Radboud University Nijmegen

Faculty of Science

Implementing and testing downgrade attacks on an LTE network

Master's Thesis Computing Science

Author:
Jeroen Wijenbergh

Supervisor:
Katharina Kohls

Second reader:
Erik Poll

January 2022

Acknowledgements

I would like to thank Katharina Kohls (my supervisor) and Thijs Heijligenberg for allowing me to use Software Defined Radios and the OnePlus 8 to test my attacks. Additionally, I want to thank the staff at the Radboud University, especially Simone Meeuwsen, to allow me to work at the university and conduct my tests during the holiday period.

Acronyms

This section consists of an overview of the acronyms used throughout the thesis. While not all acronyms are used often, they are still important to know when reading related work.

Notation	Description	Page List
3GPP	3rd Generation Partnership Project	9, 10, 24, 25, 34, 43–46, 48–51, 54, 56, 57
ADB	Android Debug Bridge	35
AS	Access Stratum	17
BSC	Base Station Controller	13
BTS	Base Transceiver Station	13
CS	Circuit Switching	12, 14
E-UTRAN	Evolved UMTS Terrestrial Radio Access Network	15, 16
eNB	Evolved Node B	16–18, 26, 28, 31, 34, 44, 52
EPC	Evolved Packet Core	15, 16, 31
EPS	Evolved Packet System	15, 18, 19, 24
ESM	EPS Session Management	29
GERAN	GSM EDGE Radio Access Network	12, 13
GSM	Global System for Mobile Communications	11–16, 21, 33, 55
HSS	Home Subscriber Server	14, 18

Notation	Description	Page List
IMSI	International Mobile Subscriber Identity	15, 21, 36, 41, 51, 52, 54
LTE	Long Term Evolution	5, 8–21, 23–25, 27, 29, 34, 36, 38, 46, 53–57
MCC	Mobile Country Code	17, 35, 36, 52, 57
ME	Mobile Equipment	15
MME	Mobility Management Entity	17–19, 29, 30
MNC	Mobile Network Code	17, 35, 36, 52, 57
MT	Mobile Termination	15
NAS	Non Access Stratum	17, 30
NAT	Network Address Translation	18
PDNs	Packet Data Networks	12, 14
PLMN	Public Land Mobile Network	12, 17, 18
PS	Packet Switching	12, 14
PSTN	Public Switched Telephone Network	12, 14
QoS	Quality of Service	19
RF	Radio Frequency	34
RNCs	Radio Network Controllers	13
RRC	Radio Resource Control	17–19, 26, 28, 36, 40, 42, 47, 52, 57
SAE	System Architecture Evolution	15
SDR	Software Defined Radio	9, 10, 31–33, 35, 36, 38, 41, 57

Notation	Description	Page List
SIM	Subscriber Identity Module	5, 9, 15, 21, 23, 26, 35, 36, 39, 47, 49–52, 56
SSH	Secure Shell	36
SSL	Secure Sockets Layer	21
SUCI	Subscription Concealed Identifier	54, 55
TAU	Tracking Area Update	23, 24, 55, 57
TCP	Transmission Control Protocol	34
TE	Terminal Equipment	15
TLS	Transport Layer Security	21
TMSI	Temporary Mobile Subscriber Identity	15, 54
UE	User Equipment	12, 15–21, 24, 29, 31, 32, 34, 36, 44, 45, 51, 52, 55–57
UHD	USRP Hardware Driver	35, 36
UICC	Universal Integrated Circuit Card	15
UMTS	Universal Mobile Telecommunication System	11–16, 21, 33, 34, 36, 55
USIM	Universal Subscriber Identity Module	15, 47, 57
UTRAN	UMTS Terrestrial Radio Access Network	12, 13, 15

Abstract

Long Term Evolution (*LTE*) is the network architecture behind the fourth generation (4G) of mobile networks. LTE is more secure than its predecessor (3G) [47], which is important as mobile networks are used by millions of people [12]. However, a weakness in LTE called a downgrade attack has been reported [17, 18, 48]. These downgrade attacks force an LTE-capable phone to (temporarily) only be able to communicate with an older communication standard such as a 3G network. As 3G is less secure than 4G, the improvements made in security for 4G are then rendered meaningless.
While these downgrade attacks have been analysed in literature, there has not been an overview of the difference in the handling of these attacks between mobile devices. Additionally, this research [17, 18, 48] is already a few years old which means that they do not test their downgrade attacks on the current latest mobile devices. Analysing the difference in the handling of these attacks between mobile phones allows us to get a better picture on how practical these attacks are. The latest mobile devices use newer LTE revisions which can mean that these downgrade attacks are made obsolete.
In this thesis, we implement LTE downgrade attacks in open-source software and then test these attacks on different devices. In total, we implement four attacks. Two of these are downgrade attacks, while the other two are variants of downgrade attacks rendering a phone unable to connect to *any* network.
The devices that we tested on are the *Nexus 5*, *OnePlus 3*, *Xiaomi Mi 9T Pro*, and the *OnePlus 8*. We find that all implemented attacks still work on the OnePlus 8 and Xiaomi Mi 9T Pro. These two devices can, however, recover to a normal state by toggling flight mode, re-inserting the *Subscriber Identity Module* (*SIM*) or rebooting the device. Additionally, we find that these devices have a timer to recover from these attacks which is set at an interval between 12 and 24 hours. On the Nexus 5 and OnePlus 3, three attacks work, where one of these attacks does not allow the device to recover using flight mode. The fourth attack does not work on these two devices. We find that the procedure used by this attack is unsupported by certain LTE revisions. Finally, we find that the fifth generation of mobile networks (5G) has insufficient countermeasures against the attacks we implemented.

Contents

Acronyms 2

1 Introduction 8

2 Preliminaries 11
- 2.1 UMTS and GSM architectures . . . 11
  - 2.1.1 Radio access network . . . 12
  - 2.1.2 Core network . . . 14
- 2.2 LTE architecture . . . 14
  - 2.2.1 High-level . . . 14
  - 2.2.2 Architecture of the UE . . . 15
  - 2.2.3 Architecture of the E-UTRAN . . . 15
  - 2.2.4 Architecture of the EPC . . . 16
- 2.3 LTE protocols & procedures . . . 17
  - 2.3.1 Radio Resource Control (RRC) . . . 18
  - 2.3.2 Connection establishment . . . 18
  - 2.3.3 States . . . 18
  - 2.3.4 Attach Procedure . . . 18
  - 2.3.5 Service Procedure . . . 19
- 2.4 Network Attacks . . . 20
  - 2.4.1 Fake base stations . . . 20
  - 2.4.2 Downgrade attacks . . . 21

3 Implementing the attacks 23
- 3.1 Choosing the attacks . . . 23
  - 3.1.1 Connection to related work . . . 23
  - 3.1.2 Denial-of-service attacks . . . 24
  - 3.1.3 Problematic nature . . . 24
  - 3.1.4 High-level message overview of the attacks . . . 25
    - 3.1.4.1 Attach Reject downgrade . . . 25
    - 3.1.4.2 Attach Reject denial-of-service . . . 26
    - 3.1.4.3 Service Reject downgrade . . . 27
    - 3.1.4.4 Service Reject denial-of-service . . . 28
- 3.2 Technical implementation details . . . 28

4 Testing environments 31
- 4.1 Overview . . . 31
  - 4.1.1 Virtual network . . . 31
  - 4.1.2 Physical setup . . . 32
- 4.2 Detailed setup description . . . 33
  - 4.2.1 Virtual network . . . 33

4.2.2 Physical network . . . 34
4.2.2.1 Setting up the mobile devices . . . 34
4.2.2.2 Setting up the 4G Network . . . 35
4.2.2.3 Setting up the 3G Network . . . 36
4.2.2.4 Final setup . . . 37

5 Testing procedure **38**
5.1 Attach Reject downgrade . . . 38
5.2 Attach Reject denial-of-service . . . 40
5.3 Service Reject downgrade . . . 41
5.4 Service Reject denial-of-service . . . 42

6 Attack Analysis **44**
6.1 Hypothesis . . . 44
6.1.1 Differences between 3GPP releases . . . 44
6.1.1.1 Improvements in handling with later 3GPP releases . 45
6.1.1.2 Service Reject denial-of-service not supported . . . 45
6.1.2 Expected result . . . 46
6.2 Results . . . 47
6.2.1 Attach Reject downgrade . . . 47
6.2.2 Service Reject downgrade . . . 48
6.2.3 Attach Reject denial-of-service . . . 48
6.2.4 Service Reject denial-of-service . . . 49
6.2.5 Overview . . . 50
6.3 Discussion . . . 50
6.3.1 Reflecting on the results . . . 50
6.3.2 Usage in practice . . . 51
6.3.2.1 Adversary difficulty . . . 51
6.3.2.2 Signal overshadowing . . . 53
6.3.3 Countermeasures . . . 54
6.3.3.1 5G improvements . . . 54
6.3.3.2 Additional countermeasures . . . 54

7 Conclusion **56**
7.1 Summary and findings . . . 56
7.2 Future work . . . 57

8 Appendix **58**
8.1 SrsRAN vs OpenLTE code changes . . . 58
8.2 Pcap 3GPP release info . . . 58
8.2.1 Nexus 5 . . . 59
8.2.2 OnePlus 3 . . . 59
8.2.3 Xiaomi Mi 9T Pro . . . 60
8.2.4 OnePlus 8 . . . 60
8.3 OnePlus 8 APN Settings . . . 61
8.4 Scat flags . . . 61

1 Introduction

In a recent study from October 2021 [12], the fraction of users that access the internet through mobile devices is over 90 percent and the total share of mobile traffic is over 50 percent. While this data is an approximation and part of it was obtained through extrapolating older data sets, we can confidently say that a large fraction of internet traffic comes from mobile devices. These mobile devices connect to the internet through a variaty of networks, such as home networks, company networks, public hotspots or mobile networks. Out of all of these options mobile networks are used the most [42]. Mobile networks is an overarching term used for different communication technologies, ranging from the second generation networks (2G), up to the fifth generation (5G). Out of these generations, the current most used generation is the fourth generation (4G), powered by the Long Term Evolution (LTE) architecture.
4G is an improvement over 3G, especially, in terms of speed [16, 58] and security [47]. Focusing on the latter, security, it is important that these mobile technologies are secure against adversaries (e.g. adversaries that tamper or eavesdrop the communication). If an adversary finds a weakness or vulnerability in the LTE standard or an implementation thereof in software, it could use this against a great number of active subscribers within range of its rogue network.
One of such weaknesses in the LTE system is described in the literature [17, 18, 48] as a downgrade attack. Downgrade attacks do not only exist in the LTE system, they are often a problem in devices and systems that have to be backwards compatible with older networks. As in a downgrade attack, the goal is to move to an inferior and/or older protocol that is easier to exploit by an adversary. In a typical downgrade attack scenario, the user is first connected to the more secure protocol and through carefully crafted messages an adversary can force the user to move to the less secure protocol. Relating this back to LTE, the goal of such a downgrade attack is to force the victim to communicate through a non-LTE network, such as 2G or 3G, even though the mobile device is compatible with LTE. As hinted before, these older generations are less secure which ultimately makes the better security guarantees from LTE meaningless.
An example of a downgrade attack in LTE is introduced by the paper *Practical Attacks Against Privacy and Availability in 4G/LTE Mobile Communication Systems* from Shaik et al [48]. This downgrade attack uses a procedure in LTE that is not integrity protected nor is it authenticated by the mobile device [48]. The main requirement for an attacker to exploit this vulnerability is to set up a rogue network which the mobile device connects to. For this to happen, the rogue network needs a way to force the mobile device to connect to the network instead of to legitimate networks. One way to achieve this, as described by the authors [48], is to configure the rogue network to use high priority frequencies.
The implementation of the rogue network in software is achieved by changing the

open source *OpenLTE* [36] software that emulates an LTE network. This changed code is compiled and tested on a laptop attached with a *Software Defined Radio* (*SDR*), the device that handles the wireless interaction between the laptop and the phone, and several LTE-capable mobile devices. The authors find that the mobile devices are unable to connect to LTE networks up until the devices are rebooted or up until the SIM card is re-inserted. Additionally, the authors state that mobile devices "having baseband from most vendors can recover by toggling flight mode" [48].

After this paper was published, there has been several follow-up research papers. One of these papers, *LTEInspector: Model-based testing* [18] introduces a model-based testing approach to find vulnerabilities in LTE. This paper has discovered 10 new attacks and 9 prior attacks. One of these prior attacks is a variant of the attack that was discovered by the previously mentioned paper from Shaik et al [48]. This variant has a different message flow and procedure that is used for the downgrade, however, the underlying problem and outcome is the same: messages are accepted without integrity protection or authentication and the service of the phone is disrupted.
Furthermore, to add to this research area, a talk was held at the *Hack In The Box Security conference* of 2016 with the title *Forcing A Targeted LTE Cellphone Into an Unsafe Network* [17] that also reports a form of downgrade attack. This attack is very similar to the previously mentioned attack [48], but it has a special addition. Instead of forcing the mobile phone to only connect to *any* non-LTE network (e.g. 2G or 3G), an adversary forces the device to connect to a *specific* 2G network, for example, its own malicious network. This is why this attack is mentioned in the talk as a *redirection* attack.

The research we discussed already explains, analyses, implements and/or tests LTE downgrade attacks. However, there are several factors involved that pique our interest to follow up on this research area. First, LTE has different standards, also referred to as *3rd Generation Partnership Project* (*3GPP*) releases, where the current latest stable release is 3GPP release 16 [3]. Every mobile phone that is used, uses a specific 3GPP release. Later releases have different handling of the procedures that are used in these downgrade attacks as compared to earlier releases. However, the authors from these papers and talks do not explicitly distinguish between them. While they have tested the downgrade attack on several mobile devices [18, 48], they do not give a clear overview of the difference in results for these various devices. Analysis of a difference in result has been done multiple times in related LTE research. For example, one paper [41] analyses the difference in result of an attack that obtains the identifier for the SIM card of the mobile devices. For downgrade attacks, a difference in result might be that one phone downgrades while the other does not. Additionally, there can be a difference in how these phones recover from these attacks.
On top of this, as the papers we discussed that mention downgrade attacks are already a few years old, they did not test these attacks on devices with LTE 3GPP releases that came out later. Therefore, we propose to follow up on this research area, by analysing how these differences in 3GPP releases and vendor implementations relate to the handling of the downgrade attacks. This will give us a better understanding of how useful these attacks are in practice.

In this thesis, we document, implement and test downgrade attacks in an open source software stack that emulates LTE: *SrsRAN* [52]. The SrsRAN software shares code with the OpenLTE codebase [49] and nowadays has more active development compared to OpenLTE (see Section 8.1). Additionally, SrsRAN currently has support for parts of 5G [52]. This allows an implementation of a downgrade attack in SrsRAN to be used to test against the 5G standard when SrsRAN has proper support for this in the future.

In total, we implement four attacks. While the first two are *denial-of-service* attacks (the mobile device is unable to connect to *any* network), the other two are downgrade attacks similar to what was previously discussed. For each type of attack there are two implementations as these denial-of-service attacks and downgrade attacks are analysed using two distinct LTE procedures: the *Attach* procedure and the *Service* procedure. The reason we decided to also test denial-of-service attacks where the phone is unable to connect to LTE and non-LTE networks, is the fact that the implementation of them does not differ much as compared to the implementation of the downgrade attacks. This was also highlighted by the paper from Shaik et al [48]. Additionally, denial-of-service attacks are arguably even more dangerous as compared to downgrade attacks as the goal is to force the victim to be unable to communicate with any outsider instead of still allowing the victim to communicate through an inferior protocol.
The testing of our implementation is conducted on four devices (the *Nexus 5, OnePlus 3, Xiaomi Mi 9T Pro* and the *OnePlus 8*) which all have different 3GPP releases (ranging from 3GPP release 9 to 15) and operating system versions. As we also want to test whether the devices can still connect to a non-LTE network, we set up the *OpenBTS-UMTS* open source software that emulates a 3G network using the Ettus USRP B210 as SDR. For the 4G network (the SrsRAN network), we use the Ettus USRP B205 mini as SDR.
Additionally, we also want to test the attack in a virtual network. Unlike OpenLTE, SrsRAN has a mobile device emulation layer which allows the user to set up a fully end-to-end network using software running on one computer. This eliminates the need for expensive SDRs and mobile devices to test the software. While testing in a virtual network does not accurately reflect the real world, this will be used to test our implementation during development.

The outline of this thesis can be described as follows.
In Chapter 2: *Preliminaries*, we give a bird's-eye view of the required knowledge to understand our research. This mainly consists of an overview of the 2G, 3G and 4G (LTE) architectures. We also go into detail for the various LTE concepts, and for each we explain how they fit into our research.
The main scientific contribution of our thesis is given in the remaining chapters. Chapter 3: *Implementing the attacks*, explains the thought process behind picking the different attacks and implementing them in SrsRAN. Then Chapter 4: *Testing environments*, explains the environments that we use to test our attacks with. The actual procedure to test these attacks is explained in Chapter 5: *Testing procedure*. A deep dive into the attacks and presenting the results is given in Chapter 6: *Attack analysis*. Additionally, this chapter discusses how practical these attack are and how they could possibly be prevented or weakened. Finally, in Chapter 7: *Conclusion*, we end with a conclusion where we summarize what we have done and what is left to be explored for future work.

2 Preliminaries

This chapter contains the preliminaries that are needed to read our research. The main focus of this section is to explain LTE: the fourth generation mobile network standard. For this, the book *An introduction to LTE* by *Christopher Cox* [11] is used.
While we expect that the reader has a basic understanding of computer networks, we intend to keep this section as high-level as possible. However, in certain parts it is crucial that we give a detailed explanation as it is more important to our implementation, testing and analysis sections.

Section 2.1 gives a recap of the different networks that LTE was based upon. These are the second and third generation of mobile networks, called *Global System for Mobile Communications* (*GSM*), and *Universal Mobile Telecommunication System* (*UMTS*) respectively.
Afterwards, in Section 2.2, we give a high-level overview of LTE and how it differs from GSM and UMTS. Section 2.3 gives a description of the various LTE protocols and procedures that are relevant for the downgrade attacks.
Finally, in Section 2.4, we give a description of common network attacks in mobile networks that are relevant for our thesis.

2.1 UMTS and GSM architectures

LTE is based upon the UMTS and the GSM network architectures. It is important that we first get an understanding of these networks, before we can dive into the inner workings of the LTE network.

UMTS and GSM have the following components:

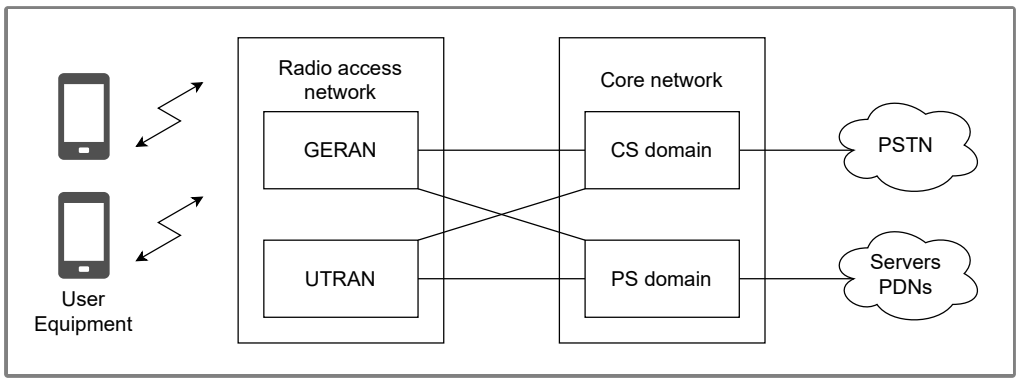


Figure 2.1: High-level UMTS and GSM overview [11]

On the left, we have the mobile phone, in mobile networks referred to as the *User Equipment* (*UE*). The UE communicates with the *core network* through the *radio access network* using the *radio interface* (also widely known as the *air interface*). This core network including the radio access network is often referenced as a *Public Land Mobile Network* (*PLMN*) that is run by a network operator such as *T-Mobile* or *Verizon*.

