

ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG  
INSTITUT FÜR INFORMATIK

Lehrstuhl für Kommunikationssysteme  
Prof. Dr. Gerhard Schneider  
Betreuer: Dr. Dirk von Suchodoletz



Master-Arbeit zum Thema:  
Open Source IMSI-Catcher

Dennis Wehrle

*Diese Arbeit wurde eingereicht als Teilleistung zur Erlangung  
des Master-Grades an der Technischen Fakultät,  
Albert-Ludwig-Universität Freiburg, 2009*



# Erklärung

Hiermit erkläre ich, dass ich diese Abschlussarbeit selbstständig verfasst habe, keine anderen als die angegebenen Quellen/Hilfsmittel verwendet habe und alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten Schriften entnommen wurden, als solche kenntlich gemacht habe. Darüber hinaus erkläre ich, dass diese Abschlussarbeit nicht, auch nicht auszugsweise, bereits für eine andere Prüfung angefertigt wurde.

Diese Arbeit entstand parallel zur Masterarbeit „Open Source GSM BTS Setup und Analyse für Demo-Zwecke“ von Holger Bertsch [2]. Aufgrund der Komplexität und der aufwendigen Einarbeitung in die Thematik war eine enge Zusammenarbeit unerlässlich. Die Beschaffung der Hardware, notwendige Modifikationen von Soft- und Hardware und die initiale Inbetriebnahme sowie die Lösung etlicher sich daraus ergebende Probleme, konnte nur in Teamarbeit bewerkstelligt werden. Infolgedessen ergaben sich inhaltliche Überschneidungen, in erster Linie in den einleitenden Kapiteln GSM, Hardware und Software. Die Ausformulierung und Erstellung der Grafiken erfolgte eigenständig, jedoch in gemeinsamer Abstimmung.

---

Freiburg, den 28. Oktober 2009

# **Danksagung**

An dieser Stelle möchte ich mich bei allen Personen bedanken, die mich während meiner Masterarbeit unterstützt haben. Besonderer Dank gilt meinem Betreuer Dr. Dirk von Suchodoletz, der mir bei der Beantwortung jeglicher Fragen geholfen hat und mich mit Ideen und Anregungen sowie mit konstruktiver Kritik bei der Ausarbeitung unterstützte. Des Weiteren möchte ich Klaus Rechert danken, der mir stets beim Programmieren weitergeholfen und sich immer die Zeit dafür genommen hat. Spezieller Dank gilt auch meinen Studienkollegen Holger Bertsch und Konrad Meier. Gerade zu Beginn der Masterarbeit war die Zusammenarbeit besonders hilfreich um Hard- und Softwareprobleme zu lösen sowie Ideen und Lösungsansätze bei anderen Problemen zusammenzutragen. Ebenso möchte ich mich bei allen Korrekturleser/-innen bedanken.

# **Abstract**

Über die Sicherheit in GSM-Netzen herrschte lange Zeit Unklarheit. Auf Grund fehlender Hardware konnten sicherheitskritische Aspekte nicht genauer untersucht werden. Dies betraf auch den IMSI-Catcher, der in der Öffentlichkeit diskutiert wird, aus sicherheitsrelevanten Gründen allerdings nicht allgemein verfügbar ist und in Deutschland nicht eingesetzt werden darf. Mit Erscheinen geeigneter Hardware sind Sicherheitsuntersuchungen und die Entwicklung eines IMSI-Catchers erstmals möglich geworden. Basierend auf einem Software-defined radio (USRP) und den Softwareprojekten GNU Radio, OpenBTS und Asterisk, soll ein prototypischer Open Source IMSI-Catcher und ein komfortables Frontend entwickelt werden. Ziel dieser Arbeit ist es, bestehende Sicherheitslücken mit Hilfe des IMSI-Catchers zu demonstrieren. Diesbezüglich wird gezeigt, wie Mobilfunkteilnehmer sich ohne ihr Wissen in den IMSI-Catcher einbuchen und überwacht werden können. Die Kontrolle, der sie hierbei unterliegen beinhaltet abgesehen vom Auslesen der IMSI- und IMEI-Nummer auch eine Auflistung aller aktuell geführten Gespräche und die Möglichkeit diese aufzuzeichnen. Neben der Entwicklung und Demonstration des IMSI-Catchers soll außerdem geklärt werden, ob der Teilnehmer einen IMSI-Catcher entdecken oder sich davor schützen kann.

