MASTER

GSM Cell Broadcast Service security analysis

Cedillo Torres, A.

Award date:
2013

Link to publication
GSM CELL BROADCAST SERVICE SECURITY ANALYSIS

ARTURO CEDILLO TORRES

Department of Mathematics and Computer Science
Eindhoven University of Technology

Supervisors:
Nicola Zannone¹
Erik Poll²
Ronald Heil³
Brinio Hond⁴

¹ Assistant Professor at Eindhoven University of Technology
² Associate Professor at Radboud University Nijmegen
³ Senior Manager KPMG
⁴ Junior Advisor KPMG
ABSTRACT

The Cell Broadcast Service (CBS), part of the GSM standard, is currently being used by governments around the world to broadcast emergency information to the population in the presence of natural or human caused catastrophes. When this service was released in the early 1990s, no specification was made available to indicate how broadcast messages should be processed and displayed by mobile phones. It was not until 2009 when a set of basic requirements was defined. This lack of provisions forced mobile phone manufacturers to implement their own CBS message processing mechanisms. This diversity of implementations raises security concerns due to a higher probability of the absence of proper CBS message validation mechanisms, in charge of ensuring the message contains valid values or a proper structure.

This thesis, with the help of OpenBTS and a USRP-1, presents the results of performing several fuzzing tests to verify the quality of GSM CBS implementations in four different mobile phones. First, the theory of how the CBS works in the GSM standard as well as the structure of CBS messages is provided. Later, this theory is used to fuzz the CBS implementation in mobile phones. Our results show that in most cases mobile phones properly handle CBS messages, even when the messages’ structure and values are not as defined in the GSM specification. The results show, however, that each mobile station processes and displays CBS messages in different manners, in some cases making it impossible to actually read the messages. Finally, we show that one mobile phone, running the latest version of Android operating system, does not properly handle one type of crafted CBS message. This situation causes the triggering of unspecified behavior in the device. Even though this behavior does not affect the general functioning of the mobile phone, it shows the lack of compliance to the latest CBS message processing requirements.
ACKNOWLEDGMENTS

I want to thank all my supervisors, Nicola, Erik, Ronald and Brinio for providing me with invaluable feedback all along this process as well as support and motivation when things did not look too bright. To all of them, my recognition and gratitude.

My recognition also goes to Fabian van den Broek for proposing this topic once my original research idea could not be done due to a lack of equipment. I will never forget the hours we spent trying to configure the equipment and figure out what was wrong when nothing worked after all that time spent. Thanks for guiding me and providing me with ideas on how to solve the different challenges I encountered.

I also want to thank my girlfriend Jennifer who showed patience and gave me support in the past six months to finish this thesis. Last but not least, I want to thank my parents, Ramon and Cristina as well as my sister Cristina for always being there for me, even when I bore them to death with the details of my research.
# CONTENTS

1 INTRODUCTION 1
   1.1 Research Question ............................ 3
   1.2 Contribution and Scope .......................... 4
      1.2.1 Contribution ............................. 4
      1.2.2 Scope .................................. 4
   1.3 Research Methodology ............................ 5
   1.4 Outline ..................................... 5

2 RELATED WORK ................................ 7

3 INTRODUCTION TO GSM 11
   3.1 Background ................................. 11
   3.2 System Architecture ........................... 12
   3.3 Interfaces ................................ 13
      3.3.1 The Air Interface ....................... 13

4 CELL BROADCAST SERVICE 21
   4.1 Background .................................. 21
   4.2 General PWS Requirements ....................... 22
      4.2.1 Public Warning Service Requirements .... 23
      4.2.2 Mobile Stations Requirements .......... 23
   4.3 Type of Messages .............................. 23
   4.4 Integration with GSM ............................ 24
      4.4.1 Communication Channels ................. 25
   4.5 Interface CBC - BSC ........................... 26
      4.5.1 CBC and BSC functionalities ............ 26
      4.5.2 Communication Protocols ............... 26
   4.6 Interface BSC - BTS ........................... 35
      4.6.1 Handling of CBS Messages ............ 35
      4.6.2 Handling of ETWS Messages .......... 36
   4.7 Interface BTS - MS ............................. 37
      4.7.1 CBS Message Transmission ............ 37
      4.7.2 ETWS Message Transmission .......... 40
   4.8 Message Reception at Mobile Stations .......... 42
      4.8.1 Duplication Detection ................. 42
      4.8.2 Search list ............................. 43

5 INTRODUCTION TO PROTOCOL FUZZING 45
   5.1 Background ................................. 45
   5.2 Types of Protocol Fuzzing ..................... 46

6 FUZZING METHODOLOGY 49
   6.1 Methodology ................................. 49
   6.2 Expected mobile station behavior ............... 50
   6.3 Fuzzing Scheme ............................... 52
   6.4 Identify Fields to Fuzz ....................... 52
      6.4.1 Block Type Fields ....................... 53
6.4.2 Serial Number .............................................. 54
6.4.3 Data Coding Scheme ....................................... 54
6.4.4 Page Parameter ............................................. 54
6.4.5 Content of Message ......................................... 55
6.5 Fuzzing approach ............................................. 55

7 FUZZING EXPERIMENTS RESULTS .......................... 57
7.1 Hardware and Software Setup ................................. 57
7.2 Fuzzing ........................................................ 59
7.2.1 Mobile Stations ............................................. 59
7.2.2 Fuzzing the Block Type Header ........................... 61
7.2.3 Fuzzing the Serial Number ................................. 62
7.2.4 Fuzzing the Data Coding Scheme ........................ 63
7.2.5 Fuzzing the Page Parameter ............................... 63
7.2.6 Fuzzing the Content of Message .......................... 64
7.3 Analysis of Results ............................................ 65

8 CONCLUSIONS AND FUTURE WORK ......................... 71
8.1 Conclusions .................................................. 71
8.2 Future Work .................................................. 73

BIBLIOGRAPHY .................................................. 75

A CELL BROADCAST SERVICE MESSAGES ..................... 81
A.1 CBCH LOAD INDICATOR .................................. 81
A.2 PAGING REQUEST TYPE 1 ................................... 81
A.3 SMS BROADCAST COMMAND ............................... 82
A.4 SMS BROADCAST REQUEST ................................ 82

B INFORMATION ELEMENTS ..................................... 85
B.1 Information Elements Normal CBS Message ............... 85
B.1.1 Serial Number ............................................. 85
B.1.2 Message Identifier ........................................ 87
B.1.3 Data Coding Scheme ...................................... 88
B.1.4 Page Parameter ............................................ 89
B.1.5 Content of Message ....................................... 89

C MOBILE STATIONS OPERATIONAL MODES ................. 91

D EXPERIMENT SETUP .......................................... 93
D.1 Hardware Requirements ...................................... 93
D.2 Software Requirements ....................................... 94

E GSM 7 BIT ALPHABET .......................................... 95
E.1 GSM 7-bit Message Encoding ............................... 95
E.2 GSM 7-bit Message Decoding ............................... 96

F TROUBLESHOOTING OPENBTS .............................. 99
F.1 Hardware and Software Configurations ..................... 99
F.2 Debugging the Air Interface ................................. 100
F.3 Analysis of OpenBTS log files .............................. 101
F.4 Spoofing a Provider’s Base Station ......................... 103
F.5 New Configuration Tests .................................... 104

GOMMENTS AND LOG FILES ................................. 57
G.1 Hardware and Software Setup ............................... 57
G.2 Fuzzing ........................................................ 59
G.2.1 Mobile Stations ............................................. 59
G.2.2 Fuzzing the Block Type Header ........................... 61
G.2.3 Fuzzing the Serial Number ................................. 62
G.2.4 Fuzzing the Data Coding Scheme ........................ 63
G.2.5 Fuzzing the Page Parameter ............................... 63
G.2.6 Fuzzing the Content of Message .......................... 64
G.3 Analysis of Results ............................................ 65

H CONCLUSIONS AND FUTURE WORK ......................... 71
H.1 Conclusions .................................................. 71
H.2 Future Work .................................................. 73

I INFORMATION ELEMENTS ..................................... 85
I.1 Information Elements Normal CBS Message ............... 85
I.1.1 Serial Number ............................................. 85
I.1.2 Message Identifier ........................................ 87
I.1.3 Data Coding Scheme ...................................... 88
I.1.4 Page Parameter ............................................ 89
I.1.5 Content of Message ....................................... 89

J MOBILE STATIONS OPERATIONAL MODES ................. 91

K EXPERIMENT SETUP .......................................... 93
K.1 Hardware Requirements ...................................... 93
K.2 Software Requirements ....................................... 94

L GSM 7 BIT ALPHABET .......................................... 95
L.1 GSM 7-bit Message Encoding ............................... 95
L.2 GSM 7-bit Message Decoding ............................... 96

M TROUBLESHOOTING OPENBTS .............................. 99
M.1 Hardware and Software Configurations ..................... 99
M.2 Debugging the Air Interface ................................. 100
M.3 Analysis of OpenBTS log files .............................. 101
M.4 Spoofing a Provider’s Base Station ......................... 103
M.5 New Configuration Tests .................................... 104

N BIBLIOGRAPHY .................................................. 75
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 1</td>
<td>GSM system architecture</td>
<td>13</td>
</tr>
<tr>
<td>Figure 2</td>
<td>GSM channel, frame and time slot</td>
<td>14</td>
</tr>
<tr>
<td>Figure 3</td>
<td>CBC integration into GSM architecture</td>
<td>25</td>
</tr>
<tr>
<td>Figure 4</td>
<td>CBS successful Write-Replace procedure</td>
<td>27</td>
</tr>
<tr>
<td>Figure 5</td>
<td>CBS Write-Replace message format</td>
<td>29</td>
</tr>
<tr>
<td>Figure 6</td>
<td>CBS Write-Replace Complete message format</td>
<td>31</td>
</tr>
<tr>
<td>Figure 7</td>
<td>CBS Write-Replace Failure message format</td>
<td>33</td>
</tr>
<tr>
<td>Figure 8</td>
<td>Successful ETWS message write operation</td>
<td>34</td>
</tr>
<tr>
<td>Figure 9</td>
<td>CBS SMS BROADCAST REQUEST operational mode</td>
<td>35</td>
</tr>
<tr>
<td>Figure 10</td>
<td>CBS SMS BROADCAST COMMAND operational mode</td>
<td>36</td>
</tr>
<tr>
<td>Figure 11</td>
<td>CBS normal CBS message format</td>
<td>37</td>
</tr>
<tr>
<td>Figure 12</td>
<td>CBS Block Type header format</td>
<td>37</td>
</tr>
<tr>
<td>Figure 13</td>
<td>Broadcasting of CBS message</td>
<td>39</td>
</tr>
<tr>
<td>Figure 14</td>
<td>CBS emergency message format</td>
<td>40</td>
</tr>
<tr>
<td>Figure 15</td>
<td>Broadcasting of emergency CBS message</td>
<td>41</td>
</tr>
<tr>
<td>Figure 16</td>
<td>Fuzzing tests methodology</td>
<td>49</td>
</tr>
<tr>
<td>Figure 17</td>
<td>CBS message fuzzing candidates</td>
<td>53</td>
</tr>
<tr>
<td>Figure 18</td>
<td>OpenBTS network architecture</td>
<td>57</td>
</tr>
<tr>
<td>Figure 19</td>
<td>CBS message when received in a Nokia 3310</td>
<td>59</td>
</tr>
<tr>
<td>Figure 20</td>
<td>Galaxy Note displaying unspecified behavior</td>
<td>65</td>
</tr>
<tr>
<td>Figure 21</td>
<td>Blackberry 9700 displaying the contents of a compressed CBS message using an 8-bit alphabet</td>
<td>68</td>
</tr>
<tr>
<td>Figure 22</td>
<td>Nokia 1600 displaying only a few characters of CBS message</td>
<td>69</td>
</tr>
<tr>
<td>Figure 23</td>
<td>Blackberry 9700 displaying three characters of CBS message</td>
<td>69</td>
</tr>
<tr>
<td>Figure 24</td>
<td>CBCH LOAD INDICATOR message format</td>
<td>81</td>
</tr>
<tr>
<td>Figure 25</td>
<td>PAGING REQUEST TYPE 1 message format</td>
<td>82</td>
</tr>
<tr>
<td>Figure 26</td>
<td>SMS BROADCAST COMMAND message format</td>
<td>82</td>
</tr>
<tr>
<td>Figure 27</td>
<td>SMS BROADCAST REQUEST message format</td>
<td>83</td>
</tr>
<tr>
<td>Figure 28</td>
<td>Format New Serial Number IE</td>
<td>85</td>
</tr>
<tr>
<td>Figure 29</td>
<td>Inner structure of Serial Number IE</td>
<td>85</td>
</tr>
<tr>
<td>Figure 30</td>
<td>Format Emergency Message Code field</td>
<td>87</td>
</tr>
<tr>
<td>Figure 31</td>
<td>Format Message Identifier IE</td>
<td>88</td>
</tr>
<tr>
<td>Figure 32</td>
<td>Format Data Coding Scheme IE</td>
<td>88</td>
</tr>
<tr>
<td>Figure 33</td>
<td>Format Page Parameter IE</td>
<td>89</td>
</tr>
<tr>
<td>Figure 34</td>
<td>Format Message Content IE</td>
<td>89</td>
</tr>
<tr>
<td>Figure 35</td>
<td>Hardware setup for fuzzing experiments</td>
<td>94</td>
</tr>
</tbody>
</table>
List of Figures

Figure 36 Mobile phones used for fuzzing testing . . . . 94
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Common GMS channel combinations</td>
<td>19</td>
</tr>
<tr>
<td>Table 2</td>
<td>Some CBS WRITE-REPLACE message information elements</td>
<td>28</td>
</tr>
<tr>
<td>Table 3</td>
<td>CBS Sequence number values</td>
<td>38</td>
</tr>
<tr>
<td>Table 4</td>
<td>Unspecified behavior expected to be triggered</td>
<td>52</td>
</tr>
<tr>
<td>Table 5</td>
<td>Software and hardware configuration</td>
<td>58</td>
</tr>
<tr>
<td>Table 6</td>
<td>Mobile stations used for fuzzing</td>
<td>60</td>
</tr>
<tr>
<td>Table 7</td>
<td>Block Type header fields' values and lengths</td>
<td>61</td>
</tr>
<tr>
<td>Table 8</td>
<td>Summary fuzzing attacks</td>
<td>66</td>
</tr>
<tr>
<td>Table 9</td>
<td>Geographic Scope field values</td>
<td>86</td>
</tr>
<tr>
<td>Table 10</td>
<td>Message Code IE Emergency User Alert and Pop-up fields values</td>
<td>87</td>
</tr>
<tr>
<td>Table 11</td>
<td>Software used for fuzzing experiments</td>
<td>94</td>
</tr>
<tr>
<td>Table 12</td>
<td>GSM 7-bit Alphabet</td>
<td>95</td>
</tr>
<tr>
<td>Table 13</td>
<td>Codification to GSM 7-bit alphabet</td>
<td>96</td>
</tr>
<tr>
<td>Table 14</td>
<td>Codification to GSM 7-bit alphabet</td>
<td>97</td>
</tr>
<tr>
<td>Table 15</td>
<td>Hardware and software configurations</td>
<td>99</td>
</tr>
<tr>
<td>Table 16</td>
<td>Block Type Decoding</td>
<td>102</td>
</tr>
<tr>
<td>Table 17</td>
<td>Information Element Decoding</td>
<td>103</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-------------</td>
<td></td>
</tr>
<tr>
<td>3GPP</td>
<td>3rd Generation Partnership Project</td>
<td></td>
</tr>
<tr>
<td>AGCH</td>
<td>Access Grant Channel</td>
<td></td>
</tr>
<tr>
<td>AuC</td>
<td>Authentication Center</td>
<td></td>
</tr>
<tr>
<td>BSC</td>
<td>Base Station Controller</td>
<td></td>
</tr>
<tr>
<td>BTS</td>
<td>Base Station Transceiver</td>
<td></td>
</tr>
<tr>
<td>BCHs</td>
<td>Broadcast Channels</td>
<td></td>
</tr>
<tr>
<td>BCCH</td>
<td>Broadcast Control Channel</td>
<td></td>
</tr>
<tr>
<td>CBC</td>
<td>Cell Broadcast Center</td>
<td></td>
</tr>
<tr>
<td>CBCH</td>
<td>Cell Broadcast Channel</td>
<td></td>
</tr>
<tr>
<td>CBE</td>
<td>Cell Broadcast Entity</td>
<td></td>
</tr>
<tr>
<td>CCCHs</td>
<td>Common Control Channels</td>
<td></td>
</tr>
<tr>
<td>CDMA</td>
<td>Code Division Multiple Access</td>
<td></td>
</tr>
<tr>
<td>CEPT</td>
<td>European Conference of Postal and Telecommunications Administrations</td>
<td></td>
</tr>
<tr>
<td>CI</td>
<td>Cell Identity</td>
<td></td>
</tr>
<tr>
<td>CMAS</td>
<td>Commercial Mobile Alert System</td>
<td></td>
</tr>
<tr>
<td>DCCHs</td>
<td>Dedicated Control Channel</td>
<td></td>
</tr>
<tr>
<td>DoS</td>
<td>Denial-of-Service</td>
<td></td>
</tr>
<tr>
<td>DCS</td>
<td>Digital Cellular System</td>
<td></td>
</tr>
<tr>
<td>EPs</td>
<td>Elementary Procedures</td>
<td></td>
</tr>
<tr>
<td>EIR</td>
<td>Equipment Identity Register</td>
<td></td>
</tr>
<tr>
<td>ETSI</td>
<td>European Telecommunications Standard Institute</td>
<td></td>
</tr>
<tr>
<td>ETWS</td>
<td>Earthquake and Tsunami Warning System</td>
<td></td>
</tr>
<tr>
<td>FACCH</td>
<td>Fast Associated Dedicated Control Channel</td>
<td></td>
</tr>
<tr>
<td>FCCH</td>
<td>Frequency Correction Channel</td>
<td></td>
</tr>
<tr>
<td>FDMA</td>
<td>Frequency Division Multiple Access</td>
<td></td>
</tr>
<tr>
<td>GMSC</td>
<td>Gateway MSC</td>
<td></td>
</tr>
</tbody>
</table>
GSM  Global System for Mobile Communications
HLR  Home Location Register
IEs  Information Elements
IMEI  International Mobile Equipment Identity
IMSI  International Mobile Subscribers Identity
KPAS  Korean Public Alert System
LAC  Location Area Code
LAI  Location Area Identification
LTE  Long Term Evolution
MCC  Mobile Country Code
MMS  Multimedia Messaging Service
MNC  Mobile Network Code
MNO  Mobile Network Operator
MS  Mobile Station
MSC  Mobile Switching Center
PCH  Paging Channel
PCS  Personal Communication Service
PDU  Protocol Data Unit
PSTN  Public Switched Telephone Network
PWS  Public Warning System
RACH  Random Access Channel
SACCH  Slow Associated Dedicated Control Channel
SCH  Synchronization Channel
SDCCH  Standalone Dedicated Control Channel
SDR  Software Defined Radio
SMS  Short Message Service
SQL  Structured Query Language
TCHs  Traffic Channels
TDMA  Time Division Multiple Access
TMSI  Temporary Mobile Subscriber Identity
UMTS  Universal Mobile Telecommunications System
USRP  Universal Software Radio Peripheral
VLR   Visitor Location Register
INTRODUCTION

Mobile communications can be considered one of the disruptive technologies of the 1990s. In that time, two different mobile technologies were launched: Global System for Mobile Communications (GSM) and Code Division Multiple Access (CDMA). The former was made available to the public for the first time in Finland in 1991. CDMA, on the other hand, was first adopted by South Korea and the United States in 1993. In the latter country, the first CDMA network was launched in 1995. Due to several factors, one of it being design and features of mobile phones, GSM networks slowly became the dominant technology in the world. By 2010, GSM networks covered 80% of the worldwide mobile subscribers [33]. The success of GSM served as the baseline for the development of the Universal Mobile Telecommunications System (UMTS) and Long Term Evolution (LTE) technologies, respectively known as 3G and 4G, which are currently gaining importance among consumers due to their improved (Internet) data rates. Still, GSM remains as the incumbent due to the large amount of mobile phones which depend on this technology. This importance keeps attracting researchers, especially from the security area, to find new vulnerabilities in the various algorithms, protocols and services that are part of this technology.

Since its conception, GSM documentation related to the technology’s architecture, services and communication protocols has been widely available to the public. Security related information such as authentication and cryptographic algorithms, however, were kept secret to avoid any possible attacks to the GSM infrastructure, including mobile phones. This situation motivated the security community to figure out how these algorithms worked and possible vulnerabilities contained in them. In 1998, researchers Briceno, et al. [15] reversed engineered and cryptanalyzed the authentication and key agreement algorithms $A_3$ and $A_8$, enabling them to find the master key used by the network and mobile phones, therefore allowing cloning of SIM cards [16]. Additionally, during the analysis of these two algorithms it was discovered the GSM network does not authenticate to mobile phones, enabling researchers to impersonate a GSM operator. Finally, in the 2000 decade, researchers reversed engineered and cryptanalyzed the $A_5$ encryption algorithm use to keep conversations secret, allowing them to find the key and decrypt the conversations [13, 14].

