UNIVERSITY OF TARTU
Institute of Computer Science
Computer Science Curriculum

Georg Šumailov

Rogue Mobile Phone Base Station

Bachelor's Thesis (9 ECTS)

Supervisor: Danielle Melissa Morgan, MSc

Tartu 2023

Rogue Mobile Phone Base Station

Abstract:

Mobile network protocols have greatly improved over the decades both in terms of speed as well as security. Nonetheless, as of 2023, the outdated and vulnerable 2G network generation is still operational around the globe. This poses a threat to the general public due to known security vulnerabilities found in 2G. While various 2G rogue base stations were tested over the years, little work was done on how they perform in a modern world.

In this study a 2G rogue base station was built using YateBTS and a BladeRF. Furthermore, its effectiveness was tested in areas where there are faster networks available. Importantly, the rogue base station was built with off-the-shelf components and open-source software limiting deployment costs to a minimum. The findings of this thesis suggest that in areas covered with faster network base stations modern phones prefer new generation networks (e.g. 4G) over 2G despite the signal strength. However, in areas with no cellular connection such as tunnels or metros 2G rogue base stations can still be effective to this day. SMS phishing, SMS interception and IMSI catching attacks were all possible once the phones connected to the rogue base station.

Keywords:

rogue mobile phone base station, IMSI-catcher, SMS-interception, cellular-site simulator

CERCS:

P170 Computer science, numerical analysis, systems, control

P175 Informatics, systems theory

Võlts Tugijaam

Lühikokkuvõte:

Mobiilsidevõrgu protokollid on aastakümnete jooksul kõvasti paranenud nii kiiruse kui ka turvalisuse osas. Sellegipoolest on 2023. aasta seisuga vananenud ja haavatavad 2G-võrgud endiselt kasutusel üle maailma. See ohustab ühiskonda 2G-s leitud teadaolevate turvaaukude tõttu. Kuigi aastate jooksul on 2G pahatahtlikke telefonijaamasid uuritud palju, siis nende võimekusest tänapäevas kontekstis on teada vähe.

Selle uurimistöö põhieesmärk on testida 2G pahatahtlikku telefonijaama jõudlust kaasaegses maailmas, kus enamik riike on võtnud kasutusele uuemad mobiilsidevõrgu tehnoloogiad. Kõige tähtsam on see, et võlts tugijaam ehitati valmiskomponentide ja avatud lähtekoodiga tarkvaraga, mis viis kulud miinimumini. Oli valitud YateBTS tarkvara koos BladeRF transiiveriga. Lõputöö järeldused viitavad sellele, et kiiremate võrgutugijaamadega kaetud piirkondades eelistavad kaasaegsed telefonid signaalitugevusest hoolimata uue põlvkonna võrke (nt 4G) 2G-le. Piirkondades, kus puudub mobiilsideühendus, nagu tunnelid või metrood, võib 2G võlts tugijaam siiski olla kasulik. Pärast telefonide ühendamist pahatahtlikku tugijaamaga viidi läbi mitmeid edukaid rünnakuid. Näiteks võimaldas YateBTS konfigureerimise võimalused SMS-ide levitamist, SMS-ide pealtkuulamise ja IMSI püüdmise.

Võtmesõnad:

pahatahtlik telefonijaam, võlts tugijaam, IMSI-püüdja, SMS-püüdmine

CERCS:

P170 Arvutiteadus, arvanalüüs, süsteemid, kontroll

P175 Informaatika, süsteemiteooria

Introduction

Wireless communication technologies like mobile networks have become an integral part of our daily lives, enabling us to stay connected and access information from anywhere at any time. Even though faster and more secure new generation mobile networks have been developed, old 2G technology remains widely supported around the world to this day [1]. Unfortunately, 2G is highly outdated in terms of security by today's standards. This poses a threat as known 2G security vulnerabilities can be exploited by malicious actors. One such threat hides itself amongst real cell phone towers, which provide the ability to make calls, send SMS and much more.

A Rogue Mobile Phone Base Transceiver Station (rogue BTS), also known as a Cellular-Site Simulator (CSS) masquerades itself as a legitimate base station to lure surrounding Mobile Stations (MSs) such as phones or smartphones to interact with it. Depending on the rogue base station's intent, attacks can differ from intercepting SMS messages to International Mobile Subscriber Identity (IMSI) catching. The latter can be used to identify individuals in specific geographical areas [2].

While rogue base stations have been a concern for some time, the access to such devices was only granted to law enforcement agencies such as the United States Secret Service [3]. Coupled with the closed-source nature of CSSs and the price of around 70,000€ it was inaccessible to the public [3]. However the emergence of open-source network protocol software together with low-cost radio hardware made it easy for individuals to set up their own rogue base stations for as little as 500€ (See Figure 7). Homemade 2G rogue base stations started gaining popularity from 2010 [4].

Despite this, there has been little practical how-to research on the effectiveness of these low-cost 2G solutions for building rogue base stations in a modern world. Nowadays, there are much faster networks available and mobile network operators have significantly upgraded their infrastructure over the years [5]. This can lead to previously known 2G vulnerabilities being much less effective in a modern context. It is therefore essential to understand the viability of cheap 2G solutions for building rogue base stations and the risks they pose to wireless network security nowadays.

This thesis aims to address this gap in knowledge by investigating the practical effectiveness of low-cost solutions for building rogue base stations. Specifically, the thesis will explore the feasibility of using low-cost Software-Defined Radios (SDRs) to build fully-fledged rogue base stations. The research will also investigate potential countermeasures that can be employed to protect against rogue base stations.

1. Background

This chapter is intended to give a brief overview of general concepts used in this thesis and background information needed to understand them. The initial section of this thesis provides information on how cellular towers communicate with phones and vice versa. It will also help to get acquainted with the terminology used throughout the thesis. Terminology varies by network generation. Therefore, for simplicity sake, 2G terms will be used to describe mobile networks. Following that, the succeeding section explains different mobile network generations in order to give context for why 2G was chosen to build the rogue BTS. Subsequently, a brief analysis of the architecture of the Global System for Mobile Communications (GSM, 2G) will be presented. In order to provide a contextual framework, a brief history of cellular-site simulators will also be included. Lastly, related work will be discussed.

1.1 Mobile Network Overview

In general a mobile network today consists of multiple base stations (BTS) operated by different network providers, such as Telia or Elisa in Estonia [6]. Each BTS, also known as cell tower, is assigned a unique Cell ID (CID) to serve in a unique geographical area called a cell. Multiple cells are then organized into larger geographical areas called Location Area (LA) (See Figure 1). Each Location Area is assigned a unique code Location Area Code (LAC). Each BTS constantly broadcasts information about the identity of the network on radio frequency so nearby Mobile Stations (MSs, e.g. phones) are notified of their presence. Broadcast information includes the aforementioned CID, LAC and also a list of neighboring cells. Mobile stations can then use these neighboring cells to determine if any other BTS is worth transitioning to (e.g. because better signal). This process is called a handover. Neighboring lists allow mobile stations to save power by avoiding doing a full scan of nearby cellular towers. Usually an MS conducts a full scan when it has lost connection to its network or after a power-up. [7]

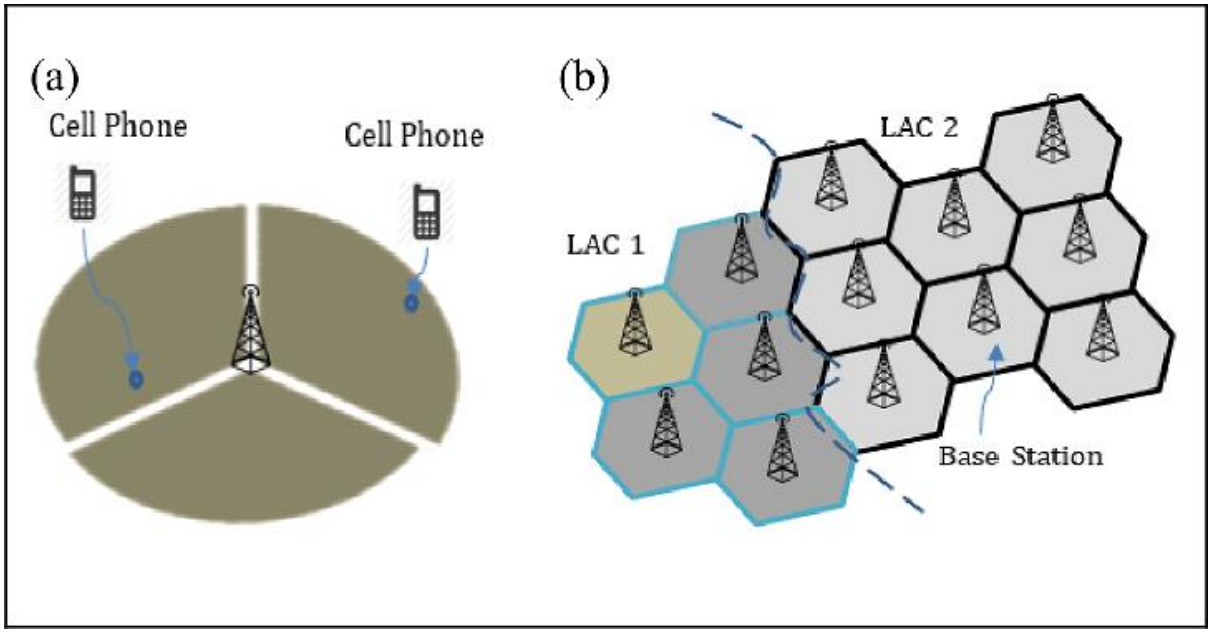

Figure 1. Base stations organized into location areas [8].

When a mobile station has picked the base station it wants to connect to it starts a registering process. The MS registers to the BTS using unique and sensitive identifiers called the International Mobile Equipment Identity (IMEI) and International Mobile Subscriber Identity (IMSI). The IMEI is used to identify hardware (e.g. phone) and IMSI to identify the user of the hardware. It is necessary for the network to know the IMSI to confirm that the user of the network has indeed paid for the service. Depending on the protocol and network generation used, a private key stored on the Subscriber Identity Module (SIM) might be needed for authentication. In the example of a GSM network, the BTS will assign an MS a Temporary Mobile Subscriber Identity (TMSI) to reduce the risk of revealing unique and sensitive IMSI information. However, the network must know where its subscribers are at any given time to be able to communicate with them. For example to send SMS or forward calls. As a result, the MS notifies a particular network of its location usually when it has moved to a new Location Area. [7]

1.2 Network Protocols

Mobile networks have undergone significant transformations since the introduction of the first-generation (1G) mobile network in the 1980s. Over time, new generations of mobile network protocols have been developed to provide faster data speeds, more advanced features and also most importantly in the context of this research - security [9]. It is important to understand key differences between various generations and their capabilities since rogue base stations will operate using a specific mobile network protocol. Currently, the latest mobile network protocol is fifth-generation (5G) technology, which promises to deliver unprecedented speeds and improved connectivity [9]. This section will provide a brief overview of the different generations of mobile network protocols, highlighting their key features and limitations.

1.2.1 0G (pre-cellular networks)

Wireless telephone began with zero generation networks shortly after the second World War. It was possible to communicate only on a predefined set of channels. Phones did not support the handover feature which meant that data (e.g. voice) could not be transferred from one base station to another. Primary users were construction workers, realtors and celebrities. Zero generation networks were used purely for basic voice communication [9]. Security was not a concern and anyone who tuned on the right frequency could eavesdrop on the conversation [10].

1.2.2 1G (first-generation networks)

The first-generation mobile network was introduced in the 1980s and became widespread in the 1990s. 1G allowed users to make voice calls using analog signals technology like 0G [9]. Analog signals meant that radio interference was a big problem and security was still not a concern. The main difference between the previous generation was the use of cells. Geographical areas were divided into cells to allow for multiple devices to operate on the same frequency without interference. This allowed for better network coverage which was a significant achievement and laid the foundation for subsequent generations of mobile networks [10]. 1G also featured handover and roaming meaning that phone communication was possible in other countries outside of network provider coverage. However, even though roaming was possible, it was not interoperable between countries. This meant that it wasn’t possible to call

someone from another country [9]. Additionally, 1G did not support sending text messages. In conclusion, the first-generation mobile network had several limitations that led to the development of newer technologies [10].

1.2.3 2G (second-generation networks)

Second-generation networks will be explained more in-depth compared to other network protocols since 2G is the main area of research in this paper.

The second-generation network was a big leap in telephone wireless communication technology when it launched in 1991. 2G introduced various features that are widely used to this day which includes Short Message Service (SMS), picture messaging and Multimedia Messaging Service (MMS). MMS messages are similar to SMS but allow for larger file sizes and richer content. [9]

With the emergence of 2G various data encryption protocols were introduced to improve the absence of security measures in the previous generations [9]. Nonetheless, a major drawback persists. 2G does not support mutual authentication. As a result, only the mobile station is authenticated by the base station but not vice versa [11]. This means that anyone can set up their own 2G base station without connecting phones doubting its legitimacy. This makes 2G a very favorable decision when choosing a rogue BTS mobile network protocol.

While there are multiple newer and faster mobile network protocols, 2G remains still widely used throughout the world thanks to its low implementation costs. This resulted in various network providers all over the world shutting down the newer 3G first and keeping 2G operational [12]. The same applies to Estonia, where current thesis research is conducted. Estonian network provider Telia started shutting down its 3G networks in 2023 while Elisa and Tele2 plan to do so in the upcoming years. 2G, however, is here to stay until at least 2025 [13].

1.2.3.1 2G - GSM

Global System for Mobile communications (GSM) is the first European standard that was developed for 2G and as a result became widespread all over the world. GSM is a circuit-switched technology that was originally designed for voice communication. It divides the available frequency spectrum into channels, with each channel reserved for a single conversation. When a user makes a call, a dedicated circuit is established between the two parties for the duration of the call. While GSM can also support data transmission, it is limited to relatively low speeds, and data services are typically charged based on the amount of time the circuit is in use [9]. GSM allowed transmission speeds of around 6-10 messages per minute [10]. The GSM protocol has been already shut down in many parts of the world and mainly newer 2G protocols such as EDGE and GPRS are the ones that remain operational to this day [14].

1.2.3.2 2.5G - GPRS

General Packet Radio Service (GPRS) is an extension of GSM to support packet-switched data transmission instead of the circuit-switched protocol. Since the frequency allocated for mobile communications is finite and expensive it made sense to make specific frequency channels usable simultaneously by various clients. With packet-switching, messages are divided into small packets which are sent independently to the destination. A message can then be

reassembled at the destination when all of the packets have been received. Packet-switched protocols include Internet Protocol (IP) which made it possible for GPRS to support internet access. Additionally, GPRS improved SMS transmission speeds to about 30 messages per minute [10].

1.2.3.3 2.75G - EDGE

Enhanced Data Rates for GSM Evolution (EDGE) or Enhanced GPRS was deployed in 2003 to provide increased data rates compared to GPRS. It is easy to tell if a phone is using EDGE by the logo in the status bar (See Figure 2).

Figure 2. Android phone using EDGE mobile network.

Technically EDGE is considered a 3G technology as it supports data rates of over 200 kbit/s. For example a typical text file of 320 kbit is transferred in only 2 seconds whereas in GPRS technology it would require around 6 seconds [9]. Later, Evolved EDGE was developed to further improve 2.75G achieving speeds of over 1Mbit/s or 1000 kbit/s [10].

1.2.3.4 GSM encryption

In GSM networks several algorithms are used for authentication and encryption. As GSM has been around for many years some of these algorithms had to be updated and replaced due to being labeled defunct [15]. Additionally, these algorithms have to be updated as advancements in hardware can quickly find the keys used in the encryption process [15]. Below the most popular algorithm used for GSM security, the A5, will be explained.

A5: The most notable GSM encryption standard is A5. It is used to encrypt wireless transmissions between the mobile station and the base station. A5/1 and A5/2 were developed in the 80s and in 1999 their protocols were reverse engineered and revealed [2].

- A5/0 - Indicates that there is no encryption used and messages are sent between the MS and BTS in plain text. This allows for anyone who listens on the specified frequency to gather sensitive information [2].
- A5/1 - The cipher used to encrypt messages with this algorithm can be cracked within seconds with today's technology [2].
- A5/2 - Extremely weak cipher that can be cracked in real-time. Discontinued since 2006 meaning no phones made since 2006 support A5/2 [2].
- A5/3 - This cipher can be cracked within 2 hours on a relatively weak PC [15]. It is unknown if anyone has successfully applied this attack in a real-life environment [2].