# **Disclaimer**

Alle in dieser Arbeit beschriebenen Verfahren und Experimente erfolgten in einem besonders abgeschotteten Bereich. Hierdurch ergaben sich keine Störungen für jegliche Teilnehmer oder vorhandene GSM-Netze. Die Verwendung der Arbeit sowie praktische Ausführungen bedürfen einer besonderen Genehmigung durch die entsprechenden Behörden und sind daher nur mit Lizzenzen gestattet.

# Inhaltsverzeichnis

<b>1 Einleitung und Motivation</b>	<b>1</b>
<b>2 GSM</b>	<b>3</b>
2.1 Historischer Überblick . . . . .	3
2.2 Frequenzen . . . . .	4
2.3 GSM-Systemarchitektur . . . . .	6
2.3.1 Mobile Station . . . . .	6
2.3.2 Base Station Subsystem . . . . .	9
2.3.3 Network Subsystem . . . . .	11
2.3.4 Operation and Maintenance Center . . . . .	15
2.4 Luftschnittstelle . . . . .	16
2.4.1 Burst . . . . .	17
2.4.2 Logische Kanäle . . . . .	19
2.5 ISO/OSI . . . . .	23
2.5.1 Layer 1 - Bitübertragung . . . . .	25
2.5.2 Layer 2 - Sicherung . . . . .	25
2.5.3 Layer 3 - Vermittlung . . . . .	25
<b>3 Hardware</b>	<b>26</b>
3.1 USRP . . . . .	26
3.2 USRP2 . . . . .	28
3.3 Hardwarekomponenten . . . . .	28
3.4 Hardwaremodifikationen . . . . .	29
3.4.1 RFX900 ISM-Band Filter . . . . .	29
3.4.2 Taktgeber . . . . .	30
3.4.3 USRP . . . . .	32
3.5 Testumgebung . . . . .	33
3.5.1 Computer . . . . .	33
3.5.2 Mobilfunktelefone . . . . .	33
<b>4 Software</b>	<b>35</b>
4.1 GNU Radio . . . . .	36
4.2 OpenBTS . . . . .	37
4.2.1 Installation . . . . .	37
4.2.2 Patch . . . . .	38
4.2.3 Konfiguration . . . . .	38
4.3 Asterisk PBX . . . . .	39
<b>5 IMSI-Catcher</b>	<b>41</b>
5.1 Vorbedingung . . . . .	42
5.1.1 Hardware . . . . .	42
5.1.2 Software . . . . .	42

5.2	GSM-Sicherheitsaspekte . . . . .	42
5.2.1	IMSI - TMSI . . . . .	42
5.2.2	Verschlüsselung der Kommunikation . . . . .	43
5.3	Funktionsweise . . . . .	43
5.3.1	Reale Funknetze simulieren . . . . .	44
5.4	Registrierung . . . . .	45
5.4.1	Registrierung – GSM . . . . .	45
5.4.2	Registrierung – Standard IMSI-Catcher . . . . .	47
5.4.3	Registrierung – Open Source IMSI-Catcher . . . . .	48
5.4.4	Erzwungener Zellwechsel . . . . .	49
5.4.5	Jammer . . . . .	51
5.5	Rufaufbau . . . . .	55
5.5.1	GSM-Rufaufbau . . . . .	56
5.5.2	IMSI-Catcher-Rufaufbau . . . . .	57
5.6	Änderungen . . . . .	57
5.6.1	SIP-Account erstellen . . . . .	58
5.6.2	Selektive Registrierung . . . . .	59
5.6.3	IMEI auslesen . . . . .	61
5.7	OpenBTS als IMSI-Catcher . . . . .	62
5.7.1	Funktionsumfang . . . . .	63
5.7.2	Aufnahme . . . . .	64
5.7.3	Inbetriebnahme . . . . .	65
<b>6</b>	<b>Fazit und Ausblick</b>	<b>67</b>
<b>Literaturverzeichnis</b>		<b>71</b>
<b>Abkürzungsverzeichnis</b>		<b>73</b>
<b>Abbildungsverzeichnis</b>		<b>75</b>
<b>A</b>	<b>Zusammenbau des externen Taktgebers</b>	<b>77</b>
<b>B</b>	<b>Installationsanleitungen</b>	<b>78</b>
B.1	GNU Radio . . . . .	78
B.2	OpenBTS / Open Source IMSI-Catcher . . . . .	80
B.3	OpenBTS-Patch . . . . .	81
<b>C</b>	<b>Code</b>	<b>83</b>
C.1	createSIP - Funktion . . . . .	83
C.2	acceptIMSI - Funktion . . . . .	84
C.3	resolveIMEI - Funktion . . . . .	84
<b>D</b>	<b>GSM900 Downlink ARFCN - Frequenz Tabelle</b>	<b>86</b>
<b>E</b>	<b>GSM1800 Downlink ARFCN - Frequenz Tabelle</b>	<b>87</b>
<b>F</b>	<b>CD-Inhalt</b>	<b>89</b>