Between the UE and the radio access network we can distinguish between two communication directions:

- *Downlink/Forward* link: From the radio access network towards the UE
- *Uplink/Reverse* link: From the UE towards the radio access network

The functionality of the core network is split into two separate domains:

- *Circuit Switching* (*CS*): Used for sending phone calls, which is communicated with the *Public Switched Telephone Network* (*PSTN*). As the name suggests, it uses circuit switching to transport the data. In circuit switching, a separate unique channel needs to be set up between each source and destination [27]. The channel is terminated when the communication is no longer needed, otherwise the channel needs to stay open.
- *Packet Switching* (*PS*) domain: Used for transporting streams of data, for example, e-mails and website traffic to the *Packet Data Networks* (*PDNs*): the internet. As the name suggests, circuit switching is not utilized, instead it uses *packet switching*. In packet switching, data is split up into fragments (*packets*). Using these packets, a dedicated unique channel does not need to be reserved for the entire communication lifetime. A communication link needs to be set up on demand, which means that there can be a wait time to send packets [27].

However, as we will see later on, only packet switching is used in LTE.
As we have now given a broad overview of the UMTS and GSM systems, we take a more detailed look by first going over the elements of the radio access network.

2.1.1 Radio access network

As we have explained, the radio access network is used as the middleman between the UE and the core network. In Figure 2.1, there are two separate radio access networks depicted, the *GSM EDGE Radio Access Network* (*GERAN*) and the *UMTS Terrestrial Radio Access Network* (*UTRAN*). GERAN and UTRAN are similar on a high-level. The

difference is that the communication technique is different; they use GSM and UMTS respectively. Additionally, they also use different terminology for their components. First, we provide a high-level overview of UTRAN. Then, briefly explain the differences in terminology with GERAN:

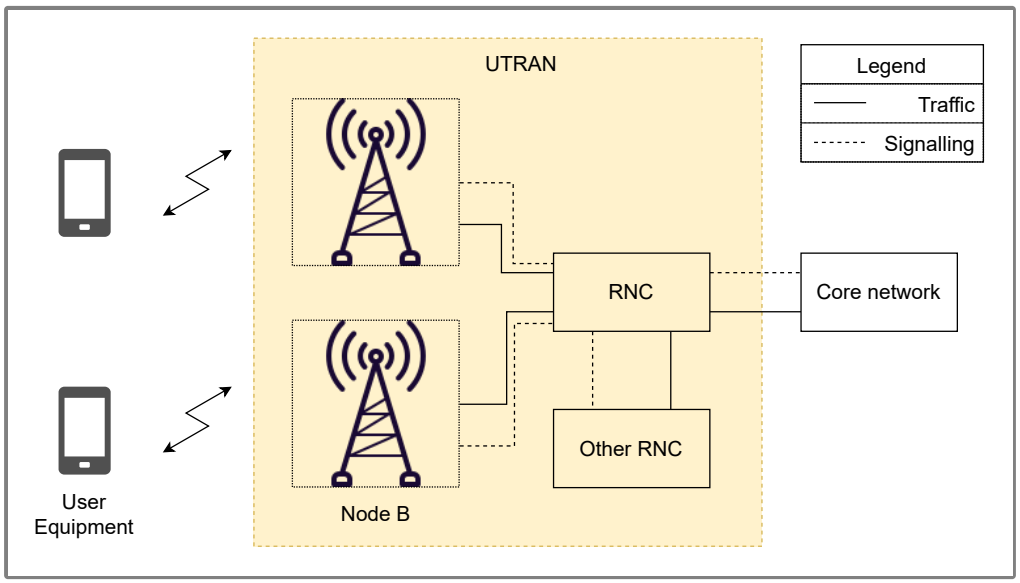


Figure 2.2: UMTS Terrestrial Radio Access Network overview including User Equipment and Core network interaction [11]

Node B stands for the base station of the network. The base station takes care of the communication with the mobile devices using *antennas*. An important term to know is *sector*, it signifies the area in which the base station communicates with the mobile device. There can be more than one sector in which the base station communicates with the mobile. Sectors are often also called *cells* and will be further referenced in our downgrade attacks.
In Figure 2.2, we see two different connections between the components: *signalling* and *traffic*. Knowing the difference in these means of communication is crucial as it is also used in LTE. Traffic is data that is sent by the user's device and forwarded by the network. Whereas signalling are messages that control communication. For example, messages belonging to the *handover* procedure.

If a mobile phone moves from one cell to another, two techniques are used. The handover technique is used if a mobile is actively communicating with the network. A technique called *cell reselection* is used for mobiles that are on standby.
An important component in these signalling protocols are *Radio Network Controllers* (*RNCs*). RNCs control a group of base stations. They pass the different data packets to the core network. They also communicate with mobile phones to e.g. handle the aforementioned handover process.

In GERAN, the radio access network looks similar to that of UMTS (UTRAN). However, as we said before, there is a difference in terminology. The base station in GERAN is known as the *Base Transceiver Station* (*BTS*) and the radio network controller is known as the *Base Station Controller* (*BSC*). To switch between UMTS and GSM networks, the *inter-system handover* process is used.

2.1.2 Core network

The core network only needs a general explanation as the only components that we will reference in our thesis are specific to the LTE system, not to GSM and UMTS. The core network is the same for UMTS and GSM:

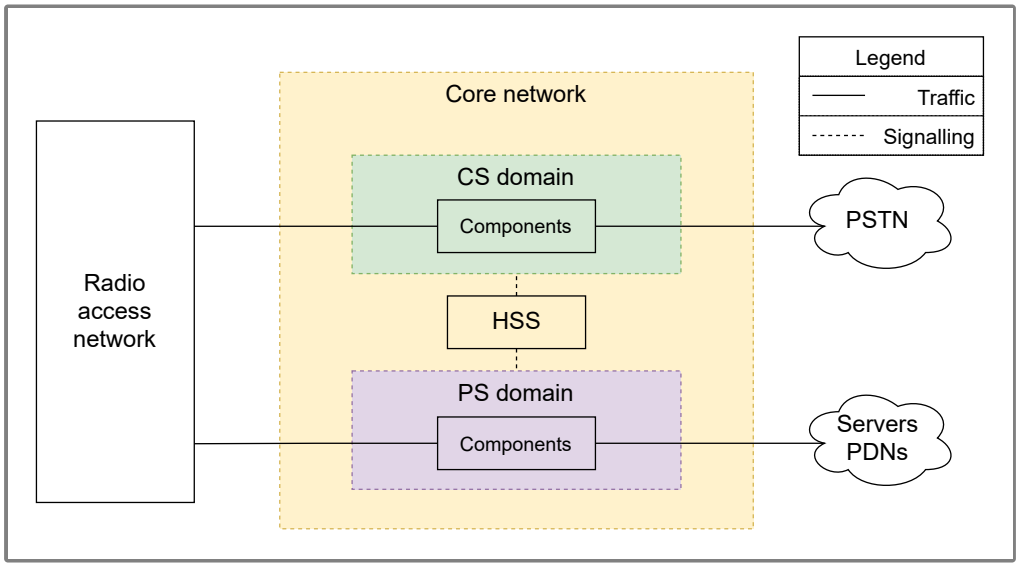


Figure 2.3: Core network architecture of UMTS and GSM including Radio access network and PDNs/PSTN interaction highly simplified [11]

Here, we see the two domains: the Circuit Switching (CS) and Packet Switching (PS) domain. This overview of the core network is highly simplified as both domains have plenty of components. In this figure, these different components are denoted as simply "components" as they are not important in order to understand LTE. The only components that we need to discuss is the *Home Subscriber Server* (*HSS*) which contains the knowledge about the subscribers of the network. This HSS is also present in the LTE architecture.
As we now have enough background information for the GSM and UMTS networks, we can now continue on with an overview of the LTE architecture.

2.2 LTE architecture

This section contains a description of the LTE architecture where we explain different terms such as *IMSI*, *TMSI*, *E-UTRAN*, *EPC*, and *eNB*. These terms are all used throughout this thesis, so it is important to explain them.

We first give a high-level overview of the different components of an LTE network. Then, we take a closer look at these components by describing what their main part is in the LTE architecture and how they interact with the other components in the LTE system. For each component, we also explain why they are relevant to our thesis.

2.2.1 High-level

Figure 2.4 provides a high-level overview of the architecture of the LTE system:

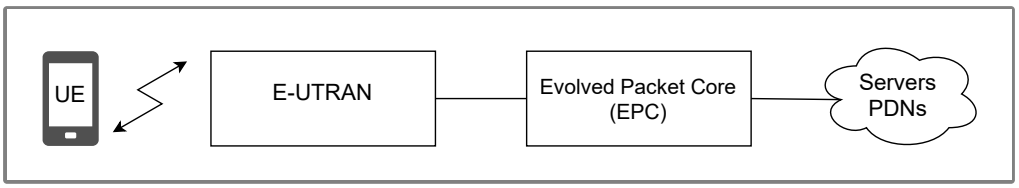


Figure 2.4: LTE architecture [11]

The *Evolved Packet Core* (*EPC*) is the replacement of the packet switching domain of the core network from UMTS/GSM. There is no circuit switching; it does not support voice calls in the same way as GSM/UMTS.

The radio access network is now called the *Evolved UMTS Terrestrial Radio Access Network* (*E-UTRAN*) and replaces UTRAN from UMTS. The acronym LTE refers to the evolution of this E-UTRAN.
Whereas the evolution of the core network (EPC) is called the *System Architecture Evolution* (*SAE*), the combined system is called the *Evolved Packet System* (*EPS*). Nevertheless, in this thesis, and in literature [11], LTE describes the whole system (a synonym for EPS).
As we have yet to touch upon the User Equipment, we thus do that first and then continue on from left to right (first the E-UTRAN and then the EPC).

2.2.2 Architecture of the UE

The User Equipment consists of the communication device and the SIM card. The technical terms for these components is the *Mobile Equipment* (*ME*) for the hardware and respectively the *Universal Integrated Circuit Card* (*UICC*) for the SIM card. The ME can be further divided into the *Mobile Termination* (*MT*) and the *Terminal Equipment* (*TE*). The exact details of these components do not have to be known in order to understand the implementation, testing and analysis of our attacks.
An important term to know, though, is the *Universal Subscriber Identity Module* (*USIM*). This is an application on the UICC that is used to handle the cryptography and data storage for the SIM card.
To identify the UICC and the USIM, the *International Mobile Subscriber Identity* (*IMSI*) is used. This IMSI must only be known between the legitimate network and the SIM card as when an adversary obtains this identifier, it can find out the physical location of the mobile device. Thus, a temporary identifier is assigned to the SIM when the UE has established a connection to the network. This temporary identifier is known as the *Temporary Mobile Subscriber Identity* (*TMSI*).
The next part in the LTE system that we need to explain is the architecture of the E-UTRAN. This is done in the following section.

2.2.3 Architecture of the E-UTRAN

The radio access network in LTE is called the E-UTRAN and has the following structure:

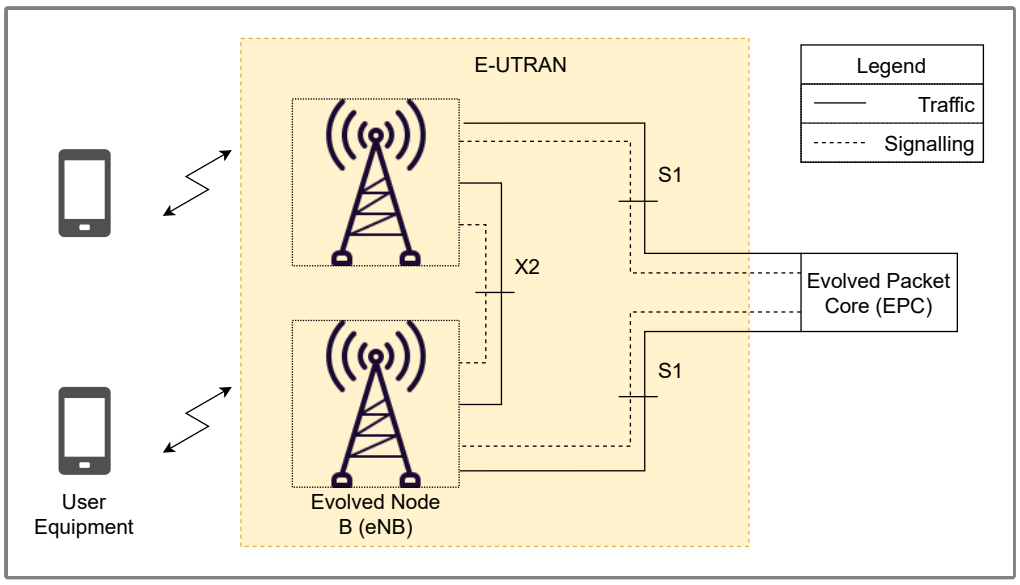


Figure 2.5: Overview of the Evolved UMTS Terrestrial Radio Access Network (E-UTRAN) including the UE and EPC interaction [11]

Because Figure 2.5 looks different from the core network in UMTS and GSM, we explain the differences and in the next section do the same for the core network.
Instead of the terminology "Node B", a base station is now called the *Evolved Node B* (*eNB*). In E-UTRAN, these base stations are not grouped through radio network controllers, but instead through *interfaces*.
In LTE, interfaces define the connections between components. Using these connections, messages can be sent between these components. This should not be confused with the more common computer networking definition of an interface: hardware that allows the computer to set up a communication link. In LTE, when we say interface we often mean the link itself, not the hardware. The *X2* and *S1* identifiers are two examples of such interfaces. It is important that we explain how they work generally as they are referenced in the code that we write for the downgrade implementation.
The X2 interfaces define the link between base stations, whereas the S1 interface defines the link between base station and the Evolved Packet Core.
The base stations, apart from receiving uplink and downlink transmissions, also control operations such as the handover process (the process which was briefly explained in UMTS and GSM). This process is now done by the *X2* interface instead of the radio network controllers. It is important to note that the *X2* interface can be seen as optional; the base stations can communicate with a neighbouring base station by forwarding the message through the core network using the S1 interface.

2.2.4 Architecture of the EPC

The Evolved Packet Core (EPC) is the replacement of the core network from GSM and UMTS. The Evolved Packet Core has multiple components that we have not seen before in UMTS or GSM:

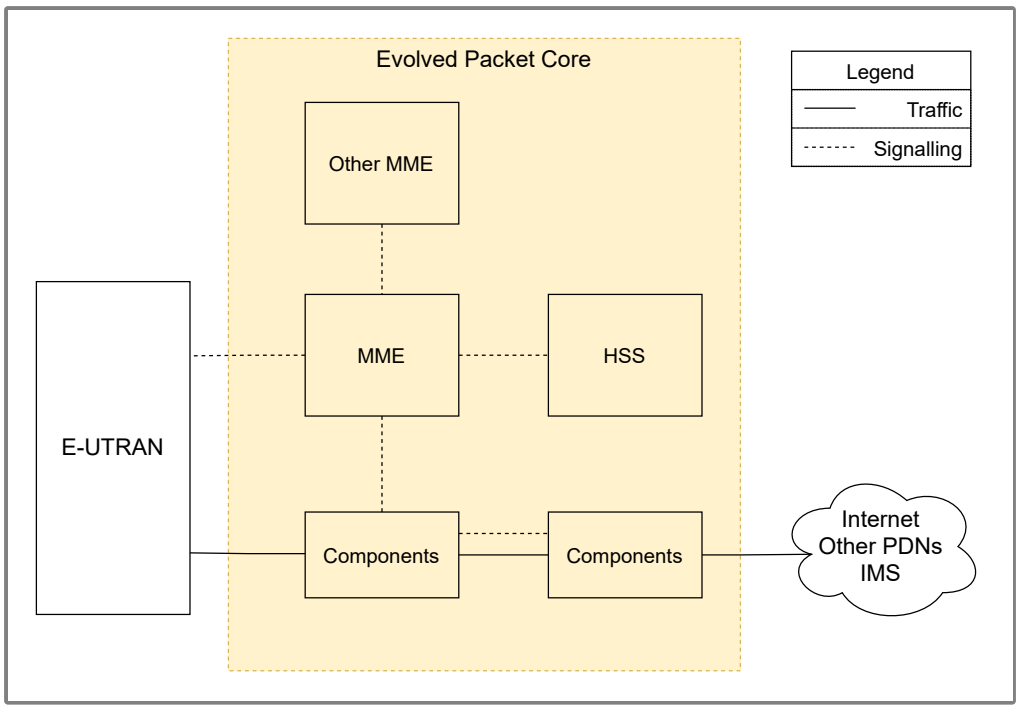


Figure 2.6: Overview of the Evolved Packet Core including the E-UTRAN and internet interaction [11]

For our thesis, it is most important to understand the *Mobility Management Entity* (*MME*) which handles the *mobility* of the UE. With mobility, we mean the fact that a mobile device will not have a connection to one specific access point. A mobile phone connects from one base station to another. The messages that the MME sends are messages not related to radio communications (e.g. security issues). Each UE is assigned to a single MME, known as the *serving* MME. The "components" boxes signify other components that are not important to know. It is most important to get a high-level overview of this core network, the exact parts of this system do not have to be known in detail. What has to be known is the *Mobile Country Code* (*MCC*) and *Mobile Network Code* (*MNC*) identifiers. Together they form the identity for the PLMN.
The core network can send signalling messages to the UE. This is done using the *Non Access Stratum* (*NAS*) signalling layer. In contrast, the signalling between the UE and the eNB is made using the *Access Stratum* (*AS*) signalling layer.

2.3 LTE protocols & procedures

This section contains information about the various LTE protocols and procedures. These are important as they are used in our downgrade attack.

We first explain the *Radio Resource Control* (*RRC*) protocol, then continue with the Attach and Service procedure. These protocols and procedures are used extensively by the downgrade attacks that we implement.

2.3.1 Radio Resource Control (RRC)

To form the communication between the User Equipment and the base station, the Radio Resource Control protocol is used. To set up this protocol, the RRC Connection Establishment procedure is initiated.

2.3.2 Connection establishment

The establishment in RRC is called the *RRC Connection Establishment* procedure and consists of three messages:

- *RRC Connection Request*: The initial request that sends the identity of the UE and a cause for establishment, for example, an establishment cause can be *Emergency* [2].
- *RRC Connection Setup*: Sent to the UE to signal that the request has been received and that the base station will now be the *serving* base station of the UE.
- *RRC Connection Setup Complete*: When the message has been received by the UE and the correct protocols have been set up from the message configuration, the UE will set its internal *state* to *RRC_CONNECTED*. The Setup Complete message is then transmitted to the base station with the following parameters: the PLMN it will connect to, the ID of the previous serving MME and an EPS message which needs to be sent to the MME.

As the state in the RRC Connection Setup Complete message is not explained yet, we do that in the next section and then continue with the Attach Procedure.

2.3.3 States

The UE has two main RRC states it can be in: *RRC_IDLE* and *RRC_CONNECTED*. The initial state of the UE is RRC_IDLE. When a connection is established with a base station, the state is set to RRC_CONNECTED. However, when the connection is released again, the UE's state is reset to RRC_IDLE. This can be modeled as a state machine. In this thesis, we reference these states often, so it is important to remember what they represent.

2.3.4 Attach Procedure

The Attach procedure has multiple purposes. It is used to register the location of the UE with a MME and to allow the UE to connect to the internet. The Attach procedure is a complex procedure that involves multiple parts of the LTE system: UE, eNB, MME, HSS. In this section, we will simplify this procedure by only looking at the UE, eNB, and MME interaction. Operations such as how exactly the location is updated, will not be discussed as it is not relevant to our attacks.

