1.0) Introduction

Wireless cellular communication technology has changed the way we communicate with each other. From the introduction of the 1G technology to the masses to the widespread adoption of 4G LTE Technology, each generation of wireless technology has improved upon the last and increased the functionality and use cases, leading to the deprecation of the previous generation over a period of time.
However the 2nd Generation of wireless cellular communication, or Global Standard for Mobile Communication (GSM) still has a large user base in many countries including India, where the majority of the cell phones are feature phones operating on a GSM network. Hence it is important to understand the technology behind this standard of wireless cellular communication, and its security (or lack thereof)

1.1) Purpose

The purpose of this project is to understand the GSM Standard (2nd Generation) of Wireless Cellular Communication. This is done by first using and learning about Software-Defined Radio (SDR), its hardware and software tools and thereafter using SDR to analyze the GSM technology and security.

1.2) Scope

The scope of the project developed now is as follows:

1. Passive sniffing of GSM Packets
2. Capturing International Mobile Subscriber Identity Numbers (IMSI)
3. Long-Term Intelligence from passive sniffing and IMSIs

However, with more resources, tools and time, the project can further be expanded to:

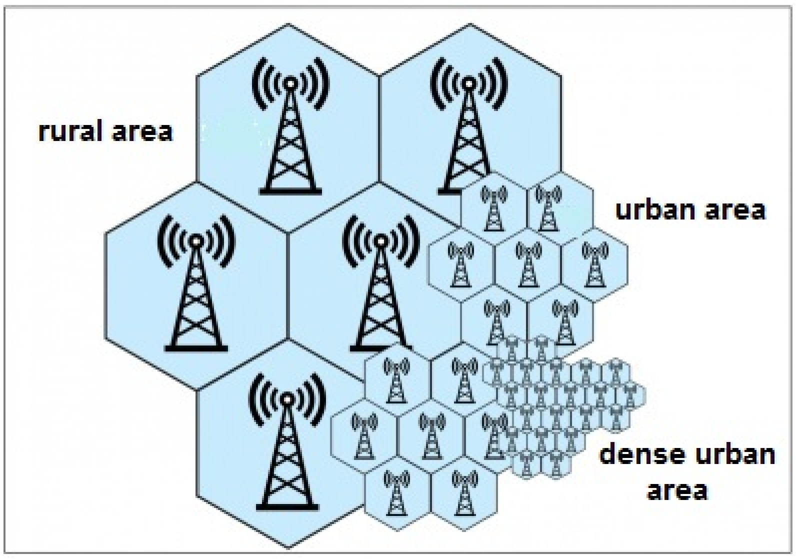
1. Active sniffing of GSM Packets / Active IMSI Catcher
2. Cracking A5/1 Encryption used in GSM
3. Study of further attacks and flaws in GSM
4. IMSI Catcher for 4G LTE Networks

1.3) Overview

1. Understanding the telecommunication landscape
2. Comparison of technology of different generations of cellular communication; 1G vs. 2G vs. 3G vs. 4G vs. 5G
3. Understanding GSM (Global System for Mobile Communication) Technology
4. Introduction to Software-Defined Radio
5. Sniffing GSM IMSIs with Software-Defined Radio

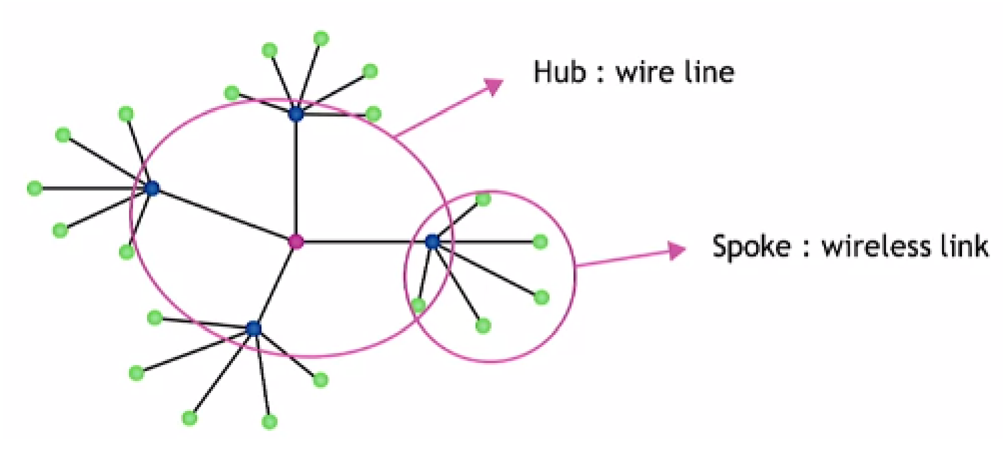
2.0) Cellular Communication Landscape

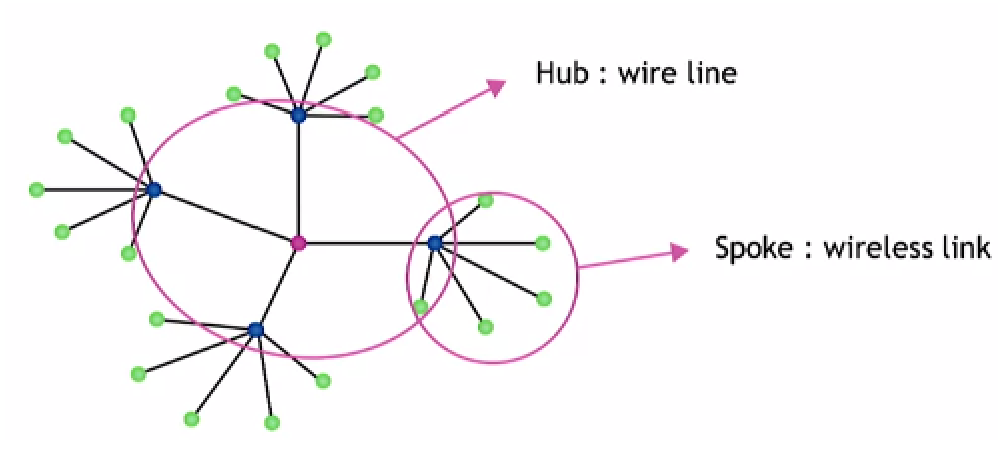
The most important development in telecommunication industry is the cellular structure. This concept and technology was developed by AT&T Bell Labs in 1970s and incorporated into the first generation wireless communication system. All the wireless communication systems still follow this model.
The cellular structure means that wireless coverage is divided into areas which a particular BTS covers and provides service to.



Notice the cellular coverage area depends on the density or the population covered

To minimize the interference and maximize the connections, a Spoke and Hub Network Model is followed.




For local connection, ie within the same cell or BTS, it is done with the via wireless links with small transmission power
For connection between far off cells, access points and wired links are used

With the understanding of the structure of the cellular communication landscape we can dive into the technology behind the different generations of wireless telecom.

3.0) Comparison between different generations of cellular communication

Services/Generation	**1G**	**2G**	**3G**	**4G**
Voice	Analog	Digital	Digital	Voice Over LTE
SMS	N/A	Available	Available	Available
Data Rate	N/A	`~300Kbps`	`~10Mbps`	`100Mbps and more`
Transmission	Analog	Digital	Digital	Digital
Multiple Access	Frequency Division	Time Division or Code Division	Code Division	Orthogonal frequency-division
Standards	NMT/TACS/AMPS	GSM; GPRS-EDGE; US-TDMA	WCDMA; CDMA-2000	LTE

4.0) Understanding GSM Technology

4.1) Overview

GSM standard was first developed as a digital, circuit switched, telecom standard deployed majorly in Europe and Asia. GSM operates on the mobile communication bands of 900 MHz and 1800 MHz in most parts of the world.
GSM standard served 80% of the mobile market, making it the most ubiquitous of the many standards for cellular networks. Currently, in many parts of the world GSM is being phased out in replacement of 4G LTE and 5G systems. However, in India, GSM still has a very large user base, mostly of feature phone users leading telecom companies to discontinue 3G, and have 2G GSM and 4G LTE running in parallel.

4.2) Architecture

GSM Consists of many functional units which can be broadly divided into:

1. The Mobile Station (MS)
2. The Base Station Subsystem (BSS)
3. The Network Switching Subsystem (NSS)
4. The Operation Support Subsystem (OSS)

For the scope of this report, the MS and BSS are relevant as they handle the user end of the connection.

4.2.1) The Mobile Station

The mobile station consists of the hand-held device and a GSM Subscriber Identity Module (SIM) Card. The SIM is an integrated circuit which stores the International Mobile Subscriber Identity (IMSI) number and its related secret key which is used to authenticate and identify users.
IMSI is 14 or 15 digit long and has the following format:

MCC	MNC	MSIN
404	10	06199XXXXX
India Country Code	Airtel - Delhi NCR Network Code	Unique MSIN

4.2.2) The Base Station Subsystem (BSS)

BSS comprises of Base Transceiver Station and Base Station Controller.