# 1 Einleitung und Motivation

GSM ist heutzutage internationaler Standard für Mobilfunknetze. Mehr als 3,5 Milliarden Menschen telefonieren weltweit, Tendenz steigend. In Deutschland gibt es seit April 2008 sogar mehr Mobilfunkanschlüsse als Einwohner, insgesamt über 100 Millionen. Statistisch besitzt somit jeder fünfte Deutsche zwei Mobilfunktelefone bzw. Mobilfunkkarten. Trotzdem liegt Deutschland bezüglich der absoluten Anzahl der Anschlüsse im weltweiten Vergleich nur auf dem siebten Platz.<sup>1</sup> GSM galt lange Zeit als sicher, weshalb die Teilnehmer unbesorgt darauf vertrauten. Trotz offener Spezifikationen wurden einige sicherheitsrelevante Aspekte unter Verschluss gehalten, getreu dem Motto: Security by Obscurity.<sup>2</sup> Dieses Prinzip funktionierte solange es keine erschwingliche Hardware für die breite Masse gab, um GSM näher zu untersuchen.

1996, fünf Jahre nach dem GSM-Netzstart, wurde der erste IMSI-Catcher vorgestellt. Ein IMSI-Catcher täuscht eine GSM-Basisstation vor und bildet die gesamte Infrastruktur nach. Das primäre Ziel eines IMSI-Catchers ist es, die Mobilfunktelefone dazu zu bringen sich aus dem offiziellen GSM-Netz auszubuchen und sich in das Netz des IMSI-Catchers einzubuchen. Ursprünglich wurde der IMSI-Catcher entwickelt, um eine zu observierende Person abzuhören. Ist die Mobilfunknummer nicht bekannt oder telefoniert die Zielperson über das Festnetz, ist es nicht möglich die Gespräche über den Provider abzuhören. In diesen Fällen wird der IMSI-Catcher eingesetzt. In einem GSM-Netz besitzt jeder Teilnehmer eine eindeutige Kennung, durch die der jeweilige Anbieter alle Teilnehmer identifizieren kann. Diese eindeutige Identifizierung ist eine Voraussetzung für den Netzzugang und ermöglicht auch eine korrekte Gebührenabrechnung. Mit Hilfe des IMSI-Catchers kann diese Kennung ausgelesen werden. Außerdem können die Gespräche der Teilnehmer abgehört oder auch deren Aufenthaltsort bestimmen werden. Darüber hinaus kann der IMSI-Catcher den Zugang zum offiziellen Mobilfunknetz blockieren. Auf diese Weise hat er die Kontrolle über die eingebuchten Teilnehmer. Die Nutzung eines IMSI-Catchers ist in Deutschland nicht legal, da dieser auf dem GSM-Frequenzband sendet, das durch Lizenzen geschützt ist. Ebenso sind Abhörmaßnahmen datenschutzrechtlich nicht gestattet, weshalb der Erwerb eines IMSI-Catchers ausschließlich Behörden vorbehalten ist.