The identified flaws in GSM’s cryptographic algorithms opened the possibility that not only that part of the system was vulnerable, but also other components as well. This led researchers to analyze the
various services offered by GSM, especially when executed in mobile phones. Each mobile phone manufacturer develops its own implementation of the GSM standard based on the services it decides to support. Moreover, some services are left to the mobile manufacturer to implement them as they see fit. This decision increases the likelihood that a GSM implementation or set of them contain some implementation flaws, especially when a manufacturer decides to support only parts of a service. This can lead to a lack of proper checking procedures of the data being received in a message or the structure of the message itself. In such cases, a mobile phone may not gracefully handle the received message, resulting in the triggering of unexpected or undefined behavior such as a Denial-of-Service (DoS), system crash or reboot caused, for example, by a buffer overflow.

At first, analyzing the GSM services in mobile phones proved to be challenging for two reasons: (1) source code of GSM’s implementation in a mobile phone is completely closed and (2) messages must be transmitted to mobile phones requiring usage of GSM infrastructure, which becomes prohibitively expensive if such a type of equipment has to be purchased. This forced researchers to develop their proprietary environments emulating a GSM network. However, this process was time consuming and difficult for other researchers to recreate. The open source community identified this problem and addressed it by developing applications which emulate a GSM base station. There are currently two projects which perform this function: OpenBSC [40] and OpenBTS [37]. The former has the main disadvantage of still requiring some type of proprietary GSM base station which are not widely available and can be quite expensive (€2000 - €5000). OpenBTS, on the other hand, can be used with a Software Defined Radio (SDR) device known as Universal Software Radio Peripheral (USRP). This device along with the transceivers and antennas required to set up a GSM base station are developed and sold by Ettus Research [45] for around €1500. The advent of this software and hardware significantly lowered the entry barrier to setup a GSM network.

With the GSM infrastructure problem solved, the issue of access to mobile phones’ source code to analyze its possible flaws still remains. Researchers overcame this challenge by using a technique known as fuzzing [32, 50]. Fuzzing is mainly used to test protocols and applications, by transmitting automatically generated random information. If proper checking procedures are not in place, such as input validation in a web site, unspecified behavior can be triggered possibly making the application vulnerable to an attack. The same concept applies for testing the GSM implementation in mobile phones. By choosing a GSM service such as Short Message Service (SMS) or Multimedia Messaging Service (MMS), messages are crafted to contain invalid data or undocumented structure, and sent to mobile phones. Again, if procedures in charge of checking the structure or contents of re-
ceived messages are not present, undefined behavior can be produced which may cause a negative effect in the mobile phone, for example, a complete system crash. Fuzzing is ideal for testing GSM implementations since access to source code is not required, in this way treating mobile phones as a black-box. Any visible undefined behavior is observed through the devices’ screen.

Previous research has extensively utilized fuzzing to test the implementation of the SMS and MMS services in mobile phones. This thesis implements fuzzing to test the Cell Broadcast Service (CBS) implementation. The CBS is used by governments in Europe, United States and Asia to rapidly communicate preventing measures to the population in the presence of natural disasters such as hurricanes or tornadoes. This service is very similar to SMS, in the sense that a text message is received at mobile phones. The difference is that all users located in a geographical area receive the same message, most likely at the same time. Hereafter, we refer to such messages as broadcast messages.

The CBS was designed in the 1990s, but it started to gain relevance after the fatal tsunami at Sri Lanka in 2004. When the service was released, no specifications were made to establish how messages have to be processed, displayed and managed at mobile phones. This forced mobile phone manufacturers to develop their own broadcast message processing algorithms. This problem was addressed in 2009, when a basic set of requirements specifying how broadcast messages must be dealt with upon reception was released. This means that all mobile phones released before 2009 and which support the CBS, contain a proprietary algorithm to process broadcast messages. Therefore, it is very likely that outdated CBS implementations do not process messages according to the current requirements and do not contain the adequate validation functions to properly handle an invalid message.

1.1 RESEARCH QUESTION

The research question which acts as the motivation for this research is the following:

Is it possible to trigger unexpected behavior such as a Denial-of-Service in mobile phones by transmitting crafted broadcast messages?

To address this question, we investigate to find programming flaws in the Cell Broadcast Service implementation of different mobile phones by fuzzing them. Fuzzing takes place by transmitting crafted broadcast messages to mobile phones and observe if unspecified behavior such as a Denial-of-Service is triggered. All of this is possible by setting up a fake GSM network, formed by OpenBTS and a USRP-1.
1.2 CONTRIBUTION AND SCOPE

This thesis first provides the theory of how the CBS works in the GSM standard as well as the structure of broadcast messages. This knowledge is then used to identify which fields are better suited for the fuzzing tests. Finally, those fields are modified in the OpenBTS source code and transmitted to mobile phones connected to our USRP-1 to observe their behavior.

1.2.1 Contribution

The main contributions of this thesis are the following:

- We examine in detail the Cell Broadcast Service, covering the requirements the GSM network must fulfill to offer this service as well as the expected behavior when a broadcast message is received in mobile phones.

- We explain the difference between normal and emergency broadcast messages, clearly stating what characterizes each type of message and how they are transmitted from the network to mobile phones.

- We provide a working setup of software and hardware that allows the testing of the Cell Broadcast Service, using OpenBTS version 2.5.4 and a USRP-1.

- To the best of our knowledge, we are the first in performing a security analysis of the Cell Broadcast Service through the use of fuzzing.

1.2.2 Scope

In this thesis, fuzzing is only performed to mobile phones capable of receiving broadcast messages. Mobile phones which do not support the CBS were discarded. The reason of this decision is due to the optional nature of the service, where if a manufacturer decides not to support it, a mobile phone is not capable of listening to the channel that carries the broadcast messages, therefore rendering futile any efforts to fuzz those mobile phones.

The Cell Broadcast Service uses two types of messages: normal and emergency. The former is transmitted using a channel called the Cell Broadcast Channel and the latter through the Paging Control Channel. Due to technical constraints, the normal type are the only broadcast messages used for the fuzzing analysis.
1.3 RESEARCH METHODOLOGY

The methodology of this research can be divided in two main stages:

- **Theoretical:** In this stage all GSM documentation related to the Cell Broadcast Service was gathered. Most of this documentation is up to date, with some documents being modified as recently as March 2013. This research is based on those current specification versions.

- **Practical:** A setup comprised of OpenBTS version 2.5.4 and USRP-1 was configured. The OpenBTS software was modified to add a command that allowed us to send broadcast messages at will. In addition, other modifications were made to this software to be able to perform the fuzzing tests with invalid messages.

1.4 OUTLINE

The rest of this thesis is organized as follows:

**Chapter 2** discusses the previous work performed in testing the quality of GSM implementations in mobile phones.

**Chapter 3** introduces GSM, covering its network architecture as well as the difference between physical and logical channels. A special mention is made to the Cell Broadcast Channel which is in charge of conveying the broadcast messages.

**Chapter 4** examines in detail the Cell Broadcast Service, how it integrates to the GSM network and the actions that each GSM node has to perform to deliver a broadcast message. Additionally, a clear differentiation is made between normal and emergency messages and the process involved to deliver each of them to mobile phones.

**Chapter 5** discusses the types of protocol fuzzing schemes along with their respective advantages and disadvantages.

**Chapter 6** presents the provisions required to perform the fuzzing experiments. An explanation of why certain fields in a broadcast message are better candidates than others for the fuzzing experiments is provided as well.

**Chapter 7** discusses the results of the fuzzing experimentation as well as other interesting findings.

**Chapter 8** presents the conclusions of this thesis and discusses the future security research we think it can be done over the Cell Broadcast Service.
Different approaches exist to find programming flaws in software applications. One of the most common is static analysis, which consists of directly analyzing an application’s source code. Tools such as SAL/PREfast [25] as well as JML annotations [30] help developers in this task by analyzing the source code syntax and how information flows among the different functions. This allows developers to find syntax mistakes and possible errors in the type of data that is provided to functions. While static analysis can be effective, it requires access to the source code, which is not feasible when analyzing GSM implementations in mobile phones. When this is the case, an analysis must be carried out that treats the application of interest as a black box.

Techniques such as equivalence partitioning, boundary value analysis [49] and state transition analysis [24] allow to find programming flaws in software applications when access to the source code is not available. The first technique consists in the design of use cases to reduce the amount of test cases that have to be performed. For example, considering a web site’s text box that only accepts numeric values of 1 to 100, one test case defines all values in that range to be valid inputs. For the test, one number in the 1 to 100 range is selected. If a different number is chosen in the same range, the outcome of the experiment will be the same. Thus a single test is necessary for valid inputs. This same procedure is done for invalid inputs. In this case, two other test cases are defined. One for all values above 100 and the other for all values below 1. Again, only one value in those ranges is chosen for the experiment and it is assumed that the result will be the same if a different value is chosen. The boundary value analysis is very similar to equivalence partitioning, the difference being that values selected for the tests must be from the edges of the range. Considering the same range of 1 to 100, the test cases will be to try numbers (a) 1 and 100, (b) 0 and 99 as well as (c) 2 and 101.

State transition analysis, on the other hand, requires modifying the sequence of messages sent to the victim, to observe if this change in order affects the device or application. Fuzzing is very similar to the aforementioned techniques. Fuzzing requires the definition of templates for generation of test cases that can lead to the triggering of unspecified or undefined behavior. Additionally, fuzzing provides the great advantage that it can be automatically implemented, increasing the execution speed of all test cases. For this reason, fuzzing has
emerged as one of the preferred methods to perform security analysis of applications when access to the source code is unfeasible.

The idea of examining GSM security through the use of fuzzing is not new and has been executed by different researchers in the past. One of the first fuzzing tests was performed by Mulliner et al. [35] in 2006. The authors focused their efforts on the security testing of the Multimedia Messaging Service (MMS). Due to the lack of software-defined radio in the time, the researchers built their own testing environment which includes a virtual MMS service where mobile phones connected to it using a wireless LAN connection instead of connecting to a GSM network. In this environment, the group of researchers fuzzed the header and body of MMS messages by putting different types of data in both parts. The altered MMS messages were sent to a Windows PocketPC mobile phone which on reception did not know how to handle these messages, leading to the triggering of several buffer overflows.

In 2009, during the BlackHat Conference, Mulliner presented a set of programming bugs found in three smart phones through the use of fuzzing of the SMS [34]. The three smart phones available for this research were an iPhone, an Android Phone and a Windows Phone. To perform fuzzing, an application was developed for each of the three platforms, which makes it possible to directly generate and inject SMS messages into the phones’ modems. Through this application, researchers were able to make the device believe that an SMS was just received from the GSM network. Like in his previous research, only this time using the SMS service, Mulliner crafted several SMS messages with different values in their body and header. As a result, each device showed different behavior ranging from a crashed module to a complete system crash.

Later in 2009, during the Chaos Communication Congress, Harald Welte [52] presented three alternatives to perform GSM fuzzing. The first fuzzing approach Welte proposes is to fuzz the network from a mobile phone. In the time it was impossible to perform such test, but now thanks to the OsmocomBB project [41] it is possible to do so. The second alternative is to fuzz mobile phones by using a rogue base station running OpenBSC or similar software. In this case the target is the GSM implementation in mobile phones. The third alternative which is also Welte’s choice, consists in implementing an injection proxy that is located between the base station and the rest of the GSM network, to fuzz both the network and mobile phones. This can be accomplished by using OpenBSC and Scapy, a program which manipulates packets of different protocols.

Both of Mulliner’s approaches have the main disadvantage that other researchers cannot recreate the experiments, since his are proprietary solutions. Welte’s approach fixes this problem by using the open source project OpenBSC. However, as it was stated before, the
hardware required for OpenBSC is quite expensive, ranging between €2000 and €5000. Both price and openness issues were overcame by Brinio Hond [27] in 2011. His research takes full advantage of software defined radio by using GNU Radio in combination with OpenBTS and a USRP-1 to find vulnerabilities in the GSM implementations of sixteen mobile phones. He fuzzed the devices using the SMS and Call Control GSM services. His findings ranged from icons displayed on the top of the mobile phones’ screen to fully Denial-of-Service, not allowing mobile phones to receive more SMS messages. This thesis takes the same approach as Hond, although the fuzzing takes place in the Cell Broadcast Service.
This chapter provides an overview of the GSM system, covering the history of this mobile communication standard and its core network. It also looks at the radio interface which makes use of physical channels (radio frequencies) which in turn are used by logical channels to convey the user and control information. All this serves as the required basis to introduce the logical channel of interest for this research, the Cell Broadcast Channel (CBCH). This channel carries the broadcast messages from the network to mobile stations and thus, it is used to perform the fuzzing tests. The CBCH is analyzed in depth in the next chapter. Readers familiar with the GSM concepts and architecture may skip this section.

### 3.1 Background

In the 1980s, the European region had several analog systems which were based on similar standards (e.g. NMT 450), but were not compatible due to the usage of different frequencies [47]. To solve this problem, in 1982 the European Conference of Postal and Telecommunications Administrations (CEPT) established the Groupe Spécial Mobile committee to develop a complete digital voice cellular system. The system soon took the name of GSM and in 1989 the Groupe Spécial Mobile was transferred to the European Telecommunications Standard Institute (ETSI) [48]. In 1998, with the desire of collaboration between the United States, Asia and Europe to create a common third generation system to allow faster data communications and improved core network, the 3rd Generation Partnership Project (3GPP) was born, which also took under its control the GSM specifications and is currently in charge of performing any upgrades [26].

Initially, the GSM system was deployed in Europe on the 890-915 MHz spectrum for uplink communication (mobile phone to base station) and 935-960 MHz for downlink (base station to mobile phone); this system is known as GSM 900. Later versions of GSM use the 1800 MHz spectrum (1710-1785 MHz uplink, 1805-1880 MHz downlink) known as Digital Cellular System (DCS) and the 1900 MHz spectrum (1850-1910 MHz uplink, 1930-1990 MHz downlink) mainly used in North America and known as Personal Communication Service (PCS) [47].

Currently, GSM networks are being complemented by future mobile generations such as third generation UMTS and forth generation...
LTE. However, GSM is still widely used worldwide covering as much as 80% of the world’s population in 212 countries [33].

3.2 SYSTEM ARCHITECTURE

Figure 1 shows the GSM system architecture. The Mobile Station (MS) communicates with the base station, called the Base Station Transceiver (BTS). The BTS contains the antennas and amplifiers required to connect with mobile stations. A BTS forms what is known as a cell and it usually covers distances between 800 meters and 4 kilometers. The Base Station Controller (BSC) is in charge of managing the radio resources that are assigned to a BTS or set of BTSs attached to it; among its main tasks are to perform the radio channel setup, handover\(^1\) and frequency hopping\(^2\). A BSC can manage tens or even hundreds of BTSs. Together, the BTS and BSC form the radio access network [19].

The Mobile Switching Center (MSC) provides all the switching features of a switch in the fixed telephone network along with extra functionalities to deal with mobile users such as authentication, management of radio resources, location updates and handover. A GSM network administered by a single Mobile Network Operator (MNO) usually has several mobile switching centers each of them managing a large geographical area such as a city. To provide its services, the MSC relies on other systems. The Authentication Center (AuC) is a database containing all the users’ security-related information such as cryptographic keys to perform operations such as authentication and encryption. The Equipment Identity Register (EIR) is a database that stores mobile devices identification information, i.e the International Mobile Equipment Identity (IMEI) numbers. Since mobile stations can be easily stolen, anyone can make use of a stolen device by inserting a valid SIM card. The EIR keeps a blacklist of all stolen or locked devices. It also keeps a whitelist of all devices allowed to access the network. The AuC along with the EIR form the management network [19, 47].

The Home Location Register (HLR) is another database which contains the mobile users’ information such as the services they are subscribed to (Internet, call forwarding), their current location and unique International Mobile Subscribers Identity (IMSI). All this data is necessary to find users in the GSM worldwide network. The Visitor Location Register (VLR) is a subset of the HLR, containing only the

---

1 In North America, this is commonly known as handoff and it is the process of assigning a new radio channel to a user that moves from one base station to another.
2 GSM is a system that utilizes two mechanisms to access the shared radio resources: Time Division Multiple Access (TDMA) and Frequency Division Multiple Access (FDMA). For the latter, a hopping frequency scheme is defined and communicated to mobile devices allowing them to use different frequencies to reduce interference effects. For the interested reader, an excellent reference is [47].
users that are currently under the management of a MSC. In the event of users joining a new MSC, the VLR copies their data (e.g. HLR address, IMSI) from the HLR [47].

Finally, the Gateway MSC (GMSC) manages all connections to the fixed telephone network, technically known as Public Switched Telephone Network (PSTN). The MSC, HLR, VLR and GMSC compose what is known as the core network [19, 47].

3.3 Interfaces

Within the GSM architecture, interfaces are defined that establish how the different nodes communicate with each other. In Figure 1 it can be seen the $A_{bis}$ interface connects the BSC and BTS while the $A$ interface connects the BSC with the MSC. The BTS and MS communicate through the $U_m$ interface also known as air interface.

The focus of this research project is on the air interface since it carries the broadcast messages that are used for the fuzzing experiments discussed in Chapter 7.

3.3.1 The Air Interface

GSM divides the radio frequency spectrum in 200 kHz channels called carriers which convey the user voice and data traffic. These channels are divided in time frames which in turn, are further divided in 8 time slots (Figure 2).

Both user and control data are transmitted in small pieces called bursts. These bursts are transmitted inside time slots. Each burst has

3 The various GSM components are usually connected by wired links, but can be connected by wireless links as well. The air interface term is used to specify the link connecting BTSs and mobile stations.
a total size of 546.5 μs and contains 148 bits. The rest of the 30.5 μs available in the time slot are used as a guard space to avoid overlapping with other bursts due to propagation factors affecting the reception of bursts at the mobile station\(^4\).

![GSM channel, frame and time slot](image)

GSM defines five types of bursts. Figure 2 shows the most common, the **normal burst** which is in charge of transmitting user data. The **frequency correction burst** allows the mobile phone to adjust the used frequency to avoid interference with other channels. The **synchronization burst** allows the BTS to synchronize the mobile stations in time. The **access burst** is used during the initial connection between a mobile station and a BTS and the **dummy burst** is used if no data is needed to be sent in that particular slot.

These bursts are transmitted through the air interface using physical channels. Physical channels represent a small piece of the frequency spectrum and are used to convey the information that is carried by logical channels. In the remainder of the section we discuss the characteristics and differences between physical and logical channels.

---

\(^4\) In wireless communications, unlike their wired counterpart, signals follow different paths, finding in their way from the source to the destination several objects which attenuate, reflect and refract them. At the destination, signals arrive at different times; the guard bands prevent the interference between received bursts that can arise for arriving with a time shift.
3.3 Interfaces

3.3.1 Physical Channels

A GSM physical channel is defined in terms of frames and time slots. A physical channel is a time slot, therefore, there are 8 available physical channels within a frame (200 kHz carrier).

3.3.1.2 Logical Channels

Within physical channels, logical channels are defined to perform different functionalities. GSM defines two logical channel categories: traffic and control channels. The former is in charge of carrying all user-related data such as speech, while the latter are used for medium access control, allocation of traffic channels or mobility management (e.g. handover) [47]. Logical channels are organized in certain combinations which are then mapped to a physical channel. More specifically, a particular combination of logical channels can be assigned to only one physical channel.

There are two types of Traffic Channels (TCHs): Full-rate TCH (TCH/F) and Half-rate TCH (TCH/H). TCH/F has a bit rate of 22.8 kbit/s while TCH/H has only a data rate of 11.4 kbit/s. When GSM was initially deployed, the voice codecs were able to create speech at 13 kbit/s, requiring a TCH/F channel for its transmission; the remaining capacity was used for error correction. Future developments allowed the creation of better codecs which compressed voice to fit in a TCH/H channel at the expense of lower voice quality. Still, mobile networks operators prefer to implement TCH/F channels5.

Control channels are more extensive than traffic channels and they are further divided in 3 main categories: broadcast, common and dedicated control channels. Below we provide a brief description of the specific channels within these categories.

Broadcast Channels (BCHs) The channels in this category are used only on the downlink, i.e. in the communication that takes place from the BTS to the mobile station. These channels broadcast information to all mobile stations. Among the information carried by BCHs are frequency, timing and system information [36].

- Frequency Correction Channel (FCCH): This channel uses the frequency burst full of all-zero data which is used for the sole purpose of allowing mobile stations to find the Broadcast Control Channel (BCCH) frequency. Once the BCCH was identified, the mobile can access the Synchronization Channel (SCH) [36].

- Synchronization Channel (SCH): The Synchronization Channel always comes after the FCCH and it carries information that identifies a particular BTS, preventing in this way interference

5 The only advantage of TCH/H channels is better bandwidth management. These channels are preferably used in special circumstances such as heavy network load.
in the event a mobile station receives two SCH channels from two cells that use the same BCCH frequency [36].

- **Broadcast Control Channel (BCCH)**: Meanwhile the FCCH and SCH provide with frequency and timing information to mobile stations, the BCCH carries the data necessary for the mobile station to use the services of the network. Among the information transmitted are the Cell Identity, Location Area Identity and BCCH frequency used in neighboring cells [36].

**Common Control Channels (CCCHs)** Channels under this category are common to both the network and mobile stations and are used on specific situations. For instance, in the uplink (communication from MS to BTS) common channels are used for random access, to request the allocation of a dedicated channel, and in the downlink to respond to a random access request by assigning a dedicated channel or to request mobile stations to initiate random access by paging them, e.g. due to an incoming phone call [36].

- **Random Access Channel (RACH)**: To be able to obtain a dedicated channel, mobile stations must request them. This is done using a random access request which is a message that is sent using an access burst through the Random Access Channel (RACH). The usage of the term random access is to characterize the fact that the network does not know when or who is going to make a request to access its services [36].