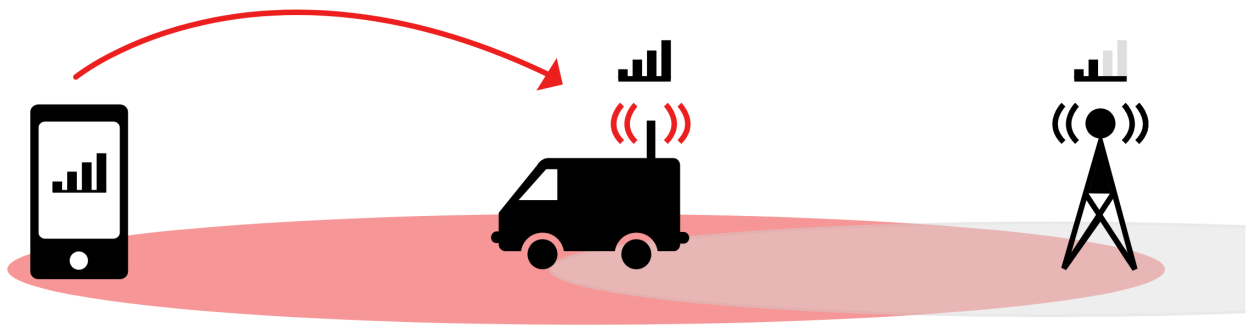
- BTS houses the radio transceiver that define a cell and handle the encoding, encryption, decoding, decryption, modulation etc functionality

- BSS manages one or more BTSs and handles the radio setup, frequency hopping and handover

4.3) Connection in GSM

Phones have the inherent functionality of connecting with a BTS which has higher signal strength and quality. Once the phone identifies the stronger base station, it starts negotiating a connection with it.

BASIC CSS: PHONE CONNECTS TO THE STRONGEST SIGNAL STRENGTH

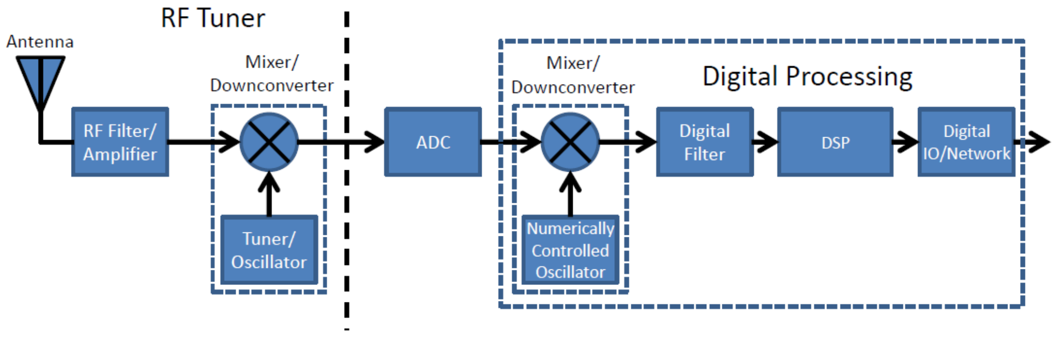


Hereafter, the cell tower decides which encryption algorithm to use or whether to even use encryption or not. This information along with an Identity Request is sent out, to which the cell phone responds with its IMSI Number.

In subsequent generations of telecommunication, the procedure has evolved and added multiple security measure leaving GSM to be the most insecure standard in use as of now.

5) Introduction to Software-Defined Radio

Software-Defined Radio (SDR) is the software implementations of hardware components. Hardware, which in traditional radio would have processed the signal now works as software components. This allows to control and make software programs having a much more control on the radio signal.



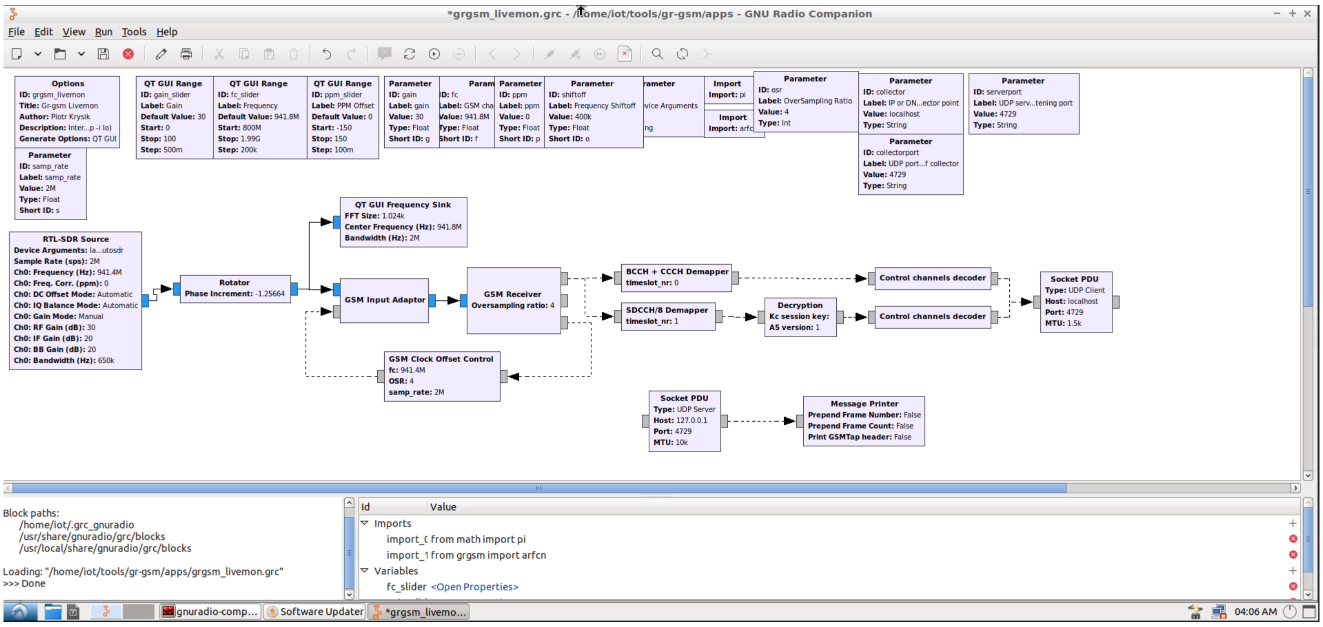
5.1) SDR Tools and Hardware

HackRF

HackRF is an open source SDR board with Receiving and Transmission Capabilities

GNU Radio

GNURadio is a SDK that provides signal processing blocks to implement software radios.


*grgsm_livemon.grc - /home/iot/tools/gr-gsm/apps - GNU Radio Companion

Options ID: grgsm_livemon; Title: Gr-gsm Livemon; Author: Piotr Krysik; Description: Inter...p -i lo); Generate Options: QT GUI

Parameter ID: samp_rate; Label: samp_rate; Value: 2M; Type: Float; Short ID: s

QT GUI Range ID: gain_slider; Label: Gain; Default Value: 30; Start: 0; Stop: 100; Step: 500m

QT GUI Range ID: fc_slider; Label: Frequency; Default Value: 941.8M; Start: 800M; Stop: 1.99G; Step: 200k

QT GUI Range ID: ppm_slider; Label: PPM Offset; Default Value: 0; Start: -150; Stop: 150; Step: 100m

Parameter ID: gain; Label: gain; Value: 30; Type: Float; Short ID: g

Param ID: fc; Label: GSM cha; Value: 941.8M; Type: Float; Short ID: f

Parameter ID: ppm; Label: ppm; Value: 0; Type: Float; Short ID: p

Parameter ID: shiftoff; Label: Frequency Shiftoff; Value: 400k; Type: Float; Short ID: o

rameter vice Arguments; ng

Import Import: pi

Import Import: arfc

Parameter ID: osr; Label: OverSampling Ratio; Value: 4; Type: Int

Parameter ID: collector; Label: IP or DN...ector point; Value: localhost; Type: String

Parameter ID: collectorport; Label: UDP port...f collector; Value: 4729; Type: String

Parameter ID: serverport; Label: UDP serv...tening port; Value: 4729; Type: String

QT GUI Frequency Sink FFT Size: 1.024k; Center Frequency (Hz): 941.8M; Bandwidth (Hz): 2M

RTL-SDR Source Device Arguments: la...utosdr; Sample Rate (sps): 2M; Ch0: Frequency (Hz): 941.4M; Ch0: Freq. Corr. (ppm): 0; Ch0: DC Offset Mode: Automatic; Ch0: IQ Balance Mode: Automatic; Ch0: Gain Mode: Manual; Ch0: RF Gain (dB): 30; Ch0: IF Gain (dB): 20; Ch0: BB Gain (dB): 20; Ch0: Bandwidth (Hz): 650k

Rotator Phase Increment: -1.25664

GSM Input Adaptor

GSM Receiver Oversampling ratio: 4

BCCH + CCCH Demapper timeslot_nr: 0

SDCCH/8 Demapper timeslot_nr: 1

Decryption Kc session key: ; A5 version: 1

Control channels decoder

Control channels decoder

Socket PDU Type: UDP Client; Host: localhost; Port: 4729; MTU: 1.5k

GSM Clock Offset Control fc: 941.4M; OSR: 4; samp_rate: 2M

Socket PDU Type: UDP Server; Host: 127.0.0.1; Port: 4729; MTU: 10k

Message Printer Prepend Frame Number: False; Prepend Frame Count: False; Print GSMTap header: False