Seit Januar 2005 wird von der Firma Ettus Research LLC<sup>3</sup> ein Universal Software Radio Peripheral angeboten. Dabei handelt es sich um eine Hardware, die es ermöglicht, nahezu jedes Funksignal zu digitalisieren. Die gesamte Signalverarbeitung wird anschließend auf einem angeschlossenen Computer durchgeführt und mittels entsprechender Software interpretiert. Daraus ergeben sich für diese Hardware vielfältige Anwendungsmöglichkeiten. Als Basis müssen lediglich verschiedene Empfangs- und Sendemodule eingebaut werden. Mit dieser Hardware ist es jetzt zum ersten Mal möglich sicherheitsrelevante Themen in GSM zu untersuchen. Während die Sicherheit im Internet bereits vielfach betrachtet worden ist, konnten eventuelle

<sup>1</sup> Mehr als 100 Millionen Mobilfunkanschlüsse in Deutschland, <http://www.handy-market.com/news/mehr-als-100-millionen-mobilfunkanschluesse-in-deutschland-5300.php> [Online; letzter Aufruf 09.10.2009]

<sup>2</sup> Ein System wird solange als sicher erachtet, solange seine Funktionsweise geheim bleibt.

<sup>3</sup> Ettus Research LLC, <http://www.ettus.com/> [Online; letzter Aufruf 20.09.2009]

Schwachstellen des GSM lange Zeit nur theoretisch untersucht werden. Aus diesem Grund ist GSM ein interessantes Forschungsgebiet, da mittlerweile geeignete Hardware zur Verfügung steht, um bestehende Sicherheitslücken zu analysieren.

Mit Hilfe der oben beschriebenen Hardware und dem Open Source Softwareprojekt OpenBTS ist es möglich eine eigene GSM-Zelle nachzubilden. OpenBTS sollte Länder mit geringer Netzabdeckung bzw. Entwicklungsländer oder auch spärlich besiedelte Gebiete kostengünstig mit GSM versorgen. Da ein IMSI-Catcher eine GSM-Zelle nachbildet, wird OpenBTS als Basis für den Open Source IMSI-Catcher genutzt.

Ziel dieser Arbeit ist es die Funktionsweise eines IMSI-Catchers zu demonstrieren, sowie einen prototypischen Open Source IMSI-Catcher unter Zuhilfenahme der frei verfügbaren Hard- und Softwarekomponenten USRP, GNU Radio, OpenBTS und Asterisk zu bauen. Darauf Aufbauend soll ein komfortables Frontend für den Open Source IMSI-Catcher entwickelt werden. Die geänderte Funktionsweise des Open Source IMSI-Catchers soll mit der des Standard IMSI-Catchers verglichen und die sich daraus ergebenden Probleme erläutert werden. Es soll überdies gezeigt werden, welche Methoden notwendig sind, um den Mobilfunkteilnehmer ohne dessen Kenntnis in den IMSI-Catcher einzubuchen. Die Verwundbarkeit von GSM wird hierbei besonders deutlich, da GSM keinen Schutz vor einem IMSI-Catcher bietet.

Da der IMSI-Catcher eine reale Funkzelle vortäuscht und die gesamte Infrastruktur in Software abbildet, werden die GSM-Infrastruktur und ihre Komponenten in Kapitel 2 näher betrachtet. Es enthält eine Einführung und einen Überblick über die Aufgabengebiete der einzelnen Komponenten und deren Funktionsweise. Hierbei sollen die Grundlagen für das Verständnis der Funktionsweise eines IMSI-Catchers vermittelt werden. Im darauffolgenden Kapitel 3 werden das USRP und zusätzliche Hardwaremodule sowie notwendige Modifikationen für den IMSI-Catcher beschrieben. Anschließend werden in Kapitel 4 die für die Ansteuerung der Hardware wichtige Softwarekomponente GNU Radio, sowie die für den Betrieb des IMSI-Catchers notwendige Software OpenBTS und Asterisk näher betrachtet. Der neu entwickelte Open Source IMSI-Catcher wird schließlich in Kapitel 5 beschrieben. Hierbei wird zunächst die Funktionsweise mit einem Standard IMSI-Catcher verglichen. Anschließend erfolgt eine Beschreibung, wie sich die Teilnehmer in den Open Source IMSI-Catcher einbuchen und wie der Rufaufbau stattfindet. Es werden verschiedene Lösungswege für die aufgetretenen Probleme diskutiert. Der Open Source IMSI-Catcher basiert auf dem Code von OpenBTS. Da OpenBTS nicht für den Betrieb eines IMSI-Catchers entwickelt wurde, ergibt sich die Notwendigkeit den Code zu modifizieren und weitere Funktionen zu implementieren. Diese Änderungen und die Bedienung des IMSI-Catchers und andere notwendige Tools werden ebenfalls beschrieben. Abschließend werden in Kapitel 6 mögliche Weiterentwicklungen aufgezeigt sowie die Ergebnisse zusammengefasst.