The A5/0 standard is especially important in the context of this research. This means that the rogue base station can communicate to the phone that it supports only A5/0 encryption.

Therefore the communication, like SMS messages and calls, can be easily intercepted without the need of further processing (e.g. deciphering).

1.2.4 3G (third-generation networks).

In the early 2000s, International Telecommunication Union (ITU) rolled out standards to label specific mobile communication technologies as 3G. On most phones, 3G can be identified by either 3G or H logo in the status bar (See Figure 3).

Figure 3. Android phone using 3G enhanced HSPA (HSPA+) mobile network.

The standard is called the International Mobile Telecommunications for the year 2000 (IMT-2000) [9]. The most important improvement in 3G is security. Compared to earlier 2G, now mutual authentication is required. With mutual authentication mobile phones verify the base station's identity with challenge-response protocol and vice versa. This is achieved by solving cryptographic challenges on both sides that require secret keys [16]. 3G has also significantly improved data transmission rates which allowed mobile television and video conferencing features. Several 3G standards and protocols exist such as Universal Mobile Telecommunication System (UMTS), High Speed Packet Access (HSPA) and many more which made downlink (from base station to phone) data rates of up to 42 Mbit/s possible [10].

1.2.5 4G (fourth-generation networks)

The 4G standard, as well as 3G, is specified by the ITU. One of the most famous standards for 4G is Long Term Evolution (LTE) and IEEE 802.16m (WIMAX). 4G can be easily identified by its logo in the status bar (See Figure 4).

Figure 4. Android phone using 4G mobile network.

The requirements for the fourth-generation mobile network data transmission speed is up to 100 Mbit/s while moving (e.g. in a car or train) and up to 1 Gbit/s while stationary [9]. 4G offers better security as opposed to 3G. 4G internally is mainly IP-based which makes it possible for more end-to-end encryption via Ipsec protocol. Ipsec provides overall more authorization by ensuring that only authorized users can access data resources [17]. In 2023, 70% of all the phone call volume in Estonia flows through 4G mobile networks making it the most popular generation in the country [13]. 4G is also the most dominant mobile network in the world [12].

1.2.6 5G (fifth-generation networks)

Global mobile network operators started launching new 5G networks in early 2019. The fifth-generation network (5G) introduced higher peak data rates up to 20 Gbit/s based on the IMT-2020 requirements. It makes 5G around 20-30 times faster than 4G. 5G also introduced counter-measures to IMSI-catchers by encrypting the sensitive user identifier SUPI (same as IMSI in 2G) [18]. However, a work published in 2021 proves that targeted tracking of specific users in 5G networks remains possible [19].

Overall, 2G was chosen as the focus of this research due to its lack of mutual authentication, broken encryption and worldwide adoption.

1.3 2G network architecture

As 2G was chosen, it is therefore important for the context of this research to understand general concepts and definitions in regards to the 2G architecture. This architecture is displayed in Figure 5, where the main components of the 2.5G are seen. The following paragraphs explain the role of each component in the mobile network.

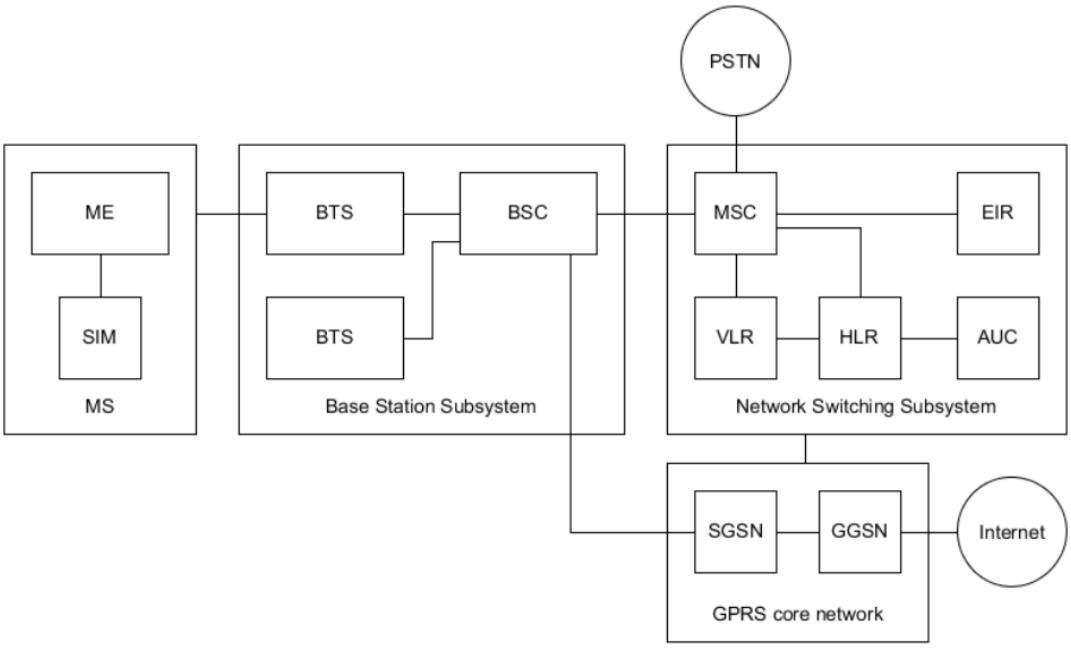

Figure 5. A simplified 2.5G network overview [10].

The **Mobile Station (MS)** is a device capable of communication using the 2G network. In order for the MS to connect to a legitimate base station a valid SIM card is required. A SIM card is a separate component of the MS which stores the IMSI to identify the user of the network. The

hardware used for receiving and sending data via 2G is called the mobile equipment (ME). The ME can also be uniquely identified thanks to the IMEI stored on it [10]. On mobile phones it is usually explicitly displayed on the backplate of the phone. The IMEI can also be viewed on most phones by dialing *#06# [20].

The **Base Transceiver Station (BTS)** is what relays information to the MS and vice versa. A BTS is also referred to as a "cell tower" and covers a certain cell in a cellular network. Grouping of the BTS in cells is determined by various factors such as population density in a given area [2].

The **Base Station Controller (BSC)** covers multiple base stations. BSC is responsible for handling handover procedures between different base stations under its control. If one of the BTS is not connected to the same BSC then the handover is handled by the Mobile Switching Center [2].

The **Mobile Switching Center (MSC)** is responsible for managing the authentication and routing of calls and SMS. As a result it needs to communicate with the Public Switched Telephone Network (PSTN). The PSTN is a combination of different landline networks used worldwide. The MSC is also responsible for inter-BSC handover procedures [10].

The **Home Location Register (HLR)** is used for storing sensitive subscriber data such as IMSI, phone number and other white-list information [2]. IMSIs can be used for location tracking of an individual, therefore this information is considered sensitive.

The **Visitor Location Register (VLR)** holds active subscriber data for MSC connections. Temporary Mobile Subscriber Identity (TMSI), which is an alias for the IMSI is one such example. TMSI makes it more difficult for malicious actors to identify network subscribers [2].

The **Authentication Center (AUC)** purpose is to manage authentication to the network. The AUC stores every registered network subscriber private key used for cryptographic authentication. The same key that is embedded in the SIM card is used to solve cryptographic challenges that are generated by the AUC [10].

The **Equipment Identity Register (EIR)** is a service that contains IMEI numbers of banned or stolen mobile equipment. The EIR is not always implemented in every network [10]. However, various internet resources exist where it is possible to check if a certain ME device is reported as stolen by providing the IMEI. The IMEI can also give out potentially sensitive information about the device such as manufacturer, model etc [20]. Since IMEI is transmitted to the BTS upon connection, it can be used for tracking as well making this data sensitive.

The **Serving GPRS Support Node (SGSN)** is responsible for handling all packet-switched data inside the network. The SGSN handles authentication of GPRS devices and routing data-packets to and from the MS. The SGSN's purpose is similar to that of the MSC. However, in contrast to the SGSN, the MSC is circuit-switched. In practice this means that MSC is mostly used for routing calls and the SGSN for managing packets of internet origin [10].

The **Gateway GPRS Support Node (GGSN)** is an interface for external packet-switched networks (e.g. the internet). The GGSN routes data coming from a mobile station to the corresponding packet-switched network. Similarly, any data coming from outside the network uses the GGSN as an entry in order for the information to reach the mobile station [10].

Now that the main components of the GSM network were described, a history of cellular-site simulators will be given. Rogue base stations were exploiting GSM security vulnerabilities for decades and the next section is going to give a brief chronological overview of that.

1.4 History and Usage of Cell-Site Simulators

The use of cell-site simulators dates back to the early 1990s when the United States Federal Bureau of Investigation (FBI) began using devices manufactured by the Harris Corporation to track the locations of suspects. In the years that followed, cell-site simulators became small enough that they could covertly be mounted on top of a vehicle [3].

In 2001 Harris Corporation filed a trademark registration for a device called the Stingray, which was one of the first portable cell-site simulators (See Figure 6). The Stingray was about the size of a briefcase and could be used to intercept cell phone signals and collect international mobile subscriber identities (IMSIs). Stingray has become a catchphrase for IMSI-catchers although Harris Corporation has introduced other models. One of such models, the Amberjack, came equipped with built-in magnets so that it could be attached to the roof of a law enforcement vehicle [3].

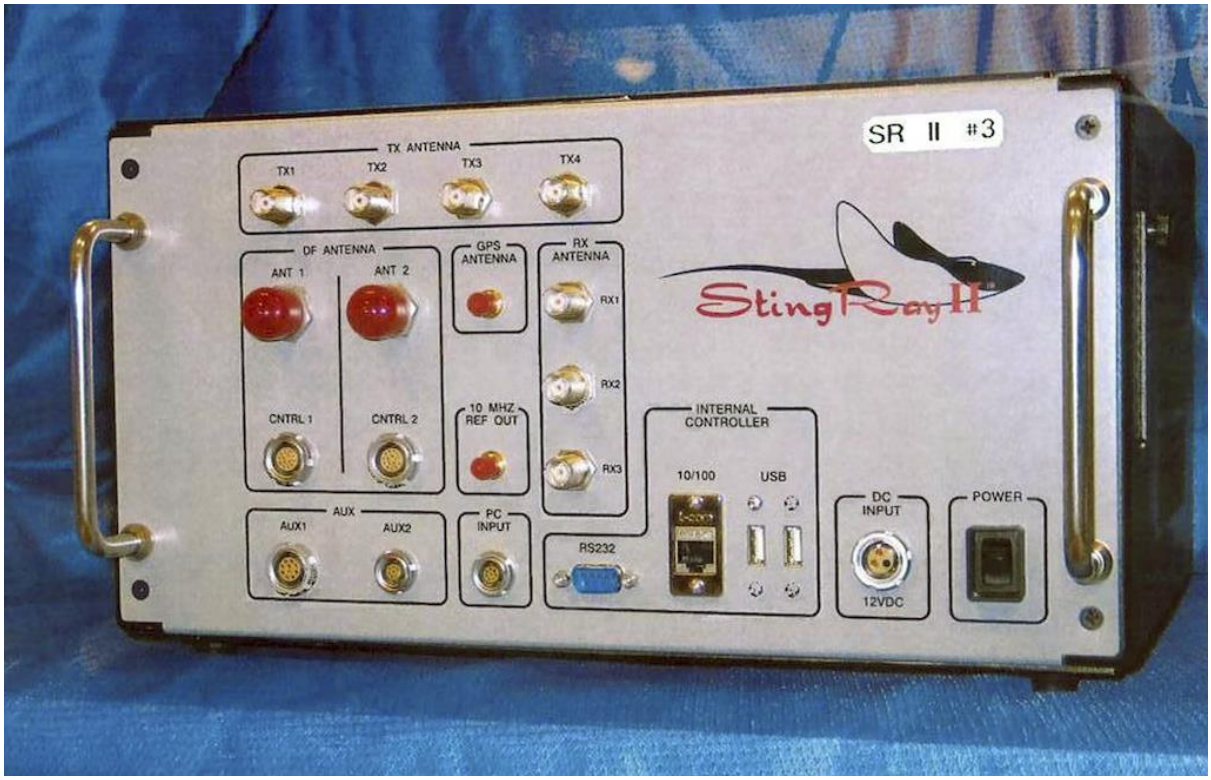

Figure 6. Stingray II manufactured by Harris Corporation [21].

The use of cell-site simulators by law enforcement agencies became more widespread in the 2010s. In 2015, the American Civil Liberties Union (ACLU) obtained documents that revealed that the FBI was using cell-site simulators to track the locations of suspects without obtaining a warrant. The use of cell-site simulators without a warrant raised concerns about privacy and civil liberties [22].

In 2015, the United States Department of Justice (DOJ) issued a policy requiring law enforcement agencies to obtain a warrant before using cell-site simulators in most cases. However, there are exceptions to the warrant requirement, such as in cases involving national security or in situations where obtaining a warrant would be impractical [23]. Despite the newly imposed regulations on the use of CSS, several law-enforcement agencies have failed to comply [24]. United States federal agencies such as the Secret Service and Immigration and

Customs Enforcement (ICE) have deployed cell-site simulators without proper authorization and in violation with the law [25].

Today, CSS are widely used by law enforcement agencies not only in the United States but also all around the world. It is also fair to assume that CSS are actively used not only for domestic surveillance but also for international operations. One notable incident happened in Estonia in 2015. Norwegian military portal Aldrimer states that Russian Special Forces operated a fake base station in Pärnu during NATO exercises [26]. An unknown mobile base station was allegedly deployed near the territory where the exercise took place. Estonian law-enforcement agencies refused to comment on the Aldrimer report and overall the article was met with criticism due to logical inconsistencies in the report [27].

In conclusion, CSS technology has been known to law-enforcement for over two decades and is being widely used around the world. Technical capabilities of commercial-grade CSS remain largely unknown due to non-disclosure agreements between vendors and buyers. Therefore it is up to researchers and interested individuals to unravel technical mysteries of commercial CSS. With the emergence of cheap hardware it became possible to build CSS in home environments. CSS usage in the hands of the public will be discussed in section "Related work".

1.5 Software-Defined Radio

Software-Defined Radios (SDRs) are a game-changer in the field of wireless communication, offering unmatched flexibility and versatility. Unlike traditional hardware-based radios, SDRs rely on software to implement radio frequency (RF) communication systems, enabling users to create customized radio systems that meet their specific requirements (See Figure 7). [28]

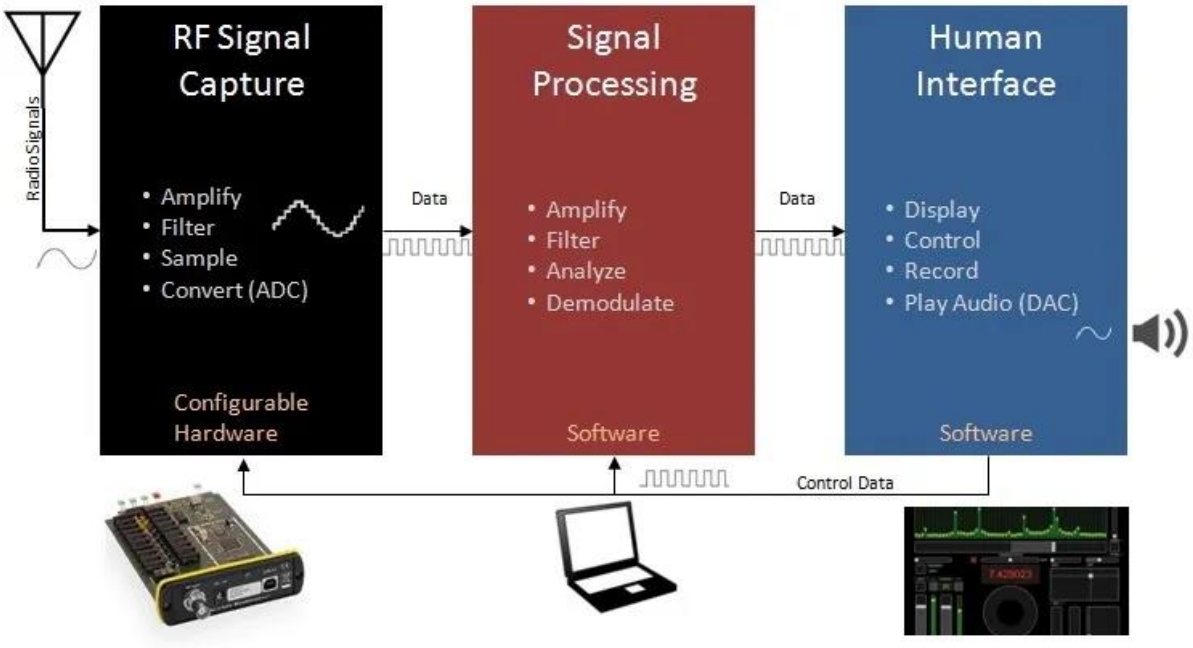

Figure 7. General overview of an SDR [29].