For the UE to connect to the internet, it needs an IP-address. This IP-address is a private IP-address (in the case of IPv4) and is mapped to a public IP-address using *Network Address Translation* (*NAT*). To obtain the *private* IP-address, the UE initiates the Attach procedure. The initiating message for the Attach procedure, is called the *Attach Request*. This message is sent along with the RRC Connection Setup Complete message, as the RRC connection first needs to be set up between the base station and the User Equipment. The base station will forward this Attach Request to the MME in an *Initial UE message*. When this Attach Request is accepted by the network, an Attach Accept message is sent to the base station and the base station and UE reconfigure

RRC. At last, an Attach Complete message is sent to the MME. Below is a figure of this high-level overview for the Attach procedure:

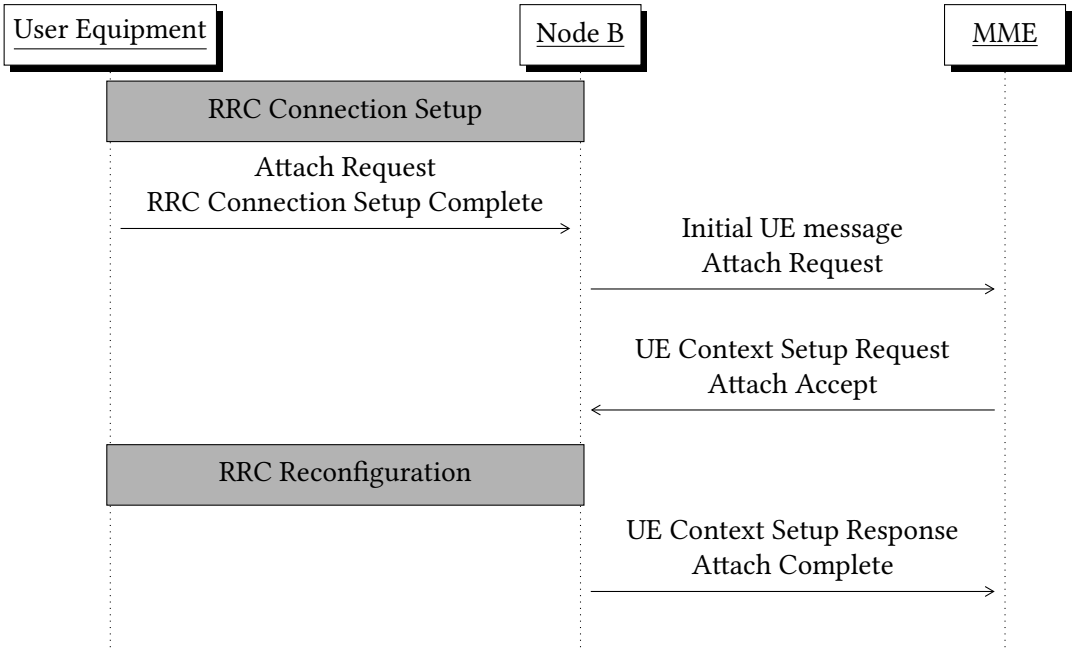


Figure 2.7: LTE Attach Procedure [11]

In reality, the Attach Procedure is much more complex. An important detail that we left out is that the UE needs to set up an *EPS bearer* with the MME. This EPS bearer is used as a communication pipe to ensure *Quality of Service* (*QoS*) and mobility. The QoS signifies the performance of the communication. For example, someone that pays a premium fee can get a connection that is more stable and faster. The RRC Reconfiguration phase, the MME, and the components we left out further sets up this EPS bearer.
It is also possible that the network does not accept the Attach Request, this is then responded with an *Attach Reject* message by the MME. Some possible causes for the Attach Reject are *congestion*, *illegal UE* (meaning that the UE cannot be authorized), and *EPS Services Not Allowed* (the UE is not allowed to use LTE services).

In the next section, we take a look at a similar procedure: the Service procedure. This procedure is also used in the attacks that we implement.

2.3.5 Service Procedure

When the UE wants to send uplink data to the network, but it is already connected and in idle state, the device will initiate the Service Procedure:

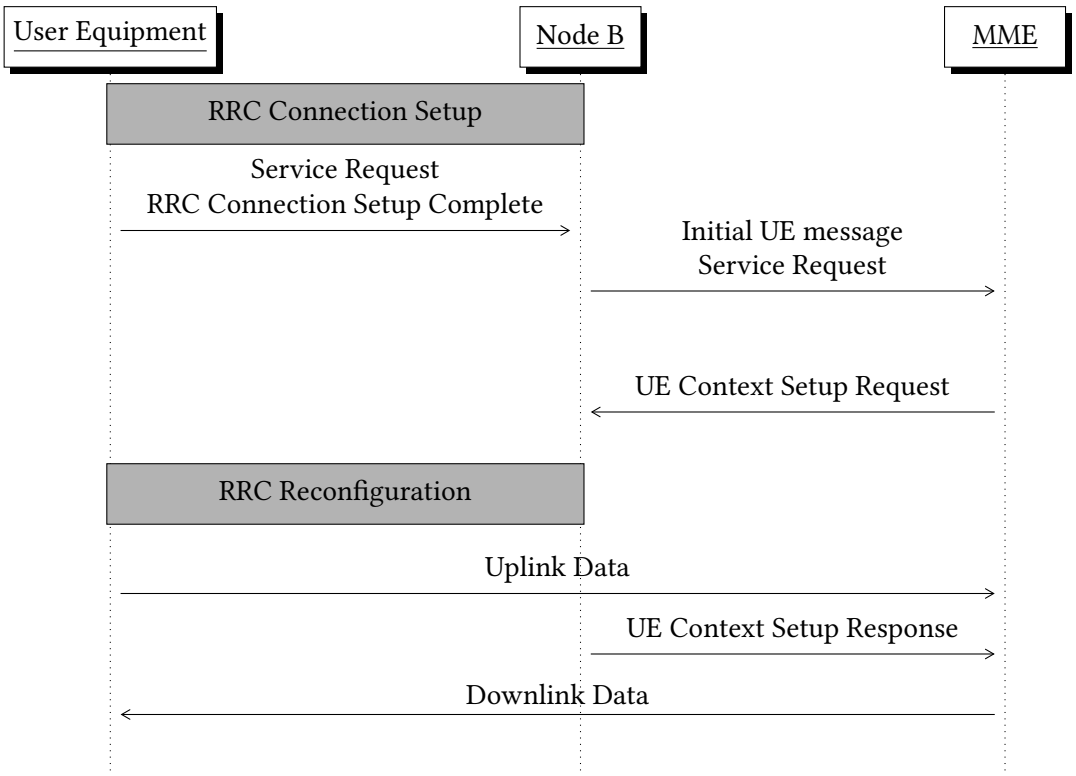


Figure 2.8: LTE Service Procedure initiated by the UE [11]

This looks familiar to the Attach procedure. However, there is no *Service Accept* or *Service Complete* equivalent to the Attach Accept and Attach Complete messages. The *Service Reject* does exist and is sent when the Service Request is not accepted by the network. The causes for this is similar as the Attach Reject message.
It is also possible that the Service Request is sent by the network when the phone is idle, but the network needs to send data to the device. This will not be further explained as it is not part of our downgrade attacks.

In the next section, we take a look at common (mobile) network attacks that are relevant for our thesis.

2.4 Network Attacks

In this section, we explain the idea behind several attacks. These all relate to the attacks that we implement and analyse. How they exactly relate, we discuss in Chapter 6.

The attacks we explain in this section are *IMSI catchers*, *man-in-the-middle* attacks, and at last, downgrade attacks. The first two attacks both are closely related to fake base stations so they will be grouped together in a single section.

2.4.1 Fake base stations

A common method for adversaries to attack victims is by acting as a base station. If the adversary sets up a fake endpoint and gets the user to connect to this endpoint, it

can employ further attacks.

One popular attack in related literature is an IMSI catcher. The goal of an IMSI catcher, as the name suggests, is to obtain the IMSI from a mobile device. These identifiers can be used for targeted attacks as they are unique to each SIM card. The IMSI catcher operates by forcing the mobile device to send the identifier to the adversary. In GSM this was trivial to do: the adversary sets up a fake base station which mobile devices can connect to. When the mobile device has been connected to the malicious base station, an identity request can be sent to the mobile device which forces it to send its IMSI to the adversary. In LTE and UMTS, catching an IMSI is still possible. The UE trusts any base station that claims it has not seen the device before [20]. A possible implementation is getting the IMSI using the Attach Request message [21].
Another attack that deals with fake base stations is the man-in-the-middle-attack. The goal of a man-in-the-middle attack is that the attacker sets up a connection in the middle of the sender and receiver without the parties knowing this.
These types of attack are often the result of lack of (proper) authentication between sender and receiver in network protocols. Possible operations an attacker can do in a man-in-the-middle attack is read or change the messages that are sent. Relating this back to mobile networks, man-in-the-middle attacks are also possible here.
GSM has no authentication from the base station to the mobile device. This made a man-in-the-middle attack trivial which resulted in the ability to tap a mobile device connection. UMTS added mutual authentication. This rendered the aforementioned man-in-the-middle useless. However, man-in-the-middle attacks can still be constructed in UMTS. There have been multiple papers that describe how to set up a man-in-the-middle attack. For example, the paper *A man-in-the-middle attack on UMTS* by *Meyer* et al. [28] describes how a man-in-the-middle attack can be set up in UMTS by saving authentication tokens and the ability to reuse them in a later connection with a GSM (backwards compatible) base station [28]. Furthermore, a man-in-the-middle attack in LTE exists by exploiting lower layer protocols [46].

2.4.2 Downgrade attacks

The goal in this thesis is to implement downgrade attacks in the SrsRAN system. While we have discussed them briefly in the introduction, we expand on them here.

In general, the point of a downgrade attack is to force the server and/or client to move to an older and/or inferior protocol version.
For example, a popular downgrade attack is the *POODLE* attack [32] that convinces the server to use the old *Secure Sockets Layer* (*SSL*) 3.0 instead of newer *Transport Layer Security* (*TLS*) versions. In this case, this is achieved by dropping connections up until the server switches to the older SSL protocol. The SSL 3.0 protocol is weak as it authenticates data before encrypting it. The downgrade attack can be used as a stepping stone for the adversary to decrypt data by executing a padding oracle attack [1].
Relating these attacks to the mobile communication systems, such as GSM, UMTS, and LTE, downgrade attacks also play an important role in literature. As was discussed in Chapter 1, the goal for these attacks is to force LTE UEs to only be able to connect to a 3G network or below. This is often achieved by having the user connect to a malicious base station and sending a carefully crafted message which forces the LTE capable UE to disconnect from the network and not reconnect to LTE capable networks up until the SIM card is re-inserted, the phone is rebooted or in some cases when flight mode is

[1] A padding oracle attack is an attack where the adversary sends messages to a padding oracle, an entity that responds if the padding is valid or not. In the case of the POODLE attack, the adversary could leverage a padding oracle attack to decrypt the ciphertext one byte at a time [32]

toggled [48]. Because this type of attack is often the result of the victim connecting to a fake base station, we could group it under the "fake base station" category. However, this is not the only way that these downgrade attacks can be performed. In Chapter 6, we expand on this further.
These downgrade attacks can be seen as a form of a denial-of-service attack. However, in this thesis when we write denial-of-service attack, we mean that the mobile device will not be able to connect to any mobile network, while in a downgrade attack the mobile will still be able to connect to 3G networks or below (e.g. 2G). We make this distinction because in related work the term downgrade attack is sometimes interchanged with denial-of-service (a 4G downgrade *denies the service* of the 4G connection).

3 Implementing the attacks

This chapter contains our thought process behind implementing downgrade and denial-of-service attacks in SrsRAN. This is split into two sections. In the first section, Section 3.1, we elucidate the approach behind picking the different attacks to implement. This also consists of giving the messages that need to be sent by a malicious entity in order to downgrade mobile devices.
The section after that, Section 3.2, contains the description on how we modified the SrsRAN codebase for these attacks. This is a technical description of the most important parts of the *C++* code implementation.

3.1 Choosing the attacks

This section reasons about which attacks we want to implement and test. First, we explain the basis of our attacks and how they link to the related work. This section is mostly a description of the downgrade attacks. The section after that, we explain how the denial-of-service attacks differ.
As it might not be immediately clear to the reader why these attacks form a problematic nature for LTE, we also detail this in a section. A further in depth analysis of these attacks given in Chapter 6.
At last, the overview of messages between the mobile device and the malicious entity is presented.

3.1.1 Connection to related work

As the goal of our thesis is to expand on existing work, we need to take a look at the downgrade attacks that have been implemented in related work. The most relevant papers and talks, we have already discussed in the introduction (Chapter 1).
Two of these [17, 48] describe an attack that involve the *Tracking Area Update* (*TAU*) procedure. More concretely, these attacks utilize the *TAU Reject* message with the cause *EPS Services Not Allowed*. This cause specifies that the mobile can only connect to 3G networks when this message is received and processed up until the SIM card is re-inserted or the phone is rebooted [4]. And additionally, Shaik et al. stated that it is possible to toggle flight mode to recover from these attacks [48]. The problematic nature according to these papers, is that this message is accepted without integrity or authentication, which means that it can be broadcast to any device in the range of a malicious entity.
Initially, we wanted to implement this attack as it was already fitted in the OpenLTE system [31], which SrsRAN shares code with [49].
However, the virtual mobile from SrsRAN currently has no support for the Tracking

Area procedures [55]. And as we want to test our attack also in a virtual network, this downgrade also needs to be triggered by this virtual mobile.

In one of the original papers that describe the TAU Reject downgrade attack, the authors stated the following: "Similar types of attacks are also possible with *Service Reject/Attach Reject messages*" [48]. Because the virtual mobile does have support for the Attach procedure [50, 51] and Service Procedure [54, 53], we want to implement downgrade attacks based on these procedures. Downgrade attacks for the Attach and Service procedures are already fitted in the OpenLTE system [29, 30]. The attack that uses the Attach procedure sends an Attach Reject with the same cause as in the TAU Reject downgrade. This Attach Reject message is sent after the receipt of the Attach Request message.
The attack in OpenLTE that uses the Service procedure, sends a Service Reject message with the same cause as before upon the receipt of a TAU Request message.

We plan to implement the Attach attack in the same way as OpenLTE. For the Service procedure, we send the Service Reject message when a Service Request message is received by the mobile phone. We act upon the Service Request message as the Service Reject message is meant to be sent after the receipt of this message when the connection should be denied.

3.1.2 Denial-of-service attacks

In total there are four attacks that we want to implement. Two of them we have already briefly explained:

1. Sending an Attach Reject message with the cause "EPS Services Not Allowed" upon the receipt of an Attach Request message
2. Sending a Service Reject message with the cause "EPS Services Not Allowed" upon the receipt of a Service Request message

We also want to explore denial-of-service attacks. The reason for this is that these attacks are similar in the related work [48]: the denial-of-service attack gets the cause "EPS Services And Non-EPS Services Not Allowed", while the downgrade attack has the cause "EPS Services Not Allowed" [1]. The remaining two attacks are thus formulated as follows

3. Sending an Attach Reject message with the cause "EPS Services And Non-EPS Services Not Allowed" upon the receipt of an Attach Request message
4. Sending a Service Reject message with the cause "EPS Services And Non-EPS Services Not Allowed" upon the receipt of a Service Request message

3.1.3 Problematic nature

As our attacks are a variation of the downgrade and denial-of-service attacks that are described in related work, we need to find if this variation also has the problematic nature of the original attacks.

The problematic nature of the TAU Reject downgrade, is that the downgrade is accepted without integrity or authentication protection by the UE [4]. Because of this, a malicious base station can target any LTE UE within its range.
We thus need to know why these downgrade attacks also work for Attach Reject and

[1] The paper from the related work [48] uses the term LTE instead of EPS for the causes. In this paper, we use the formulation from the 3GPP specification.

Service Reject messages. In other words, we need to check if this aforementioned problematic nature, is also present in the Attach Reject and Service Reject messages. The 3GPP LTE specification supported by our newest device in the upcoming experimental setup states that the same problem indeed holds for Attach Reject and Service Reject messages [4]:

> *EMM messages:*
>
> - *IDENTITY REQUEST (if requested identification parameter is IMSI)*
> - *AUTHENTICATION REQUEST*
> - *AUTHENTICATION REJECT*
> - *ATTACH REJECT (if the EMM cause is not #25)*
> - *DETACH ACCEPT (for non switch off)*
> - *TRACKING AREA UPDATE REJECT (if the EMM cause is not #25)*
> - *SERVICE REJECT (if the EMM cause is not #25)*
>
> *NOTE: These messages are accepted by the UE without integrity protection, as in certain situations they are sent by the network before security can be activated.*

While this only explicitly mentioned integrity protection, these messages are accepted *before security can be activated* which means that there is also no authentication present [48]. How the handling of these messages exactly differ between each 3GPP release, we discuss in Chapter 6. For now, we continue on with a high-level message diagram for each attack in the next section.

3.1.4 High-level message overview of the attacks

As we have four attacks that we implement, we have four sections that give a message diagram for an attack. We start with the Attach Reject message diagrams and then continue on with the Service Reject diagrams.

3.1.4.1 Attach Reject downgrade

The high-level overview for the Attach Reject downgrade attack is the following:

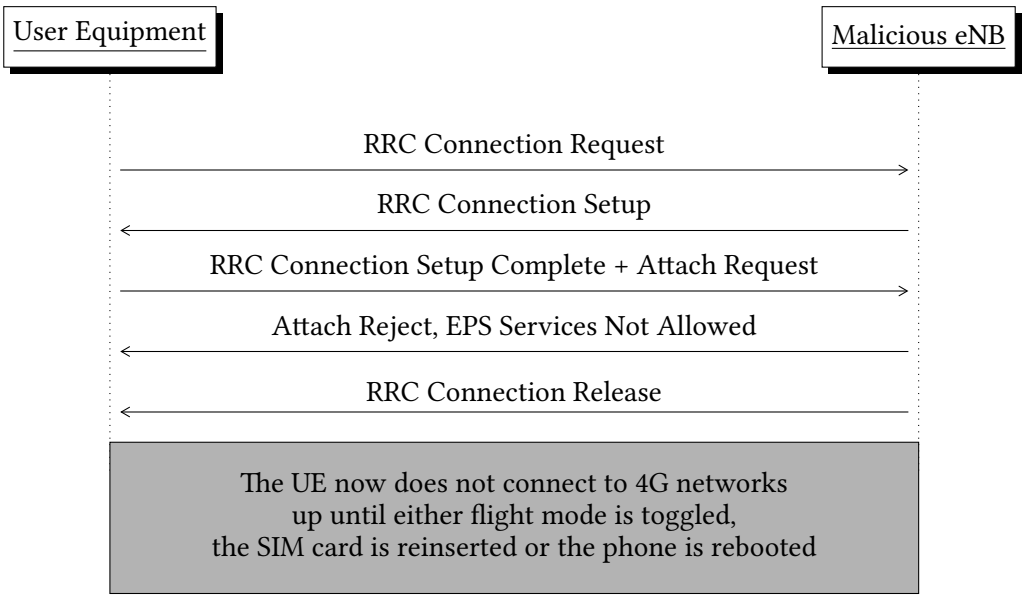


Figure 3.1: High-level overview of the Attach Reject downgrade

The first two messages are the setup of the mobile connection. When this is completed, indicated by the RRC Connection Setup Complete, the Attach Request message is sent along. The eNB sends back the Attach Reject with the downgrade cause and then releases the RRC connection using an RRC Connection Release message. If the attack works, then the result is that the phone cannot connect to 4G networks, up until either flight mode is toggled, the SIM card is reinserted or the phone is rebooted.