- **Access Grant Channel (AGCH)**: The AGCH carries the reply to a request made by the mobile station on the RACH channel. A successful reply involves the allocation of a dedicated channel on the AGCH. The allocated channel is for the mobile station’s exclusive use. Some of the information contained in the allocation reply are the frequency, timeslot, channel mode (control or traffic channel) and the random reference containing the frame number and random number used in the random access request [36].

- **Paging Channel (PCH)**: This channel is used to call for the attention of mobile stations to inform of an incoming attempt of communication, e.g. an instant message or phone call. The data carried on the PCH mainly includes the mobile station identity, in the format of a Temporary Mobile Subscriber Identity (TMSI) or IMSI. Since mobile stations cannot know when a message or call is coming, the mobile station must listen to all paging channels at all times [36].

- **Cell Broadcast Channel (CBCH)**: Even though the term “broadcast” is contained in the name of this channel, it is considered a
common control channel. The CBCH channel transmits information such as weather or traffic reports to all subscribers within a cell. Currently, it is also used to broadcast emergency information in case of natural or human disasters. This channel is assigned to the SDCCH to transmit the broadcast messages [36]. This channel is the focus of this thesis project and the details of how it is used to provide the Cell Broadcast Service are discussed in the following chapter.

**Dedicated Control Channel (DCCHs)** These channels are used to achieve different types of tasks. For example, the signaling required to set up a traffic channel is done using dedicated control channels. Furthermore, once a traffic channel has been allocated, it is necessary to ensure the channel has acceptable operational levels such as signal quality; this is performed by exchanging measurements data through DCCHs. Also, there are cases where a mobile station does not require a traffic channel to perform certain kind of operations like locations updates, which are done through dedicated channels [36].

- **Standalone Dedicated Control Channel (SDCCH):** This channel is used for the exchange of signaling data between mobile stations and the network. This signaling data may be used to set up a traffic channel. An SDCCH channel is allocated to a mobile station as a successful reply to a random access request done on the RACH channel. Once the mobile stations performed the desired task (allocate traffic channel or send location update), the allocated SDDCH channel is immediately released [36].

- **Slow Associated Dedicated Control Channel (SACCH):** Once a mobile station has a traffic channel allocated, the exchange of signaling data takes place on a SACCH channel. It is referred as slow, because the data rate requirements are of such nature, needing 480 ms to deliver a single message. Examples of the type of information exchanged on this channel are the signal quality measurements sent from the mobile stations to the BSC which in return provides power control information to the MS. In addition, based on measurements supplied by a mobile station, this channel is used if a handover must take place [36].

- **Fast Associated Dedicated Control Channel (FACCH):** There are occasions where the exchange of signaling data needs to be performed in a fast manner, for example, a handover of a mobile station located in a high speed vehicle. In this case, a SACCH channel is not enough, thus a Fast Associated Control Channel is used. A FACCH makes use of a normal burst to convey the required signaling information. When this is done, the transmission of user data is interrupted and a “stealing” flag is set in the normal burst to indicate mobile stations that
signaling data is being carried. Considering this behavior, it can be said that all traffic channels can be used as FACCH channels [36].

3.3.1.3 Logical Channel Combinations

As it was seen in the previous section, logical channels are used to accomplish different tasks. The GSM specification [6] defines several logical channels combinations that are allowed to be used in a time slot (TS).

One important example is time slot TS 0 because it carries the BCCH channel which conveys the required information for mobile stations to identify a particular cell. Time slot TS 0 can have two possible channel combinations [36]:

- **Non-combined**: In this scheme, the FCCH + SCH + BCCH + CCCH channels are contained in TS 0 (on different frames) and SDCCH/8 is contained in TS 1. It is referred as non-combined due to the separation of SDCCH channels from BCCH and CCCH channels, each of them occupying a different slot.

- **Combined**: In this configuration, the FCCH + SCH + BCCH + CCCH are contained in TS 0 along with SDCCH channels, i.e. the BCCH/CCCH channels are combined with SDCCH channels in the same time slot.

The above are just two examples of the possible combinations that can be done in time slot 0. Table 1 shows other possible combinations and the time slots where they can be used [36]

It is worth noting that the channel of interest, the CBCH channel is not present in any of the combinations. This is due to the fact that it uses the same resources as the SDCCH channel, more specifically, it “steals” four time slots of the SDDCH channel. Therefore, it can be seen that in combination one from Table 1, the SDCCH/4 channel then would be a CBCH channel since those 4 time slots will be used by the latter to send a broadcast message. Details of how a message is broadcast using the CBCH channel are discussed in the next chapter.

---

6 Here CCCH indicates that any common control channel can be used.
7 The number after the diagonal refers to the amount of time slots used by the channel.
   This means that in 8 consecutive frames, time slot 1 will contain a SDDCH channel
<table>
<thead>
<tr>
<th>Channel Combination</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FCCH + SCH + BCCH + CCCH + SDCCH/4 + SACCH/4</td>
<td>Combined configuration which is applicable to TS 0</td>
</tr>
<tr>
<td>SDDCH/8 + SACCH/8</td>
<td>Combinations that can be done in any time slot and is the standard SDCCH channel configuration</td>
</tr>
<tr>
<td>TCH/F + FACCH/F + SACCH/TF</td>
<td>Combination that can take place in any time slot and it is referred as the full-rate channel. The FACCH channel is included in the description, to indicate that at any moment a TCH can be used to transmit control data.</td>
</tr>
<tr>
<td>TCH/H + FACCH/H + SACCH/TH</td>
<td>Combination that applies to any time slot and is referred as the half rate channel.</td>
</tr>
</tbody>
</table>

Table 1: Common GMS channel combinations
This chapter focuses on the Cell Broadcast Service that allows the creation, transmission and delivery of broadcast messages to mobile stations. The first section introduces the relevance of the CBS for governments and civil protection organizations to broadcast emergency messages to a region’s population in the presence of catastrophes. Later, the requirements that must be fulfilled by mobile stations when receiving an emergency message are presented. These requirements function as a baseline to the analysis performed in Chapter 7. Next, a definition of the control messages used by the CBS is given. In addition, a description of the nodes which are added to the GSM architecture to be able to offer the CBS is provided. Finally, the last three sections of this chapter provide a detailed explanation of the process involved to deliver both normal and emergency broadcast messages to mobile stations. This process is lengthy and comprises a large amount of information. For this reason, the reader is advised to read all of the sections but to remember the details of the Interface BTS-MS section, since it contains the necessary information to understand the remaining of this document.

4.1 BACKGROUND

The Cell Broadcast Service was developed in parallel to the SMS service as a response of mobile developers to the competing paging services being offered in 1990 [51]. The SMS service became very popular among mobile users allowing them to exchange messages with a single user or a group of them. On the other hand, mobile operators did not find a business case for the CBS, making both mobile network operators and mobile developers to neglect the implementation of the service in their equipment. However, this service has been gaining importance in the last decade due to the need of governments to broadcast information in the presence of an emergency by relying in this communication channel, besides the usual media such as radio and television.

The CBS is already being used in different regions around the world [46]. During hurricane Sandy, the United States government used the CBS to send evacuation messages to citizens in the New York area. In Sri Lanka, after the Indian Ocean Tsunami disaster of

---

1 Examples of business cases were to provide news, weather and traffic information to mobile users. Due to the current CBS gain in popularity, these services are again being considered to be offered together with broadcast of emergency information.
2004, the government decided to implement an emergency system that enabled the broadcasting of warning and disaster information; the system has been operational since January 2009. In Europe, the Netherlands performed a CBS technology test from 2005 to 2007 [28] to evaluate the broadcasting messages penetration in different geographical areas. In November 2012 the NL-Alert website\(^2\) went live to inform mobile users how to configure their devices to receive broadcast messages; the service started operations in mid-2013. Japan and South Korea also implement the CBS to inform its citizens to evacuate an earthquake affected area or before a tsunami occurs.

Each of the above countries defines and implements its own warning systems that rely on the CBS to deliver emergency information. In Japan, the warning service is known as Earthquake and Tsunami Warning System (ETWS), while in the US it is called Commercial Mobile Alert System (CMAS). In South Korea, the system receives the name of Korean Public Alert System (KPAS). Whereas, the European Union named its emergency service EU-ALERT, where every country appends its own country acronym to identify it; for example, NL-ALERT for the Netherlands, UK-ALERT for the United Kingdom and so on.

Due to the diversity of technical specifications of each warning system, the ETSI with the aid of the 3GPP developed a standardized system known as Public Warning System (PWS). The PWS is based on Japan’s ETWS, with just minor modifications in how messages are delivered to mobile stations [46]. The initial goal of the PWS was to bring a standard emergency and warning communication infrastructure as well as specific technical requirements for mobile phones within the European Union. However, due to its standardized nature this system and its accompanying protocols can now be implemented worldwide. This allows roaming users to receive broadcast messages no matter what their location is, as long as they are in a GSM coverage area.

4.2 GENERAL PWS REQUIREMENTS

Like every other service, 3GPP defines general requirements that must be fulfilled by the warning system as well as by the mobile devices. For the latter, features such as behavior when receiving a warning or emergency message and battery usage expectations are defined. This section describes some of the requirements for the PWS system as well as for mobile devices which later serve as a baseline to determine if they are fulfilled when the mobile devices are fuzzed.

\(^2\) http://www.nederlandveilig.nl/nl-alert/
4.2.1 Public Warning Service Requirements

The PWS is in charge of broadcasting Warning Notifications to mobile stations camping\(^3\) in different cells in certain geographical areas. Among its features are the capability to send Warning Notifications to several mobile devices simultaneously; on reception, the PWS must not expect an acknowledgment from the mobile stations. Furthermore, the PWS must be able to support simultaneous broadcast of various Warning Notifications and their transmission is performed on a first in, first out basis. Finally, Warning Notifications must be transmitted only in situations where considerable human and property damage may occur and action on part of the population is required [1].

4.2.2 Mobile Stations Requirements

Among the features that must be supported by mobile stations are that Warning Notifications must be displayed without requiring user interaction and delivery of the Warning Notification must not interfere with an ongoing voice or data session. Moreover, mobile stations in idle mode must be able to receive Warning Notifications and implement duplication detection, allowing the discarding of messages already received by the mobile station. Finally, mobile stations must not allow the forwarding of received Warning Notifications to other mobile stations [1].

One interesting remark in the specification is the recommendation of not including URLs in Warning Notifications which redirect mobile users to a website containing information regarding how to act in case of emergency. Even though this is not recommended, it is still possible to do it, posing a clear security risk, since an attacker could redirect users to a malicious website containing misleading information or a virus, infecting a great amount of devices at once.

4.3 Type of Messages

The Public Warning System supports the broadcast of messages in CMAS, ETWS, KPAS and EU-ALERT formats. Each of those platforms define different categories and priorities to their messages. ETWS implements the most straightforward categories and are widely used through 3GPP specifications as a reference. For this reason, ETWS message categories are used for the rest of this research document, unless stated otherwise.

ETWS defines two message types: ETWS Primary Notifications (or simply Primary Notifications) and Secondary Notifications. The for-
The primary category is used to carry small amount of information which can be transmitted rapidly through the network. The type of information contained in this category should only indicate the imminent occurrence of a disaster, for example, a tsunami.

Secondary notifications, on the other hand, contain large amounts of information in the form of text or audio to indicate mobile users how to act after a disaster or catastrophe occurred. A map can also be provided to signal possible evacuation routes to be followed as well as to indicate locations where shelter and food can be found.

Primary Notifications have a higher priority than Secondary Notifications. The former must have a maximum transmission delay from the network to mobile stations of 4 seconds even in congested conditions and Secondary Notifications follow after 20 seconds of the primary notification being transmitted [46]. It is important to point out the usage of these notifications is optional, i.e. both notifications do not have to be used; in some instances Primary Notifications are enough and in others Secondary Notifications are sufficient to carry the required warning information. The implementation depends on the regulatory requirements of a certain country or region. For example, the EU-ALERT system only implements Secondary Notifications.

Finally, Secondary Notifications are divided in two categories: normal and emergency. This differentiation is done by changing a bit in the broadcast message and does not affect its priority, i.e. Secondary Notifications of type emergency are not transmitted faster than the normal type. Rather, this distinction is made to indicate mobile stations how to handle broadcast messages upon reception. More details of these differences are provided in Appendix B.1.1.

4.4 INTEGRATION WITH GSM

In GSM, the PWS service is provided by the Cell Broadcast Service which is implemented at the Cell Broadcast Center (CBC). The CBC is in charge of receiving the messages sent by the Cell Broadcast Entity (CBE). On reception, the CBC formats the messages by adding certain parameters and forwards it to the BSC which in turn sends the message to a specific set of BTSs that finally broadcast the message to the mobile stations currently camping on those BTSs. Figure 3 shows the architecture with a cell broadcast center integrated into the GSM core network.

Interfaces define how communication takes place among the different components when a CBC node is added to the GSM core network.

---

4 This system defines several categories within Secondary Notifications. One example is the Amber Alerts, used to inform the population when a child goes missing. However, the various EU-ALERT message categories are transposed to ETWS types to remain within the PWS standard.
The communication between the CBC and the BSC is established by 3GPP TS 23.041 [3], between BSC and BTS by 3GPP TS 48.058 [8] and between the BTS and mobile stations by 3GPP TS 44.012 [4]. The interface between the CBE and CBC is not standardized, leaving its implementation to each mobile network operator.

4.4.1 Communication Channels

Recall there are two categories of channels in GSM: traffic and control channels (Section 3.3.1). The CBS relies on control channels, making the service reliable even under heavy network traffic conditions since Warning Notifications are not broadcast on channels where user voice and data are transmitted.

The Cell Broadcast Channel (CBCH) is in charge of carrying the Secondary Notifications from the BTS to the mobile stations. As it was mentioned before (Section 3.3.1.3), the CBCH, in turn “steals” physical resources assigned to the Standalone Dedicated Control Channel (SDCCH) to deliver messages. Primary Notifications, on the other hand, are transmitted through the Paging Channel (PCH), allowing a faster delivery since mobile devices must listen to this channel continuously.

There are two types of CBCH channels: basic and extended. The latter channel is optionally supported in both the mobile device and the network, thus making the basic channel the most used\(^5\). The only advantage of supporting the extended channel is to increase throughput by being able to broadcast two Secondary Notifications simultaneously, one in the basic channel and the other on the extended channel. Further details on how CBCH and PCH channels are used to deliver Warning Notifications are provided in Section 4.7.

---

\(^5\) If a mobile station’s GSM implementation does not support the CBS, then it will never listen to either the basic or extended channel.
4.5 INTERFACE CBC - BSC

Both, the CBC and BSC are expected to perform certain operations when dealing with broadcast messages. The following subsections describe the functionalities of both entities along with the protocols used between them to allow proper dealing of Warning Notifications.

4.5.1 CBC and BSC functionalities

A CBC can be connected to several CBEs as well as to various BSCs allowing a mobile operator to cover various cells in a wide geographical area. Once a message is received from CBEs, the CBC must prepare the broadcast messages before they are transmitted. Moreover, the CBC must perform other managerial tasks such as communicating the BSC its desire to delete or modify a message waiting to be transmitted; specify to the BSC the geographical area to broadcast a message; determine the amount of messages to be transmitted and how often they should be transmitted as well as differentiate between ETWS and CBS messages, so the BSC provides the proper transmission priorities.

On the other hand, a BSC can only be connected to one CBC while being linked to several BTSs. The BSC must offer certain services to the CBC, among them scheduling the CBS messages to be transmitted on the CBCH channel; indicate to the CBC when it is not possible to execute the specified message repetition period; forward broadcast messages to the relevant BTSs and provide feedback to the CBC in case of successful or failed operations.

4.5.2 Communication Protocols

When the CBC and BSC communicate under normal operational conditions, the CBC will always start communication; in case of failure or error, the BSC will be the one initiating communication. The CBS specification, defines several Elementary Procedures (EPs) to indicate to either the CBC or BSC the type of operation to be performed. Elementary procedures are divided in two categories: Class 1 and Class 2. The EPs in the former class are always sent by the CBC and in return, the CBC expects a successful or unsuccessful response from the BSC. Class 2 EPs, on the other hand, are always sent by the BSC and a response message is not expected from the CBC. The EPs of classes 1 and 2 are available in [3] and [7].

---

6 In the GSM Cell Broadcast Service documentation, the nomenclature changes when referring to Primary and Secondary Notifications; “ETWS message” is used to referred to the former and “CBS message” to the latter. For the remainder of this document, this terminology is used to make a differentiation between them and “broadcast message” is used as a generic term.
For the purpose of this thesis, the focus is on the class 1 *Write-Replace* elementary procedure because it functions as the starting point to a request of the CBC towards the BSC to broadcast a message. While the other elementary procedures are important as well, their main function is to complement the Write-Replace operation, by signaling a successful or failed operation, to end an ongoing transmission or to request channel load information.

### 4.5.2.1 Write-Replace Procedure

The purpose of the Write-Replace procedure is to signal the BSC when the CBC desires to send a new message or replace a current message being broadcast with a new message.

**Successful Write-Replace Procedure** The communication protocol between the CBC and BSC when a successful Write-Replace operation takes place, is depicted in Figure 4.

The CBC initiates communication through the *Write-Replace* procedure by sending a WRITE-REPLACE message to the BSC. The WRITE-REPLACE message contains the request to transmit a new broadcast message or to replace an ongoing message with a new broadcast message. Depending on the type of message to be broadcast, the WRITE-REPLACE message will contain different Information Elements (IEs) to differentiate between a CBS and ETWS message; however, regardless of which type of message is broadcast, the communication protocol remains the same between these two entities. Table 2 lists some of the IEs contained in a CBS and ETWS WRITE-REPLACE message.

![Figure 4: A successful CBS Write-Replace procedure](image)
For both, CBS and ETWS WRITE-REPLACE messages, the presence of the \textit{Old Serial Number} IE determines the type of operation the BSC must perform, i.e. write or replace. If the \textit{Old Serial Number} is not present, the BSC interprets this as a write operation where the CBC desires to transmit a new message. In this case, a new CBS message is identified by the \textit{Message Identifier} IE, the \textit{New Serial Number} IE, the \textit{Cell List} IE and the value in the \textit{Channel Indicator} IE (basic or extended); a new ETWS message is identified by the same IEs in a CBS message minus the \textit{Channel Indicator} IE. This combination of IEs which uniquely identify both CBS and ETWS messages is referred to as a \textbf{message reference}. These message references are built by the BSC for every cell contained in the \textit{Cell List} IE of the received WRITE-REPLACE message.

If both, the \textit{Old Serial Number} and \textit{New Serial Number} IEs are present in the WRITE-REPLACE message, the BSC understands this as a replace operation, where a current broadcast message being transmitted will be replaced by a new broadcast message. A replaced CBS message is identified by the \textit{Message Identifier} IE, the \textit{Old Serial Number} IE, the \textit{Cell List} IE and the \textit{Channel Indicator} IE; a replaced ETWS message is identified by the same IEs of a replaced CBS message minus the \textit{Channel Indicator} IE. For the replace operation, the BSC must first kill the broadcasting of the CBS or ETWS message to be replaced before it can start transmitting a new message. To be able to perform this action, the BSC consults its list of message references. If terminating the broadcasting of an ongoing message fails, the BSC must send a WRITE-REPLACE FAILURE message back to the CBC.

On a successful write or replace operation, the BSC must broadcast the CBS or ETWS message to all cells contained in the \textit{Cell List} IE. In the case of a CBS message, the message must be transmitted according to the values specified in the \textit{Number of Broadcasts Requested} IE and the \textit{Repetition Period} IE. The former indicates the amount of messages to be transmitted while the latter specifies how often those messages should be broadcast. If the value contained in \textit{Number of Broadcasts Requested} IE is “0”, then the message must be transmitted until the CBC signals the BSC to stop, otherwise, the CBS message should be broadcast for the amount specified in the IE. If the BSC has

<table>
<thead>
<tr>
<th>Normal message</th>
<th>Emergency message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Identifier</td>
<td>Message Identifier</td>
</tr>
<tr>
<td>Cell List</td>
<td>Cell List</td>
</tr>
<tr>
<td>New Serial Number</td>
<td>New Serial Number</td>
</tr>
<tr>
<td>Old Serial Number (optional)</td>
<td>Old Serial Number (optional)</td>
</tr>
<tr>
<td>Channel Indicator</td>
<td>Emergency Indicator</td>
</tr>
</tbody>
</table>

Table 2: Some Information Elements of normal and emergency WRITE-REPLACE message
to broadcast different CBS messages, the BSC decides the order on which these messages will be transmitted.

When the BSC broadcasts an ETWS message, this message optionally contains two IEs: the Warning type IE and the Warning Security Information IE which contain extra information regarding the emergency situation. An ETWS message must be broadcast for the period of time specified in the Warning Period IE or until the CBC signals the BSC to stop. Only one ETWS message can be transmitted at one point in time. If a BSC receives a write operation request for a cell where an ETWS message is being broadcasted, the BSC will reply with a WRITE-REPLACE FAILURE message.

Figure 5 shows the structure of a WRITE-REPLACE message. The numbers at the top of the figure represent one byte (8 bits). Inside the parenthesis, the second parameter indicates the amount of bytes occupied by an information element and the letter represent its presence requirement in the message: mandatory (M), optional (O) or conditional (C1,C2). For example, the Old Serial Number is optional and occupies 3 bytes of the WRITE-REPLACE message.