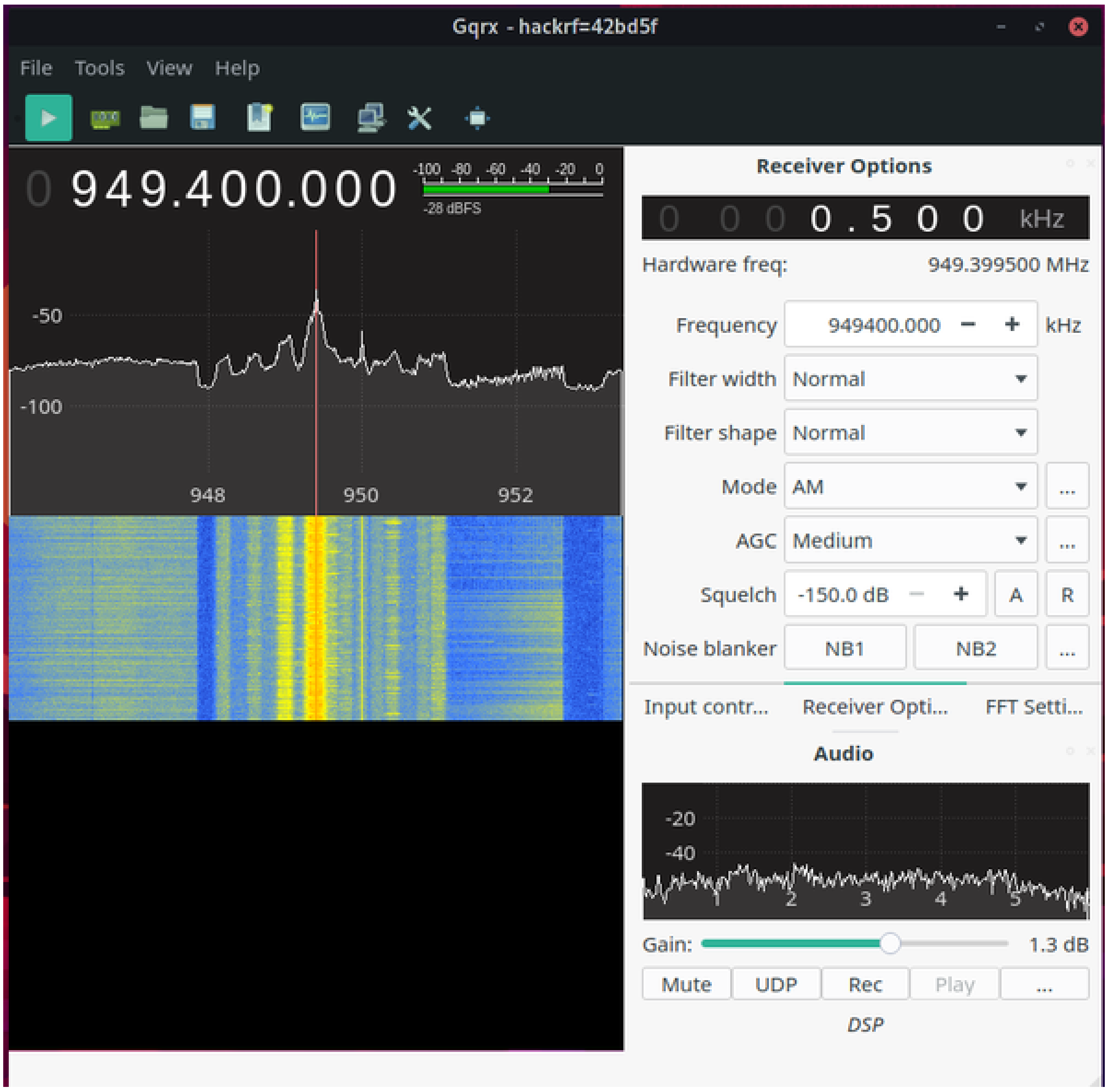
```
Block paths:
    /home/iot/.grc_gnuradio
    /usr/share/gnuradio/grc/blocks
    /usr/local/share/gnuradio/grc/blocks

Loading: "/home/iot/tools/gr-gsm/apps/grgsm_livemon.grc"
>>> Done
```

Id	Value
Imports	
import_0	from math import pi
import_1	from grgsm import arfcn
Variables	
fc_slider	<Open Properties>

gqrx

gqrx is used for basic radio tuning and to get a rough overview of the transmissions around.


Gqrx - hackrf=42bd5f

0 949.400.000   -28 dBFS

Receiver Options

0 0 0 0.500 kHz

Hardware freq: 949.399500 MHz

Frequency: 949400.000 kHz

Filter width: Normal

Filter shape: Normal

Mode: AM

AGC: Medium

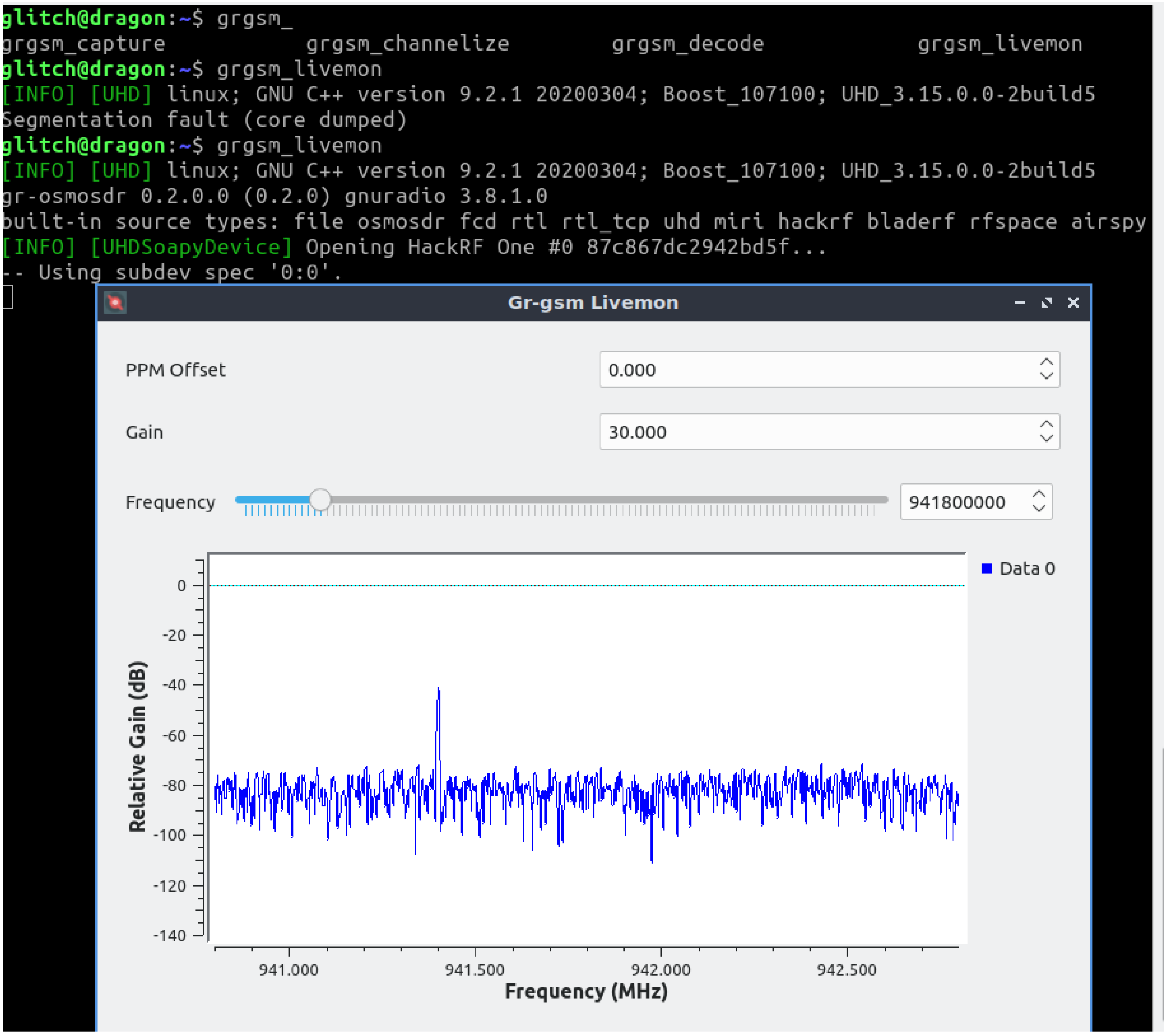
Squelch: -150.0 dB

Noise blanker: NB1 NB2

Gain: 1.3 dB

`gr-gsm`

`gr-gsm` is a set of tools used to receive GSM transmissions and works with any SDR capable of tuning into the GSM frequency.



```
glitch@dragon:~$ grgsm_
grgsm_capture          grgsm_channelize        grgsm_decode            grgsm_livemon
glitch@dragon:~$ grgsm_livemon
[INFO] [UHD] linux; GNU C++ version 9.2.1 20200304; Boost_107100; UHD_3.15.0.0-2build5
Segmentation fault (core dumped)
glitch@dragon:~$ grgsm_livemon
[INFO] [UHD] linux; GNU C++ version 9.2.1 20200304; Boost_107100; UHD_3.15.0.0-2build5
gr-osmosdr 0.2.0.0 (0.2.0) gnuradio 3.8.1.0
built-in source types: file osmosdr fcd rtl rtl_tcp uhd miri hackrf bladerf rfspace airspy
[INFO] [UHDSoapyDevice] Opening HackRF One #0 87c867dc2942bd5f...
-- Using subdev spec '0:0'.
```