3.1.4.2 Attach Reject denial-of-service

The second high-level overview describes the Attach Reject denial-of-service attack:

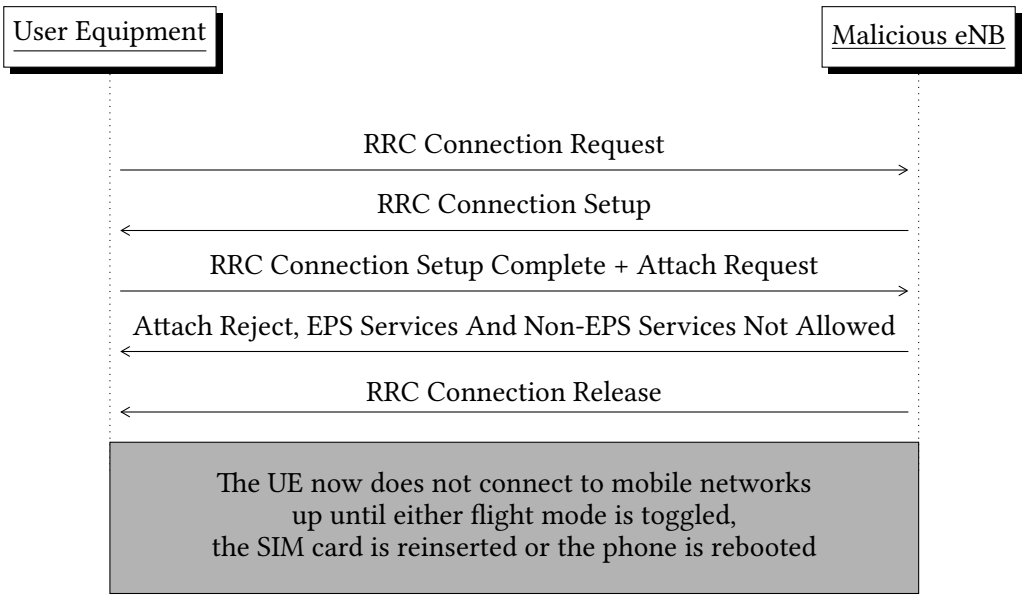


Figure 3.2: High-level overview of the Attach Reject denial-of-service

The difference here is the specified cause and result. As said before, the cause "EPS Services And Non-EPS Services Not Allowed" is used. This forces the mobile phone to be unable to connect to LTE and non-LTE networks.

3.1.4.3 Service Reject downgrade

The high-level message overview Service Reject downgrade is similar to the Attach Request/Attach Reject downgrade:

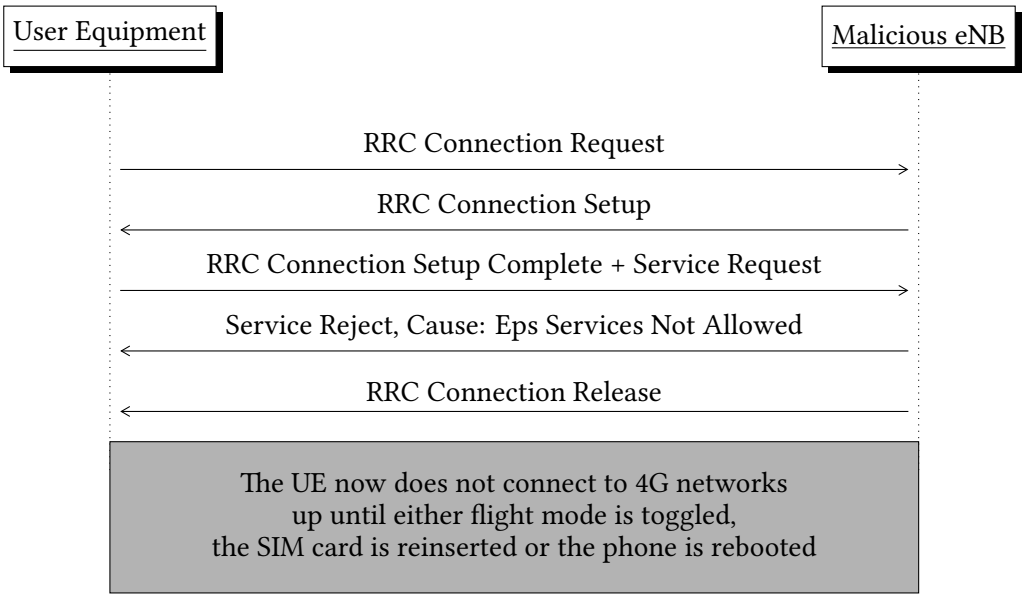


Figure 3.3: High-level overview of the Service Reject downgrade

This downgrade does not act upon the Attach Request message, but instead checks for the receipt of the Service Request message. In place of the Attach Reject message, we send an accompanying Service Reject message which indicates that the core network does not accept the Service Request. This downgraded cause and result is the same as the Attach Reject downgrade attack.

3.1.4.4 Service Reject denial-of-service

The high-level message overview Service Reject denial-of-service is similar to the Attach Request/Attach Reject denial-of-service:

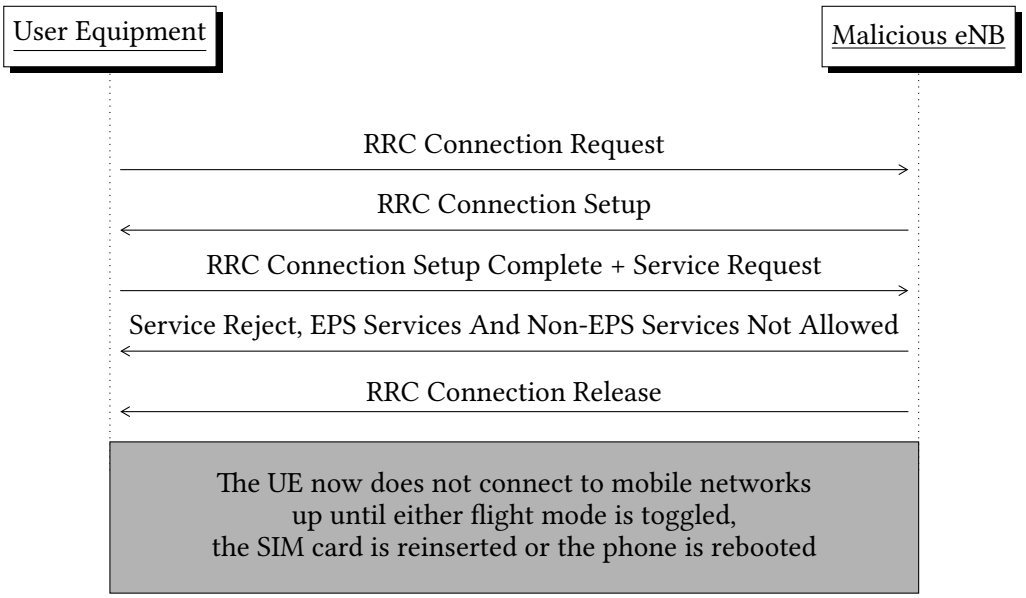


Figure 3.4: High-level overview of the Service Reject denial-of-service

Similarly to Attach Reject denial-of-service, the cause is set to "EPS Services And Non-EPS Services Not Allowed". The result is in line with the Attach Reject denial-of-service attack as the only difference is that we use the Service procedure instead of the Attach procedure.

In this section, we have given an overview diagram for each of the attacks. We can now continue on with the technical description of how we modified the SrsRAN codebase.

3.2 Technical implementation details

This section describes the steps to change the SrsRAN codebase. It does not go into detail for the full changes that were made but instead it gives the most important parts of our implementation and how they tie together.

As can be deduced from Section 3.1.4, the attacks we want to implement all act after the receipt of the RRC Connection Setup Complete message by the eNB.
The code that needs to be changed in order for this to be realized handles the receipt of this RRC Connection Setup Complete message.
We get this to work by parsing the message header in this function and then acting

upon the message type that is passed. For example to check if an Attach Request message has been received, the following code block achieves this:

```
1  // Parse header
2  uint8_t pd;
3  uint8_t msg_type;
4  liblte_mme_parse_msg_header((LIBLTE_BYTE_MSG_STRUCT*)pdu.get(), &pd, &msg_type);
5
6  if (msg_type == LIBLTE_MME_MSG_TYPE_ATTACH_REQUEST)
7  {
8    // Received Attach Request
9    ...
```

Listing 1: Checking for an Attach Request message

In this code branch, we can then execute the Attach Reject attack by first preparing the message and then sending it. Preparing the message is done by using the SrsRAN library functions for packing the message. This is followed by sending the downgraded message. The functions for this are the same functions that are used by the base station when a message is received by the MME that needs to be forwarded to the UE.
While the full code can be found on our Git repository [25], we want to explain the packing procedure as it can be confusing to look at.

Packing the Attach Reject message is done using the following snippet of code:

```
1  LIBLTE_MME_ATTACH_REJECT_MSG_STRUCT attachrej_struct;
2  attachrej_struct.emm_cause           = cause;
3  attachrej_struct.esm_msg_present     = false;
4  attachrej_struct.t3446_value_present = false;
5
6  // Pack the attach reject with the specified struct to the buffer
7  liblte_mme_pack_attach_reject_msg(&attachrej_struct,
   ↪  (LIBLTE_BYTE_MSG_STRUCT*)attach_rejmsg.get());
```

Listing 2: Packing an Attach Reject message

The `esm_msg` attribute is set to `false` as our Attach Reject message does not consist of an *EPS Session Management* (*ESM*) message.
The `t3446_value_present` attribute, which stands for timer 3446 is also set to `false`. In LTE, this timer is set whenever the Attach Reject is sent due to congestion (the cause then is set to `LIBLTE_MME_EMM_CAUSE_CONGESTION` in SrsRAN) [4] [2]. So here, we need to set it to `false`.

Packing the Service Reject message is similar but still different in the values that are passed:

[2] This timer is currently called T3346 [7]

```
1  // Set up the service reject struct using the cause we defined
2  LIBLTE_MME_SERVICE_REJECT_MSG_STRUCT servicerej_struct;
3  servicerej_struct.emm_cause           = cause;
4  servicerej_struct.t3442_present       = false;
5  servicerej_struct.t3446_present       = false;
6
7  // Pack the service reject with the specified struct to the buffer
8  liblte_mme_pack_service_reject_msg(&servicerej_struct,
9  LIBLTE_MME_SECURITY_HDR_TYPE_PLAIN_NAS,
10 0,
11 (LIBLTE_BYTE_MSG_STRUCT*)service_rejmsg.get());
```

Listing 3: Packing a Service Reject message

The timer `t3442` is not present, because it is only started when the Service Reject cause is *#39: CS Service temporarily not available* [4]. The `t3446_present` is set to `false` for the same reasoning as in packing the Attach Reject message.
The `liblte_mme_pack_service_reject_msg`, has more parameters that are not immediately clear as compared to the Attach Reject message.
The `LIBLTE_MME_SECURITY_HDR_TYPE_PLAIN_NAS` indicates that the Service Reject message is a plain NAS message. In other words, this message is not security protected. The `0` argument stands for `count` in SrsRAN. When the MME packs a Service Reject message this is set to `0`, indicating that it is the first downlink NAS message.

The full code for each of our four attacks can be found on our Git repository [25].

4 Testing environments

This chapter contains our approach to setting up the testing environments. We describe two environments: the virtual network setup and the physical network setup. This consists of first giving an overview of these two environments that we want to implement, and secondly going into the technical details to achieve this.

4.1 Overview

In this section, we explain the components in two different testing environments: a virtual network setup, and on the other side: a physical mobile phone coupled with a Software Defined Radio (SDR). The virtual network setup is only used for testing the attacks during development. The actual downgrade attacks and denial-of-service attacks are tested on the physical network setup.
First we go over the virtual network in Section 4.1.1 and then continue with the physical setup in Section 4.1.2.

4.1.1 Virtual network

SrsRAN contains the following components:

- *SrsUE*: Emulates the UE which can connect to an eNB
- *SrsENB*: Emulates the eNB and which communicates with the EPC
- *SrsEPC*: Emulates the EPC which communicates through the SrsENB

These components need to be connected together to form a virtual network. The structure of such a network looks like the following

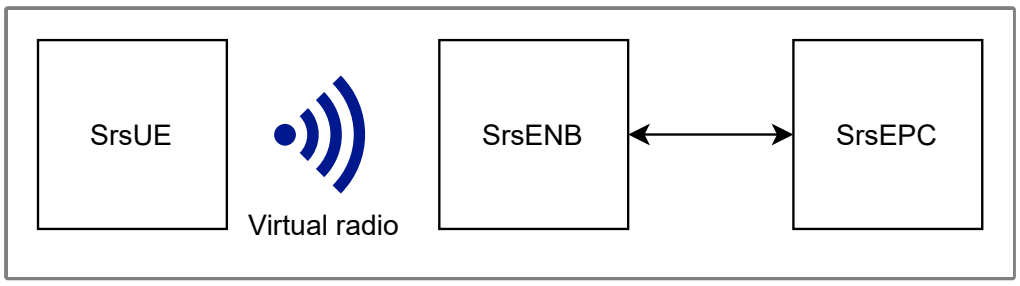


Figure 4.1: Overview of the different components in SrsRAN and how they interact in the virtual network setup

With a virtual network setup we mean a network that is implemented using only software. The difference in the physical setup is that the mobile device is not emulated using software, but is a real UE coupled with a software defined radio. Additionally, instead of using a SDR to transfer radio samples, a virtual radio implemented in software is used.

4.1.2 Physical setup

In the setup with the SDR and the UE, we plan to set up the following network:

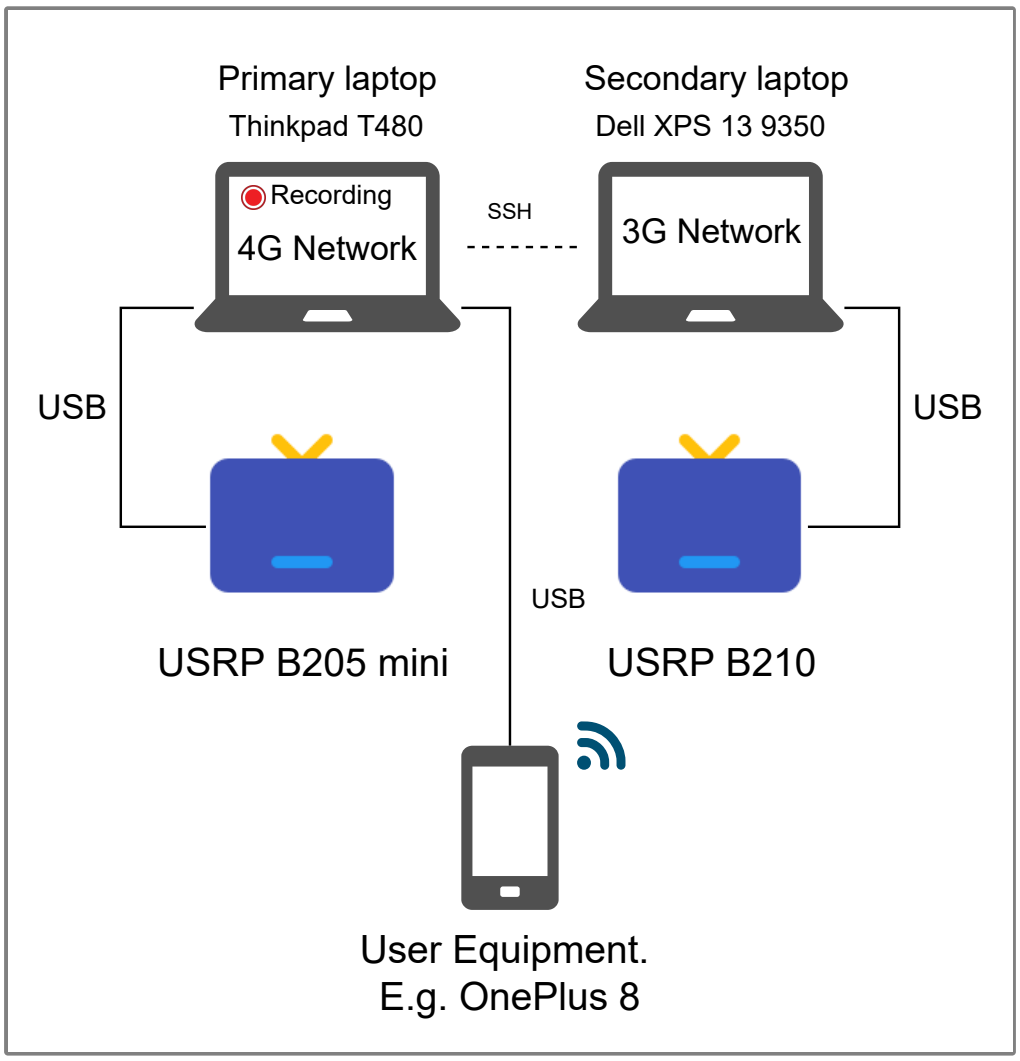
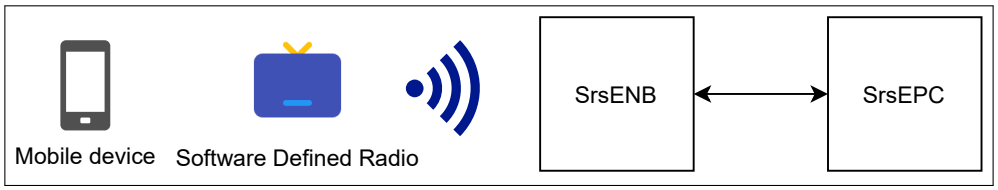


Figure 4.2: Overview of the complete 3G and 4G network setup including the UE and SDR

As can be seen in this figure, we have two laptops, a primary and a secondary laptop. The primary laptop is where we interact with the mobile device, record the screen, and start the 4G network. The secondary laptop only has the 3G network on it and is connected with the primary laptop.

We can thus further split up this figure in the 4G network and the 3G network separately.



(a) 4G network setup

(b) 3G network setup

Figure 4.3: The 4G network setup (a) and (b) the 3G network setup

As can be seen in the figure, to form the UMTS 3G network we use software called *OpenBTS-UMTS* [35]. It is coupled with the rest of our setup in the same way as the 4G network. OpenBTS-UMTS is developed by *RangeNetworks* and is Open Source [45]. The name comes from the fact that it is built upon the OpenBTS software that emulates a 2G GSM network.
The 4G network looks like the network from the virtual network, however, SrsUE is replaced by the mobile device and a SDR.

Note that SrsRAN and OpenBTS-UMTS can be interchanged with any other software that uses 4G[1] and 3G respectively. We choose SrsRAN as it is the most up to date open source 4G network. OpenBTS-UMTS is chosen also for its open-source nature and the fact that this software has clear instructions to set up a network.

4.2 Detailed setup description

This section explains the detailed steps that we take to set up the two testing environments. This is given in the following two sections, where we first go over the virtual network and afterwards continue on with the setup for the physical network. The physical network is a big network so we split it in multiple sections.

4.2.1 Virtual network

To connect the multiple components (SrsUE, SrsENB and SrsEPC) in the SrsRAN network we use the containerization software called *Docker*. Each component in our setup is a separate *container*. These multiple containers are chained together using *Docker Compose* [40]. Our Docker Compose configuration is based on an existing configuration which can be found on GitHub [43]. This image already has the different components of SrsRAN in it. However, there are a few changes that are needed in order to use this with the latest version of SrsRAN. We explain the changes we make to

[1]The 4G network has to be open source as the attacks need to be implemented there which was explained in the previous section Section 3.2

this Docker configuration and in the next section continue with the physical network setup using software defined radios and mobile phones.