Software-defined radio was chosen for implementing a rogue base station because of the advantages it has over traditional radio:

- SDRs are cheap (See Table 1)

- Availability of open-source mobile network implementations [30], [31]
- Previous work [2], [10]
- SDR parameters are easily configurable which makes it useful for setting up own base station [32]

The specific SDR used for this research will be explained in section 2.2.2.

1.6 Related Work

One of the most notable first public demonstrations of the effectiveness of homemade IMSI catchers was done by Chris Paget in 2010 at DEF CON 18. Chris used affordable off-the-shelf hardware and open-source software to operate a rogue BTS. Chris demonstrated how easy it was to spoof a legitimate cell tower due to security vulnerabilities in 2G [4].

In 2016, Kenneth Van Rijsbergen from the University of Amsterdam conducted practical research to test the effectiveness of a homemade IMSI catcher built with YateBTS and BladeRF x40. BladeRF is the software-defined radio platform and YateBTS is the software mobile network implementation for it (See Chapter 2). As a result, Kenneth concluded that with such a setup it was tricky at best to make 3G and 4G capable phones connect to a 2.5G rogue base station. Newer phones are more selective towards base stations they connect to and they prefer faster protocols even if 2G has a better signal. However, if a phone does connect with the IMSI catcher then all internet traffic and outgoing SMS can be intercepted [2].

2 years later in 2018, Thomas Veens further explored the capabilities of the Nuand BladeRF x40 towards automated 2G traffic interception and penetration testing of 2G embedded devices. Thomas concluded that several active attacks such as Man-in-The-Middle (MiTM) on data services, DNS spoofing and scanning for open ports on the device were possible if it was connected to the rogue base station [10].

In 2019, a report was published by Electronic Frontier Foundation to explain how IMSI catchers exploit cellular networks [33]. The article explained in theory how various attack methods work in different mobile network generations as well as how to detect them. The report concluded that it is rather complicated to have a defense strategy against state-deployed CSS based on assumptions on how commercial rogue base stations work. It is unknown how state-deployed CSS work and our knowledge relies on research findings [33].

2. Research Setup

Research at hand will focus on easy-to-setup rogue base station implementation. Therefore a minimum amount of hardware will be selected. Since SDRs are a lot cheaper to implement than traditional hardware radio equipment, an SDR will be chosen. SDRs also provide more flexibility since they support a large variety of different protocols and applications [28]. Additionally, software used should be preferably open-source to lower costs of the rogue base station deployment.

2.1 Software

In software-defined radios software is, of course, the most important part as it implements various network protocols, does signal processing and much more. The SDR software must be supported by the underlying operating system as well. This section explains what BTS software was chosen and why as well as the computer operating system that supports it.

2.1.1 Base Station Software

There are various options that provide a software-defined 2G protocol implementation. However, the list of options was limited to BTS software that provided the following functionality:

- Quick to set up (e.g. compile and configure)
- Be open-source
- Be compatible with chosen hardware
- Support GSM protocols

For consideration, two open-source protocol implementations were chosen, the YateBTS and OpenBTS.

2.1.1.1 OpenBTS

OpenBTS is the first free open-source implementation of a base station. It was also the choice of the aforementioned Chris Paget DEF CON presentation in 2010 [4]. OpenBTS supports GSM and GPRS but not EDGE. Initially it was designed for the Ettus Research USRP board but over time the list of supported devices grew, including Nuand BladeRF [10]. The last major source code updates were made 7 years ago in 2016 and the official website *openbts.org* seems to be non-functional at the moment of writing [30].

2.1.1.2 YateBTS

Yet Another Telephony Engine Base Transceiver Station (YateBTS) was released in 2014 as a fork of the OpenBTS project. The goal of YateBTS was to improve on OpenBTS by removing external dependencies and introducing modular design for easier functionality extension [10]. Like OpenBTS, YateBTS does not support EDGE mobile network protocol. YateBTS offers a Network in a PC (NIPC) demo javascript application which makes it easy for GSM BTS configuration. YateBTS also provides superb support for the Nuand bladeRF [34]. YateBTS official website as of date is operational, unlike OpenBTS. The website also mentions a Github repository with a last commit made in 2021 [34]. There are various research articles [2], [10]

and community content available [35] which makes it a viable choice for a BTS setup. After careful consideration, YateBTS was chosen for the 2G architecture software implementation.

2.1.2 Operating System

For the operating system DragonOS was chosen. In 2020 a user by the username of cemaxecuter decided to make an out-of-the-box Lubuntu based x86_64 operating system DragonOS for anyone interested in software defined radios. It leverages the portability, security, and power of Lubuntu Linux and contains a pre-installed suite of the most powerful and accessible open source SDR software. DragonOS has verified support for a range of inexpensive and powerful SDR hardware, including RTL-SDR, HackRF One, BladeRF and many more. [35]

2.2 Hardware

To build the BTS, two pieces of hardware components had to be selected. First of all a host device (e.g. laptop) was selected that is powerful enough to facilitate BTS calculations as well as other hardware component connectivity. Secondly, a software-defined (SDR) radio was chosen which is going to be responsible for transmitting and receiving signals simultaneously. In addition, it might be useful to monitor signals sent from the rogue BTS on the electromagnetic spectrum. Rogue BTS SDR component is not capable of monitoring itself while operating because it will be dedicated to serve BTS purposes only. Therefore additional SDR for monitoring was required.

2.2.1 Host Device

For the host device an HP Elitebook 840 G6 laptop with Windows 10 installed was chosen due to convenient availability. The Elitebook has 16 GB of RAM and an Intel i5 CPU which is sufficient for running YateBTS. The aforementioned laptop also provides 3.0 USB ports which is recommended to ensure smooth BladeRF operation [32]. Since chosen BTS software must run on Linux, Windows operating system was not sufficient. Therefore a free virtualization tool OracleVM was installed with 5GB of memory and 4 processors allocated for the DragonOS operating system. The OracleVM free alternative VMware Workstation Player does not support USB devices connectivity in the trial version which makes it unusable for this setup.

2.2.2 SDR

When BladeRF launched in 2013 as a kickstarter project it was one of the most powerful software-defined radios available at the time. 10 years have passed since its kickstarter launch and the company behind it, Nuand, has made several improvements on the BladeRF as well as introduced new versions of it [36]. One of the most key-features of the BladeRF is that it can operate in full-duplex mode [37]. This implies that the device can send and receive signals in parallel, which is essential when building a BTS.

BladeRF was chosen as the SDR due to its excellent compatibility with YateBTS [34] and previous work availability [2], [10]. There are, however, different versions of BladeRF. The main price difference lies in the size of their Field-Programmable Gate Array (FPGA). An

FPGA is a programmable digital hardware component that can be used to implement various signal processing functions, such as filtering, demodulation and channel coding [10]. In BladeRF it is used in conjunction with a high-speed Analog-To-Digital Converter (ADC) to digitize the Radio Frequency (RF) signal received by the BladeRF antennae [37]. In practice, having a larger FPGA means more available resources for complex signal processing algorithms. Only the cheapest BladeRF devices will be compared since there are no intentions to write custom software for the FPGA (See Table 1). In addition, YateBTS comes bundled with FPGA firmware images that work with both versions.

Table 1. BladeRF x40 and xA4 comparison, specifications taken from official webpage [38].

Device name	**Nuand store availability**	**Price**	**2G frequency range support**	**USB 3.0 support**	**Release date**
BladeRF x40	Out of stock	520€	Yes	Yes	2013
BladeRF xA4	In stock	540€	Yes	Yes	2018

Both BladeRF models are suitable for building a rogue base station. For the availability reasons, BladeRF xA4 was chosen for the build.

2.2.3 Monitoring SDR

In order to monitor outgoing Transmission (TX) signals from the BTS, the HackRF was used due to convenient availability. However, cheaper options like RTL-SDR would suffice as well [39]. Unlike BladeRF, HackRF can only work in half-duplex mode meaning it is not able to transmit and receive signals at the same time [40]. Half-duplex mode is enough for monitoring purposes.

2.3 Base Station Setup Configuration

In this section steps to configure the BTS will be outlined. Before any experiments can be conducted a minimal setup needs to be achieved. This involves installing all the necessary software and making sure that software is working along hardware as intended.

2.3.1 Host Operating System

First of all, Oracle VM VirtualBox manager (Virtualbox) needs to be installed. A step-by-step guide for Virtualbox setup is out-of-scope for this research thesis. For installation instructions refer to Virtualbox's official webpage [41]. For the guest operating system Linux was selected. The guest operating system does not matter since it will be overridden by DragonOS. Next, the DragonOS optical disk image was attached to the guest operating system and the virtual machine was booted up. The DragonOS distribution version used for this research is R28, which can be downloaded from the sourceforge webpage [42].

DragonOS offers a large plethora of pre-installed SDR software that comes bundled with the operating system. This means that the BTS setup does not require the compilation and installation of needed software which speeds up the setup configuration process. The pre-installed software suite includes an FPGA firmware image specifically for the BladeRF xA4, BladeRF software as well as YateBTS itself. To ensure that DragonOS has the correct software installed that is compatible with selected hardware it is necessary to check software versions in the command-line (CLI) and test their functionalities.

2.3.2 SDR Software Compatibility With Hardware

Secondly, the BladeRF software offers a command-line tool to interact with the BladeRF SDR. With *bladeRF-cli* it is possible to flash firmware files, load FPGA bitstreams, probe for BladeRF devices and look up firmware versions [43] (See Listing 1).

```
georg@dragon:~$ bladeRF-cli --interactive
bladeRF> probe

  Description:    Nuand bladeRF 2.0 (currently open)
  Backend:        libusb
  Serial:         ********************************
  USB Bus:        2
  USB Address:    4

bladeRF> info

  Board:                     Nuand bladeRF 2.0 (bladerf2)
  Serial #:                  ********************************
  VCTCXO DAC calibration:    0x20bb
  FPGA size:                 49 KLE
  FPGA loaded:               yes
  Flash size:                32 Mbit
  USB bus:                   2
  USB address:               4
  USB speed:                 SuperSpeed
  Backend:                   libusb
  Instance:                  0

bladeRF> version

  bladeRF-cli version:        1.8.0-0.2021.10-2
  libbladeRF version:         2.4.1-0.2021.10-2

  Firmware version:           2.4.0-git-a3d5c55f
  FPGA version:               0.11.0 (configured by USB host)
```

Listing 1. Checking device connectivity and version info with *bladeRF-cli* tool.

The *--probe* command allows testing device connectivity. If successful, the *bladeRF-cli* will be able to identify the SDR and print out its information. The *info* command is useful to make sure that the FPGA image was loaded for smooth BTS operation. The *bladeRF-cli* offers a *version* command as well which is helpful in case of troubleshooting.

2.3.3 Loading FPGA Image

Thirdly, the FPGA image for the BladeRF is loaded. To ensure smooth operation of the device-specific FPGA image should be chosen. See Listing 2 for FPGA image loading with *bladeRF-cli*. It is not recommended to rely on the autoload feature and therefore the FPGA image should be loaded after every device boot.

```
georg@dragon:~$ cd /usr/src/yate-rc-3/
georg@dragon:/usr/src/yate-rc-3$ ls -la
total 23944
drwxr-xr-x   2 root root     4096 okt   11  2022 .
drwxr-xr-x 136 root root     4096 apr   17 14:50 ..
-rw-r--r--   1 root root  3571462 okt   11  2022 hostedx115.rbf
-rw-r--r--   1 root root  1191788 okt   11  2022 hostedx40.rbf
-rw-r--r--   1 root root  2632660 okt   11  2022 hostedxA4.rbf
-rw-r--r--   1 root root  4244820 okt   11  2022 hostedxA5.rbf
-rw-r--r--   1 root root 12858972 okt   11  2022 hostedxA9.rbf
-rw-r--r--   1 root root     2360 okt   11  2022 README
georg@dragon:/usr/src/yate-rc-3$ bladeRF-cli -l hostedxA4.rbf
Loading fpga...
Successfully loaded FPGA bitstream!
```

Listing 2. BladeRF micro 2.0 xA4 FPGA firmware image loading with *bladeRF-cli* tool.

In Listing 2 various FPGA images are visible. All the FPGA images listed came with YateBTS. For this research *hostedxA4.rbf* was chosen because the SDR is BladeRF xA4. For example, if a different BladeRF was chosen then the procedure would be identical except for the target FPGA image.

2.3.4 Testing SDR Signal Reception

After the FPGA image is loaded, the ability of the BladeRF to receive RF signals is tested with pre-installed open-source software Gqrx (See Figure 8). Gqrx offers useful visual representation of received signals by plotting them on the waterfall (bottom visualization) and FFT plot (upper visualization) [44]. The receiver works if the plots are constantly changing and audible noise is being generated by the application.

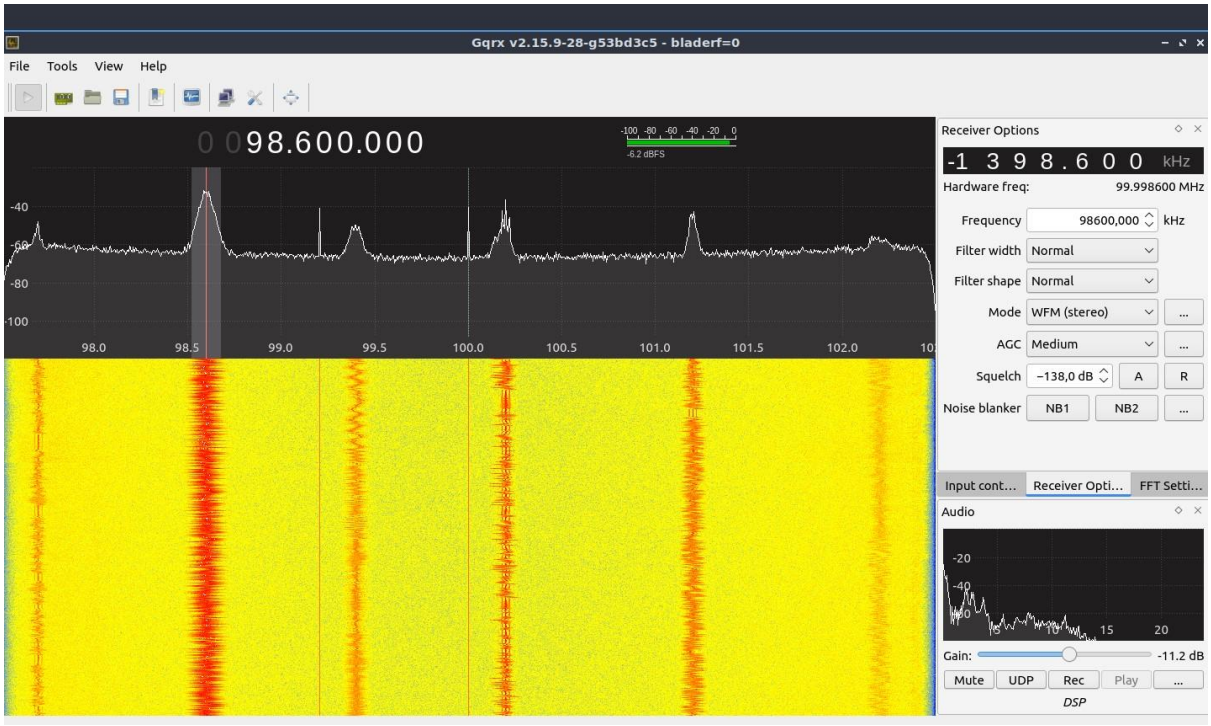

Figure 8. BladeRF listening on 98.6MHz (RetroFM station [45]) in Gqrx.

Testing BladeRF signal reception capabilities is needed because the base station will need to be able to receive signals from the phone. Successful signal reception raises confidence in the overall setup compatibility.

2.3.5 Configuring SDR Software

Finally, before the configuration can take place, the YateBTS software needs to be verified. In order to check YateBTS version see Listing 3. Note that the *yate* command does not need to be run in the *yate-rc-3* folder.

```
georg@dragon:/usr/src/yate-rc-3$ yate --version
Yate 6.2.1 devel1 r
```

Listing 3. Show the Yate version in the command-line.