Gr-gsm Livemon

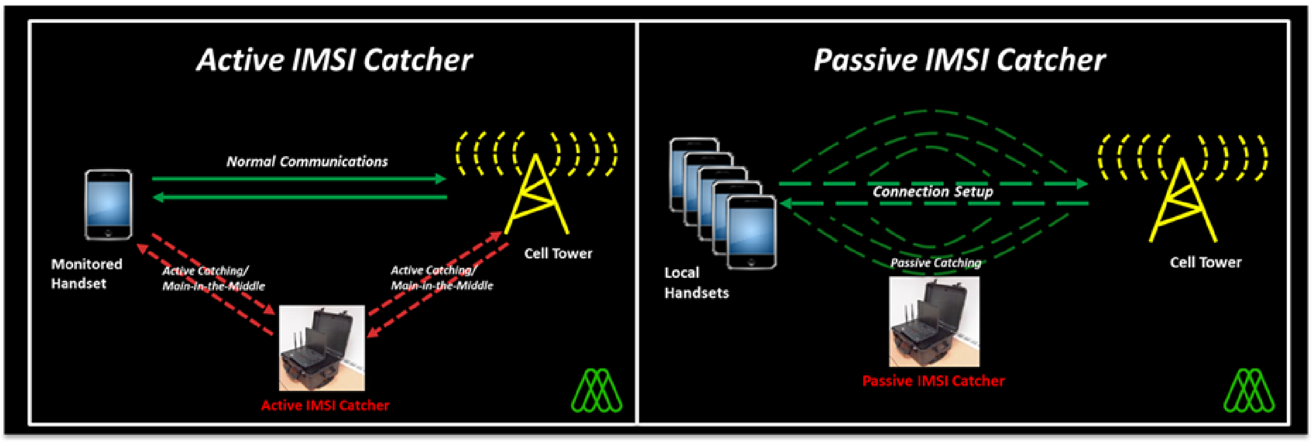
PPM Offset 0.000

Gain 30.000

Frequency 941800000

6) A note on IMSI Catchers

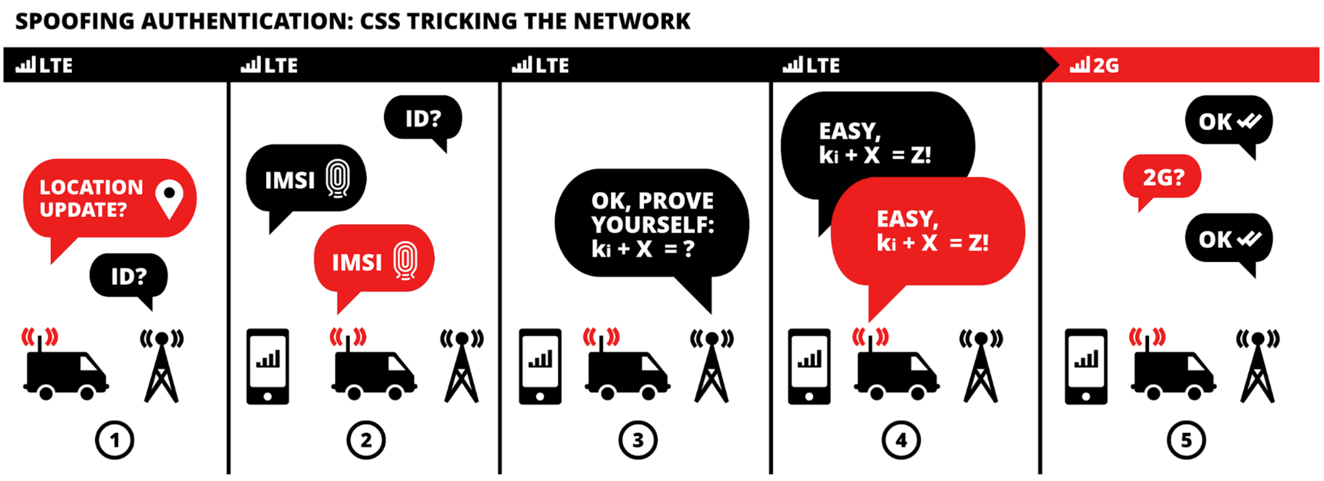
GSM IMSI are unique to each individual GSM user. It is meant to be private as it links the user and his/her physical location, SMS, voice call and data.



6.1) Active IMSI catchers

Active IMSI Catchers function as Fake BTSs and establish a connection with cellphones. They advertise a higher strength connection and offer connection to cellphones around that region. Then, during the connection negotiation, it disables encryption (something which is a possibility in only GSM) and then sends an Identity Request to capture the IMSI. From hereon, the Fake BTS can read all the data since it is unencrypted

In successive generations, ie 3G and 4G LTE, the attack gets more complex. The attacker with his fake BTS would try to spoof the connection request and act Man-in-the-Middle. Then the service would get downgraded to the non-encrypted GSM instead of more secure 3G/4G



6.2) Passive IMSI Catchers

With the help of SDR, one can develop a passive IMSI Catcher. Here there is no interaction with the cellphones

Whenever a cellphone connects with a new cell tower, the connection procedure takes place. This can easily be captured as this is unencrypted traffic and the response to the Identity Request is sent without encryption.

---

7) Passive IMSI Catching with SDR

> For the sake of this report, the installation of the tools shall not be covered

To first listen in on the GSM network, we find out the down link frequency of any nearby base station. In India, the GSM network follows the PGSM-900 and GSM-1800 standard. Meaning, the GSM Connections are of either 900MHz or 1800MHz.

To tune in to the frequency we use our SDR hardware and `gqrx` to find the exact down link.

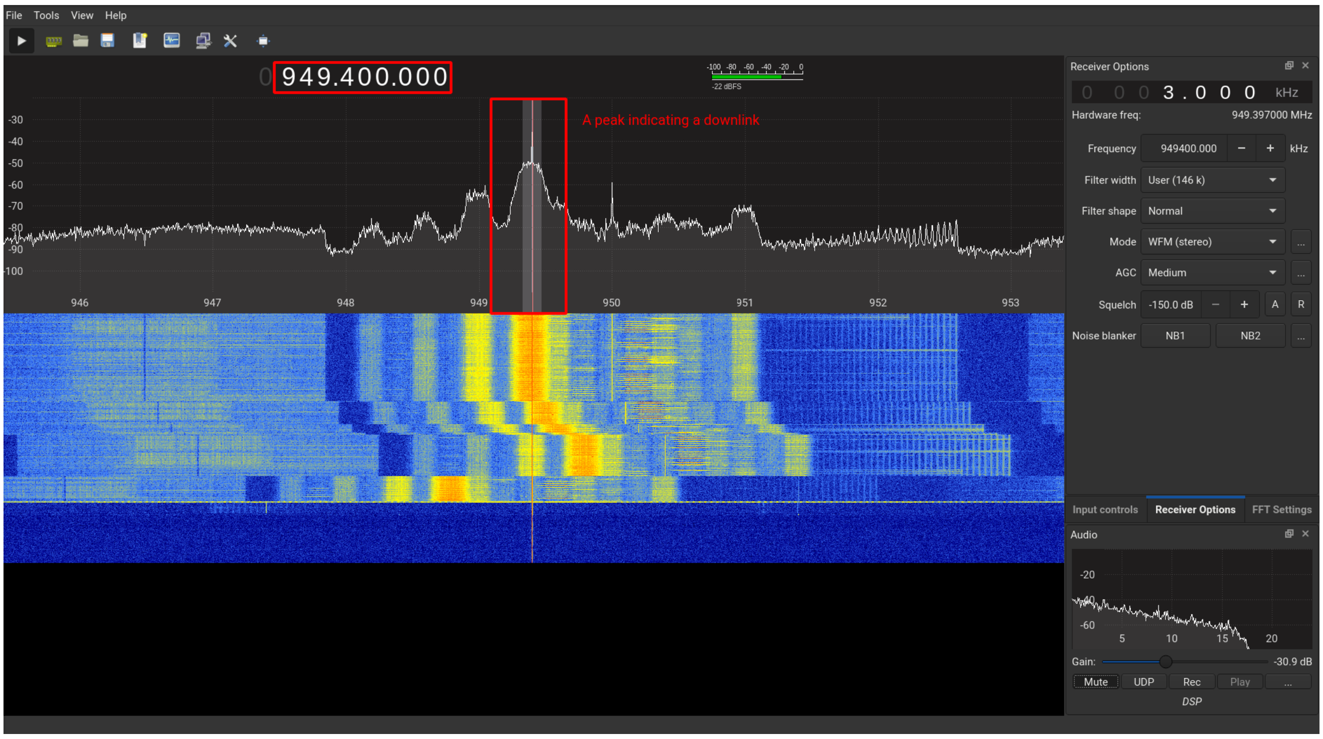
We first connect the HackRF and add the kernel module with

```
sudo modprobe hackrf
```

Then we fire up our tool;

```
gqrx
```

Then we slowly browse the range of 900MHz - 960MHz to find any peaks.

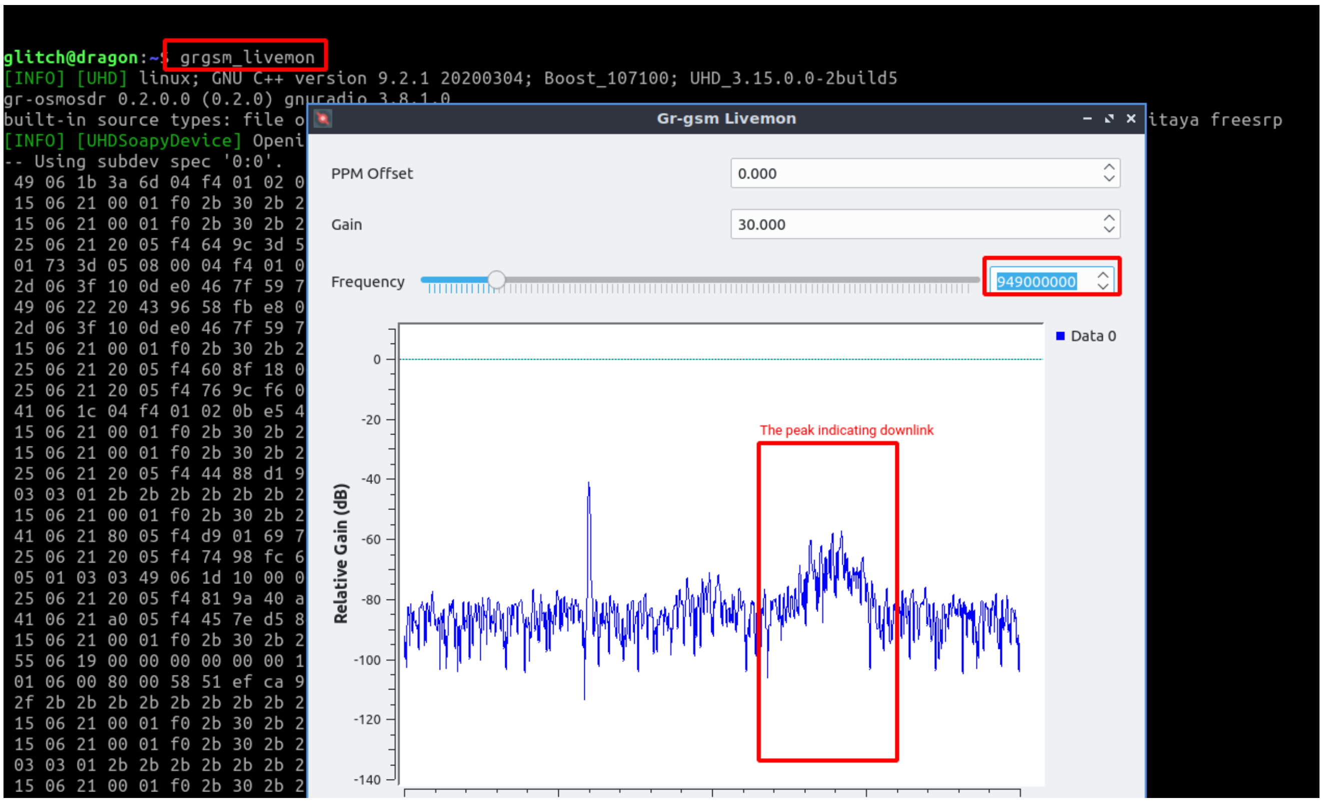


```
Hardware freq: 949.397000 MHz
Frequency      949400.000 kHz
Filter width   User (146 k)
Filter shape   Normal
Mode           WFM (stereo)
AGC            Medium
Squelch        -150.0 dB
Noise blanker  NB1  NB2
Gain: -30.9 dB
```