The original Docker image [43] is already rather complete. However, it uses an outdated SrsRAN version. The first step we do is update the commit in the Docker file to reference the latest SrsRAN commit. Additionally, the original Docker image uses a mechanism called *FauxRF* to communicate between the SrsUE and the SrsEPC containers. FauxRF emulates the *Radio Frequency* (*RF*) by using shared memory [2] to send messages. This is not sufficient because it relies on a fork of SrsRAN as the original SrsRAN codebase does not have support for this feature [9]. In later SrsRAN versions, a virtual radio is built-in using the *ZeroMQ* [61] library. The further instructions for setting up SrsRAN using ZeroMQ can be found in the documentation of SrsRAN [62]. In essence, we have to create a separate subnet in the Docker Compose file which the UE and the eNB can use to send messages over the *Transmission Control Protocol* (*TCP*) protocol using ZeroMQ.
The changed Docker image and associated scripts can be found on our GitHub [24].

4.2.2 Physical network

In this section, we go over the setup that was used to test our downgrade attack with a physical mobile phone.
Unlike SrsUE, the mobile phones that we test both have support for UMTS and LTE. To test whether these UEs can still connect to UMTS networks after the downgrade attack, we also need to set up a 3G UMTS network.

4.2.2.1 Setting up the mobile devices

As the goal is to test our attacks for different LTE revisions, we need to test it on devices that that support different 3GPP releases. The support of a 3GPP release is based on the baseband processor which is part of the modem on the mobile devices. This modem can be integrated into the CPU chip or be a separate entity. To choose the mobile devices, we thus look for differences in the CPU and/or modem that is used. In the end we chose the following mobile devices:

Phone	Android Version	Central Processing Unit (CPU)	3GPP Release
OnePlus 8	11	Qualcomm Snapdragon 865	15 revision 7
Xiaomi Mi 9T Pro	11	Qualcomm Snapdragon 855	14 revision 6
OnePlus 3	11	Qualcomm Snapdragon 820	11 revision 3
Nexus 5	6	Qualcomm Snapdragon 800	9 revision 1

Table 4.1: The different mobile devices for our physical network

As can be seen in the table, we also noted the exact 3GPP revision that is used on these devices. To find this out, we use traffic information that is generated by these devices. More specifically, the mobile devices send a *UECapabilityInformation* message to a network which includes a field called *accessStratumRelease*. This field lists the exact 3GPP release and revision. The details of this message for each of the four devices, can be found in the appendix (Section 8.2).

[2] In shared memory a memory buffer is created between the different containers which they can use to pass messages

Capturing these messages is achieved by using the *Qualcomm Diagnostic Mode* which can be enabled by using the following set of commands in an *Android Debug Bridge* (*ADB*)[3] shell:

```
[adb shell] $ su -
[adb root shell] $ setprop sys.usb.config diag,adb
```

Listing 4: ADB shell commands to enable Qualcomm Diagnostic mode using root and ADB

As can be deduced from the `su -` command, these phones need root access in order to enable this Diagnostic mode. This is achieved by flashing *Magisk* [57].
While we have enabled this Diagnostic mode, we still need a way to find and parse messages from this mode so that we can view it in a more human-readable matter. A tool to do this, is a *Python* [44] script called *Scat* [13]. It has a command line interface to save the traffic information to a file:

```
$ python3 scat.py -t qc -u -a "001:021" -i 0 -F phone.pcap
```

Listing 5: Shell command to capture traffic with scat as output file phone.pcap

For an explanation of these options, we refer to the appendix at Section 8.4.
This method of capturing is also used in our experimental testing to verify if the messages that are sent are what we expect them to be.

Another tool that we use is Scrpcy [14]. This tool can be used to mirror the phone's screen on the host computer for controlling and recording it conveniently. After we install Scrcpy using a package manager, all four mobile devices' screens are able to be mirrored on the primary laptop by invoking the `scrcpy` command in a terminal when the phone is connected to the laptop and ADB is allowed to access it.

As we have explained how we set up the mobile devices with the necessary tools, we can go over how we managed to set up the SrsRAN 4G network with the Software Defined Radio in the next section. Before we can do that, however, we still need to explain how these mobile devices can discover and connect to our own network. This involves configuring these mobile devices with the identifiers for our own SrsRAN network. To do this, we needed to edit the access point settings in the SIM card [4]. In these access point settings, the most important values are the MCC and MNC identifiers for the network. These values need to be the same as our own SrsRAN network, which is explained in the next section. This allows the phone to start the Attach procedure to the network.

4.2.2.2 Setting up the 4G Network

Setting up the 4G network for the physical network involves a way to communicate with the mobile device through our software defined radio. Looking back at Figure 4.1, the change that we need to make is to remove the SrsUE component and replace it with the SDR and mobile device. The SDR that we use for the 4G network is the *ETTUS B205-mini* which needs the *USRP Hardware Driver* (*UHD*) to function. This is supported by

[3] ADB is used to communicate with an Android device [15]
[4] See the appendix for a full screenshot for the settings on the OnePlus 8 (Section 8.3).

changing the Docker image that we mentioned previously to exclude the SrsUE software component and add support for the B205-mini using UHD. The changed Docker image can be found on the *sdr-uhd* branch of our GitHub repository [24].
However, during testing we do not use this Docker image as build times are slower than without using Docker. Making modifications to the code, such as switching the Git branches of our SrsRAN attack implementations, then results in a long wait time. As we have limited available time to use the SDRs in the tests, we have also used scripts to set up this network without Docker. These can be found on our GitHub as well [22]. To easily set up this network, however, we recommend to use the Docker image.
The configuration files we use for SrsRAN are based on the default configuration files. For example, the MCC and MNC identifiers that we configured for our mobile devices are the same values as the default configuration file: `001` and `01` respectively. Our complete configuration files can be found on our GitHub [22].
Additionally, we need a way for UE devices to be able to connect to the internet through the laptop's wireless card interface (similarly to the Docker image). The following command achieves this in the root directory of the SrsRAN codebase:

```
$ sudo ./srsepc/srsepc_if_masq.sh networkcard
```

Listing 6: Shell command ran in the SrsRAN project root directory to masquerade the interface denoted as `networkcard`

For internet access, another step needs to be taken. SrsRAN has a database file with credentials (such as the IMSI) of the phones that are allowed to fully connect to the network. For a basic Attach Request this was not needed, but for our methodology of testing (see Chapter 5) and the Service Request procedure this is required.

4.2.2.3 Setting up the 3G Network

Another component that we need in the physical network setup is a 3G network. This network is run on a separate laptop that is connected to the laptop that runs the 4G network. Connecting these two laptops is set up using the *Secure Shell* (*SSH*) protocol [37].
While setting up the 3G network on the secondary laptop we discovered a problem. The OpenBTS-UMTS software that we use is currently not in active development. And additionally, it does not support the latest UMTS/LTE type of SIM card [35]:

> *USIMs (e.g. 3G SIMs) are not currently supported by the OpenBTS-UMTS implementation. They require different authentication algorithms than GSM SIMs; these algorithms are not supported in the public release of OpenBTS-UMTS.*

However, the only aspect we needed from this 3G network is that it is able to pick up Attach Request/RRC messages from the UE. Full internet access is not needed as the only thing we want to test is if the UE is still able to send messages to the 3G network after the downgrade attack has been initiated.
Due to the lack of development, it is difficult to get this software working on modern versions of compilers. Docker is once again used to set this software up in a reproducible manner. The resulting Docker image is published on our GitHub [23].

4.2.2.4 Final setup

This setup section concludes with a diagram that shows the complete setup with all the hardware.

Figure 4.4: Camera picture of the complete 3G and 4G network setup including the OnePlus 8

In the next chapter, we explain the testing procedure. This procedure is executed on this setup in a conference room at the Radboud University where we got permission to work.

5 Testing procedure

In this chapter, we describe the testing procedure. With the "testing procedure", we mean the exact steps to execute our tests.

As we have implemented four downgrade attacks, our plan is to test each mobile device against each of these four attacks. In total, we thus have 16 tests. To reiterate, the downgrade attacks we test for each device is the following:

- Attach Reject with cause: "EPS Services Not Allowed" (downgrade to 3G)
- Service Reject with cause: "EPS Services Not Allowed" (downgrade to 3G)
- Attach Reject with cause: "EPS Services And Non-EPS Services Not Allowed" (denial-of-service)
- Service Reject with cause: "EPS Services And Non-EPS Services Not Allowed" (denial-of-service)

To give a clear overview of the steps to be executed during testing, we present a flowchart for each of these attacks in the following sections. Additionally, we also explain why we chose the steps that are depicted in the flowchart. Each of these tests will be repeated multiple times to make sure that we do not get a downgrade or denial-of-service that worked because of external variables, such as the SDR or software failing.

5.1 Attach Reject downgrade

The attack which downgrades the LTE connection using the Attach Reject consists of the following testing steps:

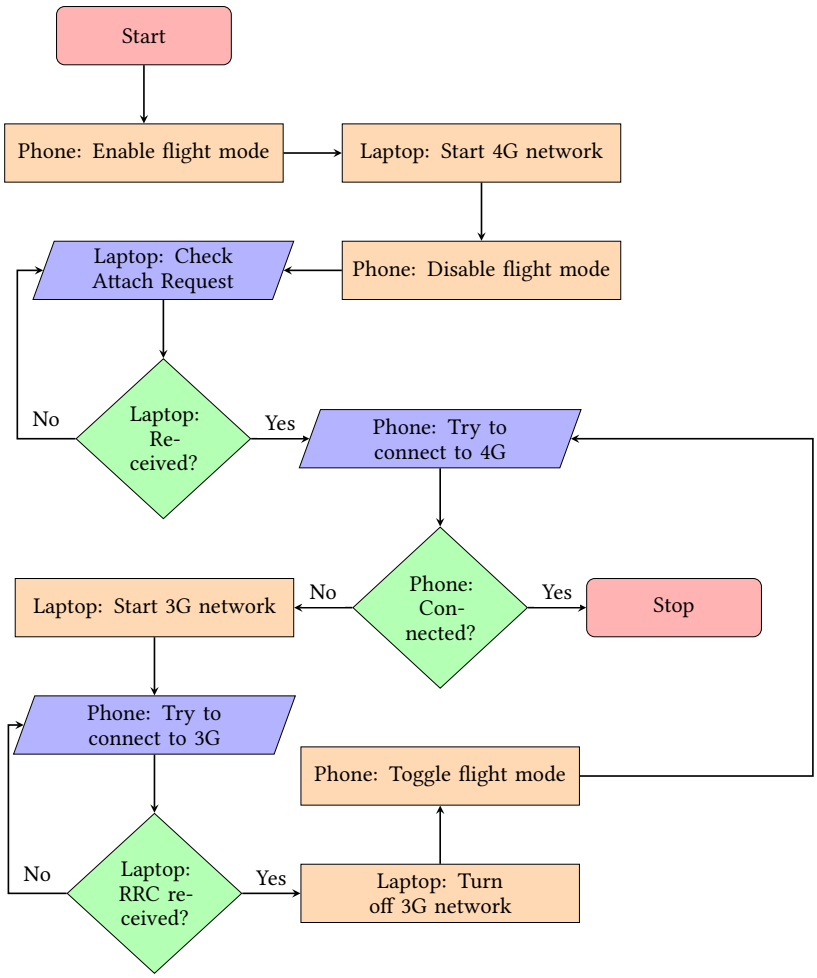


Figure 5.1: Attach Reject downgrade flowchart

We start the test by making sure that flight mode is on so that the mobile device does not attempt a connection to any mobile network. When the 4G network on the laptop has been started, we can disable flight mode such that the device connects to the malicious 4G network. When it has tried to connect to the malicious network, the mobile device sends an Attach Request message that is responded with the Attach Reject including the downgraded cause. If the attack has succeeded, the phone is unable to connect to the 4G network. However, on the contrary, when it does connect after the Attach Reject message has been received, we stop the test as that means that the mobile device is not in a downgraded state which indicates that the test has failed. If it does downgrade, we try to connect to the 3G network. Note that OpenBTS-UMTS does not fully support our mobile device due to the SIM card, so we only watch out for the receipt of the RRC messages by the laptop. When these messages have been received, the 3G network can be turned off, and we toggle flight mode again to test if the mobile device can regain control to connect to the 4G network. Toggling flight mode is done here as in the related work it stated that most mobile devices are able to regain control after flight mode has been toggled [48]. If toggling flight mode does not work we resort to re-inserting the SIM card. This also holds for the tests in the following sections.

5.2 Attach Reject denial-of-service

The next Attach Reject attack is a denial-of-service attack with the following steps:

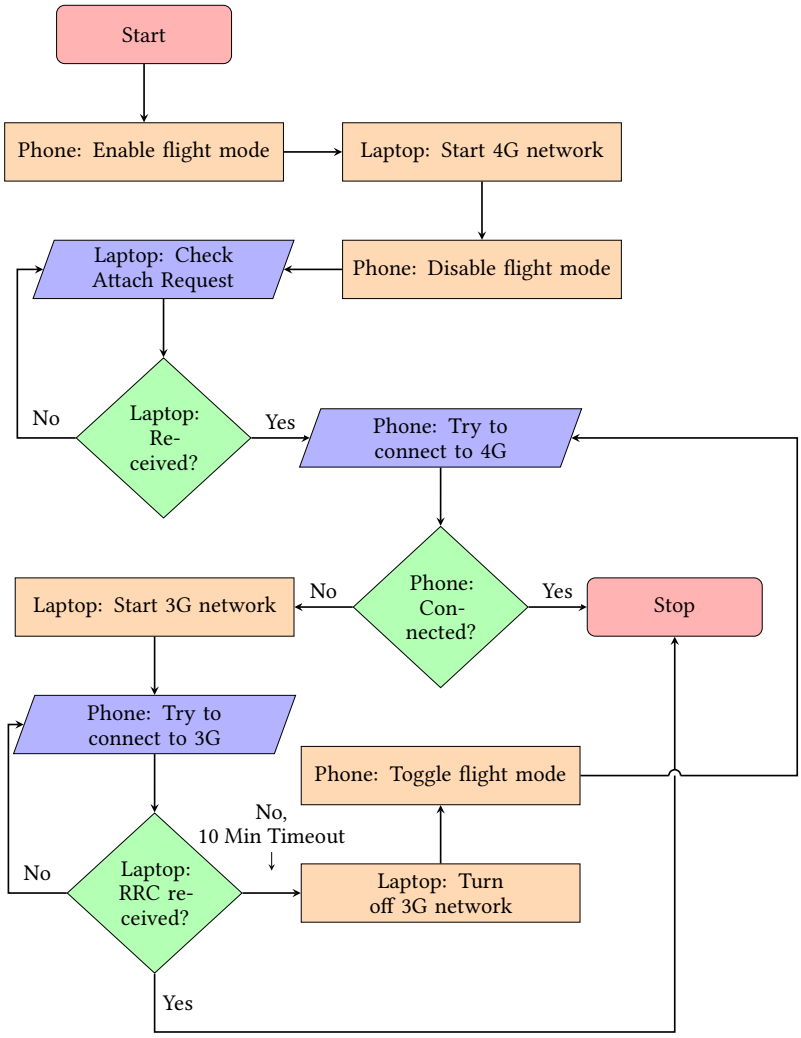


Figure 5.2: Attach Reject denial-of-service flowchart

The Attach Reject denial-of-service attack has a similar execution as the previous Attach Reject test. The only difference is that we are not able to connect to the 3G if the attack succeeds. It can be seen in the flowchart that when we do get RRC messages we stop the test as this means that this device does not react on the denial-of-service the way we expected it would. The phone is often immediately connects to the 4G network as the SrsRAN software is rather stable in our testing. The 3G network, however, often fails to immediately establish a connection with the mobile devices. This is why we wait 10 minutes in the test so that we can verify whether the phone can actually connect to the 3G network within this timeframe. After this timeframe has elapsed, we assume that the phone cannot connect to the network, and we continue by toggling flight mode. The timeframe of 10 minutes is chosen due to the fact that in our testing, the phone would not immediately connect to the 3G network. This took

at maximum a couple of minutes.

5.3 Service Reject downgrade

The downgrade attack which uses the Service Reject message has the following flowchart:

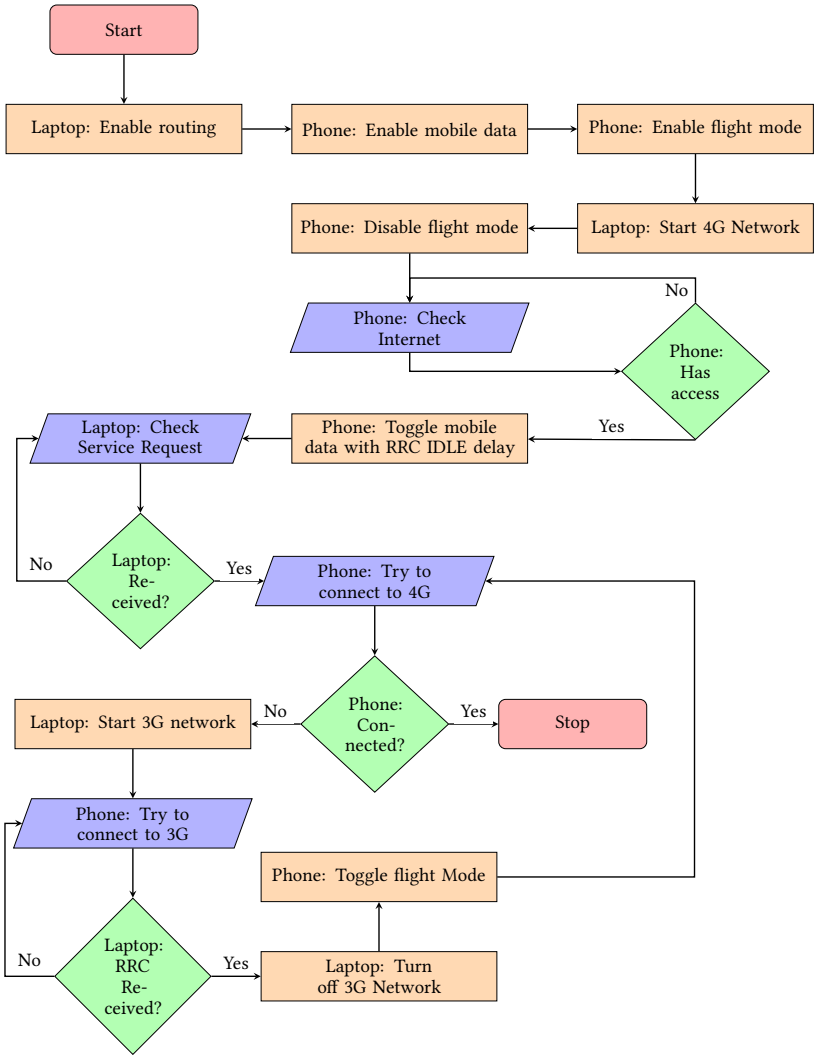


Figure 5.3: Service Reject downgrade flowchart

This attack is most similar to the Attach Reject downgrade attack as it has the same specified cause of "EPS Services Not Allowed". However, the way the Service Request message is triggered is different from the Attach Request message.
The way we test the Service procedure, it needs the phone to be able to connect to the internet. This needs the IMSI configured in the SrsRAN's user database file and the Access Point in the phone's settings. Additionally, which can be seen in the first step with the label "Enable routing", we need to masquerade the laptop's internet interface in order for the phone to actually get the internet connection from the laptop through the SDR.

When the device is inactive on the network, the phone enters RRC IDLE state. When we then reconnect to the network, the device sends a Service Request message. This is why we "toggle mobile data with RRC IDLE delay". This essentially means that we first disable mobile data, then wait for RRC IDLE, and then enable mobile data again to trigger a Service Request. The RRC IDLE state can be observed in the SrsRAN logs.