Before starting the SDR software YateBTS, it can be configured with the NIPC Web GUI application. NIPC application makes it easy to configure BTS parameters such as broadcasting frequency, signal strength and much more. NIPC application will be available at *http://localhost/nipc/* after the apache server has been started (See Listing 4).

```
georg@dragon:~$ sudo /etc/init.d/apache2 start
Starting apache2 (via systemctl): apache2.service.
```

Listing 4. Starting apache2 service.

The most important 2G parameters that are tunable through the NIPC, as shown in Figure 9, will be discussed below:

- **Radio.Band** is the frequency range on which the BTS will operate. Valid values are GSM850, EGSM900, DCS1800 and PCS1900. Numbers in the frequency band usually refer to approximate frequencies in MHz on which the tower operates. For example EGSM900 will start at 935 MHz downlink and 890 MHz uplink. All listed options are suitable for GSM. Explanation was taken from the NIPC application [34].
- **Radio.C0** is the Absolute Radio Frequency Channel Number (ARFCN) and is a unique number for the channel on a radio band (e.g. GSM850). If the ARFCN is known, it is possible to calculate the downlink frequency and the the uplink frequency of a BTS. [10]
- **Identity.MCC** is the Mobile Country Code (MCC) which represents the country of origin of the base station. The MCC is always three digits in length. For Estonia, the MCC is 248 (See Figure 12). Not to be confused with calling code which is used to identify the country to which the call is being made (e.g. +372 in Estonia).
- **Identity.MNC** is the Mobile Network Code (MNC) which is a two or three digit code that is used to identify mobile network operators in a specific country. For example an Estonian network operator Elisa is 02.
- **Identity.LAC** is the Location Area Code (LAC) which is a two byte value that identifies a cell group. The LAC is important for geographical tracking. [7]
- **Identity.CI** is what identifies the BTS within a LAC. [7]
- **Identity.ShortName** is used to display network names when display space is limited. In YateBTS version 6.2.1-devel1 it is not used.
- **Radio.PowerManager.MaxAttenDB** is the maximum attenuation of the transmission signal strength from the full output power. This parameter sets the minimum output power of the SDR. Values can range from 0 dB to 80 dB. The bigger the value the less output power. [10]
- **Radio.PowerManager.MinAttenDB** is the minimum attenuation of the transmission signal strength from the full output power. This parameter sets the maximum output power of the SDR. Possible values are the same as in maximum attenuation parameter. To achieve maximum output power of the SDR both MaxAttenDB and MinAttenDB need to be set to 0. [10]

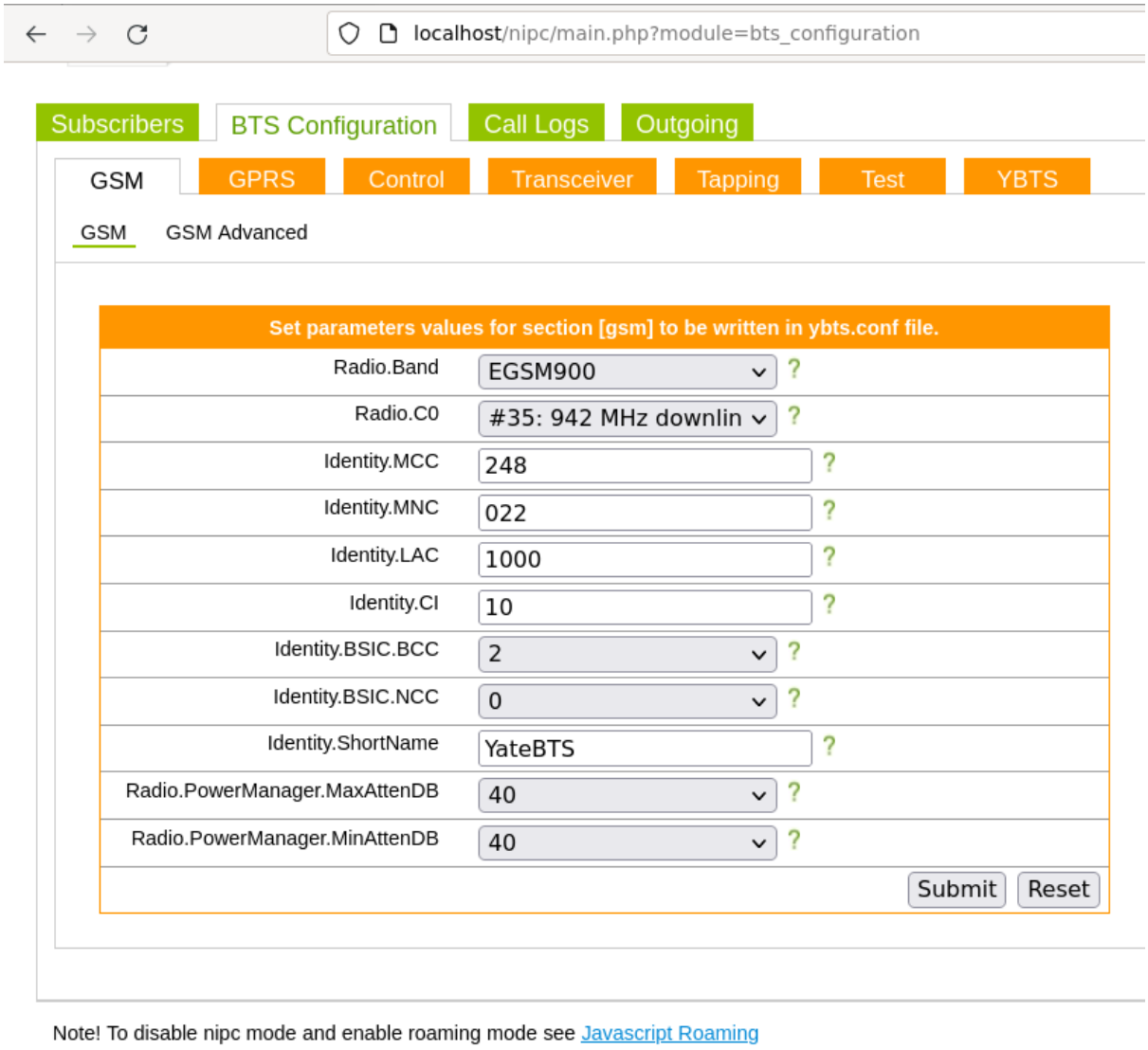

Figure 9. GSM parameters for YateBTS.

The ARFCN parameter was chosen by looking at the radio frequency spectrum with the monitoring SDR. An empty ARFCN channel was picked for the rogue BTS. Before starting the BTS, a test IMSI filter (See section 2.4.1) was configured. Lastly, Listing 5 shows how the yate command was used to start up the base station and the output of the command when it is successful.

```
georg@dragon:~$ sudo yate
[sudo] password for georg:
Yate (4119) is starting Thu Apr 27 00:11:52 2023
Loaded module Analyzer
Loaded module iLBC - based on iLBC reference library
Loaded module YRTP
...
2023-04-27_00:12:02.481239         <mbts:NOTE>          proc          4148
TRXManager.cpp:153:sendCommandPacket:       thread       139896820266816:
this:0x5618d143f500 response RSP RESET 0 to command CMD RESET 1
2023-04-27_00:12:02.481285         <mbts:NOTE>          proc          4148
TRXManager.cpp:127:sendCommandPacket:       thread       139896820266816:
this:0x5618d143f500 command CMD STATISTICS OFF
```

Listing 5. Starting up YateBTS.

After starting the BTS, the HackRF SDR was used to see outgoing transmission (downlink frequency) signals from the base station. Since it was not possible to transmit and receive signals on different devices simultaneously on one host operating system, an additional laptop with Linux operating system and Gqrx installed was used. BTS was clearly visible on the visualized graph provided by the Gqrx (See Figure 10).

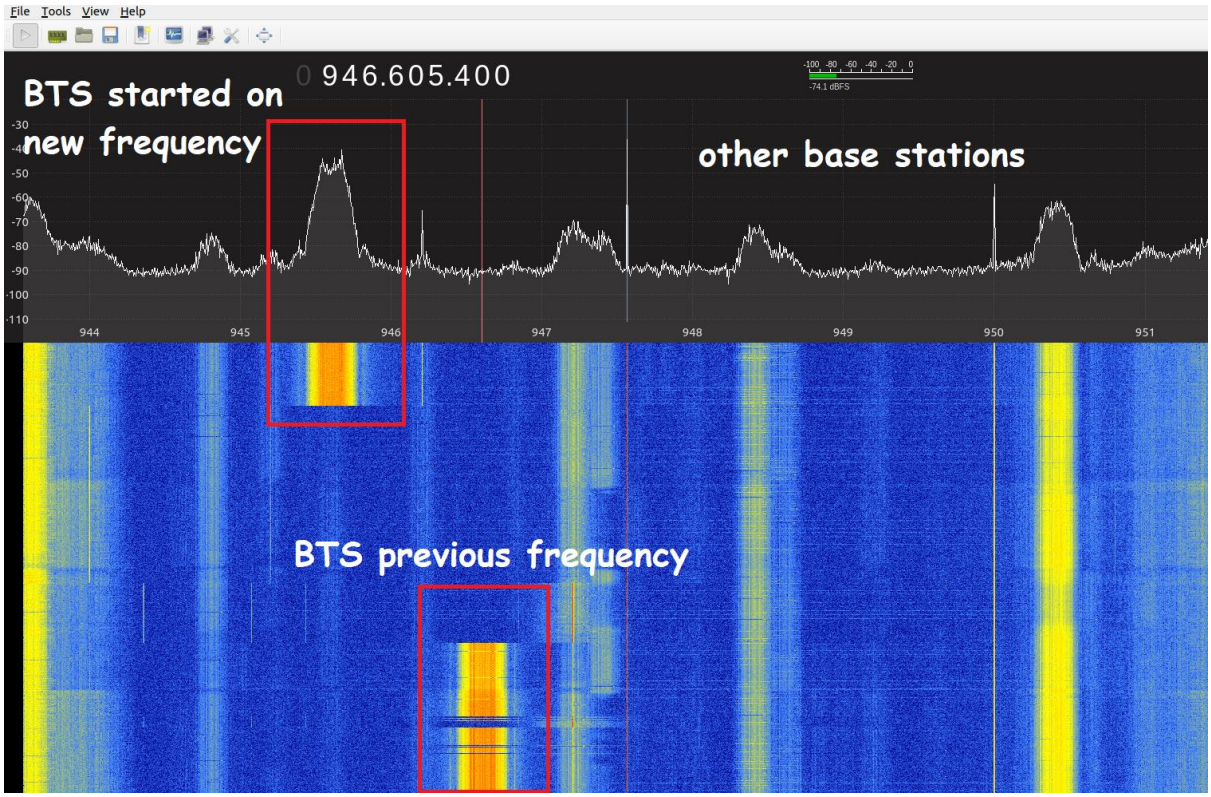

Figure 10. YateBTS changing from one frequency to another.

On Figure 10 it is visible how YateBTS ARFCN parameter affects the frequency that the BTS operates on. While monitoring SDR was running YateBTS was started and then stopped. After stopping the BTS, ARFCN parameter was changed through NIPC Web GUI and then YateBTS was started back again.

2.4 Ethics

Since experiments are conducted on the shared radio frequency spectrum it is important to ensure that only devices participating in this thesis are affected. This section will go over methods used to prevent interference with non-participating devices.

2.4.1 IMSI filter

The base station will start broadcasting its information to nearby devices after startup [7]. As a result, devices which are not participating in this thesis may be affected (e.g. connecting to test BTS). In order to ensure that only test devices are able to connect to the rogue base station, IMSI filter is used. YateBTS software NIPC GUI provides support to configure subscribers which are allowed to connect to the base station manually. YateBTS uses two filtering methods, by regular expression and manually inserting subscribers [34]. Since in the context of this research, the test device amount is limited to only a few mobile stations, a manual method was chosen.

2.4.1.1 Obtaining IMSIs

In order to manually insert IMSI codes to the YateBTS configuration they need to be known. There are several ways to obtain an IMSI and they will be discussed in the following sections.

2.4.1.1.1 With phone application

If the IMSI of the SIM card at hand is unknown, it is worth trying to acquire the IMSI with certain phone software. However, this method largely depends on the underlying phone operating system (See Figure 11).

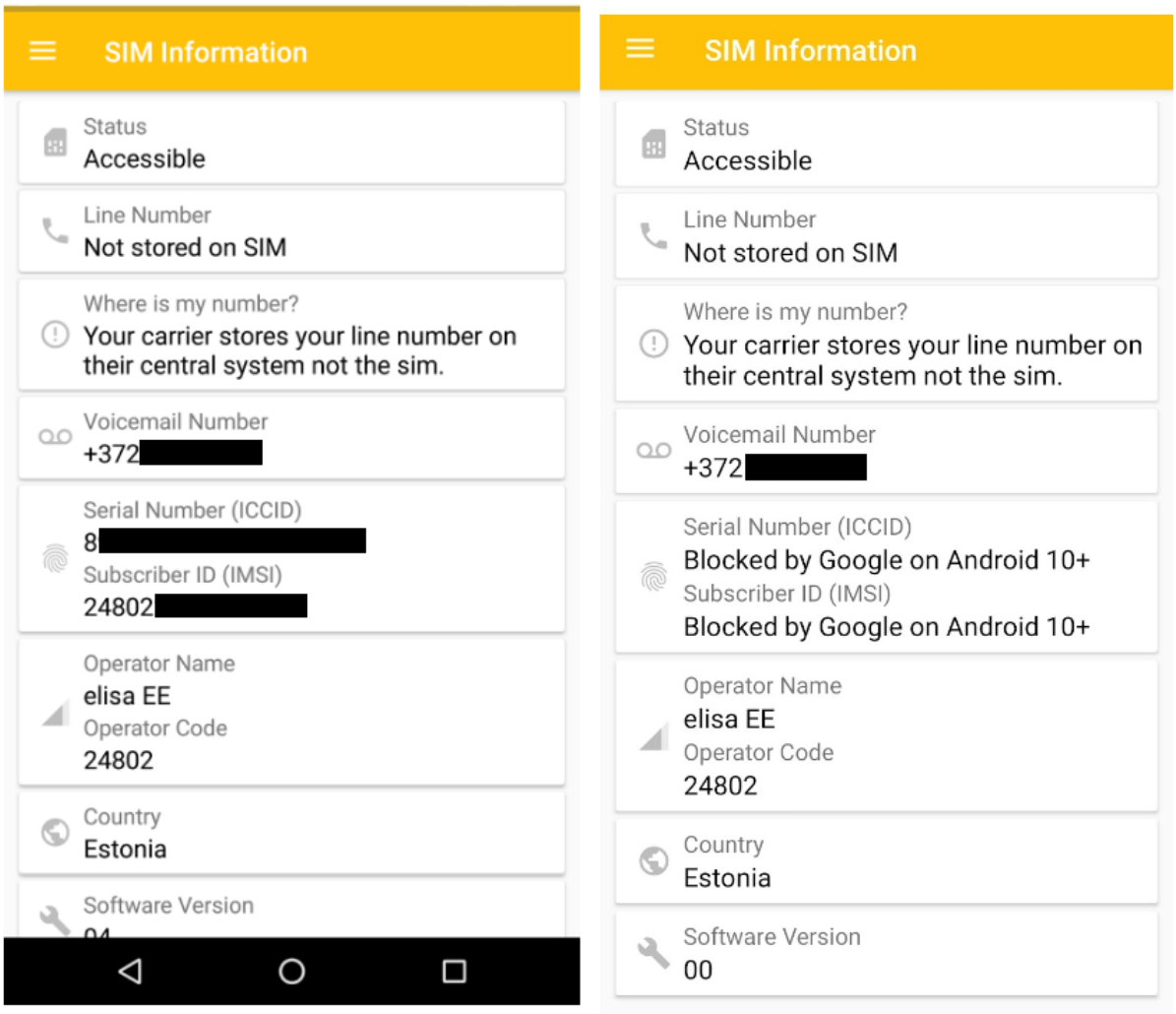

Figure 11. SIM information viewed through SIM Card Info application on different phones

In the example above IMSI and ICCID were blocked by Android version 10+ (Figure 11 right). On Android 10+ this information is considered sensitive and is blocked from the viewer. With the older Android, on the other hand, this information is clearly visible (Figure 11 left). Therefore if an old android phone is available IMSI information can be viewed with SIM Card Info application [46].

2.4.1.1.2 With pySim

It is also possible to obtain IMSI from a SIM card with python scripts. In particular, the python tool program pySim supports this very purpose [47]. This section serves as a guide on how to read the IMSI from the SIM card as it was the method used in this research.