Once found, we note down the frequency.
Next, we start up our GSM monitor;

```
grgsm_livemon
```

Now we tune into the frequency we found above until we notice GSM output;



```
glitch@dragon:~$ grgsm_livemon
[INFO] [UHD] linux; GNU C++ version 9.2.1 20200304; Boost_107100; UHD_3.15.0.0-2build5
gr-osmosdr 0.2.0.0 (0.2.0) gnuradio 3.8.1.0
built-in source types: file o                                        itaya freesrp
[INFO] [UHDSoapyDevice] Openi
-- Using subdev spec '0:0'.
 49 06 1b 3a 6d 04 f4 01 02 0
 15 06 21 00 01 f0 2b 30 2b 2
 15 06 21 00 01 f0 2b 30 2b 2
 25 06 21 20 05 f4 64 9c 3d 5
 01 73 3d 05 08 00 04 f4 01 0
 2d 06 3f 10 0d e0 46 7f 59 7
 49 06 22 20 43 96 58 fb e8 0
 2d 06 3f 10 0d e0 46 7f 59 7
 15 06 21 00 01 f0 2b 30 2b 2
 25 06 21 20 05 f4 60 8f 18 0
 25 06 21 20 05 f4 76 9c f6 0
 41 06 1c 04 f4 01 02 0b e5 4
 15 06 21 00 01 f0 2b 30 2b 2
 15 06 21 00 01 f0 2b 30 2b 2
 25 06 21 20 05 f4 44 88 d1 9
 03 03 01 2b 2b 2b 2b 2b 2b 2
 15 06 21 00 01 f0 2b 30 2b 2
 41 06 21 80 05 f4 d9 01 69 7
 25 06 21 20 05 f4 74 98 fc 6
 05 01 03 03 49 06 1d 10 00 0
 25 06 21 20 05 f4 81 9a 40 a
 41 06 21 a0 05 f4 45 7e d5 8
 15 06 21 00 01 f0 2b 30 2b 2
 55 06 19 00 00 00 00 00 00 1
 01 06 00 80 00 58 51 ef ca 9
 2f 2b 2b 2b 2b 2b 2b 2b 2b 2
 15 06 21 00 01 f0 2b 30 2b 2
 15 06 21 00 01 f0 2b 30 2b 2
 03 03 01 2b 2b 2b 2b 2b 2b 2
 15 06 21 00 01 f0 2b 30 2b 2
```

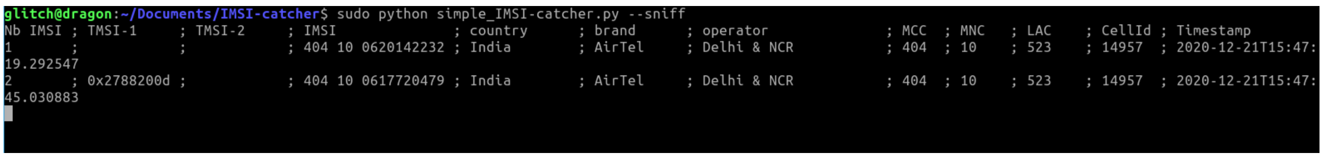
Gr-gsm Livemon

```
PPM Offset   0.000
Gain         30.000
Frequency    949000000
```

In another terminal, we start our IMSI Catcher with;

```
sudo python simple_IMSI-catcher.py --sniff -t log.txt
```

Now, we wait for some new connections to be established on the cell tower on whose downlink we are listening in on;



```
glitch@dragon:~/Documents/IMSI-catcher$ sudo python simple_IMSI-catcher.py --sniff
Nb IMSI ; TMSI-1     ; TMSI-2     ; IMSI              ; country      ; brand      ; operator                 ; MCC  ; MNC   ; LAC    ; CellId ; Timestamp
1       ;            ;            ; 404 10 0620142232 ; India        ; AirTel     ; Delhi & NCR              ; 404  ; 10    ; 523    ; 14957  ; 2020-12-21T15:47:
19.292547
2       ; 0x2788200d ;            ; 404 10 0617720479 ; India        ; AirTel     ; Delhi & NCR              ; 404  ; 10    ; 523    ; 14957  ; 2020-12-21T15:47:
45.030883
```

After some time, we can notice that we have captured the IMSI and TMSI of some new connections.

Ideally, you shouldn't be able to do anything with capturing *just* IMSI, but problems arise when you start to associate an IMSI with an individual. Usually law enforcement has access to this kind of database linking users and IMSIs.
One often overlooked method is through mobile applications. Apps can access a device's IMSI number (on Android), which is sometimes collected and used to serve as a unique user identifier. IMSIs can be stored alongside names, phone numbers and email addresses in customer databases. Customer databases are commonly sold to 3rd parties for marketing purposes, and of course have potential to be stolen. Now all of the sudden you have databases with IMSI numbers and names on the internet, which is bad.

---

8) Data Aggregation and Long-Term Intelligence

Background

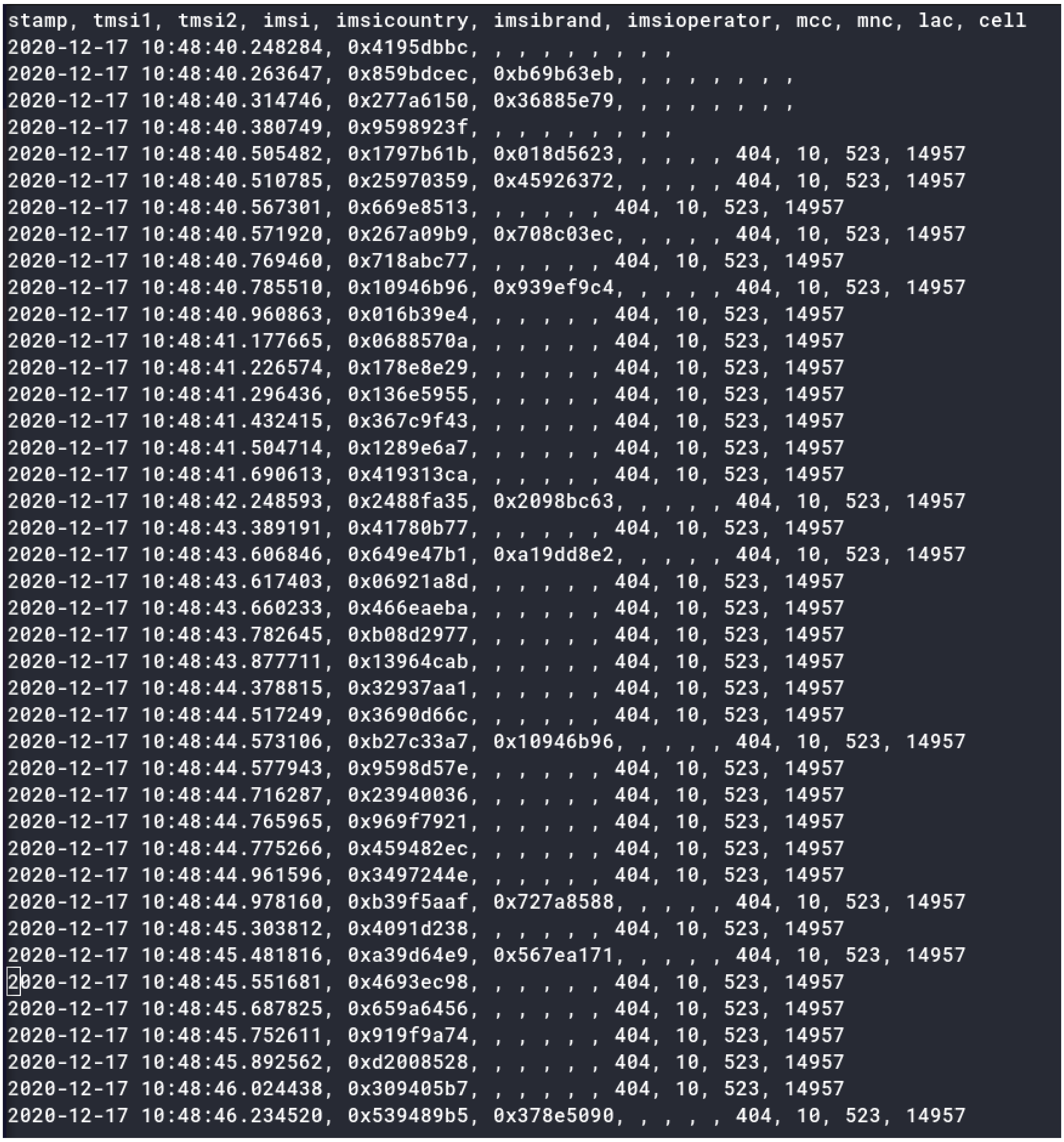
The HackRF One was set up in a residential area with moderate movement. One should expect to see low activity during the late hours and higher activity during the day. Moreover, if the device had been set up in a very active area, near a main road or highway, one might see large sets of IMSIs.
The downlink frequency I could tune into was `949MHz`

While Passive IMSI Catching won't do any harm, one can extract quite useful insights by collecting logs and using a data aggregator tool.