5.4 Service Reject denial-of-service

The last attack, denial-of-service using the Service Reject message has the following execution:

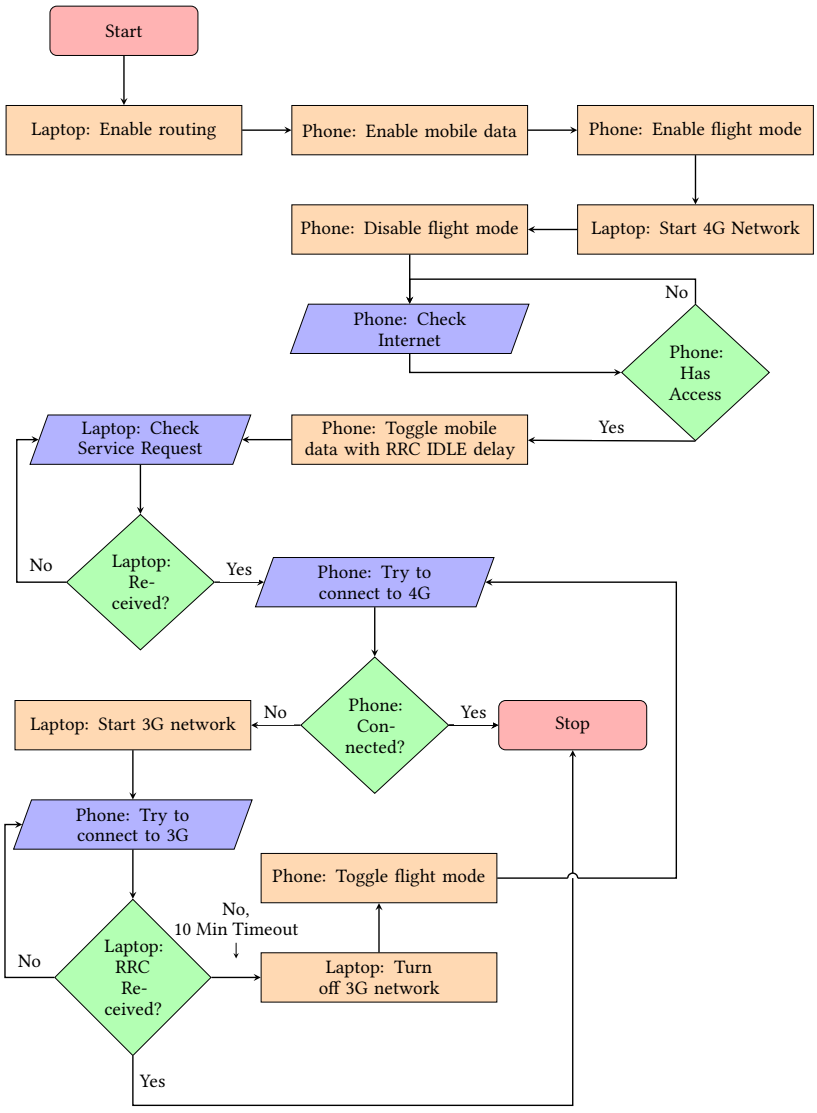


Figure 5.4: Service Reject denial-of-service flowchart

We can see that the changes from the Service Reject downgrade attack to the Service Reject denial-of-service attack is similar to that of the Attach procedure. The same

timeout of 10 minutes is used to verify that we really cannot connect to the 3G network.

We have now given an overview of the steps in our tests. The following chapter, Chapter 6, includes an analysis of these attacks using the 3GPP releases, presents the results and discusses these attacks and results.

6 Attack Analysis

In this chapter, the goal is to analyse the attacks on the different mobile devices of our physical network setup. To do this, we first build a hypothesis on what we expect to happen when our testing procedure is executed on these devices. Afterwards, we present the outcome of our downgrade and denial-of-service attacks. In the last section of this chapter, we discuss the outcome of these attacks. Additionally, in this discussion section we analyse how practical these attacks are and give possible countermeasures that can be implemented in upcoming mobile network standards.

6.1 Hypothesis

This section builds the hypothesis for the experiments of the downgrade and denial-of-service attacks. We do this by first looking at the different 3GPP releases of the mobile devices that we test on. These 3GPP releases give a theoretical overview of what these devices *should* be doing. Whether they actually do this is not discussed here, this is analysed in the next section where we present the results.

6.1.1 Differences between 3GPP releases

This section further expands on the analysis of the attacks based on the various 3GPP releases. In Section 3.1.3, we already went over the problematic nature of these attacks[1] and how this can be found in the 3GPP releases. While this is a good starting point, we have not yet given a breakdown on the differences between the 3GPP releases. The differences that are important are the ones that influence the behaviour of the UE when it receives a downgrade or denial-of-service message. If there are no differences, then testing on various devices would not be so interesting any more.

In total, we could find two differences that have an impact on the handling of the downgrade or denial-of-service messages. These are discussed in two separate sections, where the structure of these sections is that we first give a high-level overview of the differences and then do a deep dive into 3GPP change request documents. These change request documents consists of changes between 3GPP revisions where further motivation is given [1].

[1] Recall that the problematic nature of these attacks is that there is no integrity protection nor are they authenticated between the UE and the eNB

6.1.1.1 Improvements in handling with later 3GPP releases

The first difference is that newer 3GPP releases have timers against the downgrade and denial-of-service attacks that we implemented. This is already highlighted in the literature by e.g. Jover [21].
By analyzing the release documents, we find that this is indeed the case: starting from 3GPP release 13.04 timer T3245 is used in the handling of downgraded messages [6]:

> *The UE shall consider the USIM as invalid for EPS services [and non-EPS services] until switching off or the UICC containing the USIM is removed* ***or the timer T3245 expires as described in subclause 5.3.7a***

However, the value of this timer is set to 12 to 24 hours and is started when such a downgrade message is received [5]. When it expires, the UE acts normally again. In older 3GPP revisions, this timer is set to 24h to 48h where it is not used in handling of these downgrade messages. It is lowered in 3GPP release 13.3.0 [5] to be more in line with the timer that is used for "forbidden location areas" [5]. It would be interesting to find out how accurate and useful this 12 to 24 hours is in practice.

6.1.1.2 Service Reject denial-of-service not supported

In this section, we highlight another difference we notice between the 3GPP releases, namely, the lack of support for the handling of the Service Reject procedure in the denial-of-service attack.

When we look at 3GPP release 14.3.0, we have the following paragraph for the handling of cause 8, EPS Services and Non EPS Services Not Allowed in the section *Service request procedure not accepted by the network*:

> *The UE shall consider the USIM as invalid for EPS services until switching off or the UICC containing the USIM is removed or the timer T3245 expires as described in subclause 5.3.7a. The UE shall enter the state EMM-DEREGISTERED.*

However, when we take a look at an earlier release such as the 3GPP release 9 standard, we see no such paragraph in the same section. We can trace the change back to 11.4.0 [8], where the reasoning for adding it is the following:

> *It has specified in TS 24.301 that the normal (not combined) EPS attach request can be rejected with cause #8 "EPS services and non-EPS services not allowed". This may happen when a UE initiated such normal attach to a vPLMN but the vPLMN is not allowed to operate either EPS or non-EPS services for this UE.*
>
> *The similar case can also happen for the normal TAU procedure. When a EPS attached UE (in EMM-REGISTERED state) moves into a new vPLMN and this vPLMN is not allowed to operate either EPS or non-EPS services for this UE, then the TAU request will also be rejected with cause #8 "EPS services and non-EPS services not allowed". Note that in the case of inter-PLMN mobility, the TAU procedure is initiated for initial registration to the new PLMN (i.e. in this case, the TAU is more liked as the EPS attach).*

In both of the 3GPP (early) release 11 and 9, the receipt of the Service Reject cause 8 by the UE is classified and handled as an abnormal case [8]. The reasoning for the change further states the following of why this is problematic:

> *The handling as an abnormal case is dangerous due to the UE will re-try for at least five times to the same PLMN and will always be rejected with cause*

#8. The unnecessary signalling is created and wastes the UE battery.

Whether this is indeed what happens in our results for the devices with earlier 3GPP releases, remains to be seen in Section 6.2 where we present the results. Before we can do that, however, we need to take a look at what we expect to happen using the information we gathered.

6.1.2 Expected result

As we have gone over the theoretical differences between our mobile devices, we can now look at how we expect that these devices react to the attacks that we test.

In the next section where we present the results, we list the attack outcome for each device and how these phones could potentially recover from this. The *expected* result is the following for the downgrade and denial-of-service attacks:

Attack	Outcome	Recovery
Attach Reject/Service Reject downgrade to non-LTE	Cannot connect to our 4G network	Toggle flight mode
Attach Reject/Service Reject denial-of-service	Cannot connect to our 4G/3G network	Toggle flight mode

Table 6.1: Expected result for the OnePlus 8 and the Xiaomi Mi 9T Pro

As these devices have a high 3GPP release (14 and 15), they support all procedures that are used in the attacks. In theory, the only LTE component that can stop these attacks in this release is a timer. However, as we noted in the first 3GPP release difference (see Section 6.1.1.1), it is set to a value between 12 and 24 hours so this should not cause an issue.

The last two phones, the OnePlus 3 and the Nexus 5, both have older 3GPP releases than the previous two phones. The Nexus 5 has a release where the Service Reject with the cause for our denial-of-service attack is not supported (see Section 6.1.1.2). The same holds for the OnePlus 3 as it has 3GPP release 11 with revision 3, which is a revision below the introduction of the Service Reject handling with cause "EPS Services and Non-EPS Services Not Allowed".

Attack	Outcome	Recovery
Attach Reject/Service Reject downgrade to non-LTE	Cannot connect to 4G	Toggle flight mode
Attach Reject denial-of-service	Cannot connect to our 4G/3G network.	Toggle flight mode
Service Reject denial-of-service	The Service Reject denial-of-service handling is unsupported. Keeps sending Service Request	-

Table 6.2: Expected result for the OnePlus 3 and the Nexus 5

The results from our tests have similar columns for each device: *Outcome* and *Recovery*. For example, if a phone is downgraded from 4G to 3G, and it can recover by toggling flight mode, this is then filled into these two columns. However, if the outcome is that the phone can still connect to 3G and 4G, the downgrade has not worked and there is no need to recover from this. The *Recovery* column is then filled with a hyphen (-). Note that we have filled in *Toggle flight mode* for each device on this column. This is because Shaik et al. [48] stated the following:

> *UEs having baseband from most vendors can recover by toggling the flight mode*

If a device cannot recover using this flight mode it deviates from our hypothesis.
For each attack, we film the screen of the laptop and the mobile phone. These recordings are edited into a demo for some scenarios. Which exact scenarios these are, is explained in the following sections. The final demo videos are published on our *YouTube* channel [59].

6.2 Results

This section contains our results for each test. Each test has a separate section where we go over these results by giving a table that is similar to our hypothesis. Additionally, we present a table with every test and outcome combined. A further in depth discussion of the results is given in Section 6.3.

6.2.1 Attach Reject downgrade

The first test, downgrading a connection using an Attach Reject message and cause "EPS Services Not Allowed", has the expected result on every device we test it on:

Phone	Outcome	Recovery	Recorded	Expected
OnePlus 8	✓	✈ or 🖴	Yes	Yes
Xiaomi Mi 9T Pro	✓	✈ or 🖴	No	Yes
OnePlus 3	✓	✈ or 🖴	No	Yes
Nexus 5	✓	✈ or 🖴	No	Yes

- ✓ means that the attack works, the phone cannot connect to our 4G network
- ✈ means that the phone can recover by toggling flight mode
- 🖴 means that the phone can recover by re-inserting the sim or rebooting the phone

Table 6.3: Attach Reject downgrade outcome

This result is also consistent when we test the attack multiple times: the phones are unable to reconnect to the 4G network up until flight mode is toggled. In the meantime, the 3G network receives RRC messages but not a full connect. As we have explained before (see Section 4.2.2.3), this is because our SIM card (an USIM) is not fully supported by OpenBTS-UMTS. We expect the phone to be fully able to connect to 3G networks or below when this network supports the phone's SIM card.
The demo video for this test is for the OnePlus 8. This can be seen by the "Recorded" column in the table.

For this test and in every next test we make a demo video for this device as it has the highest 3GPP release out of all the devices we test on. This means that, in theory, it should be the most resilient against these attacks. If we have an interesting result, meaning one that that deviates from our hypothesis, we make a demo video to highlight how the outcome differs. We make sure to mention this in the next sections which exact scenarios can be watched on the YouTube channel [59].

6.2.2 Service Reject downgrade

The next test, downgrading the mobile phones using the Service Reject message upon the Service Request has a similar result to the previous test.

Phone	Outcome	Recovery	Recorded	Expected
OnePlus 8	✓	✈ or 🖴	Yes	Yes
Xiaomi Mi 9T Pro	✓	✈ or 🖴	No	Yes
OnePlus 3	✓	✈ or 🖴	No	Yes
Nexus 5	✓	✈ or 🖴	No	Yes

- ✓ means that the attack works, the phone cannot connect to our 4G network
- ✈ means that the phone can recover by toggling flight mode
- 🖴 means that the phone can recover by re-inserting the sim or rebooting the phone

Table 6.4: Service Reject downgrade outcome

Toggling flight mode is yet again the solution to make the phones able to connect to the 4G network again. To verify that the 4G network is not at fault, we also try to reboot the 4G network off camera to test whether the phones could reconnect again. The same is done for the previous test.
There was no odd outcome in this attack scenario so we only recored the demo for the OnePlus 8 again.
As we have now briefly presented the results of the two downgrade attacks, we continue with the results for the denial-of-service attacks. These are presented similarly in the next two sections.

6.2.3 Attach Reject denial-of-service

The denial-of-service attack that sends an Attach Reject message upon the receipt of the Attach Request message has the following result:

Phone	Outcome	Recovery	Recorded	Expected
OnePlus 8	✓	✈ or 🖴	Yes	Yes
Xiaomi Mi 9T Pro	✓	✈ or 🖴	No	Yes
OnePlus 3	✓	🖴	Yes	No, cannot recover with ✈
Nexus 5	✓	🖴	No	No, cannot recover with ✈

- ✓ means that the attack works, the phone cannot connect to our 3G and 4G networks
- ✈ means that the phone can recover by toggling flight mode
- 🖴 means that the phone can recover by re-inserting the sim or rebooting the phone

Table 6.5: Attach Reject denial-of-service outcome

Here we have an interesting result. We find that the OnePlus 3 and the Nexus 5 cannot recover by toggling the flight mode on the device. This was not part of our hypothesis as we do not know in which exact scenarios flight mode helps to recover from the attack. On top of the demo video for the OnePlus 8, a demo video for the OnePlus 3 is recored as it has a higher 3GPP release than the Nexus 5.
To connect to the networks again we had to re-insert the SIM card for these phones or do a full reboot of the devices. The OnePlus 8 and the Xiaomi Mi 9T pro had the expected result, the phones cannot connect to our 3G and 4G networks and they can recover by toggling flight mode, re-inserting the SIM card or rebooting the phone.

6.2.4 Service Reject denial-of-service

The last attack, the Service Reject denial-of-service attack upon the receipt of the Service Request message has the following outcome:

Phone	Outcome	Recovery	Recorded	Expected
OnePlus 8	✓	✈ or 🖴	Yes	Yes
Xiaomi Mi 9T Pro	✓	✈ or 🖴	No	Yes
OnePlus 3	✗	-	Yes	Yes
Nexus 5	✗	-	No	Yes

- ✓ means that the attack works, the phone cannot connect to our 3G and 4G networks
- ✗ means that the attack does not work, the phone keeps sending a Service Request message
- ✈ means that the phone can recover by toggling flight mode
- 🖴 means that the phone can recover by re-inserting the sim or rebooting the phone

Table 6.6: Service Reject denial-of-service outcome

In this table, we see that the OnePlus 3 and the Nexus 5 indeed do not support the handling of this Service Reject message with cause "EPS And Non EPS Services Not Allowed". Instead of handling this message, the phones keep sending the Service Request message up until the network is shut off. On top of the demo video that we always make for the OnePlus 8, we decide to also make one for the OnePlus 3 as it has the highest 3GPP release that does not support our denial-of-service attack.

6.2.5 Overview

We have given a separate table for each attack in the previous sections. However, this might not be a clear overview of the final results. For this, we present a table that summarizes the results:

			Downgrade to non-LTE		*Denial-of-service*	
Phone	**CPU**	**3GPP Release**	**Attach Reject**	**Service Reject**	**Attach Reject**	**Service Reject**
OnePlus 8	Qualcomm Snapdragon 865	15.7	✓✈	✓✈	✓✈	✓✈
Xiaomi Mi 9T Pro	Qualcomm Snapdragon 850	14.6	✓✈	✓✈	✓✈	✓✈
OnePlus 3	Qualcomm Snapdragon 820	11.3	✓✈	✓✈	✓	✗
Nexus 5	Qualcomm Snapdragon 800	9.1	✓✈	✓✈	✓	✗

- ✓✈ means that the attack works and the phone can recover by toggling flight mode, re-inserting the SIM or rebooting the phone
- ✓ means that the attack works and the phone can recover by re-inserting the SIM or rebooting the phone
- ✗ means that the attack does not work and the phone keeps sending a Service Request message

Table 6.7: Overview of our downgrade and denial-of-service attacks based on the Attach Reject and Service Reject messages

To expand upon these results we reflect on them in the next section.

6.3 Discussion

In this section, we reflect on the results, take a more detailed look at the attacks by discussing how practical they are, and consider possible countermeasures for the attacks. This is discussed in the following sections. In the next chapter (Chapter 7), we wrap up this thesis by explaining what we have done and what is left for future work.

6.3.1 Reflecting on the results

In the previous section, we have seen the outcome for our downgrade and denial-of-service attacks for the different devices. We believe this gives a good overview that has not been done in previous research. Most notably, we deduce from these results that toggling flight mode does not always work to recover from the attacks. While it is briefly mentioned in *Practical attacks against privacy and availability in 4G/LTE mobile communication systems* [48] that not all devices have support for flight mode recovery, they do not explicitly mention which devices these are. We find that in the specific case of a Attach Reject denial-of-service, the Nexus 5 and OnePlus 3 need to be reset by reinserting the SIM card or rebooting the phone. This is an interesting

result as this shows that toggling flight mode is not a reliable way to recover from all denial-of-service attacks.
Additionally, we find that a device using 3GPP release 15 (the OnePlus 8) has no notable countermeasures against these attacks. While in theory it could recover from these attacks with a timer that is set between 12 and 24 hours, we did not test for this long to notice whether this timer is properly configured. We also think this timer is too high to be really useful.
For the Service Reject denial-of-service attack, the handling is unsupported for devices using 3GPP release 11.4.0 and below which we could find in the standards and also observe during our tests. Instead of re-trying for at least five times, as was noted in the specification (see Section 6.1), our Nexus 5 and OnePlus 3 keep sending the Service Request message up until the network is shut down. This can be very problematic for these devices, for example, this attack could be used to attack the victim's battery.

6.3.2 Usage in practice

This section contains the description of how useful these attacks are in practice for an adversary. We provide the steps that are needed for an attacker to execute these attacks. We also relate them to several other attacks such as IMSI catchers and explain how this can be used to create targeted attacks. Additionally, we have a section about *signal overshadowing* attacks that have an alternate approach to attacks based on fake base stations. We also explain how this signal overshadowing can be combined with the attacks that we present in this thesis.

6.3.2.1 Adversary difficulty

While we have talked about the fact that these attacks are problematic, we have not yet given an overview of how difficult these attacks are to execute for an adversary. To explain this, we take a look at the steps that an adversary needs to take to employ these attacks and go over possible challenges that come with it.