First of all, several hardware components serve as a prerequisite for this method:

- SIM card
- pySim compatible smart card reader

It is also important to have the SIM card's protected PIN1 or PUK1 at hand, otherwise it will be impossible to obtain IMSI information from the SIM due to authorization issues. Furthermore, the pySim python tool needs to be installed. See Appendix 1 for pySim installation.

After installation, the pySim interactive shell (*pySim-shell.py)* needs to be started with the *-p0* flag so that script scans for smart cards (See Listing 6). If the smart card reader uses a different slot, then a different flag might be needed.

```
georg@dragon:~/code/pysim$ python3 pySim-shell.py -p0
Using PC/SC reader interface
Waiting for card...
Autodetection failed
Warning: Could not detect card type - assuming a generic card type...
Info: Card is of type: UICC-SIM
AIDs on card:
 USIM: a0000000871************* (EF.DIR)
 ADF.ISD: a000000003000000
Welcome to pySim-shell!
pySIM-shell (MF)>
```

Listing 6. Starting the pySim shell.

Now it is possible to authenticate to a SIM card and print out the IMSI. Listing 7 shows the steps required to obtain the IMSI:

1. The *verify_chv* command is used to authenticate with PIN1 of *1234*
2. The *dir* command is used to list directories present
3. The *select* command is used to navigate to the directory where the IMSI is located
4. Lastly, *read_binary_decoded* command is used to read contents of the file

```
pySIM-shell (MF)> verify_chv 1234
CHV verification successful
pySIM-shell (MF)> dir
MF
3f00
 .           ADF.USIM    DF.GSM      EF.ARR      EF.ICCID    EF.UMPC
 ADF.ISD     DF.EIRENE   DF.TELECOM  EF.DIR      EF.PL       MF
12 files
pySIM-shell (MF)> select ADF.USIM
...
pySIM-shell (MF/ADF.USIM)> dir
MF/ADF.USIM
...
 EF.CNL             EF.IMSI            EF.PNNI
 EF.DCK             EF.IPS             EF.PSLOCI
 EF.ECC             EF.Keys            EF.PUCT
105 files
pySIM-shell (MF/ADF.USIM)> select EF.IMSI
...
pySIM-shell (MF/ADF.USIM/EF.IMSI)> read_binary_decoded
{
    "imsi": "24802**********"
}
```

Listing 7. Authenticating to a SIM card and obtaining the IMSI.

In Listing 7, the IMSI was intentionally hidden due to it being sensitive information. PySIM also features *pySim-read.py* which can be used to give a summary of the card. However, it does not support authentication and therefore *pySim-shell.py* should be used.

2.4.1.2 Testing IMSI filter

To ensure that only the test equipment would participate in this research, an IMSI filter was tested. The test IMSI of a non-registered subscriber was picked. After deploying the rogue base station, the phone was started so that it would choose the rogue BTS immediately. Section 3.2.1.2 explains how to configure BTS in order for the phone to connect to it immediately. As a result, the phone found the rogue BTS but failed to register to it. Failed registration attempt can be seen in Listing 8.

```
georg@dragon:/usr/local/share/yate/scripts$ telnet localhost 5038
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
YATE 6.2.1-devel1 r (http://YATE.null.ro) ready on dragon.
nipc list rejected
IMSI            No attempts register
--------------- ---------------
24802********** | 1
```

Listing 8: YateBTS showing rejected registration attempt.

The NIPC command *nipc list rejected* showed one rejected registration attempt after booting up the phone. The IMSI from the command-line was identical to IMSI gathered via the pySim tool.

To validate results, the same test was conducted once again but this time with a registered user. As a result, the phone successfully connected to the BTS. This meant that the IMSI filter was working as intended by allowing only registered IMSIs to access the network.

2.4.2 Faraday Cage

The rogue base station transmits signals on the shared radio frequency spectrum which may interfere with already existing legitimate base towers. Additionally, it might be useful to ensure better connectivity of rogue BTS with test equipment (e.g. phone would not switch to a real base station). In order to fulfill the above goals, a faraday cage was built. A faraday cage is a protective enclosure (e.g. a box), which prevents certain electromagnetic radiation types from entering or exiting [48]. Radio wave is an electromagnetic radiation type which makes a faraday cage a suitable solution for the aforementioned problems.

2.4.2.1 Building

Considering the fact that professional faraday cages are expensive (between 1000€ and 2000€) [10] and cheaper laptop bag options cost more than 100€ [49], an attempt to build a homemade faraday cage was made. A NATO ammunition box was chosen as it is spacious enough for both phone and the BTS and costs around 10€. Next, the rubber seal was removed and the seal area was sanded down to bare metal. Box lip was also sanded down to bare metal in order to achieve electron flow from the lid of the box to the main case. That is important as NATO ammunition boxes are painted which act as an insulator and affect electron flow. Then the seal area was stuffed with aluminum foil and cardboard insulation was added on the insides of the cages in order to reduce equipment coming in contact with the box (See Figure 12).

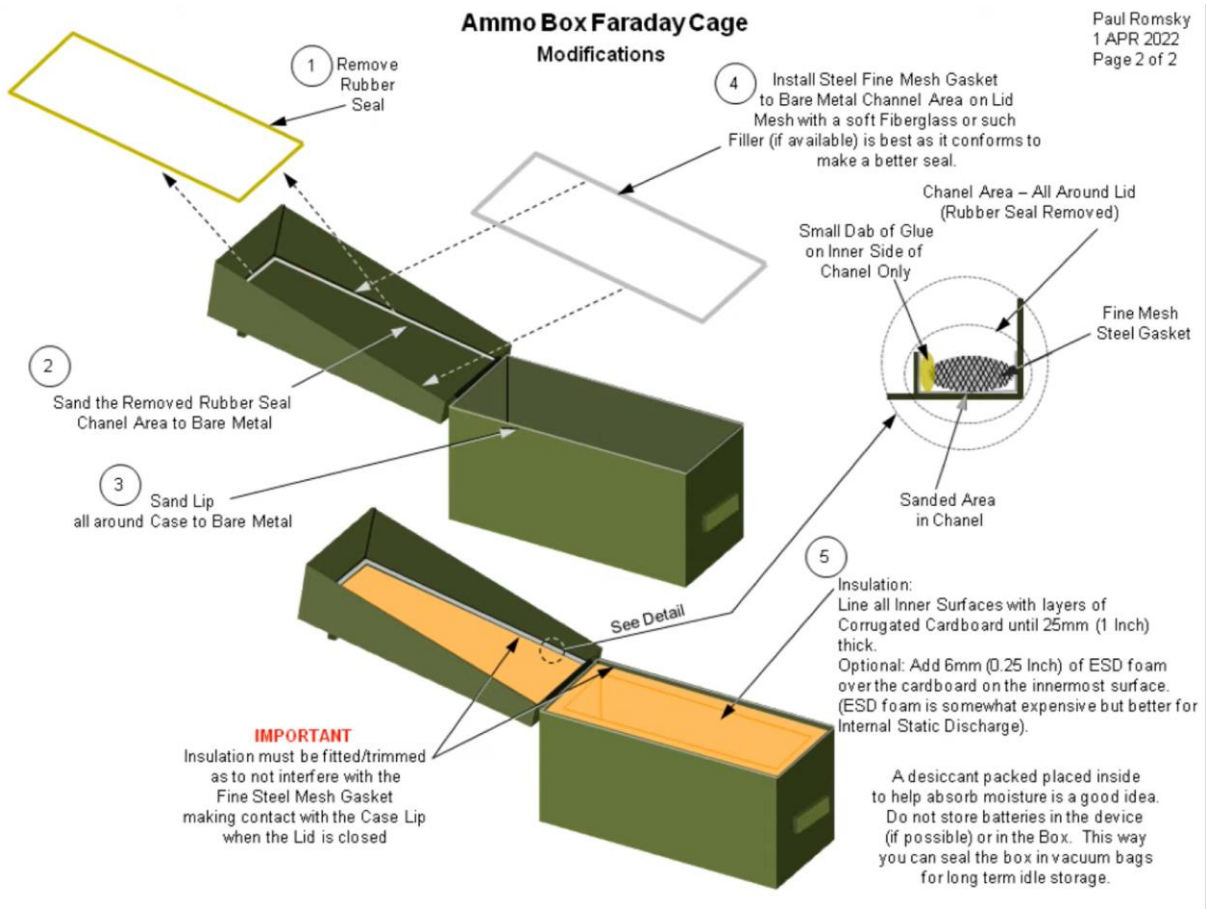

Figure 12. Ammo box faraday cage step-by-step tutorial [50].

It was not possible to place the BladeRF alongside the phone in the cage because the SDR would need to be constantly connected to the laptop. In order to address that issue 2 holes were drilled into the faraday cage to facilitate cable management. Thomas Veens, in his research, has brought up the rule of thumb for "significantly small" holes that are allowed inside the faraday cage [10]. He concluded that holes in the cage can be a maximum of 1/10th of the wavelength. Within the context of this research this resulted in about 1 cm wide holes [10].

Two holes were drilled, one for a USB Type-C phone and the other for the BladeRF SDR. The phone cable hole allowed the phone to be connected to the laptop. With Scrcpy software it is possible to screencast phone screens to the laptop which makes it more convenient to see the current status of the phone screen [51]. BladeRF SDR USB 3.0 SS cable was significantly larger than the USB Type-C cable because of the port housing. To mitigate the problem and stay in the rough 1cm requirements a hole for the port was drilled rather than the housing. This solution didn't allow for the cable to fully go inside the cage, however it was still possible to connect the BladeRF provided it was close to the cage wall.

2.4.2.2 Testing MS and BTS signal reception

Now with passages for the equipment cables made, it was possible to test the whole setup (See Figure 13). First of all, it was tested by placing a 4G phone inside of it with an active screen recording. Footage confirmed that the 4G phone lost signal completely. However, occasionally the phone would connect to the 2G and 3G towers. Stuffing the holes with aluminum foil

increased stability of blocking incoming signals. However, cables still needed to be present in order for the rogue base station to operate. After connecting all the cables to the laptop and closing the faraday cage a major difficulty emerged. Cables were acting as an antennae and routing radio frequency signals inside the faraday cage which made the phone to have a stable 4G connection with a legitimate base tower. Thomas Veens faced the similar issue in his research [10].

To mitigate this issue he used ferrite cores which helped reduce external radio frequencies being channeled into the faraday cage. Unfortunately in the context of this research off-the-shelf ferrite core availability was highly limited. Therefore cheap (2.50€ per piece) and only available off-the-shelf ferrite beads were used. A total of four such beads were used around the cables in various combinations. Unfortunately, they did not yield any result, the mobile phone was still connected to a legitimate base tower without any issues. The same result appeared with BTS signal reception. The BTS was tested with Gqrx software by listening on nearby radio stations. Radio station signal strength appeared to be the same on the visualized graph inside or outside the faraday cage. For additional insulation more aluminum foil was placed all around the insides of the faraday cage in order to reduce external signals present in the cage.

Figure 13. The faraday cage with equipment inside.

Aluminum foil was positioned behind the cardboard on all sides of the faraday cage so that equipment would not come into contact with the foil. Sadly, this did not affect the connection either.

2.4.2.3 Testing BTS signal transmission

Even though the faraday cage did not succeed in blocking external signals with cable attached it was still unknown how the cage affected outgoing signals. In order to test this, the HackRF

SDR was set up to monitor the rogue base station transmission signal strength. Unfortunately it was not possible to operate both the monitoring SDR and base station on the virtual machine simultaneously. The monitoring SDR would not pick up the rogue base station transmissions. To mitigate this, the HackRF was connected to an additional laptop running Linux with Gqrx installed. With a new host device for the monitoring SDR it was possible to see outgoing signals from the BladeRF in Gqrx. For testing purposes, BladeRF maximum attenuation parameter was set to 45 and minimum attenuation parameter to 40. After starting up the BTS inside the faraday cage, HackRF picked up signals in Gqrx. Next, the BTS was placed outside the faraday cage with the same distance from the monitoring SDR. Now it was easy to compare the signal strength of the BTS inside and outside the faraday cage. As shown in Figure 14, signals emitted by the BTS appear to be of the same strength outside and inside the faraday cage.

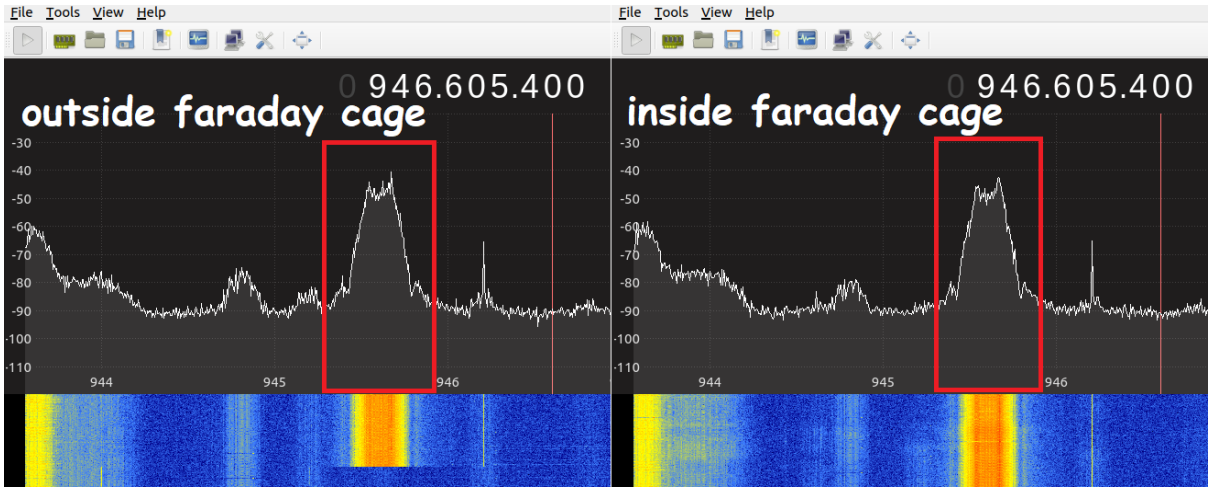

Figure 14. BTS signal strength inside and outside the faraday cage.

Unfortunately, the faraday cage did not affect incoming signals nor outgoing signals with cables attached to the equipment inside.

2.4.2.4 Conclusion on the effectiveness of the faraday cage

In conclusion, construction of a faraday cage that is capable of blocking all incoming mobile network signals is possible. However, if a device inside the faraday cage needs to be physically connected to the outside world then it significantly complicates blocking outgoing and incoming signals. For further research it would be interesting to investigate different ferrite cores on built faraday cage and compare their abilities in blocking cables acting as an antennae. Since the faraday cage is still relatively effective if no cables are attached, a mobile phone was placed inside the faraday cage for upcoming experiments in order to reduce external signals.

3. Experiments and Investigation

3.1 User Equipment

All the experiments were conducted with the author's equipment only. In order to simulate different possible mobile stations three phones were selected. This section will provide their relevant technical specifications as well as reasons for why they were picked.

3.1.1 Modern phone

Considering the fact that 4G is the most popular mobile network in the world as of 2023 it will be necessary to represent that user group in this research. For this reason the Xiaomi Mi 9 Lite was selected. It supports all mobile network generations up to 4G. Furthermore, it is possible to manually select preferred mobile network protocol. This can aid in stimulating environments where faster mobile networks such as HSPA and LTE are not available. For the SIM card Elisa prepaid card was selected.

3.1.2 Old phone

In the context of this research a phone is considered old if it supports only 2G networks. In particular, Motorola C115 and Estar X18 were selected. The Motorola features dual-band GSM. It is important to note that it does not support 2.5G (GPRS) which is the mobile network protocol implemented by the YateBTS [34]. However, the YateBTS should still provide services to phones of older second-generation mobile networks than the GPRS with backward compatibility. Such phones will not be able to have internet access. The Estar X18, on the other hand, does support the GPRS technology. The Estar phone is still manufactured to this day and can be bought in various mobile phone stores. For the SIM cards, Elisa prepaid cards were selected.

3.2 Experiments

This section will cover all the experiments conducted with the aforementioned rogue base station and user equipment. Various functionalities of the YateBTS will be tested to see how viable these solutions are in the real world.

3.2.1 Spoofing Legitimate BTS

In the spoofing attack, the target mobile phone is led to believe that the rogue base station is in fact a legitimate base station [33]. By configuring certain parameters of the base station it might be possible to mimic the original base station and force the phone to connect to rogue BTS instead of the original one.