Für ein besseres Leseverständnis wurden alle Ordner, Dateien sowie Befehle die im Text verwendet werden, kursiv hervorgehoben. Befehle sowie Konfigurationsparameter, die außerhalb des Fließtextes erscheinen, wurden in Maschinenschrift geschrieben. Überdies erscheinen selbstgeschriebene Codefragmente in sog. Listings. Der vollständige Code, sowie weiter Dateien befinden sich auf der CD, dessen Inhalt dem Anhang F entnommen werden kann. Für eine verkürzte Schreibweise in den Pfadangaben wird die Bezeichnung „osic“ für den Open Source IMSI-Catcher verwendet.

## 2 GSM

Das Global System for Mobile Communication – kurz GSM – ist ein standardisiertes, vollständig digitales Mobilfunknetz, das zunächst für den europäischen Markt entworfen wurde, heute allerdings auf der ganzen Welt eingesetzt wird. Rechts abgebildet ist das offizielle Logo der Vereinigung der GSM-Mobilfunkanbieter, der 1987 gegründeten GSM-Association. Ziel der Standardisierung war es, die analogen Mobilfunknetze der ersten Generation (in Deutschland A,B und C) abzulösen. Daher wird GSM auch als 2G Standard bezeichnet. Im Jahr 2001 wurden die Lizenzen für die dritte Generation (3G) – besser bekannt als UMTS – vergeben. Bisher konnte sich UMTS jedoch nicht flächendeckend durchsetzen. Aus diesem Grunde ist GSM das bisher weitverbreitetste Netz. Es zählt wie das Festnetz zu den leitungsvermittelnden Kommunikationsnetzen und wurde ursprünglich für die Übertragung von Sprache ausgelegt. GSM wurde allerdings stetig erweitert, so dass der Zugang zum Internet mit Hilfe des GPRS-Datendienstes möglich ist.



Offizielles GSM-Logo

### 2.1 Historischer Überblick

1982 wurde eine Arbeitsgruppe für Mobilfunk mit dem Namen „Groupe Spécial Mobile“ damit beauftragt einen pan-europäischen Mobilfunkstandard zu entwickeln. Das Ziel war ein mobiles Telefonystem anzubieten, welches den Teilnehmern europaweite Mobilität erlaubt und mit ISDN oder herkömmlichen analogen Telefonnetzen kompatibel ist. Die Nutzung zur Datenübertragung stand dabei zunächst nicht im Mittelpunkt, wurde aber bis heute durch Zusatzspezifikationen hinsichtlich der Datenrate stetig verbessert. Das Problem der alten analogen Mobilfunknetzen (in Deutschland A,B und C), die innereuropäisch unterschiedlich waren, sollte durch die Standardisierung gelöst werden. Daher unterzeichneten 17 GSM-Betreiber aus 15 verschiedenen europäischen Ländern am 7. September 1987 das GSM MoU (Memorandum of Understanding). Die erste Spezifikation für GSM wurde 1990 fixiert und bildete somit die Grundlage für das GSM-Netz, welches erstmalig 1991 in Finnland in Betrieb genommen wurde. In Deutschland und anderen europäischen Ländern begannen 1992 einige Betreiber mit einem kommerziellen Netzstart. Ein großer Vorteil der Standardisierung stellt die Möglichkeit dar, GSM-Komponenten unterschiedlicher Hersteller in verschiedenen Ländern miteinander zu kombinieren. Infolgedessen wurden 1993 die ersten internationalen Roamingabkommen abgeschlossen.[25] Daher ist jeder Teilnehmer über seine Rufnummer in einem Land, das ein Roamingabkommen mit dem Anbieter abgeschlossen hat, erreichbar. Dies alles trug dazu bei, dass GSM heutzutage der weitverbreitetste Mobilfunkstandard der Welt ist. Mehr als 220 Länder haben sich für GSM entschieden und betreiben über 860 GSM-Netze.[12] Seit Juni 2009 telefonieren überdies mehr als 3,5 Milliarden Menschen weltweit mit einem Mobilfunktelefon oder nutzen Datendienste, Tendenz weiter steigend.[19] Der jährliche Umsatz mit GSM-Technik liegt laut einer Studie der Deutschen Bank bei geschätzten 277 Milliarden Dollar. Die wichtigsten Ereignisse und weitere Meilensteine der Mobilfunktechnik können der Tabelle 2.1 entnommen werden.