In order for an adversary to execute these attacks, it needs roughly the following components:

1. A rogue base station for which the UE is more willing to connect to than surrounding base stations
2. In case of a targeted attack: IMSIs for each victim's SIM card
3. Malicious code to run on the base station that executes the attacks with optionally for the provided IMSIs
4. Trigger an Attach Request or Service Request by the victim's UE depending on the attack

While this is not an exhaustive list, it gives a general impression what the adversary needs in order to execute the attacks in practice.

We assume the most difficult step for an adversary is step 1. A way for an adversary to build this rogue base station is to create one that has higher power than the surrounding legitimate base stations. However, the paper *Practical Attacks Against Privacy and Availability in 4G/LTE Mobile Communication Systems* which was mentioned in the introduction (Chapter 1) [48], notes that this can be problematic. If a mobile device is in close proximity to the base station it is currently connected to, the serving eNB, it does not scan other surrounding base stations. The authors further noted that it is possible to circumvent this problem by using *absolute priority based cell reselection.* While

these UEs do not scan all base stations when they are close to the serving eNB, they do periodically connect to base stations that operate on high priority frequencies [48]. The adversary can thus exploit this priority system by looking for these priorities and then configuring the rogue base station with them. This, however, is an extra step, and it remains to be seen in practice how difficult this is.
Another difficulty is that we need this rogue base station to be able to impersonate a real base station such that the phone actually tries to connect to the base station. Shaik et al. further states that this is possible by broadcasting the same MCC and MNC identifiers as the real network operator's base station. Note that in our experimental set up we did not do this. For the mobile device, we preconfigured our mobile network into the SIM card access point names settings menu.
The second category, obtaining the IMSIs for the victims, can be accomplished by using an IMSI catcher. A challenge that we can see though is that if the attacker wants to target a specific individual, it might be hard to accurately get the IMSI. This is because in an IMSI catcher it is possible that we catch multiple IMSIs as there can be more than one mobile device within the cell. Eliminating the IMSIs we do not need then makes further analysis necessary.
The third component, is not hard for an adversary to obtain. As noted before, there are multiple attack implementations in the OpenLTE [29, 30, 31] software stack. Additionally, as we described in the technical implementation section (Section 3.2), the changes that need to be made to a different open source software stack such as SrsRAN are not substantial. Even if a motivated attacker does not have access to any implementation, we think this would not be an obstacle for it.
At last, the adversary needs a way to trigger the accompanying Request message from the UE. For the Attach Request message this is trivial as it is sent to establish a connection with the fake base station. The Service Request message is sent when the device has data to forward to the base station but is in RRC IDLE state. However, an additional way to force the sending of a Service Request message is to initiate the Paging procedure [18]:

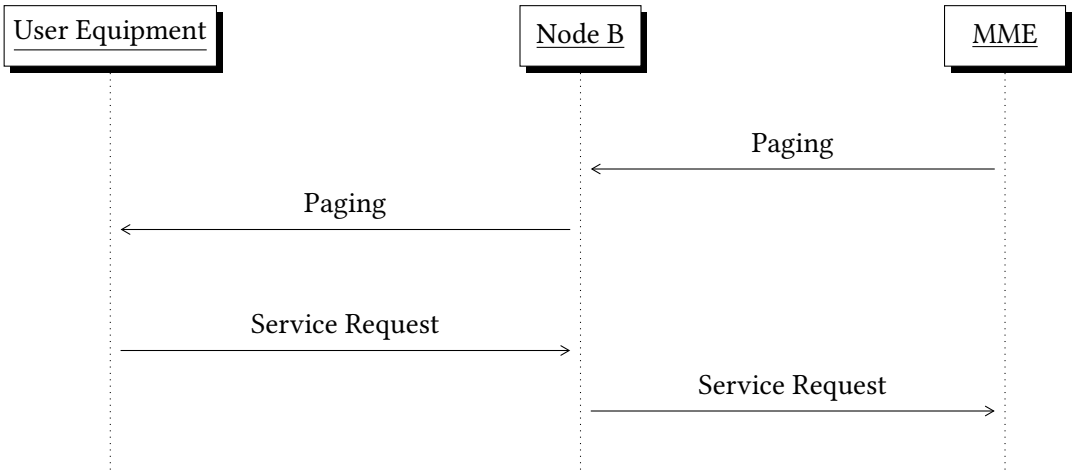


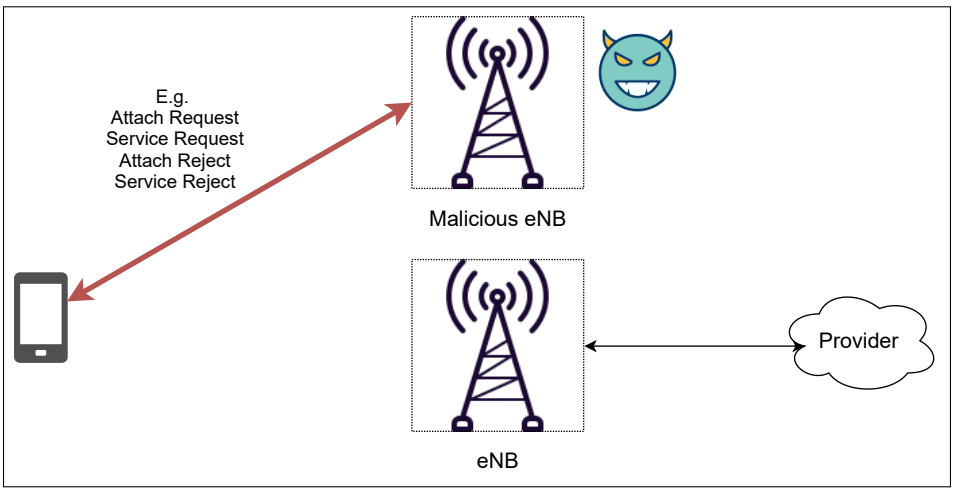
Figure 6.1: LTE Service Request send by the UE due to Paging

With Paging, in practice, the base station has to send a Paging message to the UE and the UE will send an integrity protected and encrypted Service Request message back to the base station. This Paging is used to instruct the UE to connect again as the network has data to send to it.

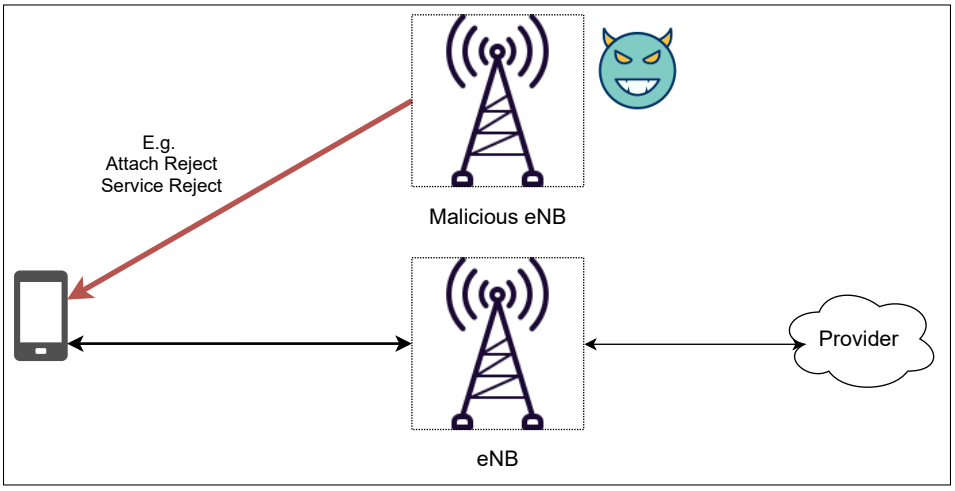
6.3.2.2 Signal overshadowing

We noted earlier that it can be problematic to set up a rogue base station. Thus, it is not always possible to employ these attacks using the fake base station approach. There is, however, a different method we can use to employ the downgrade and denial-of-service message which is highlighted in this section.

As certain messages (e.g. broadcast, Attach Reject and Service Reject messages) are accepted without integrity protection in LTE, we can use a technique called *signal overshadowing* to alter the messages that are sent. The paper that introduces this overshadowing is called *Hiding in Plain Signal: Physical Signal Overshadowing Attack on LTE* by Yang et al. [60]. The difference between a signal overshadowing attack and an attack with a fake base station is the following:



(a) Fake base station



(b) Signal overshadowing

Figure 6.2: Fake base station (a) vs Signal Overshadowing (b) in a downgrade setting adapted from [60]

This has the advantage that in a signal overshadowing attack, the adversary does not

need the victim to have an established connection with its network. So in theory, an adversary can use this to send the Attach Reject and Service Reject messages.
Additionally, the authors state that this can have a higher chance of succeeding than an attack using a fake base station:

> *With a 3 dB power difference from a legitimate signal, the SigOver attack demonstrated a 98% success rate when compared with the 80% success rate of attacks achieved using a fake base station, even with a 35 dB power difference [60].*

6.3.3 Countermeasures

In this section, we highlight possible countermeasures against the attacks we analysed. First, we look at the 5G standards to see whether things have improved since the LTE standards. Then, we analyse other countermeasures that have not been implemented into the 3GPP standards yet.

6.3.3.1 5G improvements

To understand which improvements 5G has in terms of these downgrade and denial-of-service attacks we need to take a look at the newer 3GPP releases starting at 3GPP release 15.
The big apparent change, which has already been highlighted by several other papers [26], is that the IMSI and TMSI that is used in LTE is replaced by the *Subscription Concealed Identifier* (*SUCI*) in 5G [26]. This SUCI is encrypted and is a temporary identifier. In 5G, the idea is that this SUCI is always used, even for connection establishment, whereas the IMSI is not used any more. The consequence is that it is now harder to implement a targeted attack on mobile devices.
The encryption of this identifier is handled by using the public key of the home network [26]. This can only work when the serving network also has 5G support, or else it does not know how to process this SUCI. However, as highlighted by Khan et al. [26], IMSI catching can still be done if the phone connects to a malicious LTE network, as a 5G phone still has LTE capabilities. This LTE network can then use a regular LTE IMSI catcher.

As can be seen by these types of attacks, backwards compatibility is often a problem in terms of security. Even worse, Reject messages (e.g. the Service Reject and Attach Reject messages) are still not integrity protected in 5G [10, 19]. This makes an adversary able to execute the same downgrade and denial of service LTE attacks that we highlighted in this paper. To explore how this can be fixed, we discuss in the next section possible other countermeasures that could be implemented in future 3GPP releases.

6.3.3.2 Additional countermeasures

In this section, we explore possible additional countermeasures that could be taken to prevent or weaken these denial-of-service and downgrade attacks. Shaik et al. [48] already stated that a possible countermeasure was to implement timers into the 3GPP releases. As we have highlighted in Section 6.1, these timers have been implemented in 3GPP release 13.04 and are set to a value between 12 and 24 hours. We think that this timer can further be decreased to weaken the impact of these attacks. However, lowering this timer too much can cause performance problems for the LTE and 5G networks. The reason that mobile devices are unable to connect to these networks up until the phone is rebooted, or as we saw by toggling flight mode, is done for performance concerns. Shaik et al. already highlighted this aspect [48]:

> *Since the network denies services for valid reject causes described in [11], the UE restricts itself from re-initiating LTE (or any mobile network) Attach procedure in order to conserve battery power. In addition, frequent unsuccessful Attach requests from UEs would increase signaling load on the network. These are the reasons why the LTE specification requires the UE to reboot or re-insert USIM to recover from reject messages. This preference of performance over security leaves LTE subscribers vulnerable to the DoS attacks*

To summarize, this handling is chosen deliberately as a security versus performance tradeoff. Additionally, the authors state that these Reject messages (here they only mention the TAU Reject message) should have some form of verification mechanism by the UE:

> *If there is an infrastructure to support distribution of operator public keys, TAU reject messages could be signed by the network and verified by UEs.*

As we have seen a similar notion in the distribution of public keys with the SUCI mechanism of 5G, we propose that reject messages are signed by the serving network and verified by the UE also using asymmetric cryptography. This does pose a challenge as ideally we want a separate keypair for signing these messages as the encryption/decryption procedure of the SUCI mechanism. This requires drastic changes in the 5G infrastructure.

However, completely fixing these issues is difficult as backwards compatibility is a crucial aspect for mobile networks to function. Even if we completely fix downgrade/denial-of-service attacks in the latest 5G standard, 5G capable phones are still free to connect to older and less secure networks such as LTE, UMTS, and GSM. If 5G capable phones do not have support for these older networks then adoption would be difficult to achieve. As of right now, the coverage for 5G networks is much lower than LTE networks, even for first world countries [33].

7 Conclusion

In this last chapter, we go over what we discussed in the thesis and give a recap of our findings. Additionally, we present recommendations on how to improve upon this research area by giving recommendations and shortcomings of this paper.

7.1 Summary and findings

In this thesis, we have analysed downgrade and denial-of-service attacks in 4G LTE. In essence, we implemented downgrade attacks in the open source SrsRAN software and tested it on multiple mobile phones with various 3GPP releases. The goal was to find and analyse the differences in handling of the attacks we implemented.
To do this, we have first given the needed background in the preliminaries. This consisted of explaining the architecture of the LTE network including procedures that are used in our downgrade attacks. Afterwards, we have explained our approach to picking the attacks and implementing them in the SrsRAN software. These attacks use the Attach Reject and Service Reject messages. Next, our hardware and software set up was further explained, and the testing procedure was presented. Finally, we analysed these attacks further by going over the difference in 3GPP releases (the LTE standards), presenting the results and discussing how practical they are for an adversary. Additionally, we also went over possible countermeasures and other scenarios the attacker could exploit.
We found that the attacks work on the OnePlus 8 and the Xiaomi Mi 9T pro and they can recover by toggling flight mode. The attacks that work for the Nexus 5 and OnePlus 3 are the Attach Reject downgrade, the Service reject downgrade and the Attach Reject denial-of-service. Where toggling flight mode was enough to recover from the downgrade attack. The Attach Reject denial-of-service on these two devices needed the SIM card to be re-inserted in order to recover from these attacks. The Service Reject denial-of-service attack does not work as these devices use a 3GPP release which does not handle this specific attack. These phones keep sending a Service Request message which can be problematic for the battery of these devices.
Furthermore, we found that a malicious base station is not the only way to employ these attacks. A technique called signal overshadowing can be used.
In 5G, these attacks are not fully prevented yet. While 5G has made targeted attacks harder by encrypting the SIM's unique identifier, it has not yet integrity protected or authenticated the downgraded messages that we used in the attacks. We proposed that these messages should be signed by the network and verified by the UE. This needs drastic changes in the 5G network architecture.

7.2 Future work

Throughout this thesis we have hinted towards several improvements that can be made in follow-up research. In this section, we present these possible improvements again in a more structured way.

First, the mobile devices that we use do not have the complete range of 3GPP releases. Our devices use 3GPP release 9, 11, 14, and 15. It can be interesting to test our attacks on even older devices or devices using 3GPP release 12, 13, and even 16. This also relates to the fact that we tested our attacks on a 4G LTE network, not on a 5G network. If SrsRAN has sufficient 5G compatibility, it is interesting to test these same attacks with a 5G capable UE and network.
Second, we have explained that our 3G network is not perfect. It uses the OpenBTS-UMTS software which does not support the cryptography of the USIM that we use. This means that we could not verify for sure that these devices were able to fully connect to a 3G network and send network packets to it. We could only see the RRC messages come in, a full Attach procedure could not be achieved. Additionally, we have not tested whether the devices can still connect to a 2G network instead of a 3G network. While it makes sense that these devices will be able to connect to a 2G network in the downgrade setting, this has not been verified. Even more interesting, showing that the devices cannot connect to a 2G network in the denial-of-service setting gives more concrete proof that the devices really cannot connect to any network. In future work, it might be interesting to use multiple different software projects for a non-LTE network, such as *Osmocom* [38] which can be used for 2G and 3G networks. This software possibly also allows for a full connection with USIMs [39].
Third, the attacks we use were chosen to use procedures that were also supported by SrsUE, the virtual mobile of SrsRAN. While in the end testing these attacks using fully virtual software was not so useful, it was mainly handy to test our attacks before we got a chance to use SDRs. In future work, one can explore the use of several other procedures such as the TAU procedure which was previously mentioned in other papers [17, 48]. Another way to create a variation of the attacks is sending a Service Reject message on the receipt of the Attach Request message instead of the Service Request message.
Fourth, a follow-up paper can look into adapting our attacks for a more traditional fake base station setting. In our setting, the mobile devices were preconfigured with the MCC and MNC identifiers for the LTE network. Additionally, the Service Request was triggered by letting the mobile phone connect to the network and then sending uplink data after the RRC IDLE state was enabled. A possible improvement could then be to look into the complete set up for spoofing legitimate MCC and MNC identifiers for the networks and getting UEs to connect to the malicious network. The Paging procedure could be used to force a UE to send a Service Request.
And at last, attacks using the signal overshadowing technique are an interesting way to follow up on our research. A possible idea is to create a setup that uses this technique and then test how different devices react on downgrade or denial-of-service messages. Additionally, this can be used to test how practical the signal overshadowing technique is as compared to the fake base station attack that we presented and discussed.