There are various mobile network operators that have multiple base stations working on a large variety of radio channels. Elisa base stations were selected because user equipment used in this research is powered by Elisa SIM cards. It is not possible to spoof multiple network operators at once with this setup. Spoofing multiple base stations would require additional SDRs, host devices and so on.

3.2.1.1 Reconnaissance

In order to mimic a specific base station it is necessary to gather data regarding that BTS. Otherwise it is unknown to what values rogue BTS parameters should be configured. For this purpose Network Cell Info Lite application was installed to Xiaomi Mi 9 Lite [52]. First of all, the modern phone preferred network type was set to GSM only. Such preference setting is necessary because rogue BTS will operate in GSM mode. Secondly, the phone was automatically connected to the legitimate Elisa base station. Thirdly, information regarding that base station was viewed in the RAW tab of the Network Cell Info Lite application (See Figure 15).

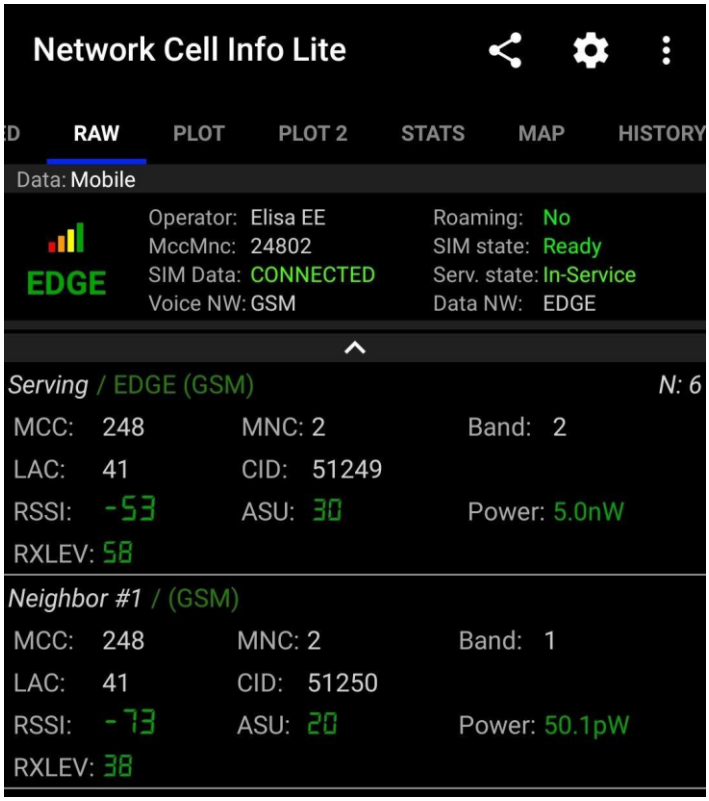

Figure 15. Elisa base station information viewed in Network Cell Info Lite

From RAW tab information it was possible to gather operator name (Elisa), mobile network protocol (2.75G EDGE), mobile country code (248), mobile network code (2), location area code (41) as well as CID (51249). Furthermore, this particular base station was also advertising a list of neighboring base stations to connect to. The most important parameters for spoofing are MCC and MNC because combined they identify a network operator as well as the country.

3.2.1.2 Spoofing

After openly available base station parameters were gathered it was possible to tune rogue BTS accordingly. In particular, the MNC, the MCC, and the LAC were configured to match the ones of Elisa's. However, to test the importance of the MNC parameter, at first, it was configured to 022 instead of 02. Therefore all the phones were tested both with the correct and incorrect MNC. In this context, incorrect means that the MNC did not match Elisa's MNC. After the

BTS was configured, an empty ARFCN channel was picked in order not to interfere with existing base stations. Picking an empty ARFCN channel involved scanning the radio frequencies with the monitoring SDR. The MS and the BladeRF were placed inside the faraday cage and the YateBTS was started.

3.2.1.2.1 Modern phone results

This section provides spoofed network tests with the real Elisa 02 MNC and a random MNC configured.

Random MNC

For this test, the MNC was set to 022. With the LTE network mode preferred (phone default setting), the modern phone did not switch to the rogue base station. Even after the power cycle (turning it off and on), Xiaomi preferred faster 4G connection. Changing location area code to trick the phone into thinking that it has moved to a new location did not work either. Legitimate base stations advertise neighboring BTS lists to help the MS to switch to another legitimate base station in case the one the phone is connected to loses connection. This neighbor list can be viewed at the bottom of Figure 15. This makes it even more difficult to make modern phones connect to the rogue BTS. Modern phones don't do full base station scans in order to increase power efficiency, they just switch to the last advertised neighbor [33].

To simulate areas with only the GSM networks available, the modern phone was set into the GSM only mode. Even then it required many attempts for the phone to pick up the rogue BTS. Changing the BladeRF transmission power with attenuation parameters did not have any effect either. After switching to airplane mode and back as well as setting network selection to manual, the modern phone finally showed the rogue BTS as a possible base station to connect to. Figure 16 shows two different states, when the rogue BTS is not running (on the left) and when the rogue BTS is booted up (on the right). When the modern phone showed the rogue BTS as a possible cellular network to connect to, for some reason, it didn't show a real Elisa base station as an option. Additionally, the rogue BTS network name appeared substantially different from other operator's base stations when viewed through phone. It turned out that YateBTS set MCC and MNC as the network name. Interestingly, the network name parameter was not a configurable option in YateBTS.

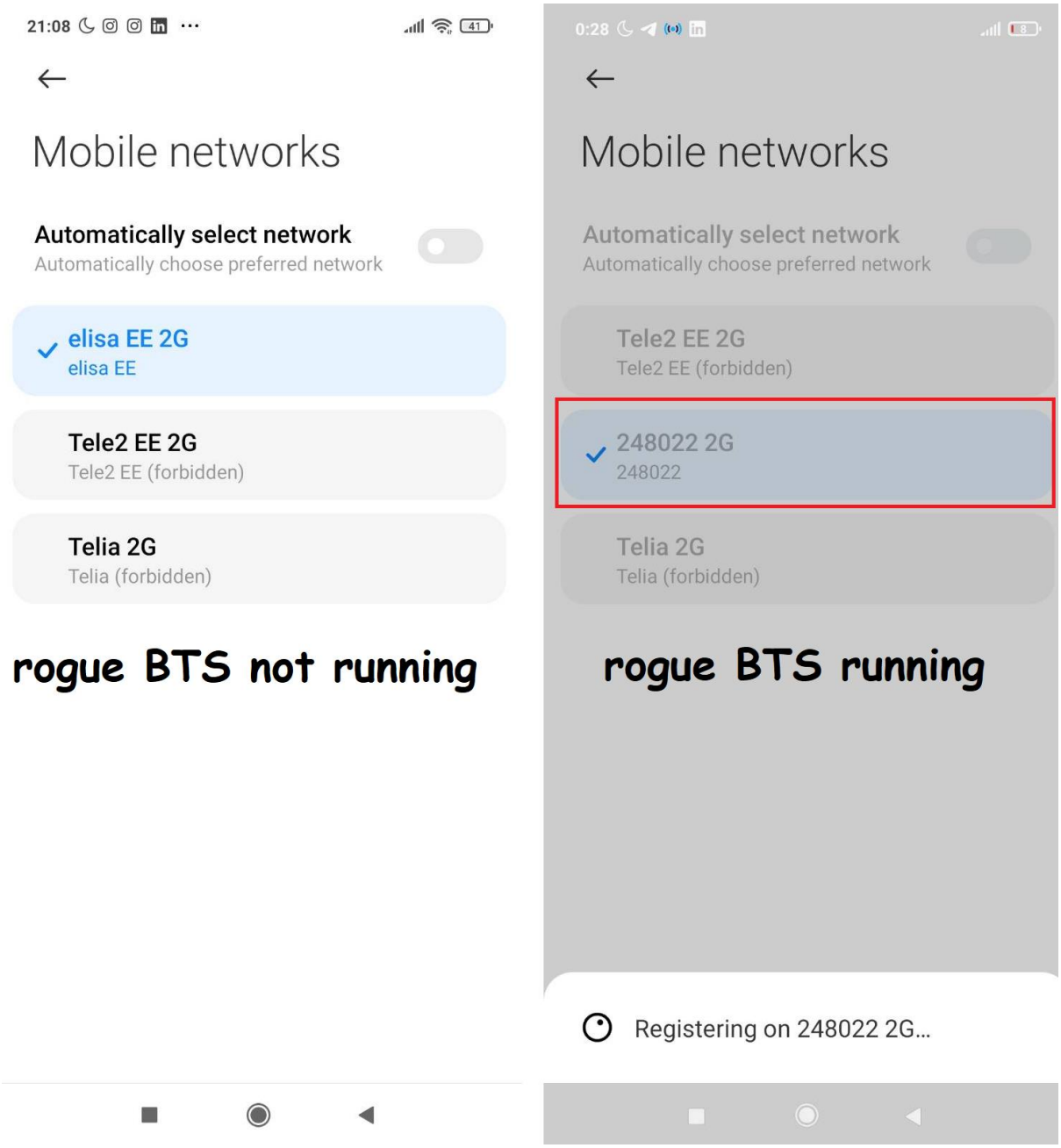

Figure 16. Modern phone registering to the spoofed BTS.

However, after the modern phone picked up the rogue BTS, it had many difficulties with registering to the BTS. Most of the time, registration either timed out or failed completely. It was, however, possible for the modern phone to connect to the rogue BTS network. A welcome SMS, shown in Figure 17, was sent from the YateBTS to confirm successful connection with the BTS.

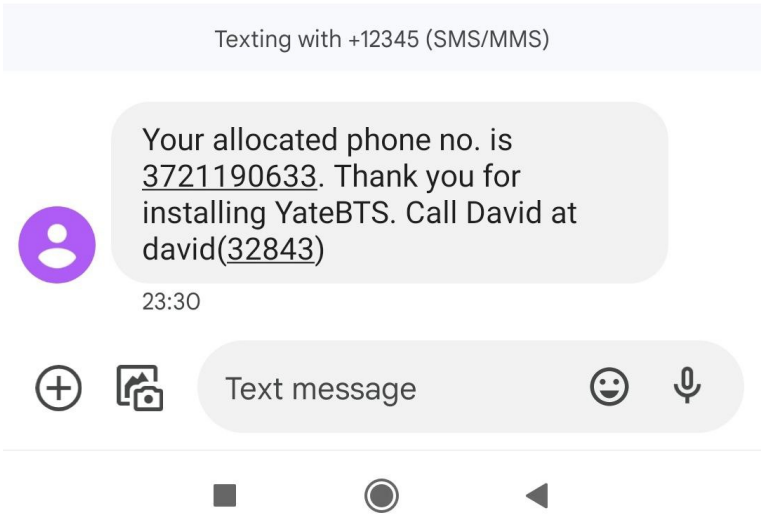

Figure 17. Welcome SMS from YateBTS.

When first registered to the network, the YateBTS provides a welcome SMS message which includes useful information for testing. Firstly, the SMS message includes an allocated phone number which can be used for calling on the network. Additionally, the YateBTS features David's phone number for testing purposes. When calling David, a prerecorded welcome audio starts playing, congratulating the user for installing the YateBTS. A mobile phone is successfully connected to the rogue BTS if David can be called.

Correct MNC

This time, the MNC was set to 02. When the modern phone was configured to LTE mode preferred, results were the same. The modern phone always preferred faster mobile networks instead of the spoofed second-generation one. Additionally, results stayed the same if the phone was already connected to the real base station or power-cycled. Based on these results, it is safe to assume that modern phones usually prefer faster generation networks if they are available.

Interesting results were obtained with GSM only mode. When booted up the modern phone automatically connected to the rogue base station instead of the real Elisa one. This was particularly interesting because Elisa supported 2.75G technology whereas rogue BTS only 2.5G. On the other hand, if the phone was already connected to the Elisa base station it did not switch to the rogue BTS even though the transmission signal was better with the latter. This time, the rogue BTS name was also indistinguishable from the real Elisa base station.

3.2.1.2.2 Old phone results

Experiment setup followed the same procedure as with the modern phone. Same ARFCN channel and rest of the parameters were used. Firstly, the test was conducted with the incorrect MNC parameter and secondly with the correct Elisa's MNC.

Random MNC

In the first test, the Estar X18 was already connected to a legitimate Elisa base station. Then, the YateBTS was started. The phone remained connected to the Elisa base station and did not switch to the rogue BTS. Changing location area code also had no effect on the phone's connection. This is most likely due to the phone not performing full surrounding network scan and therefore it did not spot the rogue BTS.

In the second test, the phone was turned off and YateBTS was started. After booting up the phone, it had successfully chosen the rogue BTS. This is highly likely due to the phone performing a full surrounding base station scan. Welcome message was received and it showed almost the same network name as on the modern phone (See Figure 18). The only difference was suffix *_elisa EE*. It is unknown why the phone decided to append the suffix to the network name. After checking manually other possible network operators on the Estar X18 it showed a legitimate Elisa network as well, whereas the modern phone did not.

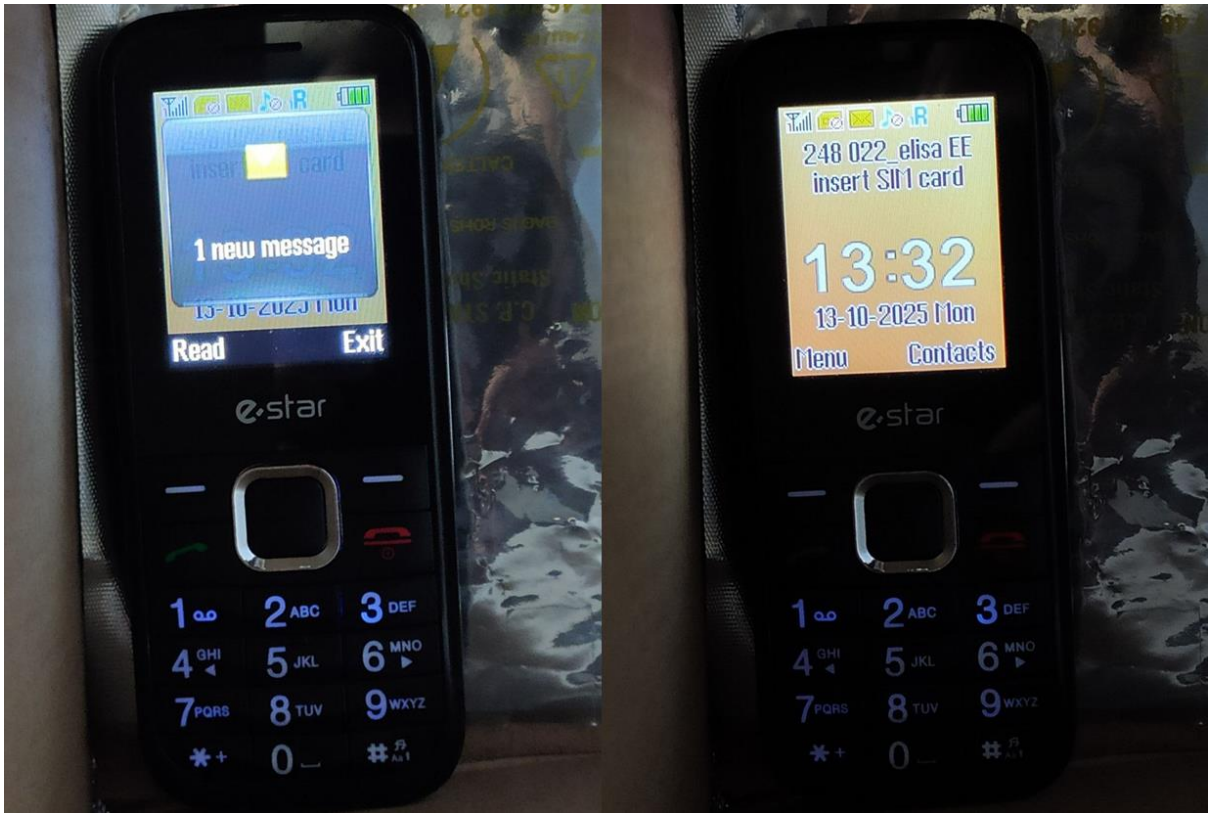

Figure 18. Estar X18 connected to the rogue base station.

Correct MNC

Changing the BTS MNC parameter to 02 and performing the test once again yielded different results. This time, the Estar X18 connected to the fake base station once again but the network name was correct and mimicked that of a real one, as shown in Figure 19.