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Type (M,1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Length Indicator (M,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Identifier (M,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New Serial Number (M,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Old Serial Number (O,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cell List (M,4 + m to 4 + mn)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel Indicator (O,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Category (C1,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repetition Period (C1,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Broadcasts Requested (C1,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Pages (C1,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Coding Scheme (C1,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Content (C1,84)[Up to 15x]</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Emergency Indicator (O,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Warning Type (C2,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Warning Security Information (C2,51)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Warning Period (C2,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 5: Write-Replace message format

All these information elements will be used by the CBC depending of the type of CBS message that needs to be sent, i.e. a CBS or ETWS message. For both types, the Message Type IE, the Length Indicator IE, the Message Identifier IE, the New Serial Number IE and the Cell List IE are required. The Message Type IE indicates the type of command that is being sent, which in this case is WRITE-REPLACE. The Length In-
**Message Identifier** IE specifies the amount of bytes comprised by all information elements that form a WRITE-REPLACE message. The **Message Identifier** IE is used to identify the source and type of a broadcast message; for example, source can be “meteorological department” and type “weather report”. The **New Serial Number** IE along with the **Message Identifier** IE, **Channel Indicator** IE and the **Cell List** IE are used by the BSC to construct the message reference that uniquely identifies a CBS message. All those IEs except for the **Channel Indicator** IE are used to identify an ETWS message. The **Cell List** IE identifies the cell or group of cells that must receive a broadcast message. If the Write-Replace procedure is executed successfully in all cells contained in the **Cell List** IE, the BSC must send a WRITE-REPLACE COMPLETE message to the CBC. However, if the broadcast message fails to be sent successfully in at least one cell, the BSC must send a WRITE-REPLACE FAILURE message to the CBC. Details of these two messages are explained later on this subsection.

Finally, the conditional presence requirements C1 and C2 indicate under what scenarios those information elements must be added by the CBC to the WRITE-REPLACE message. The information elements with presence C1 are only added when the CBC desires to transmit a CBS message. On the other hand, the IEs labeled with presence C2 are only added when the CBC needs to broadcast an ETWS message.

**Write-Replace Complete Message** When all cells in the **Cell list** IE of a WRITE-REPLACE message receive the broadcast message, the BSC sends to the CBC a WRITE-REPLACE COMPLETE message including the **Message Identifier** IE, the **New Serial Number** IE and if received in the WRITE-REPLACE message, the **Channel Indicator** IE.

Depending on the type of request, the WRITE-REPLACE COMPLETE message contains either the **Cell List** IE or the **Number of Broadcasts Completed List** IE. If the WRITE-REPLACE COMPLETE is sent to the CBC as a response to:

1. a write request, i.e. the CBC desires to transmit a new CBS message, then the **Cell List** IE is included containing the Cell Identifiers of every cell that received the broadcast message.

2. a replace request to terminate the broadcasting of an ongoing CBS message, then the **Number of Broadcasts Completed List** IE is included containing the number of broadcasts of the replaced CBS message for each cell.

3. a replace request to terminate the broadcasting of an ongoing ETWS message, then the Cell Identifier of each cell that received the ETWS message is included in the **Cell List** IE.

---

7 If the **Channel Indicator** is not present, the BSC assumes a value of basic CBCH channel.
The Old Serial Number IE is included in the WRITE-REPLACE COMPLETE message only in cases two and three of the above list.

When the Number of Broadcasts Completed List IE is included in the WRITE-REPLACE COMPLETE message (case 2) and the amount of messages broadcast to a specific cell is unknown, then the Number of Broadcasts Completed field must be set to “0” and the Number of Broadcast Info field is set to “unknown”. On the other hand, if the amount of messages broadcast to a specific cell is greater than the value registered in the Number of Broadcasts Completed field, then the value of Number of Broadcast Info field must be established to “overflow” to indicate an overflow in the counter of that particular cell. Figure 6 shows the structure of a WRITE-REPLACE COMPLETE message.

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Type (M,1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Length Indicator (M,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Identifier (M,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New Serial Number (M,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Old Serial Number (O,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Broadcasts Completed List (O,4 + m + 3 to 4 + (m + 3)n)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cell List (O,4 + m to 4 + mn)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel Indicator (O,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 6: Write-Replace Complete message format

The WRITE-REPLACE COMPLETE message does not include several information elements contained in a WRITE-REPLACE message but keeps the mandatory IEs, modifies the presence requirement of the Cell List from mandatory to optional and adds the optional Number of Broadcasts Completed List IE. The latter information element is used only when the WRITE-REPLACE COMPLETE message functions as a response to a replace operation of a CBS message. In this case, the Number of Broadcasts Completed List IE contains the amount of replaced CBS messages broadcast to each cell and the Old Serial Number IE refers to the replaced message that was sent successfully to a set of cells.

The Cell List is only used under two scenarios: as a response to a write operation or as a response to a replace operation of an ETWS message. For both cases, the Cell List IE contains the Cell Identifiers registered in the Cell List of the received WRITE-REPLACE message. For the latter case, the Old Serial Number IE must be included as a reference to the cells contained in the Cell List IE.

The Channel Indicator IE is included in the message only if it is contained in the received WRITE-REPLACE message.
FAILED WRITE-REPLACE PROCEDURE  If the Write-Replace procedure fails in one of the cells contained in the Cell List IE of the WRITE-REPLACE message, the BSC must send to the CBC a WRITE-REPLACE FAILURE message including the Failure List IE which contains the cell or set of cells where the Write-Replace procedure failed along with a Cause value, specifying the reason of the failure for each cell.

Just like in a successful write or replace operation where the IEs contained in a WRITE-REPLACE message determine the type of IEs that will be included in the WRITE-REPLACE COMPLETE message, the IEs in a WRITE-REPLACE FAILURE depend on the contents of the WRITE-REPLACE message sent by the CBC. If the WRITE-REPLACE message contains:

- only a New Serial Number IE which indicates a write operation, where the CBC desires to send a new message and a message reference that is already in use at the BSC, then the Write-Replace procedure must be considered as failed for this particular cell or set of cells. In this case, the BSC must send a WRITE-REPLACE FAILURE to the CBC including the Failure List IE containing the Cell Identifier of the specific cell(s) along with the Cause value “Message-reference-already-used”. No additions are made in the Number of Broadcasts Completed List IE or Cell List IE for this cell or group of cells.

- both a New Serial Number IE and an Old Serial Number IE, indicating the BSC a replace operation and a message reference that is unknown to the BSC, then the BSC must consider the Write-Replace operation as failed for this particular cell or set of cells. Under this scenario, the BSC must send a WRITE-REPLACE FAILURE message to the CBC including the Failure List IE containing the Cell Identifier of the specific cell(s) along with the Cause value “Message-reference-not-identified”. The Old Serial Number is also contained in the response message as a reference to the message that failed to be terminated in the cells included in the Failure List IE.
  No additions are made in the Number of Broadcasts Completed List IE or Cell List IE for this cell or group of cells.

- both a New Serial Number IE and an Old Serial Number IE as well as a message reference that is already in use at the BSC, then the BSC must consider the Write-Replace procedure as a failure for this particular cell or set of cells. In this case, the BSC must send a WRITE-REPLACE FAILURE message including the Failure List IE containing the Cell Identifier of the specific cell(s) along with the Cause value “Message-reference-already-used”. An entry at the Number of Broadcasts Completed List IE is made for the terminated CBS message for this cell or group of cells.
The New Serial Number IE and the Old Serial Number IE are both included in the response message as well. The former serves a reference to the new message that failed to be transmitted in the cells included in the Failure List IE and the latter as a reference to the message that was successfully terminated in the cells listed in the Number of Broadcasts Completed List IE.

Figure 7 shows the structure of WRITE-REPLACE FAILED message.

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Type (M,1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Length Indicator (M,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Identifier (M,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New Serial Number (M,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Old Serial Number (O,3)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Failure List (M,4 + m + 1 to p + m + 1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Broadcasts Completed List (O,4 + m + 3 to 4 + (m + 3)n)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cell List (O,4 + m to 4 + mn)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel Indicator (O,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 7: Write-Replace Failure message format

The WRITE-REPLACE FAILURE message has the same structure of the WRITE-REPLACE COMPLETE; the only difference is the addition of the mandatory Failure List IE. Moreover, depending on the desired operation, write or replace of a CBS or ETWS message, the contents of the WRITE-REPLACE FAILURE message change accordingly. If the failure message is a response to a write request, the Failure List IE contains the set of cells where the write operation failed and for each cell a Cause value is added describing the reason of the failure. If the Cell List IE is included in the failed response, it contains the set of cells where the write operation was successful.

If the WRITE-REPLACE FAILURE message is a response to a replace request, the Failure List IE contains the set of cells where the kill operation failed to terminate an ongoing broadcast message and for each cell, a Cause value is attached to describe the reason of the failure. If the Number of Broadcasts Completed List IE is present, it contains the group of cells where the kill operation succeeded to terminate an ongoing normal CBS message and the Old Serial Number is included as a reference to the replaced CBS message that was successfully broadcast to a set of cells. Finally, if the Cell List IE is present, it contains the set of cells where the kill operation succeeded to terminate the broadcast of an ongoing ETWS message and the Old Serial Number IE makes a reference to the cells contained in the Cell List IE.
Figure 8: Successful ETWS message write operation

The WRITE-REPLACE message includes all the mandatory IEs: 
*Message Type IE, Length Indicator IE, Message Identifier IE, New Serial Number IE and Cell List IE*. On receipt, the BSC realizes the CBC desires to send an ETWS message due to the presence of the *Emergency Indicator IE*, thus also expecting the presence of the *Warning Type IE*, the *Warning Security Information IE* and *Warning Period IE*. Finally, before responding to the CBC, the BSC constructs the message reference and broadcasts the CBS message to the cells indicated in the *Cell List IE* (not pictured).

After a successful write operation, the BSC builds the WRITE-REPLACE COMPLETE message to be sent to the CBC. In this example, the BSC sends a response including the same mandatory IEs from the WRITE-REPLACE message. Except for the *Length Indicator IE*, the rest of IEs in the response message contain the same information received in the information elements of the WRITE-REPLACE message. On reception of the WRITE-REPLACE COMPLETE message, the CBC sends an acknowledgment to the corresponding CBE.
So far the communication protocol used between the CBC and BSC to broadcast a new message (Write-Replace operation) has been analyzed. This section goes a step further, by examining the protocol and messages exchanged between the BSC and BTS.

4.6.1 Handling of CBS Messages

On reception of a WRITE-REPLACE message, the BSC determines the CBC’s desire to transmit a CBS message due to the presence of the Channel Indicator information element which differentiates it from an ETWS message along with the Cell List IE, Repetition Period IE, Number of Broadcast Requested IE, Message Content IE and other information elements (Figure 5).

Immediately after, the BSC uses the information available in the Cell List IE to identify the cells that must receive the CBS message. Subsequently, it sends 4 SMS BROADCAST REQUEST messages (Appendix A.4) or 1 SMS BROADCAST COMMAND message (Appendix A.3) to the involved BTSs. With SMS BROADCAST REQUEST operational mode the BSC splits the 88 bytes long CBS message into four 22 bytes long frames, adds a sequence number to each frame and transmits them to the BTS using four SMS BROADCAST REQUEST messages which in turn are sent to the mobile stations by the BTS through the CBCH channel (Figure 9).

When the BSC utilizes SMS BROADCAST COMMAND operational mode, the entire 88 bytes long normal CBS message is transmitted to the BTS in a SMS BROADCAST COMMAND message. On reception, the BTS splits the message in 22 bytes frames, adds a sequence num-
bers to each of them and transmits the resulting frames to the mobile stations on the CBCH channel (figure 10).

![Diagram](image)

**Figure 10:** SMS BROADCAST COMMAND operational mode [3, 8]

On both operational modes, the BSC is in charge of scheduling, queuing and transmitting the CBS messages taking in consideration the traffic load at the CBCH channel while trying to comply with the Repetition Period and Number of Broadcast Requested values established by the CBC. If the load in the CBCH channel is too high, the BTS can send a CBCH LOAD INDICATION (Appendix A.1) message to the BSC, indicating an overflow state and the amount of $m$ message slots it has to wait before it can resume sending CBS messages again. In case the BTS can handle more messages, it can send a CBCH LOAD INDICATION message, specifying an underflow state and the amount of $m$ scheduled CBS messages the BSC can immediately transmit. The BSC then proceeds to send the amount of requested messages and it later continues transmitting CBS messages according to its timetable. If the BTS requests more messages than those in the BSC queue, the BSC must transmit only the amount of messages it possesses [8].

The usage of SMS BROADCAST COMMAND or SMS BROADCAST REQUEST messages to transmit CBS messages is left as an implementation decision to each mobile network operator.

### 4.6.2 Handling of ETWS Messages

In this scenario, the BSC receives a WRITE-REPLACE message with the Emergency Indicator IE set to differentiate it from a CBS message along with the Cell List, Warning Type and Warning Period information elements. Soon after, the BSC consults the information contained in the Cell List to identify the cells that shall receive the ETWS broadcast message. Next, the BSC creates the ETWS message that is broadcast through the PCH channel to the mobile stations. Details of this process are described in the following section.
4.7 INTERFACE BTS - MS

This interface is by far the most important for this project, since it is the one that defines and carries the broadcast messages that are received by mobile devices. The knowledge of how this interface works is fundamental to understand and identify the possible flaws that can be exploited through fuzzing.

To facilitate the understanding of how broadcast messages are transmitted to the mobile stations, a clear distinction is made between the processes involved for CBS and ETWS messages.

4.7.1 CBS Message Transmission

In the previous section, it was established that a normal CBS message is 88 bytes long and it is transmitted from the BSC to the BTS in its entirety or in blocks of 22 bytes. However, how this message is created at the BSC from the information received in a WRITE-REPLACE message was not defined. Figure 11 shows the structure of the CBS message that is built at the BSC.

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Serial Number (2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Identifier (2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Coding Scheme (1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Page Parameter (1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Content of Message (82)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 11: Normal CBS message format

The Message Identifier IE, Serial Number IE and Data Coding Scheme IE are obtained from the WRITE-REPLACE message sent by the CBC to the BSC. The Page Parameter IE indicates the page currently being sent and the amount of pages comprising the CBS message; a normal CBS message can contain up to 15 pages. Refer to Appendix B for the details of the IEs composing the a CBS message.

The above CBS message is appended to 4 SMS BROADCAST REQUEST messages or to 1 SMS BROADCAST COMMAND message and sent to the BTS. On reception, the BTS retrieves the CBS message, splits the message in four frames (in case SMS BROADCAST COMMAND is used), appends a Block Type header on top of each frame to identify the message being sent and broadcasts the frames to mobile stations. Figure 12 depicts the format of the Block Type header.

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spare</td>
<td>LPD</td>
<td>LB</td>
<td>Sequence Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 12: Block Type header format
The Spare bit is always set to zero by the sender. It is kept in the specification for future improvements and the message must not be discarded if the bit is set to one.

The Link Protocol Discriminator (LPD) is always set to “01”, which is the ID identifying a CBS message. If these bits are set to other values, the message must be discarded by the MS since it is expecting a cell broadcast message.

The last frame in a CBS message containing user information is signaled by the Last Block (LB) bit. When the LB is set to zero, it indicates the upcoming frame contains information; if the bit is set to one, the following frame(s) do(es) not contain information.

Lastly, the sequence number indicates the type of frame being transmitted. Table 3 lists the possible sequence number values.

<table>
<thead>
<tr>
<th>Block type</th>
<th>Bit Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>First block</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>Second block</td>
<td>0 0 0 1</td>
</tr>
<tr>
<td>Third block</td>
<td>0 0 1 0</td>
</tr>
<tr>
<td>Fourth block</td>
<td>0 0 1 1</td>
</tr>
<tr>
<td>First schedule block: it contains</td>
<td>1 0 0 0</td>
</tr>
<tr>
<td>scheduling information</td>
<td></td>
</tr>
<tr>
<td>Null message: contains invalid CBS</td>
<td>1 1 1 1</td>
</tr>
<tr>
<td>message</td>
<td></td>
</tr>
</tbody>
</table>

Table 3: Sequence number values for different block types

Once the frames are ready to be transmitted, the BTS must indicate the mobile stations that it desires to transmit a CBS message and they should start listening on the correct channel. This is accomplished by first sending a SYSTEM INFORMATION TYPE 4 message to the mobile stations on the BCCH channel, which contains among several parameters the CBCH Channel Description IE and the CBCH Mobile Allocation IE. The former information element allows the mobile station to locate the CBCH channel, i.e. which SDCCH time slots are being used by the CBCH while the latter specifies the frequency where the CBCH is to be found. Now the mobile stations are ready to receive the CBS message. Figure 13 illustrates a complete transmission of a CBS message, from the time the CBE requests a message transmission until the message is delivered.

The CBE, besides the message it desires to deliver, must provide the geographical area (cells) where the CBS message must be received. On message reception, the CBC takes that information and forms a Cell List containing the list of cells that will broadcast the message. Before forwarding the request to the BSC, the CBC adds other important information to the WRITE-REPLACE message, such as a unique
Figure 13: Successful broadcasting of a CBS message

identifier (*Message Identifier* IE and *New Serial Number* IE), the channel where the message is to be transmitted (basic or extended defined in *Channel Indicator* IE), how many and how often messages must be transmit (*Number of Broadcast Requested* IE and *Repetition Period* IE, respectively), the language of the message (*Data Coding Scheme* IE), the actual contents of the message (*Message Content* IE), among others. The BSC in turn receives this information and uses the *Message Identifier* IE, *New Serial Number* IE, *Data Coding Scheme* and *Message Content* IE to build the CBS message to be sent to the mobile stations; it also adds the *Page Parameter* IE. Next, the BSC sends this message to the BTS on a SMS BROADCAST COMMAND message, which on reception it realizes the BSC is requesting to send a new CBS message. This triggers the BTS to send a SYSTEM INFORMATION TYPE 4 message on the BCCH channel to inform the mobile stations a new CBS message will be broadcast. Once the mobile stations are listening to the CBCH channel, the devices receive the message in four frames which were created by the BTS. On reception, mobile stations consult their topic list to verify if the users desire to read messages coming from this specific source (CBE). If the topic is registered in the mobile stations, then they ensure the received message is not a duplicate...
and notify the users a new message has been received (details of how
the topic list is used and how duplication detection is performed are
provided in Section 4.8).

Since the BSC is in charge of the scheduling and transmission of the
CBS message, it also utilizes the Repetition Period and Number of Broad-
cast Requested information received on the CBC request to transmit
the CBS message in the periodicity and amount specified by those
two values. Finally, the BSC notifies the CBC of a successful opera-
tion by sending a WRITE-REPLACE COMPLETE message. The CBC
in turn informs the CBE of the successful operation with an acknowl-
edgment (ACK).

CBS messages must be received by mobile devices only when they
are in idle mode. If the mobile device is busy on a voice call or trans-
ferring data packets, the CBS message is received once those con-
nections are terminated. This contrasts with ETWS messages, which
must be received both, in idle and connected modes (an explanation
of mobile operational modes is available in Appendix C).

4.7.2 ETWS Message Transmission

As it was mentioned above, ETWS messages have a higher priority
and thus are required to be transmitted faster than their counterpart,
the CBS messages. The GSM specification requires ETWS messages to
be transmitted on the Paging Channel (PCH) which implies the usage
of different radio resource management and messaging between the
BSC, BTS and MS. This section explores in detail how ETWS messages
are transmitted to mobile stations.

Just like CBS messages, the BSC builds an ETWS message which is
the one sent by the BTS and received at mobile stations. The format
of an ETWS message can be observed in Figure 14.

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial Number (2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Identifier (2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Warning Type (2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Warning Security Information (50)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 14: CBS emergency message format

The Serial Number IE and Message Identifier IE uniquely identify an
ETWS message. All of the information elements are obtained from
the received WRITE-REPLACE message sent by the CBC to the BSC.
The Warning Security Information IE must be ignored by the mobile
station but it is still included for backwards compatibility [3]. Once
the message is formed, the BSC puts it in the P1 Rest Octets IE of a
PAGING REQUEST TYPE 1 message (Appendix A.2) and broadcasts
the latter on the paging channel of a cell or groups of cells. The paging
message is broadcast for a period specified by the *Warning Period* IE of the received WRITE-REPLACE message.

If the ETWS message is too large, the message can be fragmented and each fragment is then transmitted in different paging messages. On reception at the mobile station, the mobile device reconstructs the original message. Figure 15 depicts the process involved in delivering an ETWS message to the mobile station.

![Figure 15: Successful broadcasting of an emergency CBS message](image)

In this scenario the CBE must provide more information than when sending a normal CBS message. This time the CBE must send emergency information such as warning type, warning message, impacted area and time period to the CBC. The *warning type* can have the values of earthquake, tsunami, earthquake and tsunami, test or other. The *impacted area information* is used by the CBC to establish which BSCs must be contacted and builds the *Cell List* for the cells that have to receive the ETWS message [3]. Next, the CBC sends a WRITE-REPLACE message to all the identified BSCs including the *Emergency Indicator* IE, allowing the BSCs to determine this is an ETWS message. Immediately after, the BSC constructs the ETWS message and appends it to a PAGING REQUEST TYPE 1 message and sends it to all the cells indicated by the CBC. The ETWS message is broadcast for the amount of time specified on the *Warning Period* IE received on the WRITE-REPLACE message.

When the *warning type* is set to “other”, the *warning message* is transmitted in a CBS message. In this case the ETWS message only carries the *warning type* information while the actual emergency information
or preventing measures are conveyed by a CBS message. On the other hand, if warning type is set to “test”, the mobile stations must ignore the message, unless they are explicitly configured to receive this type of messages.