The tool in use also supports log creation with different levels of verbosity,

```
sudo python simple_IMSI-catcher.py -a --sniff -t imsi00.txt
```

> `-a` denotes that GSM Packets with IMSI would also be saved
> `-t imsi00.txt` denotes that all the output would be saved in a file in CSV format



```
stamp, tmsi1, tmsi2, imsi, imsicountry, imsibrand, imsioperator, mcc, mnc, lac, cell
2020-12-17 10:48:40.248284, 0x4195dbbc, , , , , , , , ,
2020-12-17 10:48:40.263647, 0x859bdcec, 0xb69b63eb, , , , , , , ,
2020-12-17 10:48:40.314746, 0x277a6150, 0x36885e79, , , , , , , ,
2020-12-17 10:48:40.380749, 0x9598923f, , , , , , , , ,
2020-12-17 10:48:40.505482, 0x1797b61b, 0x018d5623, , , , , 404, 10, 523, 14957
2020-12-17 10:48:40.510785, 0x25970359, 0x45926372, , , , , 404, 10, 523, 14957
2020-12-17 10:48:40.567301, 0x669e8513, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:40.571920, 0x267a09b9, 0x708c03ec, , , , , 404, 10, 523, 14957
2020-12-17 10:48:40.769460, 0x718abc77, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:40.785510, 0x10946b96, 0x939ef9c4, , , , , 404, 10, 523, 14957
2020-12-17 10:48:40.960863, 0x016b39e4, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:41.177665, 0x0688570a, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:41.226574, 0x178e8e29, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:41.296436, 0x136e5955, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:41.432415, 0x367c9f43, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:41.504714, 0x1289e6a7, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:41.690613, 0x419313ca, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:42.248593, 0x2488fa35, 0x2098bc63, , , , , 404, 10, 523, 14957
2020-12-17 10:48:43.389191, 0x41780b77, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:43.606846, 0x649e47b1, 0xa19dd8e2, , , , , 404, 10, 523, 14957
2020-12-17 10:48:43.617403, 0x06921a8d, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:43.660233, 0x466eaeba, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:43.782645, 0xb08d2977, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:43.877711, 0x13964cab, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:44.378815, 0x32937aa1, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:44.517249, 0x3690d66c, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:44.573106, 0xb27c33a7, 0x10946b96, , , , , 404, 10, 523, 14957
2020-12-17 10:48:44.577943, 0x9598d57e, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:44.716287, 0x23940036, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:44.765965, 0x969f7921, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:44.775266, 0x459482ec, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:44.961596, 0x3497244e, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:44.978160, 0xb39f5aaf, 0x727a8588, , , , , 404, 10, 523, 14957
2020-12-17 10:48:45.303812, 0x4091d238, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:45.481816, 0xa39d64e9, 0x567ea171, , , , , 404, 10, 523, 14957
2020-12-17 10:48:45.551681, 0x4693ec98, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:45.687825, 0x659a6456, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:45.752611, 0x919f9a74, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:45.892562, 0xd2008528, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:46.024438, 0x309405b7, , , , , , 404, 10, 523, 14957
2020-12-17 10:48:46.234520, 0x539489b5, 0x378e5090, , , , , 404, 10, 523, 14957
```

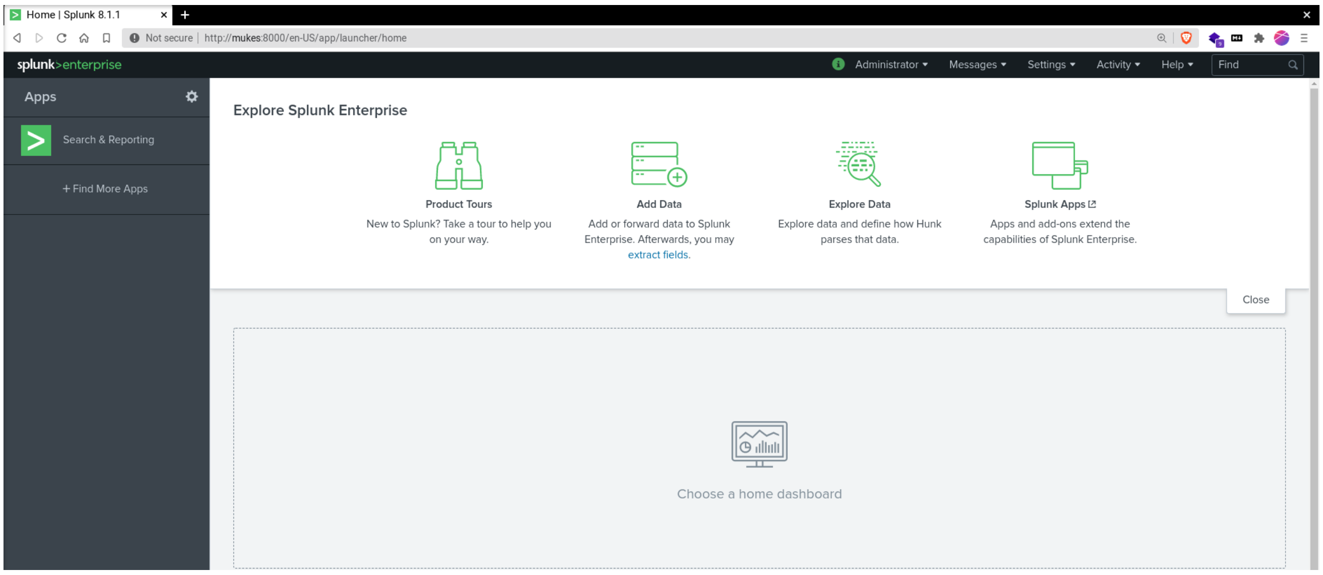
Using shall use Splunk as a data aggregator.
We can install Splunk on Arch-based systems with

```
sudo pacman -S splunk
```

And start it with;

```
sudo /opt/splunk/bin/splunk start
```

We get a web interface like;


Home | Splunk 8.1.1

Not secure | http://mukes:8000/en-US/app/launcher/home

splunk>enterprise

Administrator ▾ Messages ▾ Settings ▾ Activity ▾ Help ▾ Find

Apps

Search & Reporting

+ Find More Apps

Explore Splunk Enterprise

Product Tours
New to Splunk? Take a tour to help you on your way.

Add Data
Add or forward data to Splunk Enterprise. Afterwards, you may extract fields.

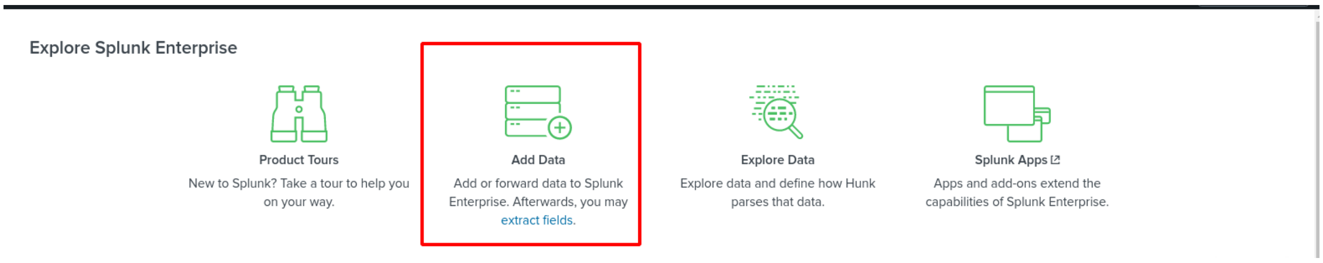
Explore Data
Explore data and define how Hunk parses that data.

Splunk Apps
Apps and add-ons extend the capabilities of Splunk Enterprise.