Jahr	Ereignis
1982	Groupe Spécial Mobile soll einen einheitlichen pan-europäischen Mobilfunkstandard entwickeln.
1987	17 GSM-Netzbetreiber aus 15 Ländern unterschreiben das GSM MoU.
1990	Fertigstellung der Phase 1 GSM900-Spezifikationen (für 900 MHz) Start der DCS1800 System-Entwicklung (Digital Cellular System, für 1800 MHz).
1991	Fertigstellung der DCS1800-Spezifikationen Die ersten lauffähigen GSM-Systeme werden auf Messen vorgeführt. Finnland startet erstes GSM900-Netz.
1992	In Europa beginnen die meisten GSM-Netzbetreiber mit dem kommerziellen Netzstart. Ende 1992 sind bereits 13 Netze in 7 Ländern verfügbar.
1993	Die ersten internationalen Roamingabkommen werden abgeschlossen, wodurch Telefonieren auch in anderen GSM-Ländern möglich wird.
1995	Short Message Service (SMS) eingeführt.
1999	Einführung von WAP (Wireless Application Protocol). Damit können erstmalig Inhalte des Internets auf mobile Endgeräte übertragen werden.
2000	Einführung von GPRS (General Paket Radio Service) als Erweiterung zu GSM, um einen drahtlosen Zugang zu paketorientierten Datendiensten mit bis zu 171,2 kbit/s zu ermöglichen.
2000	Einführung eines neuen Mobilfunkstandards der dritten Generation (3G): UMTS (Universal Mobile Telecommunication System)

Tabelle 2.1: Wichtige Meilensteine der Mobilfunkkommunikation [16], [26]

## 2.2 Frequenzen

Die Kommunikation bei GSM erfolgt über Funkkanäle in bestimmten Frequenzbereichen, auch Frequenzbänder genannt, wobei für den Uplink (Mobilfunktelefon zum Netz) und den Downlink (Netz zum Mobilfunktelefon) jeweils unterschiedliche Frequenzen genutzt werden. Diese zwei Frequenzbänder gewährleisten den Duplexbetrieb, der dem Mobilfunkteilnehmer gleichzeitiges Reden und Hören ermöglicht. Die Vergabe der Lizenzen für die Frequenzbänder ist nicht global geregelt. In Deutschland ist hierfür die Bundesnetzagentur<sup>1</sup> zuständig. Verwendet werden in Deutschland zwei Frequenzbereiche, zum einen GSM900, welches das Frequenzband um 900 MHz nutzt und die Grundlage für die heutigen D-Netze (T-Mobile und Vodafone) bildet, zum anderen das DCS1800 (Digital Cellular System) – auch bekannt als GSM1800 –, das wiederum die Grundlage für die E-Netze (E-Plus und O2) bildet und auf dem Frequenzband um 1800 MHz operiert. Für den Uplink stand bei GSM900 zunächst der Frequenzbereich von 890 bis 915 MHz und für den Downlink von 935 bis 960 MHz zur Verfügung. Im Jahr 2005 wurde das Frequenzband um jeweils 10 MHz auf 880 bis 915 beziehungsweise 925 bis 960 MHz erweitert. Da niedrigere Frequenzen bei ihrer Ausbreitung weniger stark gedämpft werden und das Mobilfunktelefon energieeffizient arbeiten muss, werden dem Uplink-Band die niedrigeren Frequenzen zugewiesen. Das GSM1800-Netz, ebenfalls um jeweils 10 MHz erweitert, arbeitet im Frequenzbereich zwischen 1710 und 1785 MHz für den Uplink, während für den Downlink die Frequenzen von 1805 bis 1880 MHz genutzt werden. Abbildung 2.1 stellt die Nutzung der