On reception, the mobile station must emit an alert and immediately display the emergency message without requiring interaction from the end user. In case the mobile station detects a duplicated message, it must ignore it.

Finally, after a successful operation, the BSC must communicate to the CBC the completion of the broadcast process by sending a WRITE-REPLACE COMPLETE message. The CBC in turn must send an acknowledgement to the CBE.

The above example only applies to mobile stations in idle mode (Appendix C). However, as a requirement all mobile stations must received an ETWS message even if they are in dedicated mode, e.g. busy on a voice call. For this to happen, the BSC appends the ETWS message to an APPLICATION INFORMATION (APDU) message, assigns a high priority to it and sends it to the mobile stations using a dedicated channel already assigned to them instead of through the paging channel. If the network has any normal priority messages ready to be sent, the APDU containing the ETWS message is transmitted first due to its highest priority and the rest of messages must wait further in the queue.

4.8 MESSAGE RECESSION AT MOBILE STATIONS

This section briefly explains some of the conditions that must be met for mobile stations to be able to receive both CBS and ETWS broadcast messages.

4.8.1 Duplication Detection

In the above sections, it is repeatedly mentioned that mobile stations discard broadcast messages that were already received. This is due to a duplication detection mechanism that is in place which considers the values of the Serial Number IE and Message Identifier IE. The former is divided in 3 fields, which together uniquely identify a broadcast message in terms of contents and geographical scope. On the other hand, the Message Identifier IE contains information regarding the topics of the broadcast messages. If a broadcast message is transmitted with certain values in these two information elements within a geographical area which were already used in a previous broadcast message in the same geographical space, the message just transmitted is discarded by mobile stations. This is an important feature since

---

9 This is usually the case of mobile stations owned by mobile network operators and used during broadcast test.
it prevents the reception of similar CBS or ETWS messages, reducing significantly the amount of messages received by mobile stations. This is unlike SMS messages, which no matter what their ID or content is, they are always received.

### 4.8.2 Search list

The topics or search list is only used by mobile stations when a CBS message is received. This list contains a set of IDs (or topics) mobile stations take in consideration when receiving and displaying CBS messages. Topics can be registered by users or mobile operators in either the memory of the mobile station or their SIM card. If a CBS message contains a topic not register in the mobile stations’ memory or SIM card, the mobile station must ignore this message and not display it to the user. Broadcast messages from different sources use different topic IDs. Usually mobile network operators decide which type of IDs are assigned to each source. This will depend on the amount and type of mobile stations that each source wishes to reach. For example, mobile stations 5 years and older only support values up to 999, the largest allowable value being 65535. For this reason, the Dutch government decided to use the topic number 919 for its NL-ALERT service, to be able to reach as many mobile stations as possible in emergency situations.

At this point, we know the format of both CBS and ETWS messages as well as how they are transmitted from the CBE to mobile stations. This knowledge will allow us to determine which information elements are good candidates for the fuzzing tests. The best candidates are determined considering certain criterion such as the range of values they accept. All this is examine in detail in Chapter 6. However, before presenting our fuzzing analysis and experimentation, we define what fuzzing is and discuss its associated advantages for a security analysis; this is presented in the following chapter.
INTRODUCTION TO PROTOCOL FUZZING

Fuzzing is the process involving the transmission of automatically generated, uncommon inputs to a target with the purpose of creating unexpected behavior [23]. The unexpected behavior can take different forms; a common example is a Denial-of-Service caused by a buffer overflow which is generated when an input is too long and the application does not have procedures in place to trim the input or report an execution error.

Fuzzing enables the possibility to find programming flaws in an application or protocol implementations, even in extreme situations where both the source code and the protocol specification are not known.

This chapter discusses protocol fuzzing along with its advantages and disadvantages considering the different approaches available, towards finding flaws in a protocol implementation.

The first section gives a short background on the origins of fuzzing and the concept of protocol fuzzing is introduced. Next, the types of protocol fuzzing are discussed along with their respective advantages and disadvantages. The chosen fuzzing scheme for this research project is discussed in Chapter 6.

5.1 BACKGROUND

Fuzzing was created at the University of Wisconsin by professor Barton Miller in 1988. During a class, Prof. Miller and his students developed a command-line fuzzer to test how reliable a Unix system was [32]. This testing technique evolved to find vulnerabilities not only in an operating system, but also to test web sites, embedded systems, networks, protocols and any other system that receives an input.

In protocol fuzzing, taking as an example TCP or IP, the fuzzer or tester analyses the structure of a TCP or IP packet, modifies the information carried by a certain field, injects the modified packet to the network and observes the reaction at the receiving end. The receiver in turn can display a normal or strange behavior. The latter can manifest as a thrown exception, DoS or any other unspecified behavior.

When unspecified behavior is triggered, the fuzzer which can be a developer, researcher or attacker has found a vulnerability which may be exploited. This opens a window of opportunity to each of the aforementioned individuals towards developing a security patch or exploiting such vulnerability to gain some type of benefit.
Considering the protocol of interest of this thesis, the GSM standard (in particular the CBS service) specifies just like TCP or IP, packets containing information elements which can be modified and sent to mobile stations (receivers). On packets reception, it is possible to observe mobile stations’ behavior when they process the packets containing data outside the GSM specification.

Fuzzing of a protocol can be performed in different ways. The chosen scheme depends on the amount of target’s information available to the fuzzer. This information can be the GSM specification itself or the source code which represents the specification’s implementation in a mobile station. The following section describes the protocol fuzzing categories considering the type of information that is available to the fuzzer.

5.2 Types of Protocol Fuzzing

There are mainly two categories of protocol fuzzing, which are distinguishable depending on the type of information available to the fuzzer. If the protocol specification is known and thus used to performed the fuzzing procedure, it is called **smart fuzzing**. However, if this information is not available it is known as **dumb fuzzing** [23]. On the other hand, when fuzzing is performed based on the knowledge of the protocol implementation source code, it is known as **white box fuzzing** while the term **black box fuzzing** is used to refer to fuzzing done in the absence of such information. Considering these fuzzing definitions, it can be said that white box fuzzing implies smart fuzzing since most of the time knowledge of source code implies a knowledge of the specification, either because the protocol specification is available or it is deducted from the available source code [27].

The above two categories of protocol fuzzing can be combined to obtain a more complete fuzzing strategy. Following is an explanation of what each combination does and their respective advantages and disadvantages:

- **Dumb black box fuzzing** is the simplest scheme due to a lack of knowledge of both the protocol specification and the source code. Even though it is the simplest, it is also the least efficient to implement since random data with no identified pattern to follow is input to the implementation. Its main advantage is the simplicity when creating the input random data that, once it has been created, it can be used to fuzz a different protocol implementation or for a completely different protocol. Unfortunately, most protocols define a certain message structure to be accepted and without the knowledge of either the specification or source code, it might require a lot of time to identify the format of the message that is accepted by the specification. Fuzzing tests per-
formed in the past document a 50% lower effectiveness of this scheme when compared with smart black box fuzzing [50].

- **Smart black box fuzzing** is an improvement over the previous scheme, since the protocol specification being used at the target is known. This allows the fuzzer to know the structure of the messages accepted by the specification and in this way modify the various message’s fields to try to find an exception. However, this could also make the fuzzer omit certain combinations or modifications believing it is useless or not necessary to do so. This is the advantage of dumb black box fuzzing over the present scheme, since in the former all random combinations are tried [27].

- **White box fuzzing** implies the knowledge of both the protocol specification and the source code. One might think this is the best fuzzing approach, since all data defining the protocol and its implementation are available. However, by knowing every path of how data flows requires to design a model for each path, to try to find vulnerabilities for every one of them. Techniques such as symbolic execution can be implemented to optimize the model design process. Nonetheless, in cases where a project’s source code is extremely large and the specification is known as well, a first step is to execute a smart black fuzzing and use the source code to see how much code is covered in a test. With this approach, the code coverage is increased in manageable chunks by every new smart black fuzzing test while allowing to find as many vulnerabilities as possible.

The main disadvantage of white box fuzzing is the amount of time required to design all inputs and execute them taking into account the available source code. Moreover, even though the obtained inputs can be ported to another implementation, code coverage is no longer guaranteed [27].

In this thesis, the focus is on fuzzing several GSM protocol implementations, i.e. how the GSM specification is implemented in different mobile stations. This analysis is important because every manufacturer implements its own interpretation of the specification based on the services that it decides to support in its mobile stations. It is in those services where fuzzing takes place, since each of them defines a set of messages and procedures that enable the provision of the service to mobile stations. For this research, the service of interest is the Cell Broadcast Service (CBS) which as stated above, has been gaining importance to broadcast emergency information to mobile stations and as far as we know, has never been examined from a security standpoint. This becomes relevant considering the fact the GSM network never authenticates to mobile stations, opening the possibility for an attacker to impersonate a base station and exploit possible
vulnerabilities contained in the implementation of the CBS in mobile stations.

The following chapter presents the provisions performed before the fuzzing tests can be carried out. Chapter 7 discusses the executed fuzzing experiments over the CBS as well as the results obtained from this experimentation.
This chapter presents and discusses the methodology implemented to perform the fuzzing experiments. The first section defines the methodology itself along with its stages. Next, we define and examine the type of behavior that we may observe on mobile stations during our fuzzing experimentation. Since we can observe any type of behavior, we take as a reference results obtained from previous fuzzing research. Later, based on the specification and source code availability, one of the fuzzing schemes presented in the previous chapter is selected. Afterwards, an analysis is carried out to determine which fields of the broadcast messages are better suited for the fuzzing experimentation. The outcome of this analysis allows us to design an approach to carry out the fuzzing tests.

6.1 Methodology

For this thesis, the main goal of the fuzzing experiments is to attempt to trigger some type of undefined behavior (e.g. Denial-of-Service) in mobile phones by sending malformed CBS messages. To achieve this objective, it is important to first design a methodology that allows us to see the big picture and defines the different phases that must be performed before, during and after the fuzzing experimentation takes place. Figure 16 shows the implemented methodology.

![Figure 16: Methodology followed to perform the fuzzing tests](image)

The methodology is divided in two main stages. In the Fuzzing Pre-requisite stage, we perform all the tasks required previous to the execution of fuzzing tests. The Fuzzing Experiments stage encompasses

---

1 In Chapter 4, we used the term information elements. Information elements are in turn formed by fields. This is explained in Section 6.4.
the execution of the fuzzing tests and the analysis of the results. This chapter covers the former stage while the following chapter deals with the latter stage.

Before performing the fuzzing tests we must have in mind what type of unexpected behavior we want to trigger, which will aid in the design of an approach that allows us to trigger such behavior. This is defined in the Expected MS behavior phase.

Having define what we expect to observe during our fuzzing, the Choose Fuzzing Scheme phase can take place. In this phase, we choose a way to carry out the fuzzing based on the amount of information available regarding the GSM specification as well as the CBS implementation’s source code.

The Identify Fields to Fuzz stage defines the criterion to identify the fields that might be good candidates to carry out the fuzzing; fixed length or limited range of values in fields are examples of parameters that are taken into account to identify the best candidates.

Finally, in the Fuzzing Approach phase we consider the structure of the broadcast messages to design the best approach to carry out the fuzzing tests which will enable us to trigger undefined behavior.

6.2 Expected Mobile Station Behavior

When a vulnerability analysis of an application that runs in a personal computer is performed, security testers have access to a wide variety of tools and information that facilitate this task. Most of the time testers know the programming language used to elaborate the application, allowing them to investigate possible vulnerabilities contained in the compiler itself. Moreover, security testers can make use of a debugger to find memory leaks that could be exploited by a malicious user. Additionally, if the application performs queries to a database using a language such as Structured Query Language (SQL), the tester can search for vulnerabilities in the way these queries are performed. With all this information available to testers, they can determine the possible consequences in the event an attacker exploits the found vulnerabilities. For example, a SQL injection that leads to the disclosure of sensitive information, the triggering of a buffer overflow that causes a Denial-of-Service or system crash or that enables the injection of malicious code.

The above tools and knowledge are not available when searching for vulnerabilities in a GSM implementation. The source code is not available and mobile stations’ platforms are closed, making impossible to use debuggers or similar tools to analyze the mobile stations’ memory. Consequently, we cannot predict what will happen or the type of vulnerabilities that will be found when mobile stations are fuzzed using broadcast messages. However, depending on the crafted broadcast message that we use, it helps to determine the possible
reasons behind the triggering of unexpected behavior. Any behavior that is observed which is not defined in the GSM specification is regarded as *unspecified* or *unexpected behavior*. This unexpected behavior can take any form. It can manifest as a Denial-of-Service, causing mobile stations to stop receiving phone calls and SMS messages or it may completely crash the mobile stations’ system. The unexpected behavior, however, does not always have to be invasive or interfere in the performance of mobile stations. It can simply be a broadcast message containing unreadable content. This is undefined behavior as well because the broadcast message is not being displayed properly.

As discussed in Chapter 2, we are not the first ones to perform fuzzing of GSM implementations in mobile stations. This gives the advantage of knowing what possible behavior we can observe during our fuzzing experimentation. For example, both Mulliner [34] and Hond [27] observed different types of behaviors in their fuzzing experiments using SMS messages. The behaviors ranged from the appearance of random icons on mobile stations’ screens to a module crash to a Denial-of-Service that affects the whole device. In his research, Mulliner triggered a Denial-of-Service in three different mobile stations, each running a version of the Android, iOS and Windows Phone operating systems. The DoS caused different disruptions in each mobile station. Among the observed failures were disconnection from the GSM network (iOS, Android), locking of the SIM card (Android) and complete crash of the graphic interface (Windows Phone). This behavior was caused due to the SMS process execution privileges. By crafting special SMS messages, he forced the SMS process to interfere with the execution of other modules, triggering in this way the Denial-of-Service. Hond, on the other hand, observed that certain mobile stations showed a series of icons on top of the screen. These icons did not degrade the performance of the devices but they could only be removed by restarting the mobile stations. Moreover, by crafting a special type of SMS, it was possible to access random parts of the memory. Every time the malicious SMS was opened, a new part of the mobile stations’ memory was displayed. The author attributed this behavior to a possible buffer overflow. Additionally, a buffer overflow was also the cause behind mobile stations system reboot.

The above examples serve as a baseline for the possible behavior that we may observe as well during our experimentation. Table 4 lists the unspecified behaviors we expect to trigger, what might cause such behaviors and the possible effect on mobile stations. From the table it can be seen that buffer overflows are the main cause behind mobile stations unspecified behavior. We expect to trigger a buffer overflow when we craft a broadcast message which contains more data than
specified in the CBS documentation. Mobile stations might not contain a checking procedure in place that verifies the length of the data carried by the broadcast message and as a consequence, the array in charge of storing the message overflows. The buffer overflow in turn might manifest as a Denial-of-Service, module or system crash, affecting in a way mobile stations. On the other hand, when mobile stations receive broadcast messages in a different sequence than expected, we might trigger a module crash if a function in charge of checking the order of such messages is not present.

<table>
<thead>
<tr>
<th>Behavior</th>
<th>Cause</th>
<th>Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>Denial-of-Service</td>
<td>Buffer Overflow</td>
<td>Affect general functioning of the device</td>
</tr>
<tr>
<td>System crash</td>
<td>Buffer Overflow</td>
<td>Reboot of mobile stations</td>
</tr>
<tr>
<td>Module crash</td>
<td>Buffer Overflow</td>
<td>Stop receiving broadcast messages</td>
</tr>
<tr>
<td></td>
<td>Out of order msgs.</td>
<td></td>
</tr>
</tbody>
</table>

Table 4: Unspecified behavior expected to be triggered

The details of how we plan to trigger the listed behaviors are explained in the last section of this chapter. First, we discuss the fuzzing scheme chosen for our research and we identify the fields of the broadcast message where fuzzing can take place.

6.3 FUZZING SCHEME

In our particular case, full knowledge of the protocol is available since we know the structure of both the ETWS and CBS messages and how they are broadcast from the base station (USRP-1) to mobile stations. However, access to mobile stations’ CBS implementation source code is not available, rendering unfeasible to know how data flows and how it is handled by each mobile station. Therefore, we cannot use white box fuzzing. Moreover, even if the source code is available, white box fuzzing would be too time consuming since a model for every path the data takes must be designed. On the other hand, we can make use of dumb black box fuzzing, since it is the simplest scheme; however, we would not be benefiting on the availability of the CBS specification. Consequently, the best scheme is smart black box fuzzing. In this way, we take full advantage of knowing the ETWS and CBS messages structure which enables us to identify the best fuzzing fields candidates.

6.4 IDENTIFY FIELDS TO FUZZ

Each information element is formed by a set of fields (Appendix B). These fields carry the data that define an information element, mak-
Identifying fields to fuzz depends on different factors. Usual fuzzing candidates are fields specifying the length of an information element. A larger or smaller value may create a buffer overflow or similar problems if proper checking procedures are not in place. Another approach is to leave the length field untouched and modify the information element contents making it to carry more or less data than the one registered in the length field. GSM specifies minimum and maximum length for most fields, thus modifying those values might trigger unspecified behavior.

Other attractive fields are those deprecated but which must be present to remain compatible with previous specifications versions. Modifying values in these fields can produce unexpected behavior in mobile stations, especially in those with newer CBS implementations. A newer implementation might not support deprecated fields and thus may react in a strange manner when receiving a broadcast message containing such type of fields.

Finally, fields with a limited range of values can prove to be useful, since values considered to be reserved for future use or not specified in the specification can be input to observe the mobile stations reaction.

Considering the above observations, Figure 17 depicts with background gray color the fuzzing candidates for a CBS message. Unfortunately, due to a lack of support of Rest Octets in OpenBTS version 2.5.4, it was not possible for us to broadcast ETWS messages, limiting the fuzzing activities to CBS messages. We leave fuzzing using ETWS messages as a possible future work.

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spare</td>
<td>LPD</td>
<td>LB</td>
<td>Sequence Number</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Serial Number (2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Identifier (2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Coding Scheme (1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Page Parameter (1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Content of Message (82)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 17: CBS message fuzzing candidates

Following is an explanation providing the reasons of why the highlighted elements of the CBS message are good candidates for the fuzzing experiments.

### 6.4.1 Block Type Fields

From Section 4.7.1, we know that a CBS message is split in four frames and a Block Type header is added to each frame, so on reception, mobile stations can identify and assemble the CBS message. Each Block Type header is comprised of a Spare bit, two bits repre-
senting the Link Protocol Discriminator (LPD), a Last Block (LB) bit and a Sequence Number formed by 4 bits. The Spare bit should always be zero; the LPD must be “01”; the LB should be one only for the last block carrying user data and the Sequence Number should follow a sequence from 0000 to 0011 (Table 3, page 38). Due to the constraints impose in the values that each field can take, it opens the possibility to try different ones for the fuzzing tests.

6.4.2 Serial Number

The two byte long Serial Number (Appendix B.1.1), is formed by the Geographical Scope (GS), the Message Code and the Update Number fields. The GS takes two bits to indicate the coverage of the broadcast message considering the geographical area. All possible values the GS can have are defined, thus discarding it as a fuzzing candidate. The Update Number takes up to four bits to specify any changes that have been made to the content of a CBS message since the first time it was transmitted. Like the GS, all values the Update Number field can take are well defined, eliminating it as a possible field for the fuzzing experiments. However, the Message Code field which occupies ten bits, can be used to differentiate between normal and emergency messages (refer to Appendix B.1.1 for details). In the former case, the Message Code indicates the source and type of a CBS message. When the latter is true, only two out of ten bits are used to specify the emergency nature of the broadcast message. The two bits are identified by the Emergency User Alert bit and the Pop-up alert bit which define how messages are displayed upon reception at the mobile stations. The remaining eight bits must be filled with a certain pattern of padding bits. This padding bits can be a good target for fuzzing, by using a different pattern of that specified in the GSM documentation.

6.4.3 Data Coding Scheme

The Data Coding Scheme (Appendix B.1.3) specifies the language of the message being transmitted. This allows mobile users to specify the broadcast messages they desire to receive based on the language. This information element may contain only a certain range of values, giving the opportunity to use values outside that range for the fuzzing tests.

6.4.4 Page Parameter

The one byte long Page Parameter (Appendix B.1.4) is used to indicate mobile station the amount of pages comprising the CBS message

---

3 The GSM specification defines that if the Spare bit has the value of 1 instead of 0, the broadcast message should be received. Still, it is good idea to check if this holds.
and the current page being transmitted. These two values offer the possibility to “trick” the mobile station by signaling a certain amount of pages while in reality a different amount of pages are being sent. Additionally, we can use this field to modify the order on which the pages comprising the CBS message are sent. For instance, indicate the current page is number two while mobile stations expect page number one.

6.4.5 Content of Message

This information element is formed by two fields: the User Information Length and Message Content (Appendix B.1.5). The former is used to indicate the length of the user data contained in the CBS message, i.e. the amount of bytes carried by the Message Content field. The Message Content field can carry up to 82 bytes of user data which equals to 93 7-bit GSM characters (Appendix E). The User Information Length field can be modified to signal a different length of what in reality is being transmitted. Additionally, the length can be left unmodified while the Message Content field is altered to carry more than 82 bytes of data to try to generate a buffer overflow or similar effect.

The Message Identifier (Appendix B.1.2) is not a good candidate since it is necessary it contains the correct data. The Message Identifier is the “topic” that mobile phones listen to. Topics are registered in mobile stations’ SIM card or main memory. Mobile stations ignore all broadcast messages containing message identifiers not registered in their memory or SIM card. Thus, if this information element is fuzzed, broadcast messages will not be received which is undesirable.