Close

Choose a home dashboard

We upload our log file to Splunk;


Explore Splunk Enterprise

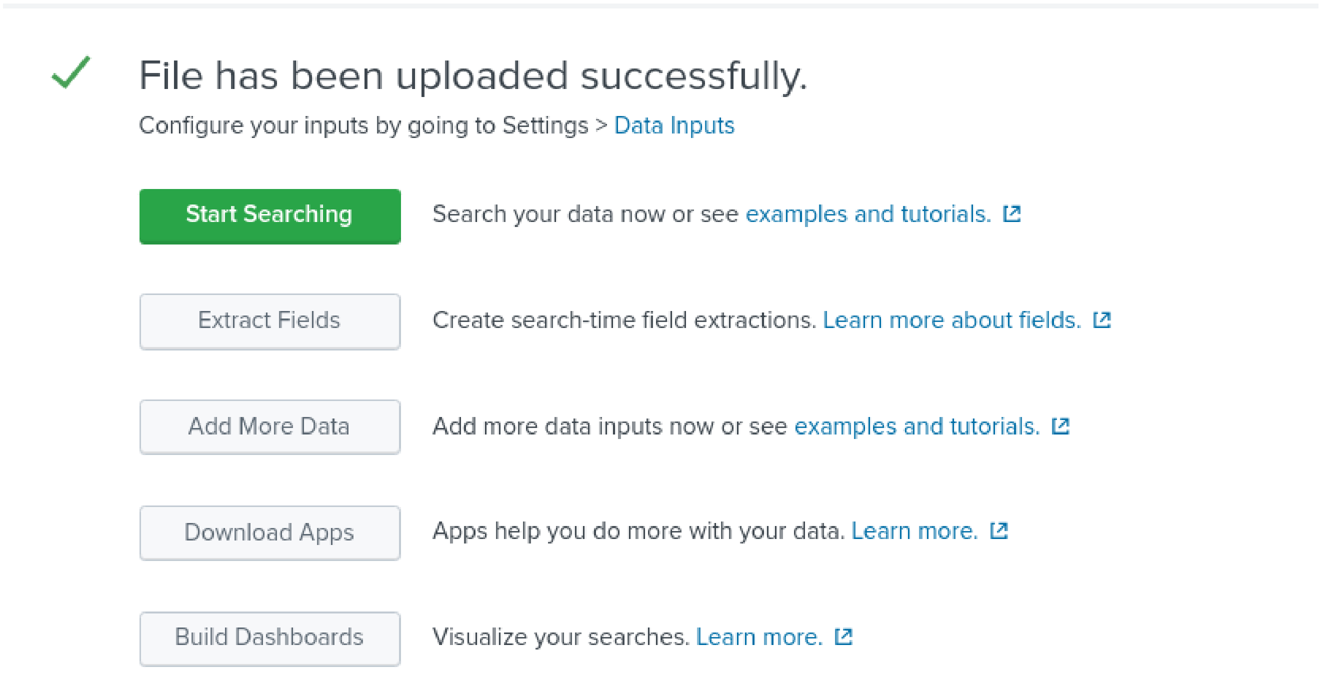
Product Tours
New to Splunk? Take a tour to help you on your way.

Add Data
Add or forward data to Splunk Enterprise. Afterwards, you may extract fields.

Explore Data
Explore data and define how Hunk parses that data.

Splunk Apps
Apps and add-ons extend the capabilities of Splunk Enterprise.

Now, we can start searching through our logs, extract fields or add more log files. All in all, we can visualise and play with our data now.


File has been uploaded successfully.

Configure your inputs by going to Settings > Data Inputs

Start Searching	Search your data now or see examples and tutorials.
Extract Fields	Create search-time field extractions. Learn more about fields.
Add More Data	Add more data inputs now or see examples and tutorials.
Download Apps	Apps help you do more with your data. Learn more.
Build Dashboards	Visualize your searches. Learn more.

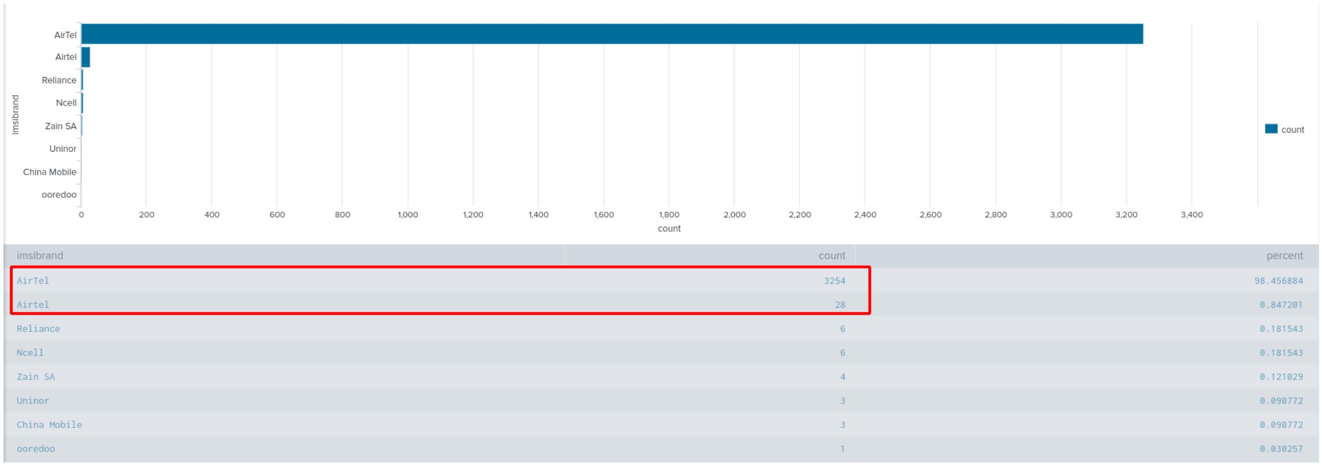
Interesting Findings

Using the Splunk Processing Language (SPL) one can query the dataset based on different parameters. Here are some interesting insights I could find.

For clarity, I ran the tool and collected logs four times, each time during different parts of the day and for different periods of time.

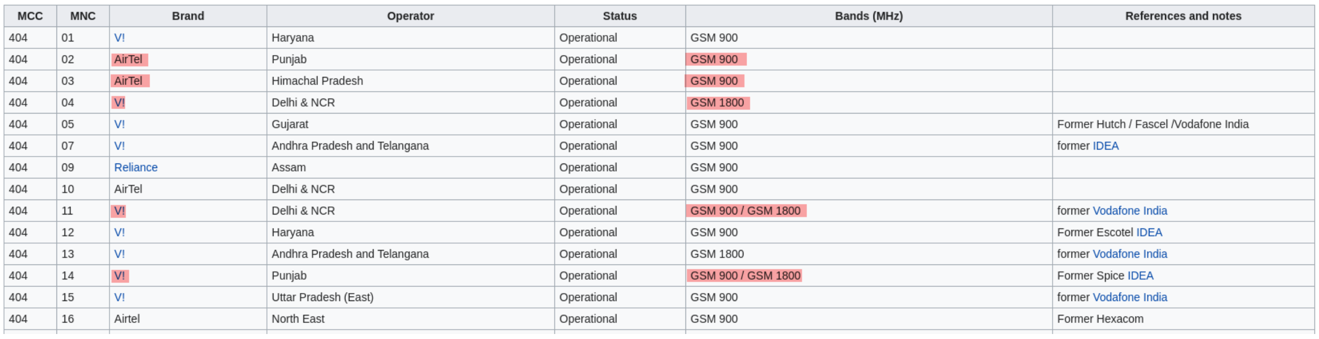
IMSI Brands - Finding #1

After going through the logs, I found that these were the top GSM brands whose IMSIs I could capture.



imsibrand	count	percent
AirTel	3254	98.456884
Airtel	28	0.847201
Reliance	6	0.181543
Ncell	6	0.181543
Zain SA	4	0.121029
Uninor	3	0.090772
China Mobile	3	0.090772
ooredoo	1	0.030257

It is a bit strange that almost all of them were of Airtel. To find out why I did a little research.
Going through the Mobile Network Code and Frequency Band list on Wikipedia, I found out this:

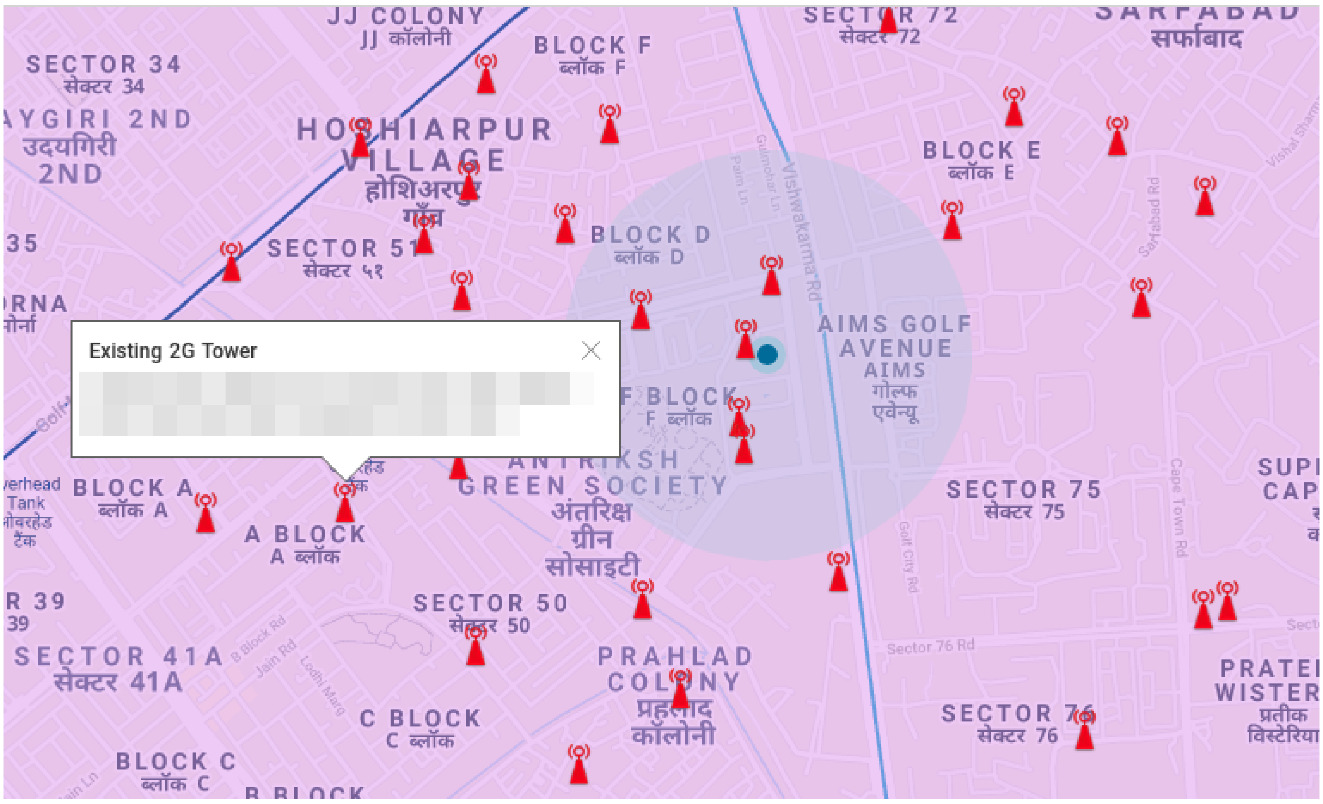


MCC	MNC	Brand	Operator	Status	Bands (MHz)	References and notes
404	01	Vi	Haryana	Operational	GSM 900	
404	02	AirTel	Punjab	Operational	GSM 900	
404	03	AirTel	Himachal Pradesh	Operational	GSM 900	
404	04	Vi	Delhi & NCR	Operational	GSM 1800	
404	05	Vi	Gujarat	Operational	GSM 900	Former Hutch / Fascel /Vodafone India
404	07	Vi	Andhra Pradesh and Telangana	Operational	GSM 900	former IDEA
404	09	Reliance	Assam	Operational	GSM 900	
404	10	AirTel	Delhi & NCR	Operational	GSM 900	
404	11	Vi	Delhi & NCR	Operational	GSM 900 / GSM 1800	former Vodafone India
404	12	Vi	Haryana	Operational	GSM 900	Former Escotel IDEA
404	13	Vi	Andhra Pradesh and Telangana	Operational	GSM 1800	former Vodafone India
404	14	Vi	Punjab	Operational	GSM 900 / GSM 1800	Former Spice IDEA
404	15	Vi	Uttar Pradesh (East)	Operational	GSM 900	former Vodafone India
404	16	Airtel	North East	Operational	GSM 900	Former Hexacom

Airtel primarily operates on GSM-900 whereas Vodafone, the other major GSM operator, operates on GSM-1800. Since I could only tune into GSM-900, almost all of the IMSIs I captured where of Airtel.

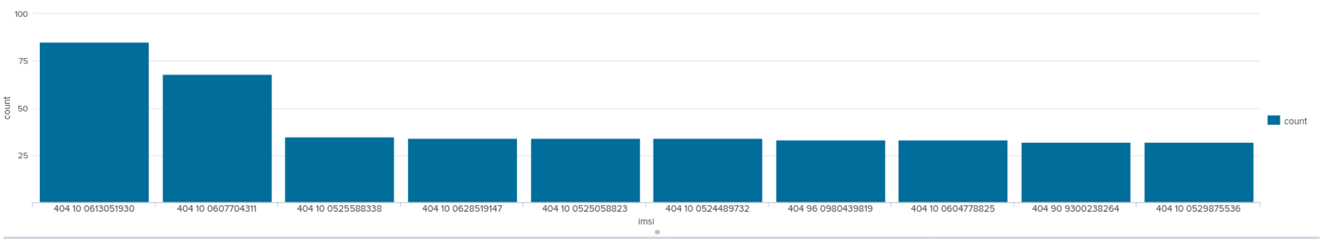
IMSI Brands - Finding #2

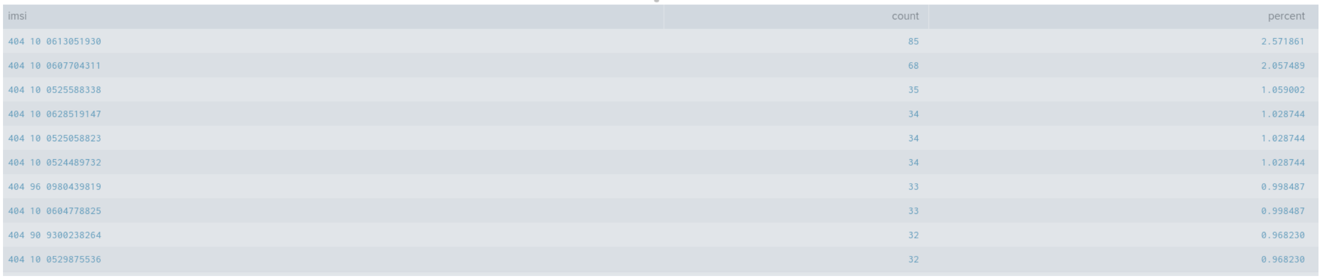
From the above, finding we get to know that an Airtel tower must be nearby. Airtel allows anyone to view all the network towers it has here.
We find out that there indeed does exist a 2G tower quite near to where we deployed our IMSI Catcher.



Popular IMSIs - Finding #3

This chart shows some IMSI numbers which were captured multiple times



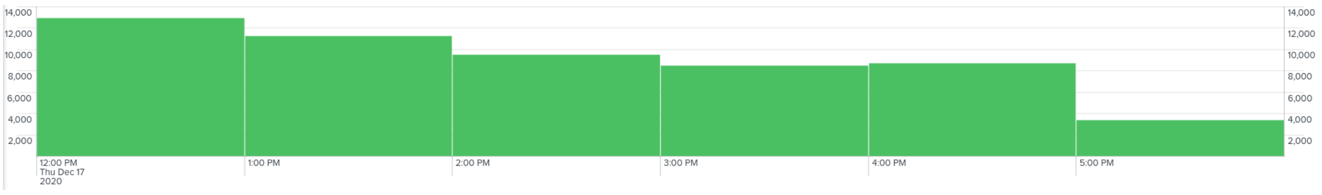


imsi	count	percent
404 10 0613051930	85	2.571861
404 10 0607704311	68	2.057489
404 10 0525588338	35	1.059002
404 10 0628519147	34	1.028744
404 10 0525058823	34	1.028744
404 10 0524489732	34	1.028744
404 96 0980439819	33	0.998487
404 10 0604778825	33	0.998487
404 90 9300238264	32	0.968230
404 10 0529875536	32	0.968230

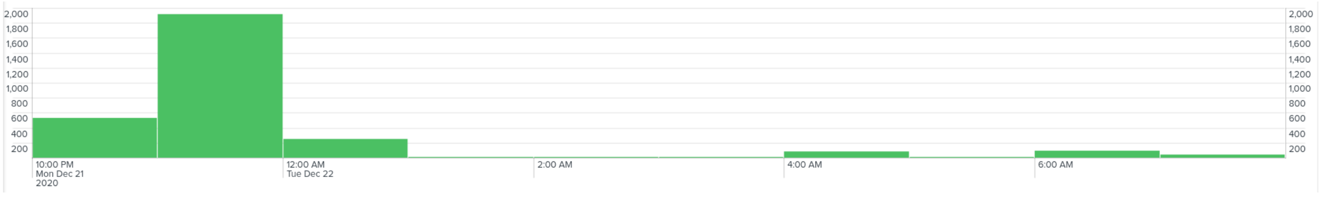
This leads us to assume that these IMSI users often went outside the range of our cell tower and connected back in. We can dig deeper and find out what time period were their IMSIs caught.
It is safe to conclude that many of these IMSI users frequent in and out of the coverage of the cell tower.

Activity - Finding #4

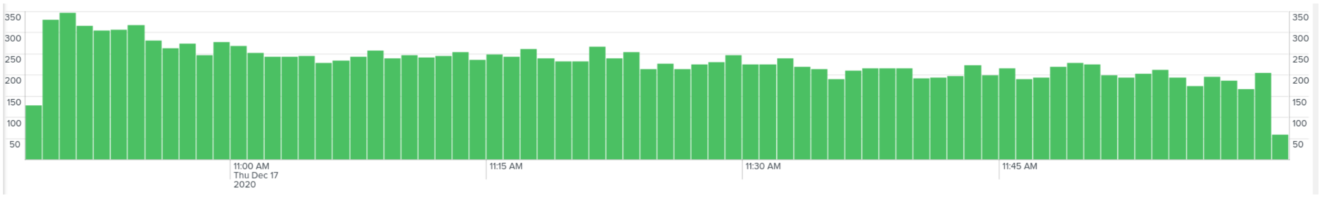
Looking through the user activity during different periods of time, we see that most IMSIs were captured during the afternoon hours, when most users were mobile and active.



The activity during late night from 10pm to 8am was significantly low



And the activity during morning hours was moderate for a short period of time



9) A Note on GSM Security

There have been multiple vulnerabilities demonstrated in different parts of the GSM Standard over the years. As we can see, with a more advanced set up and more resources, one can demonstrate and make a full fledged IMSI Catcher. However, the encryption algorithms, A5/1, A5/2, A5/3 have been proven to be weak and breakable in real time.
In fact, with an A5/1 rainbow table and some more processing power, one can crack the A5/1 encryption in real time, over the air.

Moreover, GSM is the only standard which allows unencrypted traffic leading to downgrade attacks, where 3G/4G connections are forced to downgrade to 2G without encryption.

With 2G GSM **still** being used in our country and some providers deprecating 3G instead of 2G, it begs to the question as to why this maybe and whether these well known GSM vulnerabilities are being exploited in regions where 2G GSM is the only offering available.

10) References

- Wireless Communications for Everyone
- Gotta Catch 'Em All: Understanding How IMSI-Catchers Exploit Cell Networks
- Python Simple IMSI Catcher
- Building a Passive IMSI Catcher
- GSM Frequency Bands