8 Appendix

8.1 SrsRAN vs OpenLTE code changes

SrsRAN nowadays has more contributions and code changes as compared to OpenLTE. As an impression of this, we have conducted the following test:

```
$ git log --since 5.year --pretty=tformat: --numstat \
| awk '{ \
  add += $1; \
  subs += $2; \
  loc += $1 - $2 \
} \
END { \
  printf "Total lines added: %s, Total lines removed: %s, \
Total growth of lines: %s\n", add, subs, loc \
}' -
```

Listing 7: Shell command to produce the last 5 years of changes in a git repository, adapted from [34]

> OpenLTE: Total lines added: 297477, Total lines removed: 62909, Total growth of lines: 234568
> SrsRAN: Total lines added: 1802357, Total lines removed: 980066, Total growth of lines: 822291

8.2 Pcap 3GPP release info

8.2.1 Nexus 5

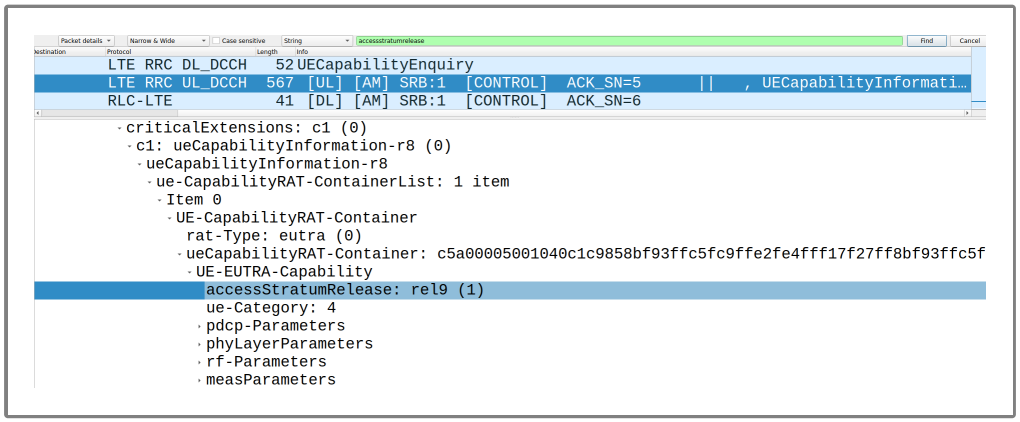


Figure 8.1: Release info for the Nexus 5

8.2.2 OnePlus 3

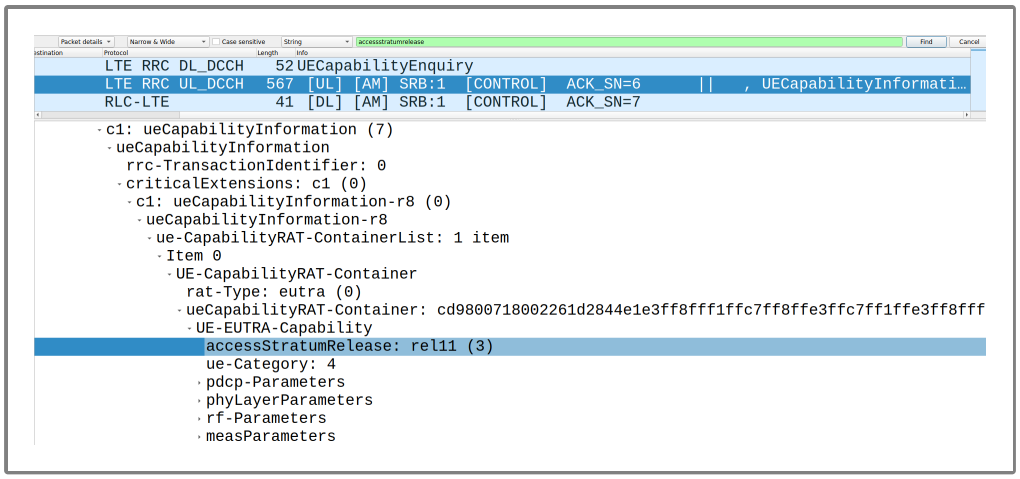


Figure 8.2: Release info for the OnePlus 3

8.2.3 Xiaomi Mi 9T Pro

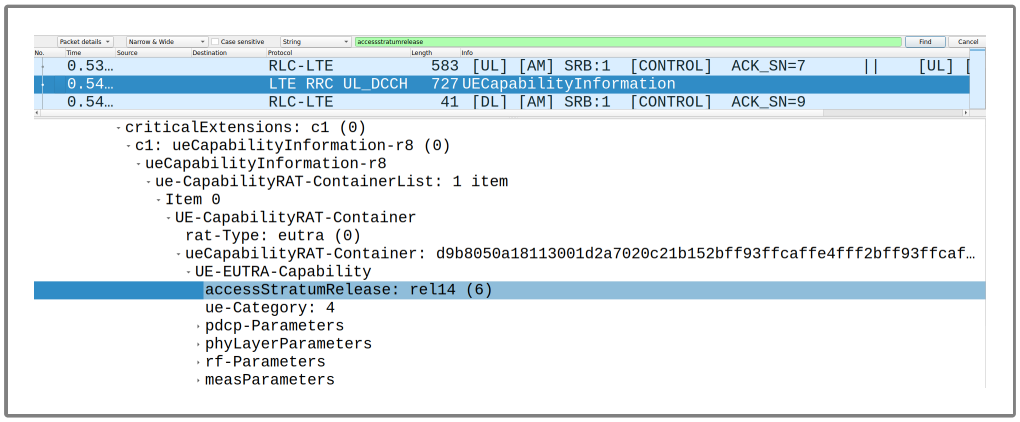


Figure 8.3: Release info for the Xiaomi Mi 9T Pro

8.2.4 OnePlus 8

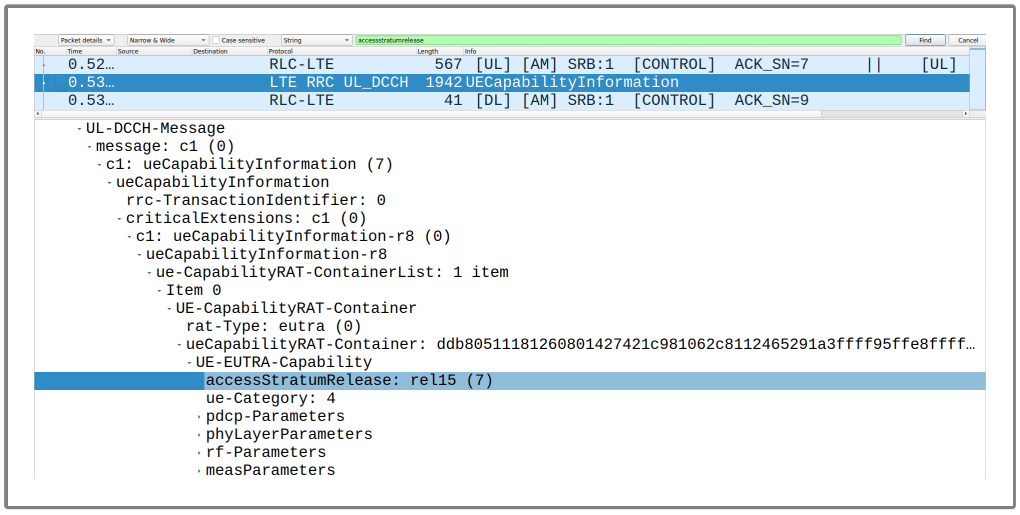


Figure 8.4: Release info for the OnePlus 8

8.3 OnePlus 8 APN Settings

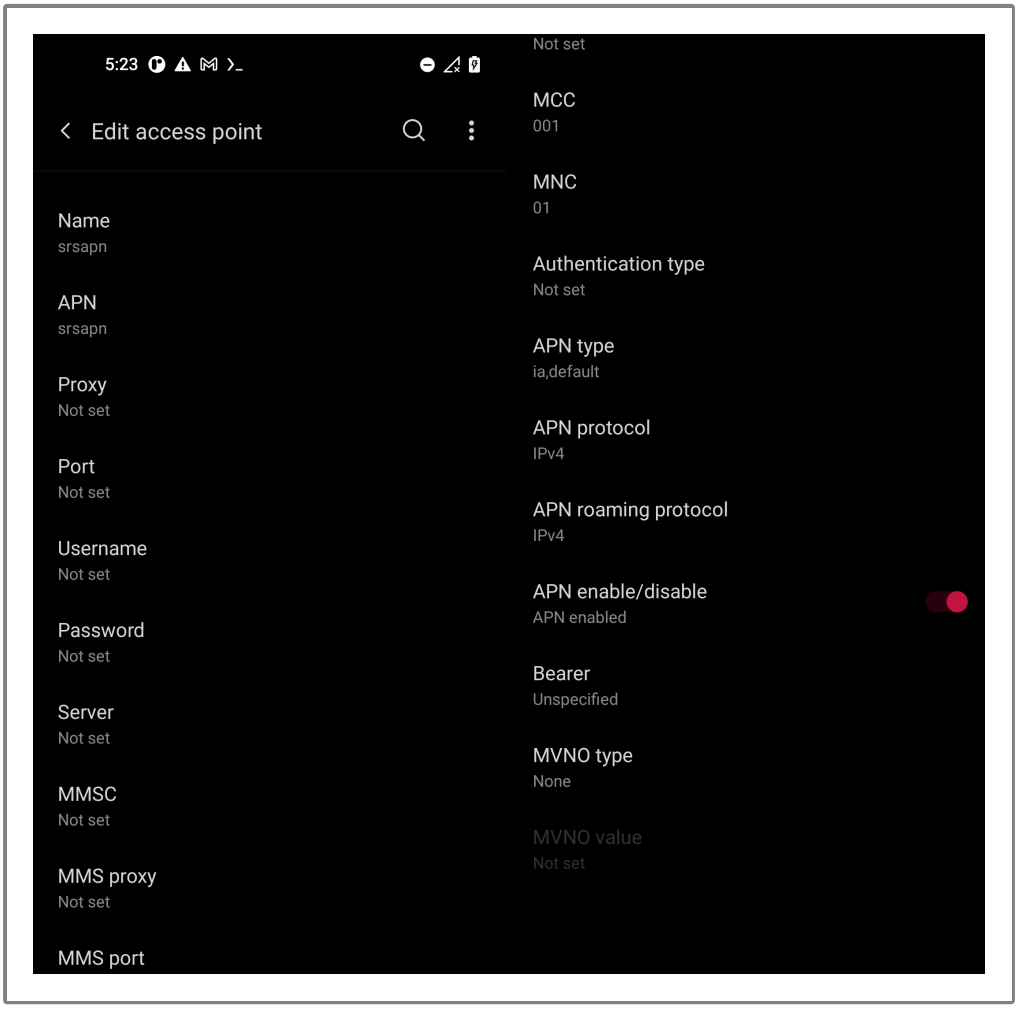


Figure 8.5: Oneplus 8 Access Point settings menu

8.4 Scat flags

The flags for Scat mean the following [13]:

- `-t`: The type of the baseband, *qc* specifies a Qualcomm baseband
- `-u`: Signifies that we are accessing the device through a USB connection
- `-a`: The USB address as gathered by the output from running the `lsusb` command
- `-i`: The USB interface to use. In our case `0` works except for the Nexus 5 where we have to use a value of `2`
- `-F`: Specifies the file to write the messages to. This file uses the *PCAP* format which can be opened and analysed by the Open Source graphical *Wireshark* [56] program

Bibliography

[1] 3GPP. *Change Requests*. https://www.3gpp.org/specifications/change-requests, Last accessed on 2022-01-12. 2021.

[2] 3GPP. *Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification*. Technical Specification (TS) 36.331. Version 14.2.2. 3rd Generation Partnership Project (3GPP), Apr. 2017. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2440.

[3] 3GPP. *Releases*. https://www.3gpp.org/specifications/67-releases, Last accessed on 2021-12-10. 2021.

[4] 3GPP. *Universal Mobile Telecommunications System (UMTS);LTE; 5G; Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS); Stage 3*. Technical Specification (TS) 24.301. Version 15.3.0. 3rd Generation Partnership Project (3GPP), June 2021. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=1072.

[5] 3GPP Portal TS 24.008 release 13. *Change Request C1-154214: T3245 timer range clarification*. https://www.3gpp.org/ftp/tsg_ct/WG1_mm-cc-sm_ex-CN1/TSGC1_95_Anaheim/docs/C1-154214.zip. Nov. 2015.

[6] 3GPP Portal TS 24.008 release 13. *Change Request C1-154699: Clarification about timer T3245 usage*. https://www.3gpp.org/ftp/tsg_ct/WG1_mm-cc-sm_ex-CN1/TSGC1_95_Anaheim/docs/C1-154699.zip. Nov. 2015.

[7] 3GPP Portal TS 24.301 release 10. *Change Request C1-112116: Replace T3446 with T3346*. June 2011.

[8] 3GPP Portal TS 24.301 release 11. *Change Request C1-124966: Reject cause #8 used for normal TAU and SR*. https://www.3gpp.org/ftp/tsg_ct/WG1_mm-cc-sm_ex-CN1/TSGC1_81_NewOrleans/docs/C1-124966.zip. Oct. 2012.

[9] andrepuschmann. *fauxRF is not supported*. https://github.com/srsLTE/srsLTE/pull/238#issuecomment-488998318. 2019.

[10] Shanay Behrad, Emmanuel Bertin, and Noel Crespi. "Securing authentication for mobile networks, a survey on 4G issues and 5G answers". In: *2018 21st conference on innovation in clouds, internet and networks and workshops (ICIN)*. IEEE. 2018, pp. 1–8.

[11] Christopher Cox. *An introduction to LTE: LTE, LTE-advanced, SAE and 4G mobile communications*. John Wiley & Sons, 2012.

[12] DataReportal. *Digital Around the World —DataReportal – Global Digital Insights*. https://datareportal.com/global-digital-overview, Last accessed on 2021-12-02. 2021.

[13] fgsect. *SCAT: Signaling Collection and Analysis Tool*. https://github.com/fgsect/scat. 2021.

[14] Genymobile. *Scrcpy: Display and control your Android device*. https://github.com/Genymobile/scrcpy. 2021.

[15] Google Developers. *Android Debug Bridge (adb) | Android Developers*. https://developer.android.com/studio/command-line/adb, Last accessed on 2021-08-04. 2021.

[16] Junxian Huang et al. "A close examination of performance and power characteristics of 4G LTE networks". In: *Proceedings of the 10th international conference on Mobile systems, applications, and services*. 2012, pp. 225–238.

[17] Lin Huang. *LTE Redirection Attack - Forcing Targeted LTE Cellphone into Unsafe Network*. https://www.youtube.com/watch?v=hNDChDM1hEE, Last accessed on 2021-08-10. May 2016.

[18] Syed Hussain et al. "LTEInspector: A systematic approach for adversarial testing of 4G LTE". In: *Network and Distributed Systems Security (NDSS) Symposium 2018*. 2018.

[19] Syed Rafiul Hussain et al. "5GReasoner: A property-directed security and privacy analysis framework for 5G cellular network protocol". In: *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. 2019, pp. 669–684.

[20] Roger Piqueras Jover. "LTE security and protocol exploits". In: *Shmoocon 2016* (2016).

[21] Roger Piqueras Jover. *LTE security, protocol exploits and location tracking experimentation with low-cost software radio*. 2016. arXiv: 1607.05171 [cs.CR].

[22] jwijenbergh. *master-thesis-resources*. https://github.com/jwijenbergh/master-thesis-resources. 2021.

[23] jwijenbergh. *OpenBTS-UMTS-Docker*. https://github.com/jwijenbergh/OpenBTS-UMTS-Docker. 2021.

[24] jwijenbergh. *srsran-docker-emulated*. https://github.com/jwijenbergh/srsran-docker-emulated. 2021.

[25] jwijenbergh. *srsran-downgrade*. https://github.com/jwijenbergh/srsran-downgrade/. 2021.

[26] Mohsin Khan et al. "Defeating the downgrade attack on identity privacy in 5G". In: *International Conference on Research in Security Standardisation*. Springer. 2018, pp. 95–119.

[27] James Kurose. *Computer networking: a top-down approach*. Boston: Pearson, 2017. ISBN: 9780133594140.

[28] Ulrike Meyer and Susanne Wetzel. "A man-in-the-middle attack on UMTS". In: *Proceedings of the 3rd ACM workshop on Wireless security*. 2004, pp. 90–97.

[29] mgp25, onkarmumbrekar, et al. *OpenLTE Attach Reject*. https://github.com/mgp25/OpenLTE/tree/attach_reject. 2021.

[30] mgp25, onkarmumbrekar, et al. *OpenLTE Service Reject*. https://github.com/mgp25/OpenLTE/tree/service_reject_on_tau. 2021.

[31] mgp25, onkarmumbrekar, et al. *OpenLTE Tau Reject*. https://github.com/mgp25/OpenLTE/tree/dos_tau_reject. 2021.

[32] Bodo Möller, Thai Duong, and Krzysztof Kotowicz. "This POODLE bites: exploiting the SSL 3.0 fallback". In: *Security Advisory* 21 (2014), pp. 34–58.

[33] nPerf. *3G / 4G / 5G dekking in Netherlands - nPerf.com*. https://www.nperf.com/nl/map/NL/-/-/signal/, Last accessed on 2021-12-03. 2021.

[34] Oh, Alexandar and Top-Master. *How to count total lines changed by a specific author in a Git repository? - Stack Overflow*. https://stackoverflow.com/questions/1265040/how-to-count-total-lines-changed-by-a-specific-author-in-a-git-repository/7010890#7010890, Last accessed on 2021-01-17.

[35] OpenBTS. *OpenBTS-UMTS - OpenBTS*. http://openbts.org/w/index.php?title=OpenBTS-UMTS, Last accessed on 2021-12-10. 2017.

[36] OpenLTE Sourceforge. *OpenLTE*. http://openlte.sourceforge.net/, Last accessed on 2022-01-15. 2022.
[37] OpenSSH. *OpenSSH*. https://www.openssh.com/, Last accessed on 2022-01-12. 2021.
[38] Osmocom. *Open Source Mobile Communications*. https://osmocom.org/, Last accessed on 2021-12-11. 2021.
[39] Osmocom. *Osmocom 3G and 2G Now Support Milenage Authentication*. https://osmocom.org/news/67, Last accessed on 2021-12-11. 2017.
[40] *Overview of Docker Compose*. URL: https://docs.docker.com/compose/.
[41] Ivan Palamà et al. "The diverse and variegated reactions of different cellular devices to IMSI catching attacks". In: *Proceedings of the 14th International Workshop on Wireless Network Testbeds, Experimental evaluation & Characterization*. 2020, pp. 80–86.
[42] Pew Research Center. *Demographics of Mobile Device Ownership and Adoption in the United States | Pew Research Center*. https://www.pewresearch.org/internet/fact-sheet/mobile/, Last accessed on 2021-12-02. 2021.
[43] pgorczak. *srslte-docker-emulated*. https://github.com/pgorczak/srslte-docker-emulated. 2020.
[44] Python. *About Python™ | Python.org*. https://www.python.org/about/, Last accessed on 2021-08-11. 2021.
[45] RangeNetworks. *OpenBTS-UMTS source code*. https://github.com/RangeNetworks/OpenBTS-UMTS. 2021.
[46] David Rupprecht et al. "Breaking LTE on Layer Two". In: *IEEE Symposium on Security & Privacy (SP)*. IEEE, May 2019.
[47] Nabil Seddigh et al. "Security advances and challenges in 4G wireless networks". In: *2010 Eighth International Conference on Privacy, Security and Trust*. IEEE. 2010, pp. 62–71.
[48] Altaf Shaik et al. "Practical attacks against privacy and availability in 4G/LTE mobile communication systems". In: *arXiv preprint arXiv:1510.07563* (2015).
[49] Software Radio Systems. *SrsLTE Release 20.10.1 Readme*. https://github.com/srsran/srsRAN/tree/release_20_10_1. 2020.
[50] Software Radio Systems. *SrsRAN Attach Reject supported*. https://github.com/srsran/srsRAN/blob/c950209902e28b7a3eb8724943ed5e3167f05d29/srsue/src/stack/upper/nas.cc#L1204. 2021.
[51] Software Radio Systems. *SrsRAN Attach Request supported*. https://github.com/srsran/srsRAN/blob/c950209902e28b7a3eb8724943ed5e3167f05d29/srsue/src/stack/upper/nas.cc#L302. 2021.
[52] Software Radio Systems. *SrsRAN Release 21.10 Readme*. https://github.com/srsran/srsRAN/tree/release_21_10. 2021.
[53] Software Radio Systems. *SrsRAN service Reject supported*. https://github.com/srsran/srsRAN/blob/c950209902e28b7a3eb8724943ed5e3167f05d29/srsue/src/stack/upper/nas.cc#L1435. 2021.
[54] Software Radio Systems. *SrsRAN Service Request supported*. https://github.com/srsran/srsRAN/blob/c950209902e28b7a3eb8724943ed5e3167f05d29/srsue/src/stack/upper/nas.cc#L352. 2021.
[55] Software Radio Systems. *SrsRAN TAU Request not supported*. https://github.com/srsran/srsRAN/blob/c950209902e28b7a3eb8724943ed5e3167f05d29/lib/src/asn1/liblte_mme.cc#L7466. 2021.
[56] The Wireshark Foundation. *Wireshark - Go Deep*. https://www.wireshark.org/, Last accessed on 2021-08-04. 2021.
[57] topjohnwu. *Magisk: The Magic Mask for Android*. https://github.com/topjohnwu/magisk. 2021.

[58] Filip Turniški et al. “Analysis of 3G and 4G download throughput in pedestrian zones”. In: *2016 International Symposium ELMAR*. IEEE. 2016, pp. 9–12.
[59] Jeroen Wijenbergh. *Master Thesis Demo Videos*. https://www.youtube.com/playlist?list=PLUwMEC686s7v2mQ3cI8VeTTjGGqaakGb9. 2021.
[60] Hojoon Yang et al. “Hiding in Plain Signal: Physical Signal Overshadowing Attack on {LTE}”. In: *28th {USENIX} Security Symposium ({USENIX} Security 19)*. 2019, pp. 55–72.
[61] *ZeroMQ | Get started*. URL: https://www.zeromq.org/get-started/.
[62] *ZeroMQ Application note — srsLTE 20.10.1 documentation*. URL: https://docs.srslte.com/en/latest/app_notes/source/zeromq/source/index.html.