6.5 Fuzzing Approach

Based on the characteristics of the CBS message fields, we devised an approach consisting of three main categories where fuzzing can be carried out. The three categories are the following:

- **Length modification**: When feasible, the length of the field can be modified to specify a different length of that in the GSM specification. Fuzzing on the field’s length is only possible when the field does not influence in the reception of broadcast messages. For example, length modification of the Block Type is not possible, since mobile stations ignore all broadcast message containing a Block Type different from one byte.

- **Data alteration**: Under this category falls the usage of reserve values and random inputs. The former is possible when fields
may only take a certain range of values, leaving all other combinations as reserve values. The second is feasible to implement on fields that carry information provided by the user. This allows to test inputs containing random data.

- **Combined:** In this category, the fuzzing is performed on the field’s length and content simultaneously. The only field where this is possible to execute is the Content of Message field. The field can carry random data while signaling a larger or shorter length of what actually is being transmitted.

The above categories encompass the fuzzing that can be done on the structure and contents of the CBS message. With this type of tests we try to trigger a buffer overflow. In all three categories we attempt to overflow the fields by either modifying their length or contents.

In Chapter 2, we made a reference to three mechanisms used to find vulnerabilities when access to the source code is not available. State transition analysis was one of them. This type of analysis implies the alteration on the sequence of messages sent to the victim, to observe if this change in order affects the device or application. This allows to verify for any functions in charge of ensuring messages are received in the correct order. If such a function does not exist, undefined behavior can be triggered since processing of the message will not be properly executed. We can make use of the same concept in our fuzzing tests, by transmitting CBS messages’ pages in a different order, as previously mentioned in the Page Parameter subsection and observe the mobile stations reaction.

At this point, we have the necessary information to start performing the fuzzing tests. The following chapter discusses the setup used to perform the different fuzzing experiments, the details of each experiment as well as the obtained results.
FUZZING EXPERIMENTS RESULTS

This chapter contains the second stage of our methodology. First, a description of the setup used to carry out the fuzzing experiments is provided. Later, the experiments performed in the various mobile stations are explained in detail. During this stage, we analyze some of the random inputs tested along with the modifications done over the identified fields from the previous chapter. Finally, the chapter closes with a discussion of the obtained results and our findings.

7.1 HARDWARE AND SOFTWARE SETUP

The hardware configuration is mainly formed by a laptop and a USRP-1. The former contains the software that enables the USRP-1 to behave as a GSM base station, presenting the $U_m$ air interface to mobile stations. The software used for this purpose is the open source project OpenBTS. The USRP-1, on the other hand, contains the hardware necessary to generate and process the radio signals used to exchange the GSM traffic and control messages between mobile stations and OpenBTS.

For its name, one might assume OpenBTS behaves as the equivalent of a GSM BTS. However, this software with the aid of other applications, enables the host device (laptop) to take the role of a GSM base station transceiver (BTS), base station controller (BSC) and mobile switching center (MSC). This allows to establish phone calls as well as to send and receive SMS messages between mobile stations connected to the USRP-1. Figure 18 shows the OpenBTS network architecture.

![OpenBTS network architecture](image)

Figure 18: OpenBTS network architecture

Asterisk [18] is the interface used to establish phone calls between mobile station in the OpenBTS network. GNU Radio, on the other
hand, functions as the interface between the USRP-1 and OpenBTS, providing the signal processing libraries and drivers to support the hardware installed in the USRP-1.

The USRP-1 makes use of daughterboards, which are transceivers that process signals at different frequencies. There are mainly two categories of daughterboards used for a GSM implementation: (1) RFX daughterboards which can function at 900 or 1800 MHz frequencies covering in this way the GSM 850/900 MHz or GSM 1800/1900 MHz spectrum, respectively and (2) WBX daughterboards, which cover the 50 to 2200 MHz spectrum, making them ideal for different radio applications, including GSM. For this research, two RFX 1800 daughterboards were utilized.

When configuring OpenBTS, it is important to consider its version, since this determines the GNU Radio version that must be used which in turn affects the hardware that is supported. For this research, OpenBTS version 2.5.4 is of interest since it is currently the only OpenBTS release supporting the CBS service. After performing several tests with different GNU Radio versions, we found that GNU Radio 3.4.0 offered the best communication in our OpenBTS network. Table 5 lists the software and hardware configuration used in this thesis.

<table>
<thead>
<tr>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ubuntu 9.04</td>
</tr>
<tr>
<td>OpenBTS 2.5.4 - SMS-CB</td>
</tr>
<tr>
<td>GNU Radio 3.4.0</td>
</tr>
<tr>
<td>Asterisk 1.4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hardware</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laptop Voodoo Intel Core Duo 1.8 GHz, 2 GB RAM</td>
</tr>
<tr>
<td>USRP 1</td>
</tr>
<tr>
<td>2 RFX 1800 daughterboards</td>
</tr>
</tbody>
</table>

Table 5: Software and hardware configuration

Once the right combination of software and hardware allows a successful communication between mobile stations and the USRP-1, it is time to test that mobile phones receive broadcast messages. To test it, two Nokia 3310 mobile stations known for supporting this service were purchased along with two prepaid SIM cards. After several attempts, both phones successfully received the broadcast messages, showing in the screen the message “Info Message Received”; when

1 There are two types of OpenBTS releases: public and commercial. The former is free and developed by the open source community while the latter is maintained by the Range Networks company [37]. The last version of the commercial release fully supports the CBS service, however, due to budget limitations we could not have access to it.
the message was opened, a string of “á” characters was displayed (Figure 19). This string came by default in the OpenBTS source code.

Figure 19: CBS message when received in a Nokia 3310

The reception of CBS messages at the mobile stations was by no means straightforward. We ran into several problems mainly for using the incorrect version of GNU Radio. Appendix F discusses the challenges we faced and how we overcame them.

7.2 FUZZING

There are several tools, known as fuzzers, which facilitate and automate the fuzzing process. Fuzzers such as Sulley, SPIKE or ProtoFuzz provide several libraries which generate different types of random data that is injected to a specified protocol (HTTP, SSH) or application. While these fuzzers are quite effective, they resulted too specialized for our needs and more importantly, did not provide support for GSM. For instance, the aforementioned tools only support generation of data in 8 bits and GSM implements a 7-bit alphabet to carry user data (Appendix E provides an example of message encoding and decoding using the 7-bit alphabet). Additionally, most fields in CBS messages only accept short binary values (maximum of 8 bits), rendering unnecessary the implementation of a specialized fuzzer. As a result, making modifications directly in OpenBTS source code was the best approach. While this course of action may seem time consuming and tiresome, OpenBTS has a very modular structure, requiring us to modify mostly three files, where the composition and transmission of broadcast messages takes place. In addition, using this approach enabled us to easily keep track of the changes made in various field, the data sent and their effect on each mobile station.

7.2.1 Mobile Stations

For the fuzzing tests we had access to eight mobile stations where two of them were of the same manufacturer. These mobile stations were donated or lent by personnel of Radboud University Nijmegen and KPMG. It is important to point out that not all mobile stations were capable of receiving broadcast messages, even in those where
the reception of CBS messages was activated. Table 6 lists the available mobile stations, their respective operating systems, indication of CBS support and if they actually receive the CBS messages.

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Model</th>
<th>Operating System (Firmware)</th>
<th>CBS support</th>
<th>CBS reception</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apple</td>
<td>iPhone 4S</td>
<td>iOS v5.5.1</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Blackberry</td>
<td>9700</td>
<td>BlackBerry OS v5.0</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Nokia</td>
<td>3310</td>
<td>5.57</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Nokia</td>
<td>1600</td>
<td>RH-64 v6.90</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Samsung</td>
<td>Galaxy Note</td>
<td>Android 4.1.2</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Samsung</td>
<td>SGH-A800</td>
<td>A80XAVK3</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Siemens</td>
<td>C55</td>
<td>v18.01.01</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Sony Ericsson</td>
<td>Xperia Ray</td>
<td>Android 2.3.4</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 6: Mobile stations used for fuzzing

From the list, the most recent mobile stations are the Samsung Galaxy Note, Sony Ericsson Xperia Ray, the Blackberry 9700 and the iPhone 4S. Of those devices, only the Galaxy Note and Blackberry support the reception of broadcast messages. At the start of this research, the Galaxy Note contained the Android version 4.0.2. This version as well as earlier Android releases do not support the reception of broadcast messages. For this reason, the Sony Ericsson Xperia Ray was not able to receive this type of messages. Fortunately, Samsung Electronics released a newer Android version 4.1.2 with CBS support, just in time for our fuzzing tests. All iterations of the iOS operating system, however, do not support the CBS. In the Netherlands, the government is currently in conversations with the Apple company to bring support for CBS2. Finally, Blackberry Ltd. supports the CBS since Blackberry OS version 4.1.

Regarding the older models, despite the reception of CBS messages was activated in the Siemens C55 and Samsung SGH-A800, both mobile stations did not receive broadcast messages3. This situation along with the mobile stations which do not support the CBS – iPhone and Xperia Ray – caused a reduction in our sample. As a result, a total of four mobile stations – two Nokias, the Galaxy Note and the Blackberry 9700 – were available to perform fuzzing tests. Still, a variety of devices and operating systems is represented by this small sample.

---

3 Several attempts were executed to try to make these two phones receive the broadcast messages. Unfortunately, they were futile and we did not find a proper explanation of why this was the case.
7.2.2 Fuzzing the Block Type Header

The GSM specification establishes the length of the fields comprising the Block Type header as well as the binary values each of them can take. Table 7 lists a summary (Section 4.7) of the fields’ lengths and allowable values.

<table>
<thead>
<tr>
<th>Field</th>
<th>Length (bits)</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Spare bit</td>
<td>1</td>
<td>Only 0</td>
</tr>
<tr>
<td>Link Protocol Discriminator (LPD)</td>
<td>2</td>
<td>Only 01</td>
</tr>
<tr>
<td>Last Block (LB)</td>
<td>1</td>
<td>1 in 4th frame; 0 otherwise</td>
</tr>
</tbody>
</table>
| Sequence Number            | 4             | Only from 0000 to 0011
Schedule message = 1000
Invalid message = 1111 |

Table 7: Block Type header fields’ values and lengths

To receive broadcast messages, mobile stations expect a Block Type header of length one byte. If the length is different from one byte, mobile stations discard the incoming frame, which is undesirable for our needs. Therefore, fuzzing can only be executed over the Block Type fields’ values. Moreover, due to the limited amount of values each field can take (binary format), the fuzzing can be performed using only reserve values.

Results When the Spare bit is changed to 1, all of the mobile stations successfully received and displayed the broadcast messages, just as it should be according to the GSM specification.

The GSM documentation specifies that a broadcast message must be ignored when the Link Protocol Discriminator (LPD) is different from “0 1”. This was the case with all mobile stations when we transmitted CBS messages with LPD values different from “0 1”.

The Last Block bit must only be 1 in the last broadcast message frame, i.e. in the 4th frame; in the other frames the value must be 0. We performed different tests by shifting the zero values for ones in the first three frames while setting the value zero for the last frame. Interestingly, the CBS message was received until the last frame labeled as 1. For example, if frame 1 had a Last Bit value of 0, frame 2 a Last Bit value of 1, and frame 3 as well as 4 a value of 0, then the CBS message was received until frame 2, since it is labeled as being the last frame; frames 3 and 4 were completely ignored. This was true for all mobile stations.

Finally, for the Sequence Number field, fuzzing was executed following two different approaches. The first approach made use of reserve values. The sequence number can only take 6 values out of 15 available, thus leaving 9 values as reserved. The second approach
consists in changing the order of the sequence number. For example, frame 1 takes the value of 0001 (instead of default 0000), frame 2 equals 0010 and so on. The latter experiment was thought out to test if some type of verification is in place that ensures frames are received in order. When reserved values were used, mobile stations completely ignored the broadcast messages. This same behavior was observed when the sequence number order was different from that specified in the GSM documentation (0000 - 0011).

7.2.3 Fuzzing the Serial Number

The structure of the Serial Number opened the possibility to try different fuzzing combinations. Unlike the Block Type, it is feasible to modify the length of the serial number, without affecting the reception of the broadcast messages. This allows to change the length of the different fields forming this information element. Moreover, the fields' values can be modified as well. However, once again the values available for fuzzing are limited to reserve values. This is due to fields accepting values only in binary format.

results The first fuzzing tests over the serial number focused in the modification of the fields' length, in this manner transmitting fewer or more bits related to the serial number. A consequence of transmitting less bits related to the serial number is a shift of bits, where bits from upcoming information elements occupy the serial number bits positions. This action completely disarranges the structure of the broadcast message. This is also true when the serial number is larger than expected. In this case, serial number bits invade bits positions corresponding to upcoming fields. The bits of the upcoming fields are then shifted towards the bits related to the user data bits. This situation could lead to a buffer overflow, since more bits than expected are being transmitted. However, the results of testing this two scenarios were that of mobile stations ignoring the broadcast messages.

The Message Code field, when used to indicate the CBS message is of emergency category, only uses two bits out of the ten bits available to signal this fact. The remaining 8 bits are padded with the bit pattern “11101110”, following the GSM specification. Any other bit pattern is reserved and should not be used. All these reserve values once again offered the opportunity to use them for the fuzzing experiments. Consequently, we proceeded to use different padding patterns for various tests, trying every time a distinct combination. For all reserve bit pattern combinations, excluding pattern “0000 0000”, CBS messages were received by all mobile stations. In addition, the BlackBerry ignored CBS messages when combination “1111 1111” was used. As a next step, we decided to completely omit the padding bits from
the Message Code field, in this way transmitting only 1 byte of serial number – 2 bits GS, 1 bit Emergency User Alert, 1 bit Pop-up alert and 4 bits Update Number. This modification caused all mobile stations to ignore the CBS messages. The Update Number field was not a good fuzzing candidate since all possible values that it can take are defined in the specification. Additionally, in our tests we detected that when this field was not modified, broadcast messages were not received due to a positive message duplication detection in all mobile stations. This is due to the serial number and message identifier being the same, causing all mobile stations to ignore the messages. It is then desirable to constantly change the value in the Update Number field.

7.2.4 Fuzzing the Data Coding Scheme

The Data Coding Scheme is divided in two parts, each four bits long. The first is used to indicate the type of coding used and the second, the language of the message. These two parts need to be present for the correct reception of broadcast messages, thus fuzzing by modifying the field’s length is not possible. On the other hand, for both parts there are plenty of reserved values. According to the GSM specification, if these reserved values are used, the mobile stations should interpret this as a CBS message using the GSM 7-bit default alphabet and language unspecified.

RESULTS After performing several tests, by trying different combinations of reserved values the Blackberry and the two Nokia phones received, in most of the cases, the CBS messages. For example, when the coding bits (bits 8–5) were set to a value of “0011” and the language bits (bits 4–1) set to any value, CBS messages were not received. In the GSM specification, this particular set of values is defined as being reserved for future use and unspecified handling, which might explain why mobile stations ignored these messages. On the other hand, when the CBS messages were actually received using reserve values different from the values mentioned above, the messages were treated as specified in the GSM documentation, i.e. using the 7-bit default alphabet to display them. Unfortunately, the Galaxy Note did not receive any CBS message containing Data Coding Scheme reserve values.

7.2.5 Fuzzing the Page Parameter

The Page Parameter length of one byte did not offer many possibilities to perform different types of fuzzing tests. In terms of the length, this value cannot be modified since it is necessary to use all bits to specify the amount of pages and current page being sent. However, these
two set of values can be modified to signal a wrong order of pages, a page different of the one currently being sent or to indicate a certain amount of pages while sending a fewer or greater quantity of them. In this field, all values are defined, thus no reserve values can be used for fuzzing.

**Results** In the first fuzzing test, the *Page Parameter* was edited to signal mobile stations that the current page being received is, for instance, the second or third page of a CBS message, but not the first. Several combinations were tried out and the broadcast messages were received only when both the number of pages and current page number were 1. For the other attempts, the broadcast messages were ignored.

The above inspired us to send multiple pages while signaling that the CBS message is formed by only one page, to possibly trigger a buffer overflow. Unfortunately, once the first page of the message was received the rest of the transmitted pages were ignored by all mobile stations. Later, we sent more than 15 pages, however, these extra pages did not have any negative effect as well. By closer examination, we could have guessed that the pages following the 15th page would be ignored since the page numeration has to start at “0000” once the limit is reached (“1111”), thus all mobile stations detect a message duplication, due to the broadcast message containing the same serial number and message identifier.

7.2.6 Fuzzing the Content of Message

This information element offered several attack avenues due to the possibility to use more than binary values, which was the case with all of the previous IEs. Moreover, the User Information Length field is also present to specify the length of the data being transmitted. Also, the Message Content field implements a padding pattern of 5 zero bits to fill up the remaining bits that are not use by user data\(^4\). All of these factors gave plenty of fuzzing opportunities.

**Results** First, we decided to change the length declared in the User Information Length field. This field was set to a random length while 82 bytes of data were always transmitted. However, no matter which length we specified, the broadcast messages were always received and displayed without problem, leading to think this value is not even considered by mobile stations. Next, we left the length field unmodified, specifying 82 bytes of data were transmitted while we sent more than 82 bytes. To do this, we added a fifth frame contain-

---

\(^4\) Since communication takes place in bytes and GSM implements a 7-bit alphabet, padding bits must be added to complete the 82 bytes of data. Refer to [2] for an explanation of how user data is arranged before it is transmitted.
ing user data as well as the padding bits. The fifth frame had the Last Bit set to 1 and sequence number of “0100” which is a reserved value. The forth frame had the Last Bit set to 0 and sequence number set to “0011”. This time, the broadcast messages were ignored only by the Galaxy Note; the two Nokias and Blackberry received the messages without troubles.

Since the above fuzzing tests did not provide the results we were looking for, we decided to experiment with the padding bits. In a first attempt, we sent 82 bytes of data without including the padding bits. The Nokias successfully received the messages and displayed their contents; the Blackberry ignored these malformed messages. The Galaxy Note, on the other hand, displayed a (No Subject) legend and as content a question mark symbol (Figure 20). This can be classified as unspecified behavior, since this behavior is not defined in the GSM documentation. Next, CBS messages again without padding bits but with more than 82 bytes of data were sent. This time, only the two Nokias received the messages. Finally, we sent 82 bytes of data with a padding of 1s instead of 0s and the Nokias once again received the CBS message.

Figure 20: Galaxy Note displaying unspecified behavior

7.3 ANALYSIS OF RESULTS

Table 8 offers a summary of all obtained results. In the majority of the fuzzing tests, crafted broadcast messages were either ignored or successfully received and displayed. However, it is important to define what ignore means in this particular case. Knowing that well structured broadcast messages are received by mobile stations means that they listen to the CBCH channel to receive them. When broadcast messages are tampered, this does not mean that messages are not re-
<table>
<thead>
<tr>
<th>Fuzzing Attack</th>
<th>Blackberry 9700</th>
<th>Galaxy Note</th>
<th>Nokia 1600</th>
<th>Nokia 3310</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Block Type</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spare bit = 0, 1</td>
<td>Received</td>
<td>Received</td>
<td>Received</td>
<td>Received</td>
</tr>
<tr>
<td>LPD != 01</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
</tr>
<tr>
<td>LB bit</td>
<td>Received until LB=1</td>
<td>Received until LB=1</td>
<td>Received until LB=1</td>
<td>Received until LB=1</td>
</tr>
<tr>
<td>Sequence Number out of order</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
</tr>
<tr>
<td><strong>Serial Number</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Different fields length</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
</tr>
<tr>
<td>Padding != 0000 0000</td>
<td>Received</td>
<td>Received</td>
<td>Received</td>
<td>Received</td>
</tr>
<tr>
<td>Padding = 1111 1111</td>
<td>Ignored</td>
<td>Received</td>
<td>Received</td>
<td>Received</td>
</tr>
<tr>
<td>No Padding</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
</tr>
<tr>
<td><strong>Data Coding Scheme</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Combination of Reserve values</td>
<td>Received</td>
<td>Ignored</td>
<td>Received</td>
<td>Received</td>
</tr>
<tr>
<td><strong>Page Parameter</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First page value greater than 1</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Ignored</td>
</tr>
<tr>
<td>Transmission of more than 15 pages</td>
<td>Received until page 15</td>
<td>Received until page 15</td>
<td>Received until page 15</td>
<td>Received until page 15</td>
</tr>
<tr>
<td><strong>Content of Message</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data = 82 bytes; Length modified</td>
<td>Received</td>
<td>Received</td>
<td>Received</td>
<td>Received</td>
</tr>
<tr>
<td>Data &gt; 82 bytes; Length unmodified</td>
<td>Received</td>
<td>Ignored</td>
<td>Received</td>
<td>Received</td>
</tr>
<tr>
<td>Data = 82 bytes; No padding</td>
<td>Ignored</td>
<td>Unspecified Behavior</td>
<td>Received</td>
<td>Received</td>
</tr>
<tr>
<td>Data &gt; 82 bytes; No padding</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Received</td>
<td>Received</td>
</tr>
<tr>
<td>Data = 82 bytes; Ones padding</td>
<td>Ignored</td>
<td>Ignored</td>
<td>Received</td>
<td>Received</td>
</tr>
</tbody>
</table>

Table 8: Summary fuzzing attacks

received. In fact, they are received, but we do not know how they are handle by mobile stations. For instance, in the case where more user data than allowed is transmitted over the Content of Message field, it
is possible that a memory corruption takes place, for example by over-
flowing an array in charge of storing the broadcast message. However,
this is not visible due to this process not affecting another module or
simply, because we are not transmitting sufficient excessive data to
trigger a Denial-of-Service or other visible event. This also applies to
fuzzing tests that modified the fields’ length or implemented reserve
values. Again, it is possible that sufficient invalid data or a combina-
tion of them was not provided to be able to trigger some undefined
behavior. Due to a lack of time we were not able to experiment with
more intricate combination of invalid values or lengths. This can be
an interesting future research.

