TB hard drive would have broken my budget
- Possible future work:
  - Re-develop attack software components
  - Expand attack to other types of data-transfers?
  - ?

References/Thanks

- https://www.youtube.com/watch?v=ZrbatnnRxFc
- http://www.cecm.sfu.ca/~lisonek/cryptography/Karsten.Nohl.GSM.pdf
- http://gsmfordummies.com/
- http://bb.osmocom.org/trac/wiki
- Thanks to:
  - Dr. Rinker for lending tools and donating time.
  - Sylvain Munaut for the helpful and prompt reply.