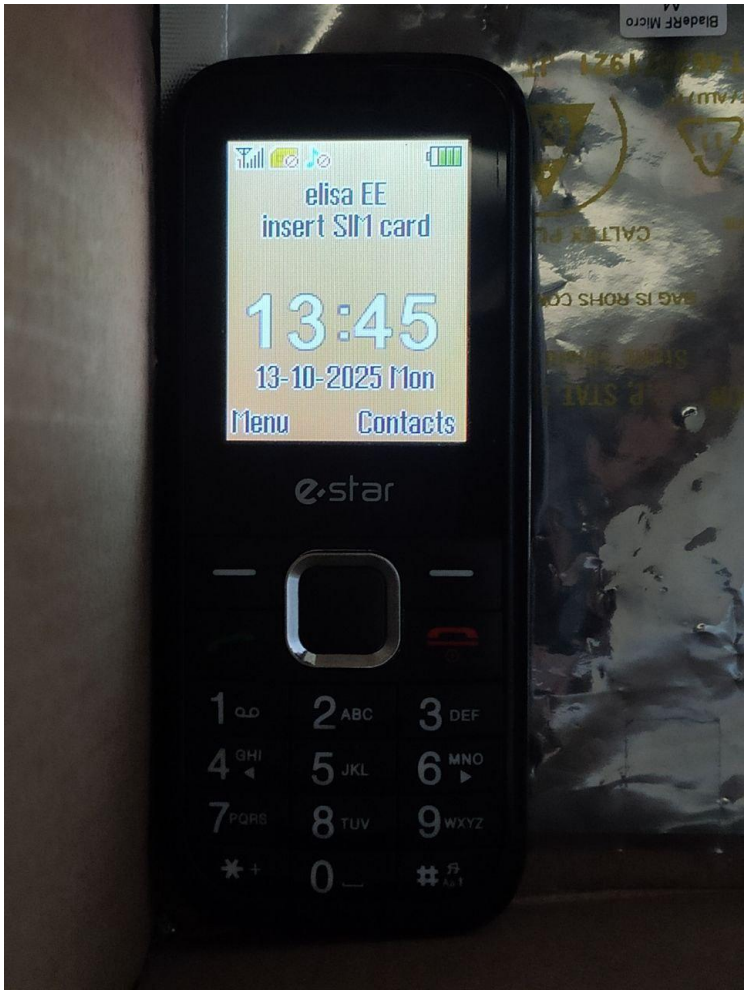

Figure 19. Estar X18 connected to the rogue base station with 02 MNC.

The Estar X18 connectivity to rogue BTS was visually indistinguishable from if it was connected to the real network. Therefore, to ensure that it was indeed a rogue BTS, the connection needed to be verified. Easiest method was to call a number provided by the YateBTS for testing purposes. Testing number is provided in the welcoming message when the phone initially connects to the network. After calling the test number, the YateBTS confirmation message was heard. This meant that the spoofing experiment was successful.

Same tests were conducted with an older Motorola C115 mobile phone. Results were mostly the same. The only difference was that it had more problems with connecting to the rogue BTS compared to the Estar even after startup. Network name was indistinguishable from the real Elisa base station as well.

In conclusion, other faster mobile networks posed a major difficulty for the phones to connect to the rogue BTS. The more the other networks were disrupted (e.g. placing phones inside the faraday cage), the easier the phones connected to the rogue BTS. This means that spoofing a legitimate base station in areas where there is no cellular connection such as tunnels or metros would be effective. Testing phone connectivity with the rogue BTS by using random MNC instead of a real Elisa's MNC showed the importance of setting the MNC correctly. Otherwise phones tend to pick real base stations instead. When connected to the rogue BTS, phones are susceptible to more sophisticated attacks such as SMS-phishing.

3.2.2 SMS-phishing

SMS-phishing, also called smishing, is a type of text message fraud which focuses on luring victims into revealing account information or installing malware. Cybercriminals use this technique by disguising themselves as a trustworthy organization (e.g. bank) or a reputable person in a text message [53]. The YateBTS is capable of sending SMS and even features a welcoming SMS message, therefore it might be possible to use YateBTS with an intent of smishing. This section explores the idea of smishing with YateBTS.

The YateBTS software has a built-in feature to send welcome SMS messages to whatever device that joins the network. It is similar to those messages that mobile phones receive when entering a foreign country. The YateBTS welcome messages tell the mobile station their allocated phone number as well as test number for calling (See Figure 20). Unfortunately, this message is not a configurable parameter. However, since the YateBTS is open-source it is possible to modify the source code in order to send out custom welcome SMS messages (See Figure 20).

```
function readConfiguration(return_subscribers)
{
    var configuration = getConfigurationObject("subscribers");
    country_code = configuration["general"]["country_code"];
    if (country_code.length==0)
        Engine.alarm(alarm_conf,"Please configure country code. See subscribers.conf or use the LMI web interface");
    gw_sos = configuration["general"]["gw_sos"];
    sms_registration = configuration["general"]["sms_text"];
    if (!sms_registration)
            sms_registration = "Your allocated phone no. is ${nr}. Thank you for installing YateBTS. Call David at david(${david_number})";

    var reg = configuration["general"]["regexp"];

    if (configuration["general"]["nnsf_bits"]!="")
        nnsf_bits = configuration["general"]["nnsf_bits"];
-- VISUAL LINE --                                                                    1         385,
```

Figure 20. Welcome message initialization in source code.

By navigating to */usr/local/share/yate/scripts/* it is possible to modify *nipc.js* in order to change the initial registration message. It is also possible to modify *subscribers.conf* to add *sms_text* parameter there instead of hardcoding it in the source code. Messages by default are sent from the preconfigured base station number found in the *nipc.js* file. This is also an important parameter to boost trustworthiness of the smishing SMS. Default number is +12345, which makes it an unusual and strange number to receive SMS from. YateBTS makes it possible for malicious actors to easily change this parameter (See Figure 21).

```
eliza_number = "35492";
david_number = "32843";
nipc_smsc_number = "12345";
sms_attempts = 3;
registered_subscribers = {};
pendingSMSs = [];
seenIMSIs = {};  // imsi:count_rejected
ussd_sessions = {};
```

Figure 21. Base station number initialization in source code.

After configuration of the SMS, it was tested on an old and modern phone by starting YateBTS and rebooting both mobile stations. Figure 22 shows a default welcome SMS message sent by YateBTS on the left (marked green) and the same message but customized on the right (marked red). It was necessary to remove aggregated IMSIs from the *tmsidata.conf* file to trick rogue BTS into thinking these devices are connecting for the first time. This was needed to test the welcome SMS messages.

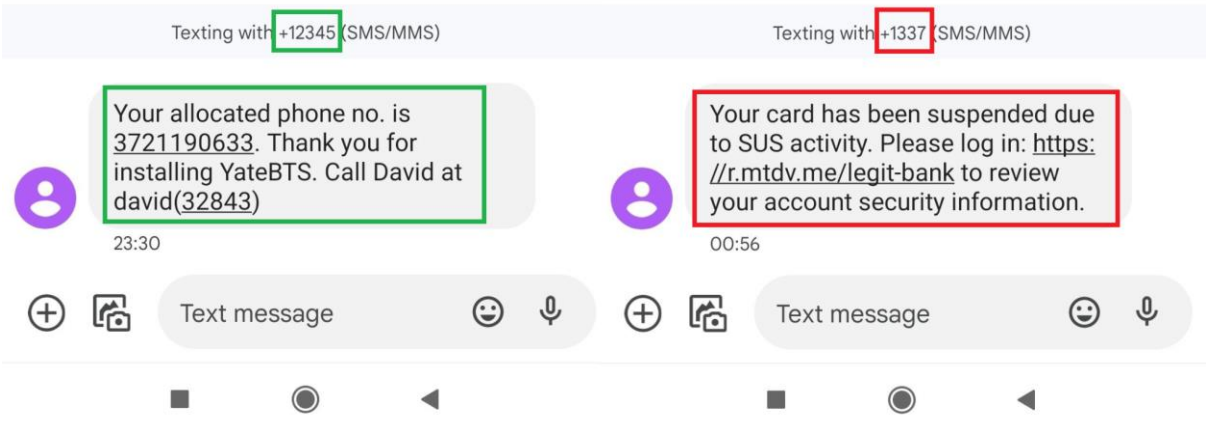

Figure 22. Comparison of welcome messages before and after configuration.

Unfortunately, configuring a unique caller ID was not possible. Substituting NIPC number with a string on source-code level resulted in a blank number. In conclusion, this experiment has proven that smishing is indeed a plausible phishing attack which is possible with YateBTS. It is however, important to keep in mind that the victim's phone must register to the rogue BTS network for the message to be received by the victim.

3.2.3 SMS interception

SMS interception is a form of Man-in-The-Middle (MiTM) attack, where an attacker eavesdrops on the messages exchanged between two parties [54]. In the context of this research, two parties are the mobile stations. This section explores possibilities of SMS traffic interception with YateBTS.

3.2.3.1 Prerequisites

First of all, several YateBTS parameters need to be configured for SMS interception to take place. By default, YateBTS does not encrypt traffic between the phone and the rogue BTS. This allows for exchanged traffic to be viewed in plaintext, including SMS. This is a configurable parameter in YateBTS so it needs to be set to default no encryption setting. Next important parameter is the GSM tapping (See Figure 23), which is turned off by default.

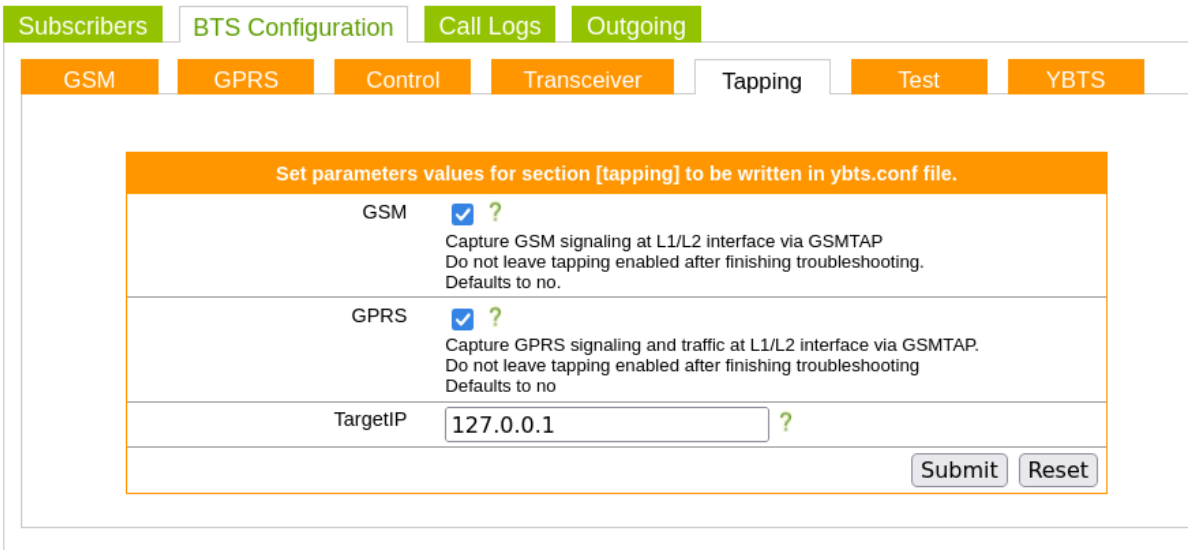

Figure 23. YateBTS GSM tapping parameters viewed in Web GUI.

This setting needs to be turned on for easy traffic interception of the GSM network. Additionally, Wireshark or Tcpdump needs to be installed for analyzing GSM traffic. Fortunately, DragonOS comes preinstalled with both aforementioned tools.

3.2.3.2 Interception testing

After YateBTS configuration, everything is ready for the SMS interception. Firstly, to capture packets sent over the GSM network Tcpdump tool was used to listen on the loopback interface port 4729 (See Listing 8).

```
georg@dragon:~/pcaps$ sudo tcpdump -i lo -w yatebts_sms.pcap port 4729
tcpdump: listening on lo, link-type EN10MB (Ethernet), snapshot length
262144 bytes
```

Listing 8: Listening on loopback interface port 4729.

Secondly, YateBTS rogue base station was started. Thirdly, two registered mobile stations were connected to the BTS. Then, one phone sent an SMS containing text to another registered phone. After closing the Tcpdump tool it was possible to analyze captured packets with Wireshark by opening the generated *.pcap* file (See Figure 24).

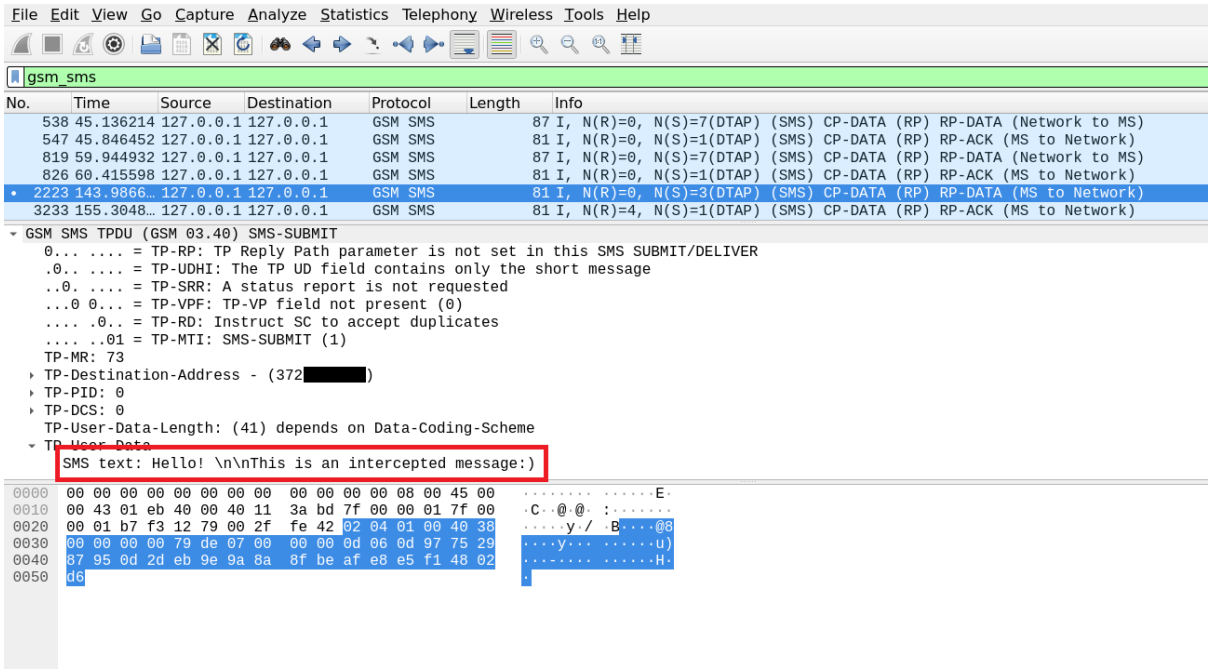

Figure 24. Captured SMS packet viewed in Wireshark.

Packets were filtered by a *gsm_sms* filter which helped to find packets containing SMS messages more easily. The message sent from one phone to another was clearly visible in plaintext in the Wireshark. Another test was conducted where the recipient phone number was not registered on the network. Messages sent over the GSM network were still easily read in plaintext. In conclusion, SMS interception in YateBTS is possible when SMS is being sent by a MS that is registered on the network.

3.2.4 IMSI-catching

IMSI-catcher's main goal is to collect the IMSIs and therefore compromise subscribers' privacy and non-traceability [33]. The IMSI is tied to the subscriber of the SIM card and therefore can be used to identify an individual. IMSI-catching attacks usually lay the foundation for further more sophisticated attacks such as tracking specific users in geographical areas, mass-surveillance, eavesdropping and many more [55]. To limit the scope of this research, only IMSI-catching capabilities of the YateBTS will be tested.

In the GSM networks, there is no mutual authentication. However, mobile stations still need to be verified by the BTS in order for the network operator to know who the subscriber is. As a result, the IMSI is sent to the BTS when MS first registers. Furthermore, BTS stores IMSI information in the VLR component. Considering that YateBTS is a full software network implementation means that whoever runs the BTS also has access to the VLR and the IMSI as a result. In order to view subscriber information, including IMSIs, *tmsidata.conf* file can be viewed as seen in Listing 9. IMSIs, as well as other sensitive information is intentionally hidden in Listing 9.

```
[tmsi]
last=007b0004


[ues]
248022*********=007b0002,35453**********,372*******,1683******,ybts/TMSI
007b0002,1682788870
248022*********=007b0002,35453**********,372*******,1683******,ybts/TMSI
007b0002,1682788870
248022*********=007b0003,35453**********,372*******,1683******,ybts/TMSI
007b0003,1682882230
248022*********=007b0004,35825**********,372*******,1683******,ybts/TMSI
007b0004,1682865809
```

Listing 9. YateBTS TMSI configuration file contents.