We believe that when crafted broadcast messages were received
and displayed, like it happened in most of the cases with the two
Nokia mobile stations, this does not mean that proper validations
procedures are implemented. Rather, it shows how the mobile manu-
facturer decided to implement the CBS, due to a lack of a specification
on how to deal with messages coming from this service. It is highly
possible the manufacturer decided to define as many cases as possible
to be valid, instead of implementing a procedure that defines if
the CBS message is valid or not. This opens the possibility to perform
other type of fuzzing tests that might encounter cases which are not
defined.

We managed to trigger only once an undefined behavior. This was
observed in the Galaxy Note, when a CBS message absent of padding
bits in the Content of Message field was received. This is undefined
behavior, since the way the broadcast message is displayed is not de-

defined on the GSM specification. However, the unexpected behavior
was not invasive enough to cause a failure in the device, since its re-
action manifested as a not properly displayed message. Still, it shows
once again that each manufacturer handles broadcast messages as
they see fit. This was evident throughout the several fuzzing tests we
performed. For example, when experimenting with the Data Coding
Scheme, we used the bit pattern “01001101”. This set of bits indicate
that CBS messages are compressed and that an
8
-bit alphabet is used.
This caused only the Blackberry to show a complete different set of
characters when the message was displayed. Both Nokia mobile sta-
tions displayed the characters based in the 7-bit alphabet. Figure 21
shows a broadcast message containing a string of 7-bit “á” charac-
ters when received in the Blackberry. From the figure, it can be seen
the “á” characters are now represented as “ÿ” since the Data Cod-
ing Scheme signals the mobile station the CBS messages’ user data is
formatted with an 8-bit alphabet instead of the default 7-bit alphabet.

Another interesting finding was to see the amount of characters
shown by the devices. In all tests, without regard of how many bytes
were sent, the two Nokia mobile stations only displayed 51 charac-
ters, which is in incorrect since 82 bytes of data correspond to 93 7-bit
characters. The Galaxy Note, on the other hand, always displayed 93 characters when it received correctly formatted broadcast messages. The Blackberry, in all scenarios, surprisingly showed only three characters on top of the main screen (Figure 23).

Regarding the reception of CBS messages based on the topic list, all mobile stations also showed different behavior from each other. According to the GSM specification, mobile stations should only receive CBS messages containing Message Identifiers (or topics) registered in their memory or SIM card. In our initial tests we used the Message Identifier value of 0 and did not registered this topic number in the mobile stations. All mobile stations except for the Blackberry received the CBS message; for the Blackberry it was necessary to register the topic number. In addition, once we changed the Message Identifier to a value different from 0, all mobile stations did not receive the CBS messages even though this time we registered the topic in the mobile stations. We could not find a reason behind this behavior.

Finally, when emergency CBS messages contained the bit value of 1 in both the Emergency User Alert and Pop-up alert fields, mobile stations showed different behavior as well. The GSM specification establishes that when the Emergency User Alert is set to 1, the emergency CBS message must be displayed immediately without requiring user intervention. Similarly, when the Pop-up alert bit is set to 1 the mobile stations must emit a unique alert – sound and vibration combination – that allows users to distinguish an emergency CBS message from a normal CBS message (Appendix B.1.1). However, when experiments used the aforementioned bit values, the GSM specified behavior was not fulfilled by any of the mobile stations. The Nokia 3310 always require user intervention to display the message and no special sound or vibration pattern was observed. The Nokia 1600 did immediately displayed the emergency CBS message, but partially, showing only a few characters (Figure 22). Furthermore, the mobile station did not offer the possibility to read the complete message. This mobile station also did not use a special type of alert. The Galaxy Note showed a similar behavior as the Nokia 3310, requiring user intervention to be able to read the emergency CBS message and no special alert was heard. Finally, the Blackberry always displayed the message on top of
the screen next to the network operator’s name, showing only three characters (Figure 23). We could not find an option that allowed us to completely read the message. In addition, like the rest of mobile station, no special type of alert was observed. Interestingly, none of the mobile stations offered the possibility to configure the alert pattern.

Figure 22: Nokia 1600 displaying only a few characters of CBS message

Figure 23: Blackberry 9700 displaying three characters of CBS message
8.1 Conclusions

The main goal of this thesis was to find programming flaws in the GSM Cell Broadcast Service implementations of mobile stations. To accomplish this objective, we relied on the fuzzing technique to modify the structure and contents of CBS messages which were transmitted over the GSM air interface towards mobile stations. The CBS message transmission was done using a very specific combination of software and hardware, mainly involving the OpenBTS software and a USRP-1. The former allowed us to create a GSM base station in charge of setting up the communication channels used to exchange the control and traffic data between the base station and mobile stations. The USRP-1, on the other hand, provided the hardware necessary to create and process the radio frequencies that carry all the GSM traffic. The configuration of these two entities proved to be difficult and time consuming. The main challenge was to find a perfect combination between GNU Radio and OpenBTS. In addition, all found online documentation and opinions indicated that OpenBTS 2.5.4 was only compatible with GNU Radio 3.2.2. This turned out to be misleading since we found out that GNU Radio 3.4.0 provided the best functionality to broadcast the CBS messages.

The fuzzing process was divided in three main stages: (1) identification of fields to fuzz, (2) design of the fuzzing approach based on the structure of the CBS message and (3) execution of the fuzzing. The first stage was relatively easy since a CBS message contains a small amount of fields and most of those fields only accept information in the form of ones and zeroes, limiting the amount of values they can take. While this situation facilitated the fuzzing approach design and fuzzing execution stages, it also limited the opportunities to find flaws. For most fields, we were able to only modify the length and try different combination of reserved values, which only had the effect of mobile stations ignoring the CBS messages. The field that allowed more fuzzing maneuverability due to its size and accepted range of values, was the Message Content field. This field accepts any type of information and can be as large as 82 bytes. This represented a splendid opportunity to inject any type of data and observed the mobile stations reactions. However, only once we were able to trigger an unspecified behavior in one of the mobile stations when the padding bytes were not present. On the other mobile stations no negative effects were observed, even when more than 82 bytes were transmitted.
In this case, mobile stations chose to ignore the extra bytes and only display 82 bytes or less.

Nonetheless, we obtained very interesting findings. For instance, old devices such as the Nokia 3310 and Nokia 1600 only displayed at all times 51 characters (around 45 bytes of data). The Nokia 1600 was even worse, since no access to the complete message was possible. In addition, none of the mobile stations complies with the specification in terms of behavior when CBS messages of type emergency are received since not one mobile station used a special type of alarm. Also, in the case of the Nokia 3310 and Galaxy Note intervention of the user was required to read the full message while the Blackberry and Nokia 1600 only showed a banner in their screens containing a few characters of the message. The Blackberry was the worst, since only three characters were shown. Additionally, the Blackberry just like the Nokia 1600 did not have the option to read the complete message and in none of the mobile stations it was possible to adjust the alarm to distinguish an emergency CBS message from other type of messages, which should be possible according to the GSM specification.

The CBS service can be very useful to communicate important information to a large base of users in just a few seconds. However, work still has to be done in both the implementation in mobile stations and the business parts. Considering the former, it is clear for us that all manufacturers do not follow the specification, but we believe this is somehow justified. During the gathering of documentation, we encountered the CBS service specification is going through a lot of changes, in many ways to improve the service. Many updates are very recent (March 2013), which serves as an indication of the increase in importance this service is currently having. This is of course a good sign, however, manufacturers cannot keep up with these changes and this is why most of them implement the service as they see fit. Still, better work has to be done on how messages are displayed and handled in mobile stations, since messages are displayed differently. This can create confusion or frustration among mobile users, which can decide to simply disable the service because they find it annoying or useless. Besides of providing emergency information, one of the CBS objectives is to supply with all types of information to mobile users, such as weather updates, news and shopping offers. If these services are abused, users can simply disable the CBS service and thus not be able to receive the emergency information when necessary. Therefore, it is important to find a balance and hopefully the CBS service will be used to fill the existing gap of rapidly communicating information to people on how to act in the presence of a catastrophe.

Finally, it is important to point out the security issue that is raised when a malicious user impersonates a GSM base station and broadcasts misleading CBS messages, possible generating panic among the
population. This is possible due to the GSM network never authenticating to mobile stations. Unfortunately, this issue will always remain open and governments around the world should be aware of this fact.

8.2 future work

During this research, an old release of OpenBTS was used. This created a series of issues, mainly related to the software and hardware supported, taking a considerable amount of time to solve them. However, this situation will be solved when the new OpenBTS iteration is released soon. Kurtis Heimerl, one of the main developers and maintainers of the OpenBTS project, announced on the OpenBTS mailing list that the CBS will be fully implemented in the next release of the software\(^1\), providing in this way a more stable and better test bed for future research.

An interesting fuzzing opportunity over the broadcast service is to manipulate the contents of the ETWS emergency message. This type of message is carried in the \textit{P1 Rest Octects} IE of a PAGING REQUEST TYPE 1 message over the Paging Channel (PCH). At this moment, OpenBTS supports Rest Octets for other type of messages but not of PAGING REQUEST TYPE 1. If at some point it is supported, it would be interesting to see the results when modifying the information elements of the ETWS message, in particular, the \textit{Warning Security Information} IE which occupies 50 bytes of the total CBS message and it must be ignored by the mobile stations. However, its presence is required to remain compatible with previous specifications.

Finally, we expect the CBS service will gain more importance in the near future due to a real need of governments to communicate emergency situations to their population. This will translate in more mobile station manufacturers to support this service, which opens the possibility to test a wider amount of devices. For this thesis, eight mobile stations were available, two of them did not support the service and the other two although they had the capability to receive broadcast messages, they never did for unknown reasons. Moreover, surprisingly the Android operating system only supports the CBS in one of its newer iteration, version 4.1.2, while Apple’s iOS does not support the CBS in all of its versions. Thus future research can be done in more recent mobile stations models once CBS support becomes more widespread.

\(^{1}\) \url{http://sourceforge.net/mailarchive/forum.php?thread_name=E17E3D00-017C-43E9-AE1E-4BF345056F8D%48floor51.com&forum_name=openbts-discuss}


[49] Vijay Shinde. What is Boundary value analysis and Equivalence partitioning? http://www.softwaretestinghelp.com/what-is-


This Appendix presents the formats of diverse control messages used during the transmission of broadcast messages. All information elements contained in the messages have different presence requirements, being mandatory (M) and optional (O) as well as certain length expressed in bytes. These two parameters are shown in a tuple next to each information element ([Requirement,Length]).

For a complete description of the most important information elements for this research, refer to Appendix B.

### A.1 CBCH LOAD INDICATOR

This message is used by the BTS to notify the BSC it can send more CBS messages when the CBCH channel is in “underflow” state or to suspend the transmission of messages for a particular period of time when the CBCH channel is under “overflow” state.

Figure 24 shows the format of the CBCH LOAD INDICATOR message.

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Discriminator (M,1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Type (M,1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Channel Number (M,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CBCH Load Information (M,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SMSCB Channel Indicator (O,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 24: CBCH LOAD INDICATOR message format [8]

### A.2 PAGING REQUEST TYPE 1

In the GSM specification, this message is used to carry different types of information for different services. For the purpose of this research, the information element *P1 Rest Octets* is the only one of interest, since it is in charge of carrying the ETWS emergency messages. The format of this message is shown in figure 25.
A.3 SMS BROADCAST COMMAND

Message used by the BSC to send a CBS message to the BTS [8]. The SMSCB Information IE carries the 88 bytes comprising a CBS message (control and user data) as well as other information elements that identify a particular SMS BROADCAST COMMAND message. Refer to Section 4.7.1 for more details on the usage of this message.

A.4 SMS BROADCAST REQUEST

Message containing the CBS message that is sent from the BSC to the BTS [8]. Four of these messages must be sent from the BSC to the BTS to deliver the CBS message to the mobile stations. The SMSCB Information IE contains the 22 byte long CBS message, its block type header as well as the Element Identifier field which is drop by the BTS before the message is broadcasted to mobile stations. Refer to Section 4.7.1 for a complete explanation on the usage of this message.
<table>
<thead>
<tr>
<th></th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>Message Discriminator (M,1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Message Type (M,1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Channel Number (M,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>SMSCB Information (M,24)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>SMSCB Channel Indicator (O,2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 27: SMS BROADCAST REQUEST message format [8]
INFORMATION ELEMENTS

This Appendix contains a detailed description of the information elements used to form the normal CBS messages which are broadcasted from the base station transceiver (BTS) to mobile stations.

An information element is formed by a group of fields. All fields in Information Elements are mandatory unless otherwise stated. Moreover, all information elements must always include the Element Identifier field which is a binary representation of the IE. The Element Identifier is never considered when the size of an information element is calculated.

B.1 INFORMATION ELEMENTS NORMAL CBS MESSAGE

B.1.1 Serial Number

The Serial Number IE is used to identify a new normal CBS message along with the Message Identifier IE, the Cell List IE and the Channel Identifier IE.

The format of the Serial Number IE is depicted in Figure 28:

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>byte 1</th>
<th>byte 2</th>
<th>byte 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element Identifier</td>
<td>Serial Number</td>
<td>Serial Number cont.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 28: Format New Serial Number IE

The IE has a length of 2 bytes (16 bits) consisting of 3 fields: the Geographical Scope (GS) indicator (2 bits), the Message Code (10 bits) and the Update Number (4 bits). The detailed structure of this IE is shown in figure 29.

<table>
<thead>
<tr>
<th>Byte 1</th>
<th>Byte 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 7 6 5 4 3 2 1</td>
<td>8 7 6 5 4 3 2 1</td>
</tr>
<tr>
<td>GS</td>
<td>Message Code</td>
</tr>
</tbody>
</table>

Figure 29: Inner structure of Serial Number IE [3]

Geographical Scope This field indicates the geographical area where a Message Code is unique and the display mode. When two CBS messages arrived in two different cells to be broadcasted, containing the same Serial Numbers/Message Identifiers, the Geographical Scope
can be used to determine if the two CBS messages are indeed the same. The format of the Geographical Scope field is as follows:

<table>
<thead>
<tr>
<th>GS Code</th>
<th>Display Mode</th>
<th>Geographical Scope</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>Immediate</td>
<td>Cell wide</td>
</tr>
<tr>
<td>01</td>
<td>Normal</td>
<td>Network wide</td>
</tr>
<tr>
<td>10</td>
<td>Normal</td>
<td>Location Area wide</td>
</tr>
<tr>
<td>11</td>
<td>Normal</td>
<td>Cell wide</td>
</tr>
</tbody>
</table>

Table 9: Geographic Scope field values [3]

The Geographical Scope (GS) code informs the mobile station if the CBS message is:

- **Cell wide**: This means that if the message is display to the end user, then the message is removed from the screen when the user changes to another cell and if a CBS message is received in the new cell, then the message must be regarded as “new”.

- **Network wide**: This means the Message Code or Update Number must change in the next cell of the network for the message to be regarded as “new”. In this case, the CBS message is relevant only to the current network being used; if the user roams to another network, e.g. going from KPN to Vodafone network, this means the CBS message is “new”.

- **Location Area wide**: A CBS message with the same Message Code and Update Number can or cannot be considered “new” in the next cell depending on the next cell being in the same Location Area as the current cell. The Location Area value is network independent. For example, if a mobile user roams from a KPN network to a Vodafone network and the Location Area value of the CBS message is the same, then the message cannot be considered as “new”.

The display mode is used to indicate if the CBS message is to be displayed all the time (“immediate”) or only when the user desires to see it (“normal”). Without regard of the display mode, the CBS message is displayed only if the Message Identifier is registered in the search or topic list of the mobile station. The display modes are intended as a suggestion for manufacturers, in this way leaving the option to them on how to implement them. Moreover, the end user should be able to activate these different modes.

**Message Code**  The Message Code field is used to differentiate CBS messages with the same Message Identifier.

The Message Code also helps to identify different situations within the same scenario. For example, for a Message Identifier value of
source equal to “Meteorological Center” and type of “Weather Report”, then “Snow in Groningen” and “Rain for North-Brabant” can be values for message codes. Message codes binary values for type and source are defined by each MNO.

The Message Code can be used to specify a CBS message as being of emergency type. In this case the Message Code can be used to instruct the mobile stations to activate the emergency alert and pop-up of messages to inform end users. For this scenario, the Message Code format is shown in Figure 30 (the structure is the same as shown in Figure 29).

<table>
<thead>
<tr>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
<th>0</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emergency User Alert</td>
<td>Pop-up</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 30: Format Emergency Message Code field [3]

The values of the Emergency User Alert and Pop-up fields is listed in Table 10:

<table>
<thead>
<tr>
<th>Field</th>
<th>Code</th>
<th>Instruction to Terminal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Emergency User Alert</td>
<td>0</td>
<td>No instruction as to emergency user alert.</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Activate emergency user alert.</td>
</tr>
<tr>
<td>Pop-up</td>
<td>0</td>
<td>No instruction as to pop-up.</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Activate pop-up on the display.</td>
</tr>
</tbody>
</table>

Table 10: Emergency User Alert and Pop-up fields values [3]

The emergency user alert must be a unique pattern of sound and vibration which must be used only when receiving CBS messages of emergency type.

**Update Number** The Update Number is used to indicate a change in the content of the CBS message, i.e. in a CBS message with the same Message Identifier, Geographical Scope and Message Code.

The Update Number can be seen as a differentiator between old and new versions of the same CBS message within a specified geographical area. A new CBS message can have the Update Number 0000 and increment by 1 every time the CBS message is updated.

**B.1.2 Message Identifier**

The Message Identifier IE is a binary number used to identify the source and type of a CBS message. For example, source can have the value of “Meteorological Center” and a type value equal to “Weather
Several CBS messages can have the same source and/or type. The Serial Number IE is used to distinguish among those messages.

A mobile station must try to receive the CBS messages with the Message Identifiers contained in its search or topic list, stored in the mobile phone’s SIM card and in its local memory. In case the mobile phone has a technical constraint which limits the amount of Message Identifiers it can search for, then the list contained in the SIM card takes a relevance over the list stored in the mobile station.

A Message Identifier is defined in hexadecimal format, occupying 2 bytes to be fully represented. For example, the value “1234” is represented by one byte being “00010010” and the other “00110100”. The structure of the Message Identifier IE is as follows:

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element Identifier</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Identifier</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Message Identifier cont.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Figure 31: Format Message Identifier IE [7]](image)

The GSM specification [3] lists all the Message Identifier values available. The interested reader is advised to consult this reference.

### B.1.3 Data Coding Scheme

This information element specifies to the mobile station how a received CBS message should be handled, its language as well as its coding and alphabet. The structure of the Data Coding Scheme IE is depicted in Figure 32:

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element Identifier</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data Coding Scheme</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Figure 32: Format Data Coding Scheme IE [7]](image)

When the SIM card contains various language configurations, the mobile station must use this information to set up language filters for the CBS message, i.e. the mobile station will display only CBS messages in the specified languages. Furthermore, the end user is also capable, if desired, to configure language filters besides those available by default in the SIM card or mobile stations’ memory.

In scenarios, where the broadcast message has a Message Identifier that classifies it as an ETWS message which is mandatory to receive (values 4370 to 4382), then the mobile station must always display the broadcast message without taking in consideration any language filter created by the end user.
When the broadcast message has a Message Identifier value ranging from 4383 to 4395, it identifies it as an ETWS message that is optional to receive. In this case, the mobile station is allowed to consider the language filters configured by the end user, to decide whether to display the message. In the event that a language is not included in the filter, the message must be discarded.

The technical specification 23.038 Alphabets and language-specific information [2] provides a full description of the format and values that can be assigned to the Data Coding Scheme IE.

### B.1.4 Page Parameter

This information element is added by the BTS to the CBS message before it is broadcast to the mobile stations. It is used to signal the total amount of pages contained in the CBS message and the page number currently being broadcast. The structure of the Page Parameter IE is shown in Figure 33:

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element Identifier</td>
<td>Page Parameter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 33: Format Page Parameter IE [7]

To accomplished its purpose, the Page Parameter IE implements two 4-bit fields. The first field (bits 1 – 4) indicates in binary representation the CBS message page number and the second field (bits 5 – 8) indicates in binary format as well, the page number currently being transmitted. The value “0000” for both fields is reserved; in case a mobile device received a CBS message with the value “0000 0000“ then it should treat as “0001 0001”, i.e. a one page message [3].

### B.1.5 Content of Message

The Content of Message IE or Message Content IE contains the user information of the CBS message.

The format of the Message Content IE is shown in Figure 34.

<table>
<thead>
<tr>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element Identifier</td>
<td>User Information Length</td>
<td>Message Content</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 34: Format Message Content IE [7]

The User Information Length field is used to represent in binary format the amount of bytes carrying user information, e.g. if Mes-
sage Content field contains 50 bytes of user information, then User Information Length is equal to "0011 0010".