First column represents subscriber IMSIs, whereas the second column represents IMEIs. IMEIs uniquely identify a mobile phone, which also can be used to track individuals. For example Russia used IMEI codes for this exact purpose. Finns were interrogated upon entering Russia to reveal their IMEI codes [56]. Later, this information can be used to track individuals by checking which base stations the phone connected to [56]. Third column shows allocated phone numbers in the YateBTS network. Fourth column is most likely used for the current epoch timestamp. Fifth column is TMSI assigned by the rogue BTS. Last column is most likely epoch time when the phone first registered.

3.3 Rogue Base Station Detection and Prevention

This research paper has shown multiple attacks that can be carried out by BladeRF and YateBTS. This section, on the other hand, will cover possible rogue BTS detection methods to increase safety of cellular network usage.

First of all, there are various BTS parameters that can differentiate from a real base station. These parameters can be easily viewed through a phone application [52]. In case of YateBTS, default parameters will lead to the network name being shown as a sequence of numbers, which can be an alarming indicator. Another important flag to look at is an empty neighboring list. Rogue BTS will most likely want to maximize the victim's connection to the BTS by broadcasting an empty list of nearby stations. A BTS in a city will most likely have multiple neighboring base stations (See Figure 15). It might also be important to be on the lookout for cellular connection in places where there shouldn't be one. For example in underground metros, tunnels and other places. Such places with no rival base stations make it easy for rogue BTS to thrive since it's the only BTS in the area.

Additionally, if possible, it is better to not use 2G technology on a phone all together because newer generations are less susceptible to various attacks. In 2022, Google introduced a 2G kill-switch on Android 12 which lets users turn off use of 2G networks by the phone [57]. Furthermore, a global phase-out of 2G networks has begun globally. This means that countries are starting to shut down 2G networks in favor of more fast and secure technologies. As of 2022, at least ten countries have switched off all of their 2G services [58]. Therefore, if a phone

switches to 2G in a country where there are no legally operated 2G base stations indicates that there is a risk of a rogue BTS.

4. Conclusion

For this thesis, various cellular network generations were discussed and analyzed in order to build a rogue base station. From them, 2G was chosen as the most suitable mobile network protocol to be used for the rogue cellular-site simulator. Furthermore, an overview of the 2G network architecture and its most important components was given. After careful consideration, base station software implementation YateBTS was chosen. As for hardware, BladeRF transceiver was selected due to its excellent compatibility with YateBTS. Thesis at hand also provided a thorough explanation of most important parameters for base station configuration available with YateBTS.

In consideration of ethics, the YateBTS software-level safeguards were tested and used to prevent non-participating cellular users from being affected by this research. An attempt to build a cheap faraday cage was also made to further increase safety of the experiments. Unfortunately, faraday cage was only able to fulfill its purpose if no cable was attached to the device. After all the safeguards were put in place and tested, the rogue base station was successfully deployed.

For the first attack, rogue BTS was configured to impersonate a legitimate Elisa cellular tower. Results showed that both modern and old phones preferred base stations that they were already connected to. However, if an old phone was power-cycled, it would immediately connect to the rogue base station due to its stronger signal. The modern phone, on the other hand, preferred faster mobile network protocols such as 4G even after it was rebooted. When a modern phone was configured to the GSM only mode, it would still usually connect to the faster Elisa 2.75G station. However, at times it would attach itself to the rogue BTS as well. These findings prove that nowadays with faster networks available it would be rather difficult to get phones to connect to the rogue base station. Nevertheless, rogue base stations can still perform well in areas with no cellular connection as it would be the only base station available.

Following successful mobile station attach procedure to the rogue base station, more sophisticated attacks were carried out. One of them was constructing an SMS phishing message upon connection to the base station. This required some source code modifications to improve legitimacy of the message. As a result, every phone that connected to the network received the aforementioned message. Furthermore, an SMS interception attack was successfully made. SMS messages were easily readable in plaintext. This includes all the messages sent over the rogue BTS network.

Possibility of IMSI-catching with the setup was also tested. Gathering an IMSI is a stepping stone to more sophisticated attacks such as tracking specific users in geographical areas. Rogue BTS obtained an IMSI and IMEI of every device that has tried to connect to the network and stored it in a configuration file where it could be easily viewed.

Current study shows that a major difficulty with this setup is to get phones to connect to the rogue BTS due to rivaling faster networks available. However, once connected, a large plethora of attacks can be deployed against the connected users. For further work, it would be interesting to explore possibilities of attacks with software implementations that support faster and newer generation networks. Additionally, it is possible to expand on the existing setup to see if it can obtain IMSIs of not only the connected users but also the ones nearby.

References

[1] R. Simon, "WHY 2G WANTS TO LIVE FOREVER," Feb. 14, 2022. [Online]. Available: https://www.cambridgewireless.co.uk/news/cw-journal/why-2g-wants-live-forever/ (10.04.2023)

[2] K. van Rijsbergen, "The effectiveness of a homemade IMSI catcher build with YateBTS and a BladeRF," 2016 [Online]. Available: https://rp.os3.nl/2015-2016/p86/report.pdf

[3] "Meet the machines that steal your phone's data." [Online]. Available: https://arstechnica.com/tech-policy/2013/09/meet-the-machines-that-steal-your-phones-data/ (15.03.2023)

[4] *DEF CON 18 - Chris Paget - Practical Cellphone Spying*. [Online]. Available: https://www.youtube.com/watch?v=fQSu9cBaojc&ab_channel=DEFCONConference

[5] "Elisa increasingly investing in 4G and 5G networks – 3G network available to customers until the end of 2023" [Online]. Available: https://elisa.com/corporate/news-room/press-releases/elisa-increasingly-investing-in-4g-and-5g-networks-%E2%80%93-3g-network-available-to-customers-until-the-end-of-2023/44259115909144/ (12.04.2023)

[6] "Telecom - Work in Estonia" [Online]. Available: https://www.workinestonia.com/living-in-estonia/telecom/ (18.04.2023)

[7] A. Dabrowski, N. Pianta, T. Klepp, M. Mulazzani, and E. Weippl, "IMSI-Catch Me If You Can: IMSI-Catcher-Catchers," in *Proceedings of the 30th Annual Computer Security Applications Conference*, New York, NY, USA, 2014, pp. 246–255, doi: 10.1145/2664243.2664272 [Online]. Available: https://doi.org/10.1145/2664243.2664272

[8] X. Chen *et al.*, "Trip-Chain-Based Travel-Mode-Shares-Driven Framework using Cellular Signaling Data and Web-Based Mapping Service Data," *Transp. Res. Rec. J. Transp. Res. Board*, vol. 2673, p. 036119811983400, Mar. 2019, doi: 10.1177/0361198119834006.

[9] S. Kumar and M. Meraj, "Evolution of Mobile Wireless Technology from 0G to 5G" [Online]. Available: https://ijcsit.com/docs/Volume%206/vol6issue03/ijcsit20150603123.pdf

[10] T. Veens, "Automated 2G Traffic Interception and Penetration Testing," 2018 [Online]. Available: https://pure.tue.nl/ws/portalfiles/portal/130177228/Thomas_Veens.pdf

[11] M. Lewis, "GSM/GPRS Traffic Interception for Penetration Testing Engagements." [Online]. Available: https://research.nccgroup.com/2016/05/19/gsm-gprs-traffic-interception-for-penetration-testing-engagements/ (25.03.2023)

[12] "2G / 3G / 4G / 5G / NB-IoT / LTE-M – Which to Choose for Your IoT Project?" [Online]. Available: https://1ot.mobi/resources/blog/gsm-3g-4g-lte-which-one-to-choose-for-your-iot-project (05.03.2023)

[13] “Telia to close its 3G network, competitor Tele2 to wait a few years” [Online]. Available: https://news.err.ee/1608854372/telia-to-close-its-3g-network-competitor-tele2-to-wait-a-few-years (20.03.2023)

[14] “CDMA vs. GSM: What’s the Difference?” [Online]. Available: https://www.pcmag.com/news/cdma-vs-gsm-whats-the-difference (10.04.2023)

[15] R. Lai, “3G GSM encryption cracked in less than two hours.” [Online]. Available: https://www.engadget.com/2010-01-15-3g-gsm-encryption-cracked-in-less-than-two-hours.html (17.04.2023)

[16] “Authentication and key agreement cheat sheets for 2G, 3G, 4G and 5G.” [Online]. Available: https://www.ericsson.com/en/blog/2021/9/authentication-and-key-agreements (17.04.2023)

[17] “Application of Firewalls in the LTE IPSec Solution.” [Online]. Available: https://support.huawei.com/enterprise/en/doc/EDOC1100087919 (17.04.2023)

[18] “Securing the 5G Era.” [Online]. Available: gsma.com/security/securing-the-5g-era/ (18.04.2023)

[19] M. Chlosta, D. Rupprecht, C. Pöpper, and T. Holz, “5G SUCI-Catchers: Still Catching Them All?,” in *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, New York, NY, USA, 2021, pp. 359–364, doi: 10.1145/3448300.3467826 [Online]. Available: https://doi.org/10.1145/3448300.3467826

[20] “CHECK IMEI NUMBER TO GET TO KNOW YOUR PHONE BETTER.” [Online]. Available: https://www.imei.info/ (25.03.2023)

[21] “HOW COPS CAN SECRETLY TRACK YOUR PHONE.” [Online]. Available: https://theintercept.com/2020/07/31/protests-surveillance-stingrays-dirtboxes-phone-tracking/ (18.04.2023)

[22] “ACLU-Obtained Documents Reveal Breadth of Secretive Stingray Use in Florida” [Online]. Available: https://www.aclu.org/news/national-security/aclu-obtained-documents-reveal-breadth-secretive-stingray-use-florida (18.04.2023)

[23] “Department of Justice Policy Guidance: Use of Cell-Site Simulator Technology” [Online]. Available: https://www.justice.gov/opa/file/767321/download

[24] “Secret Service and ICE Did Not Always Adhere to Statute and Policies Governing Use of Cell-Site Simulators” [Online]. Available: https://www.oig.dhs.gov/sites/default/files/assets/2023-03/OIG-23-17-Feb23-Redacted.pdf

[25] “ICE and the Secret Service Conducted Illegal Surveillance of Cell Phones” [Online]. Available: https://www.eff.org/deeplinks/2023/03/report-ice-and-secret-service-conducted-illegal-surveillance-cell-phones (18.04.2023)

[26] “Russian special forces operate in Estonia” [Online]. Available: https://www.aldrimer.no/claims-russian-special-forces-are-operating-inside-estonia/ (18.04.2023)

[27] “Experts say lion’s share of NATO leak is hot air” [Online]. Available: https://news.postimees.ee/3680481/experts-say-lion-s-share-of-nato-leak-is-hot-air (18.04.2023)

[28] “Understanding the difference between SDR and HDR.” [Online]. Available: https://www.barrettcommunications.com.au/news/understanding-the-difference-between-sdr-and-hdr/ (24.04.2023)

[29] “Introduction to SDR (Software Defined Radio).” [Online]. Available: http://play.fallows.ca/wp/radio/software-defined-radio/introduction-to-sdr-software-defined-radio/ (24.04.2023)

[30] “OpenBTS Github Repository.” [Online]. Available: https://github.com/RangeNetworks/openbts (15.03.2023)

[31] “YateBTS Github Repository.” [Online]. Available: https://github.com/yatevoip/yatebts (15.03.2023)

[32] “Setting up Yate and YateBTS with the bladeRF.” [Online]. Available: https://github.com/Nuand/bladeRF/wiki/Setting-up-Yate-and-YateBTS-with-the-bladeRF (18.04.2023)

[33] “Gotta Catch ’Em All: Understanding How IMSI-Catchers Exploit Cell Networks.” [Online]. Available: https://www.eff.org/wp/gotta-catch-em-all-understanding-how-imsi-catchers-exploit-cell-networks (10.03.2023)

[34] “YateBTS documentation.” [Online]. Available: https://yatebts.com/documentation/ (18.04.2023)

[35] *DragonOS Focal YateBTS Calls + SMS w/ BladeRFxA4 (Yate RC2).* [Online]. Available: https://youtu.be/JGRWjo1Niho

[36] “BLADERF 2.0 MICRO IS SMALLER, MORE POWERFUL.” [Online]. Available: https://hackaday.com/2018/08/30/bladerf-2-0-micro-is-smaller-more-powerful/ (18.04.2023)

[37] “bladeRF 2.0 micro xA4.” [Online]. Available: https://www.nuand.com/product/bladerf-xa4 (18.04.2023)

[38] “Nuand Webpage.” [Online]. Available: https://www.nuand.com (18.04.2023)

[39] “RTL-SDR.COM.” [Online]. Available: https://www.rtl-sdr.com/ (18.04.2023)

[40] “HackRF FAQ.” [Online]. Available: https://hackrf.readthedocs.io/en/latest/faq.html (18.04.2023)

[41] “Download Virtualbox.” [Online]. Available: https://www.virtualbox.org/wiki/Downloads (18.04.2023)

[42] “DragonOS.” [Online]. Available: https://sourceforge.net/projects/dragonos-focal/ (18.04.2023)

[43] “bladeRF-cli - command line interface and test utility.” [Online]. Available: https://manpages.ubuntu.com/manpages/bionic/man1/bladeRF-cli.1.html (25.04.2023)

[44] “Welcome to gqrx.” [Online]. Available: https://gqrx.dk/ (25.04.2023)

[45] “LEVIALA” [Online]. Available: https://sky.ee/retrofm (25.04.2023)

[46] H. Gonzalez, “SIM Card Info.” [Online]. Available: https://play.google.com/store/apps/details?id=me.harrygonzalez.simcardinfo&hl=en&gl=US&pli=1

[47] “pySim - Read, Write and Browse Programmable SIM/USIM Cards.” [Online]. Available: https://github.com/osmocom/pysim

[48] “What is a Faraday cage?” [Online]. Available: https://www.livescience.com/what-is-a-faraday-cage

[49] “GoDark® Faraday Sleeve Case for Laptops.” [Online]. Available: https://godarkbags.com/products/godark-faraday-sleeve-for-laptops

[50] *Ammo Box Faraday Cage*. [Online]. Available: https://youtu.be/D0e_FJR7CuQ

[51] “scrcpy (v2.0).” [Online]. Available: https://github.com/Genymobile/scrcpy (25.04.2023)

[52] “Network Cell Info Lite & Wifi.” [Online]. Available: https://play.google.com/store/apps/details?id=com.wilysis.cellinfolite&hl=en&gl=US

[53] “What is Smishing?” [Online]. Available: https://www.barracuda.com/support/glossary/smishing (25.04.2023)

[54] “Man in the middle (MITM) attack.” [Online]. Available: https://www.imperva.com/learn/application-security/man-in-the-middle-attack-mitm (25.04.2023)

[55] S. Mjølsnes and R. Olimid, “Easy 4G/LTE IMSI Catchers for Non-Programmers,” Feb. 2017, pp. 235–246, doi: 10.1007/978-3-319-65127-9_19.

[56] “Russia tracks visiting Finns via smartphone IMEI codes, Yle finds” [Online]. Available: https://yle.fi/a/74-20028795 (25.04.2023)

[57] “EFF praises Android’s new 2G kill switch, wants Apple to follow suit.” [Online]. Available: https://arstechnica.com/gadgets/2022/01/eff-praises-androids-new-2g-kill-switch-wants-apple-to-follow-suit/ (25.04.2023)

[58] “2G and 3G Shutdowns Continue.” [Online]. Available: https://blog.telegeography.com/2g-and-3g-shutdowns-continue (25.04.2023)

Appendix

1. PySim Installation

```
sudo apt-get install --no-install-recommends \
      pcscd libpcsclite-dev \
      python3 \
      python3-setuptools \
      python3-pyscard \
      python3-pip
pip3 install --user -r requirements.txt
```

License

Non-exclusive license to reproduce thesis and make thesis public

I, **Georg Šumailov**,

1. herewith grant the University of Tartu a free permit (non-exclusive license) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Rogue Mobile Phone Base Station,

   supervised by Danielle Melissa Morgan.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons license CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive license does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Georg Šumailov
09/05/2023