The maximum amount of user information that can be carried is 82 bytes. In case not all the 82 bytes are occupied with user data, then padding bits must be added following the guidelines in [2].
Depending on the actions performed by mobile stations or lack of, they are considered to be in a certain operational state. The GSM specification defines five types of mobile stations operational modes:

- **Idle mode**: The mobile station does not have a dedicated channel allocated; it periodically listens to the PCH and BCCH channels.

- **Dedicated mode**: The mobile station has at least two dedicated channels assigned to it, one of them being a SACCH.

- **Packet idle mode**: This mode only applies to mobile stations that support packet switch transmission, e.g., GPRS. The mobile station listens to the packet switch equivalent of CCCH and BCCH, these being PCCCH and BCCCH, respectively. If those two latter channels are not available at the BTS, then the MS listens to the CCCH and BCCH.

- **Packet transfer mode**: The mobile stations have radio resources assigned to transmit data packets known as Protocol Data Unit (PDU).

- **Dual transfer mode**: Mobile stations get allocated radio resources so they can simultaneously receive voice and data packets. The network is in charge of coordinating the allocation of the radio resources to avoid the utilization of two different frequencies.
This appendix lists the hardware and software required to perform the Cell Broadcast Service fuzzing tests. This is provided for the interested reader in building a similar setup to recreate the experiments.

D.1 HARDWARE REQUIREMENTS

The main pieces of hardware required are the USRP-1, one laptop, antennas and daughterboards. The daughterboards must be of frequencies supported by the mobile stations. If daughterboards will be purchased, then it is better to acquire RFX1800 daughterboards since by simply flashing the firmware, they can operate in the 900 MHz frequencies and turn back to 1800 MHz [42]. However, the opposite it’s not true. If it is required that a RFX900 daughterboard functions in the 1800 MHz, then a filter must be installed in the board [31]. On the other hand, if the daughterboards will be used for other projects besides GSM, then it is better to purchase WBX daughterboards, since the range of frequencies goes from 55 MHz to 2200 MHZ, making them ideal for other projects.

In our particular case, we used a USRP-1 with a 52 MHz clock\(^1\) and two RFX1800 daughterboards, which were purchased from Ettus Research [45]. The laptop was a Voodoo Intel Core Duo with 2 GB of RAM. The laptop computer will execute the GNU Radio and OpenBTS software, which will enable the USRP to behave as a GSM base station. For this reason, the laptop must contain sufficient RAM memory, otherwise timing problems may appeared when running OpenBTS, causing problems in the communication between the mobile stations and USRP.

Finally, unlock mobile stations and SIM cards are required to perform the fuzzing tests. In addition, they must support the reception of broadcast messages. We found that old phones usually support it and recent versions of Android as well. Moreover, it is recommended to use unlock mobile stations to facilitate the swap of SIM cards. The SIM cards can be of any operator. We used SIM cards from two different mobile network operators and obtained the same results.

Figures 35 and 36 show the whole setup and the mobile stations used in this research, respectively.

---

\(^1\) The clock is not available in the Ettus Research website, however, a quick online search returns several shops which sell this component.
D.2 SOFTWARE REQUIREMENTS

The software used for this research is listed in Table 11.

<table>
<thead>
<tr>
<th>Software</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ubuntu</td>
<td>9.04</td>
</tr>
<tr>
<td>GNU Radio</td>
<td>3.0.4</td>
</tr>
<tr>
<td>OpenBTS</td>
<td>2.4.5 - CBS Service</td>
</tr>
<tr>
<td>Boost</td>
<td>1.49.0</td>
</tr>
<tr>
<td>oSIP</td>
<td>3.3.0</td>
</tr>
<tr>
<td>oRTP</td>
<td>1.7</td>
</tr>
<tr>
<td>Asterisk</td>
<td>1.4</td>
</tr>
</tbody>
</table>

Table 11: Software used for fuzzing experiments

Other dependencies must be installed to be able to install GNU Radio. This can be found in [43]. Excellent guides to install GNU Radio and OpenBTS in legacy systems such as Ubuntu 9.04 are [11, 12].
Most computers implement the ASCII alphabet to communicate and display information to users. GSM, on the other hand, implements its own alphabet which uses 7 bits to represent characters of messages sent using the SMS or CBS services.

Following, is the table showing how to map a character to its corresponding 7-bit value.

<table>
<thead>
<tr>
<th>b7</th>
<th>b6</th>
<th>b5</th>
<th>b4</th>
<th>b3</th>
<th>b2</th>
<th>b1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>@</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>£</td>
<td>_</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>$</td>
<td>Φ</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>3</td>
<td>¥</td>
<td>Γ</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>ë</td>
<td>Λ</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
<td>é</td>
<td>Ω</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>ü</td>
<td>Ï</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>ï</td>
<td>Ψ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>ò</td>
<td>Σ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>9</td>
<td>Ç</td>
<td>Θ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>10</td>
<td>LF</td>
<td>Ξ</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>11</td>
<td>Ø</td>
<td>t</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>12</td>
<td>ø</td>
<td>À</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>13</td>
<td>CR</td>
<td>æ</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>14</td>
<td>Â</td>
<td>β</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>15</td>
<td>à</td>
<td>Ë</td>
</tr>
</tbody>
</table>

Table 12: GSM 7-bit Alphabet

### E.1 GSM 7-BIT MESSAGE ENCODING

Taking as an example the string “hola”, the encoding is performed as follows:

1. Find the character of interest. In this case we considered “h”.

---

1 The string “hola” was used in different tests when we troubleshooted the communication between the USRP-1 and mobile stations. Refer to Appendix F for more details.
2. Identify the b7, b6 and b5 bits. For “h” this equals to b7 = 1, b6 = 1 and b5 = 0.

3. Next, pinpoint the b4, b3, b2 and b1 bits. For this example b4 = 1, b3 = 0, b2 = 0 and b1 = 0.

4. Finally, append these values, resulting in 110 1000

Following the above procedure for the rest of the characters, the following bit sequence is obtained (Table 13):

<table>
<thead>
<tr>
<th>Character</th>
<th>b7</th>
<th>b6</th>
<th>b5</th>
<th>b4</th>
<th>b3</th>
<th>b2</th>
<th>b1</th>
</tr>
</thead>
<tbody>
<tr>
<td>h</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>o</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>l</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>a</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Final</td>
<td>110100011011111011001100001</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 13: Codification to GSM 7-bit alphabet

The final bit sequence is obtained by appending all obtained bits, starting with character “h” and bit b7. The remaining characters are appended in the same fashion, starting with bit b7.

For transmission, however, GSM takes the bit stream and divides them in sets of 4 bits, resulting in the following bit sequence:

1101 0001 1011 1111 0110 0110 0001

Taking each set of 4 bits and mapping them to their decimal equivalent, yields the hex-bytes “d1bf661”. The next section shows how the decoding takes place.

E.2 GSM 7-BIT MESSAGE DECODING

This section shows how to decode the hex-bytes “d1bf661” from the previous section, to obtain the original “hola” string.

The first step is to convert each character to their binary representation; the obtained bit stream is the following:

1101 0001 1011 1111 0110 0110 0001

Next, this stream is divided into sets of 7 bits. This grouping is displayed in Table 14:

Finally, with the help of Table 12, we map each bit stream to a letter. Considering the first row of Table 14, this process is performed as follows:

1. Take bits b7, b6, b5 and identify them in Table 12.
2. Next, take the remainder four bits, b4, b3, b2, b1 and pinpoint them in Table 12 as well.

3. Lastly, the number or letter found in the crossing of those two sets of bits in Table 12, is the desired character. In this case, the character equals to letter “h”.

The above procedure is performed with every row, finally yielding the “hola” string.
In this appendix we discuss the different issues we encountered during the hardware and software configuration stage of this thesis. This is showed so future researchers who desire to use a similar setup can take advantage of the tools available to troubleshoot OpenBTS, especially the GSM air interface.

F.1 HARDWARE AND SOFTWARE CONFIGURATIONS

During the research, we used different combinations of hardware and software, until we found the one that could deliver the broadcast messages to mobile stations. Table 15 shows the various setups configured through our research.

<table>
<thead>
<tr>
<th>Software</th>
<th>Configuration A</th>
<th>Configuration B</th>
<th>Configuration C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ubuntu</td>
<td>9.04</td>
<td>9.04</td>
<td>10.04</td>
</tr>
<tr>
<td>OpenBTS</td>
<td>2.5.4</td>
<td>2.5.4</td>
<td>2.6</td>
</tr>
<tr>
<td>GNU Radio</td>
<td>3.2.2</td>
<td>(3.2.2)[a] 3.4.0</td>
<td>3.4.0</td>
</tr>
<tr>
<td>oSIP</td>
<td>libosip 3.3.0</td>
<td>libosip 3.3.0</td>
<td>libosip 3.3.0</td>
</tr>
<tr>
<td>oRTP</td>
<td>libortp7</td>
<td>libortp7</td>
<td>libortp8</td>
</tr>
<tr>
<td>Boost</td>
<td>1.44.0</td>
<td>1.49.0</td>
<td>1.53.0</td>
</tr>
<tr>
<td>Asterisk</td>
<td>1.4</td>
<td>1.4</td>
<td>1.6</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Hardware</th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Daughterboards</td>
<td>1 WBX</td>
<td>2 RFX1800</td>
</tr>
</tbody>
</table>

[a] In a first instance, this version of GNU Radio was used.

Table 15: Hardware and software configurations used at different stages in the research

At the beginning of this project, we had access to an USRP-1 and one WBX daughterboard. With this hardware we proceeded to create Configuration A; however, after several attempts, mobile stations could not see and connect to the OpenBTS network. After consulting several online documentation, comments and experiences from other users, we found that OpenBTS 2.5.4 can only be used with GNU Radio 3.2.2. However, this GNU Radio version does not support WBX daughterboards, which started to be supported until release 3.3.0 – somehow indicating why mobile stations could not connect to the OpenBTS network. This situation led us to install the newer release 3.3.0 of GNU Radio at the expense of not supporting OpenBTS version 2.5.4. This new scenario required us to transfer the CBS service
source code from OpenBTS version 2.5.4 to version 2.6\(^1\). After a successful transfer, we proceeded to test our new configuration. Unfortunately, mobile stations still could not connect to the OpenBTS network. After some research, we decided to install the newer GNU Radio version 3.4.0. This upgrade allowed us to establish a good communication between the network and mobile stations, however, broadcast messages were still not being received. This situation led us to perform different troubleshooting actions to detect the issue of why this was the case. As a first measure we decided to debug the air interface to see if the CBCH channel was actually being used and carrying broadcast messages. At this point in time, we were using Configuration C from Table 15.

### F.2 Debugging the Air Interface

One of the first mobile stations we had available to perform the fuzzing tests was a Nokia 3310, which is known for supporting the reception of CBS messages. This mobile station has a special bus connector under the battery which when used with a particular data cable and connected to a computer, gives users the possibility to performed different tasks such as downloading ring tones to the phone or even make the mobile phone operator free. Nokia used this port to debug the phone’s GSM firmware. The company, however, forgot to disable this functionality when phones went into production. This allowed researchers and hobbyist to activate this feature with the usage of special software, enabling the phone to observe and record the GSM air interface.

For the debugging tests, a purpose specific data cable was purchased\(^2\); the gammu \([22]\) as well as gsmdecode \([17]\) programs were downloaded and installed in our host system (laptop). The former program is used to catch the messages exchanged between the mobile station and BTS, while the latter decodes those messages displaying the channels used and the information carried by them.

In the first debugging test, we sent a CBS message containing 93 “à” characters. This string came as a default in the OpenBTS software and is a good choice for debugging purposes due to its GSM 7-bit representation\(^3\) being a series of “ff” characters, making it easy to identify them in the output file created by gsmdecode. With this in mind and the phone plugged to the host system, gammu as well as

---

1 OpenBTS newest version is 2.8, however, the structure of the source code differs in great manner from version 2.5.4, reason why we decided to transfer the CBS service source code to an OpenBTS release as similar as possible in structure to version 2.5.4. This version was 2.6.

2 [http://www.nowgsm.com/3310cable.htm](http://www.nowgsm.com/3310cable.htm)

3 The GSM standard utilizes a system where each character is represented by 7 bits instead of 8 bits (ASCII) which are normally used in computer systems. For more information refer to Appendix E.
OpenBTS are started and the phone is connected to the base station. Once a sufficient amount of GSM traffic data is captured, it is input to gsmdecode so it becomes human readable. After analyzing the data, no data indicating the CBCH channel is being used was encountered.

Due to the unsuccessful first attempt, we decided to try again, but this time by transmitting a crafted message, to examine if this made any difference. The message sent in the second trial was “hola”, which in the GSM 7-bit alphabet system is represented by the hex-bytes “d1bf661” (refer to Appendix E.2 to see why this is the case). This time, when debugging was executed, gammu threw a Segmentation fault error, in the moment the mobile station successfully connected to the base station⁴. This was good news since it indicates that data is being transmitted on the CBCH channel and the sent message contains too many or too few bytes, causing an erroneous segmentation of the CBS message. Recall that a CBS message is transmitted in four 22-byte frames from the BTS to mobile stations.

The results of this debugging process opened the possibility of an error in the way OpenBTS creates, segments and transmits CBS messages. This led us to analyze OpenBTS log files.

F.3 ANALYSIS OF OPENBTS LOG FILES

OpenBTS records all its operations for debugging purposes, allowing the operator to see general channel information or errors that might be produced during OpenBTS execution. When analyzing these logs, an entry related to the establishment and usage of the CBCH channel is present along with the four frames carrying the content of the broadcast message, confirming once more our believes that information is being sent on the CBCH channel. Part of these log entries are shown below:

[trimmed]
mCBFrame1 raw=(201ee600010111d1bf661ffffffffffffffffffffffffffff)
mCBFrame2 raw=(21ffffffffffffffffffffffffffffffffffffffffffff)
mCBFrame3 raw=(22ffffffffffffffffffffffffffffffffffffffffffff)
mCBFrame4 raw=(33fffffffffffffffffffffffffffffffffffffffff000)

[trimmed]

In the above, frame mCBFrame1 starts with the 201ee600010111 hexadecimal bytes, which is followed by our data d1bf661 and a continuous repetition of the string “ff”. The remaining frames, mCBFrame2, mCBFrame3 and mCBFrame4 start with the characters 21, 22 and 33, respectively; the string “ff” is also present filling up the rest of the frames’ contents, which we already know they represent the charac-

---

⁴ By default, OpenBTS is programmed to send CBS messages in the moment mobile stations connected to the USRP-1.
ter “à”. To determine what the rest of characters mean, we have to look at the structure of the CBS message.

From Section 4.7.4, we know that a CBS message is split in four frames and a Block Type header is added to each frame, so on reception, mobile stations can identify and assemble the CBS message. Each Block Type header is comprised by a Spare bit, two bits representing the Link Protocol Discriminator (LPD), a Last Block (LB) bit and a Sequence Number formed by 4 bits. By converting the first two bytes at the beginning of each frame to binary, allows us to determine if these values are correctly formatted. The result of this process is shown in Table 16.

<table>
<thead>
<tr>
<th>Byte</th>
<th>Spare</th>
<th>LPD</th>
<th>LB</th>
<th>Sequence Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>0</td>
<td>01</td>
<td>0</td>
<td>0000</td>
</tr>
<tr>
<td>21</td>
<td>0</td>
<td>01</td>
<td>0</td>
<td>0001</td>
</tr>
<tr>
<td>22</td>
<td>0</td>
<td>01</td>
<td>0</td>
<td>0010</td>
</tr>
<tr>
<td>33</td>
<td>0</td>
<td>01</td>
<td>1</td>
<td>0011</td>
</tr>
</tbody>
</table>

Table 16: Bit representation of frames’ Block Type

Recall that the Spare bit should always be zero; the LPD must be “01”; the LB should be one only for the last block carrying user data and the Sequence Number should follow a sequence from “0000” to “0011” (Table 3). Considering these conditions, it is clear the Block Type of each frame is formatted correctly. The Spare bit is always zero, the following two bits are always “01” indicating this is a CBS message; moreover, the LB is only one in the last frame as it should be and the Sequence Number bits are correctly formatted as well. All of the above indicate OpenBTS is sending the correct per frame block type information to mobile stations.

Following the same procedure, it is possible to determine what the set of hex-bytes 1ee600010111 mean. From Figure 11, we know the order on which the CBS message information elements are sent and the amount of bytes required by each of them, this being two bytes for Serial Number IE and Message Discriminator IE, respectively; two bytes comprising both the Data Coding Scheme IE and Page Parameter IE and finally, 82 bytes for Content of Message IE. With this in mind, we can split the string of hex-bytes and convert them to their binary representation; by doing so, we obtain the results shown in Table 17.

Considering the structure of each information element (Appendix B) we can establish if the information elements are being correctly formatted and transmitted. Starting with the Serial Number IE (Appendix B.1.1), the first 2 bits (00) indicate a cell wide Geographical Scope (GS). The next 2 bits (01), part of the Message Code field, signal the mobile station this is a CBS message of emergency type and as such, it must be displayed to users without requiring their intervention. Due
to the emergency nature of the message, the GSM specification requires for the usage of padding bits to fill the rest of this field; in this case, these are represented by bits “1110 1110”. Finally, the Update Number field is defined by the last four bits (0110).

Moving to the Message Identifier IE, these two bytes signal a value of 1. This IE is used to indicate mobile stations the type of the message being broadcast. For example, the message identifier for a tsunami alert is different of that of an earthquake alert. For further details, refer to Appendix B.

The byte representing the Data Coding Scheme IE, carries two types of control data. The first nibble (bits 5 – 8) indicates the type of coding used in the CBS message while the second nibble (bits 1 – 4) denotes the language of the message. In this case, the former is represented by “0000” signaling mobile phones the CBS message contains GSM 7-bit characters and the latter value of “0001” indicates the CBS message is in English language\(^5\) [2]. The attentive reader might noticed the character “à” is not part of the English language. This does not affect the reception of messages, only the way they are displayed.

The Page Parameter IE, also carries two types of information. The second nibble (bits 1 – 4) represents the amount of pages comprising the CBS message and the first nibble (bits 5 – 8) indicates the page currently being sent. In this example, the message contains a single page which is the only one that is being transmitted.

Up until now, we know information is being correctly transmitted by OpenBTS on the CBCH channel; unfortunately, broadcast messages are still not received at the mobile stations using Configuration C. This leaves us with the last resort before reviewing in depth the source code: spoof a provider’s GSM base station to determine if broadcast messages can be received when the mobile station is connected to its home network.

\(^5\) Even though the “hola” string is in Spanish, it is considered to be in English since no special characters from Spanish language are being used.

### Table 17: Bit representation of CBS Message Information Elements

<table>
<thead>
<tr>
<th>Information Element</th>
<th>Hex-bytes</th>
<th>Bit Representation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Serial Number</td>
<td>1ee6</td>
<td>0001 1110 1110 0110</td>
</tr>
<tr>
<td>Message Identifier</td>
<td>0001</td>
<td>0000 0000 0000 0001</td>
</tr>
<tr>
<td>Data Coding Scheme</td>
<td>01</td>
<td>0000 0001</td>
</tr>
<tr>
<td>Page Parameter</td>
<td>11</td>
<td>0001 0001</td>
</tr>
</tbody>
</table>

F.4 spoofing a provider’s base station

Every GSM base station must transmit on its BCCH channel its Cell Identity (CI) and Location Area Identification (LAI), which help mobile stations to identify a BTS and know the geographical area where...
they are currently located. The LAI is composed by the Mobile Country Code (MCC) used to uniquely identify a country; the Mobile Network Code (MNC) which identifies a certain network within the GSM worldwide network and the Location Area Code (LAC) that can take a value of up to $65,536$ to identify a specific geographical area within a MNC.

To spoof a provider’s base station, it is only required to use its MCC and MNC values, which is easy to do in OpenBTS by modifying a few lines of its configuration file. In our case, we used the MCC and MNC values of 204 and 08, respectively, to impersonate a KPN base station in the Netherlands.

After carrying out several tests, we were unable to successfully receive a broadcast message in the mobile stations. This result could mean problems with the mobile devices, which is unlikely, since they are known for receiving CBS messages or that messages are being transmitted with slight timing shifts, preventing a correct reception at mobile stations. The latter is possible due to the implementation of a single WBX daughterboard, which is in charge of transmitting and receiving GSM traffic. For this reason, it was decided to purchase two RFX1800 daughterboards, to enable a better data transmission and reception, in this way discarding any possible hardware issues. Moreover, with this new hardware configuration, the original version of OpenBTS supporting the CBS service can be used along with GNU Radio 3.2.2, known for supporting RFX daughterboards and being compatible with OpenBTS version 2.5.4. This new hardware and software combination gave rise to an initial Configuration B (Table 15).

E5 New Configuration Tests

In Configuration B, as it was expected, mobile stations connected right away to the OpenBTS network. However, broadcast messages were still not received. This situation led us to do the only thing we did not try before: install a version of GNU Radio different from release 3.2.2. Since we knew version 3.4.0 worked successfully in Configuration C, we decided to install it. For our surprise, CBS messages were immediately received when we performed the first test with this new configuration. Therefore, Configuration B was the setup used for the fuzzing tests which are discussed in detail on Chapter 7.

For us, it was quite surprising how such a small detail, that of using a certain version of GNU Radio which is recommended by the OpenBTS community, did not offer the expected functionality. Even more unexpected was the fact that a version of GNU Radio, which in theory is not supported by OpenBTS version 2.5.4, provided the proper libraries that allow the broadcasting of messages. This comes to show that it is important to never discard certain scenarios and to take comments and suggestions just as guidelines.