<sup>1</sup> Die Bundesnetzagentur, <http://www.bundesnetzagentur.de> [Online; letzter Aufruf 30.09.2009]

Up- und Downlink-Frequenzbereiche für GSM900 und GSM1800 übersichtlich dar. Auf Grund der zwei verschiedenen Frequenzen von GSM900 und GSM1800 werden in Deutschland Dual-Band-Mobilfunktelefone bzw. für andere GSM-Netze wie zum Beispiel das GSM1900-Netz in den USA Tri-Band-Mobilfunktelefone benötigt, damit diese unabhängig vom Netzbetreiber funktionieren. Die verschiedenen Frequenzen haben jedoch auch Einfluss auf die Ausbreitung der Mobilfunkwellen. In der Praxis spielt die Ausbreitung der Mobilfunkwellen eine wichtige Rolle, wobei sie abhängig von Hindernissen wie beispielsweise Gebäuden oder Bäumen bzw. deren Überwindung ist. Außerdem lassen sich mit größeren Wellenlängen größere Reichweiten erzielen. Bei GSM900 ergibt sich auf Grund der niedrigeren Frequenz eine größere Wellenlänge als bei GSM1800. Maximal können bei GSM900 Mobilfunkgeräte in einem Radius von 35 km beziehungsweise 8 km bei GSM1800 versorgt werden.[7] Die Versorgung von großflächigen und eher dünn besiedelten Regionen ist folglich bei GSM900 kosteneffizienter. Heutzutage hat sich die ursprüngliche Verteilung der Bänder auf das D- beziehungsweise E-Netz allerdings geändert. So haben T-Mobile und Vodafone zusätzlich Frequenzbereiche im GSM1800-Netz lizenziert, um Engpässe in den GSM900-Frequenzbereichen auszugleichen und in Orten mit hoher Mobilfunklast mehr Kapazität zur Verfügung zu stellen. E-Plus und O2 haben ebenfalls durch die Bundesnetzagentur 2006 Bereiche im GSM900-Netz zugeteilt bekommen. Weltweit existieren noch weitere Frequenzbänder, wie beispielsweise für die Bahn ein R-GSM (Railway GSM) und in den USA ein zusätzliches GSM850-Netz. Eine genauere Übersicht über die verwendeten Frequenzbänder kann der Tabelle 2.2 entnommen werden.

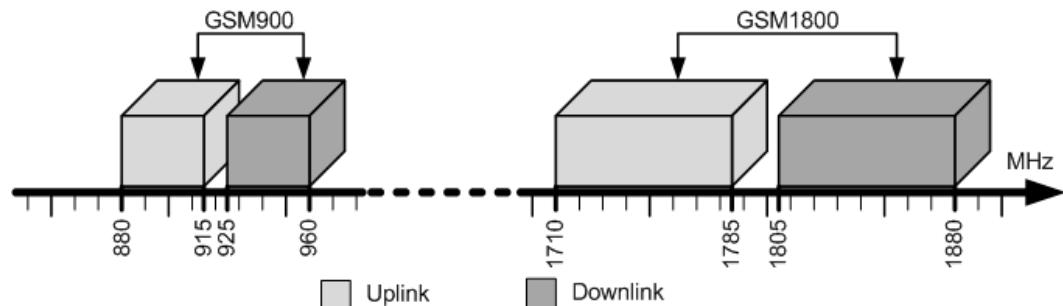


Abbildung 2.1: Frequenzbänder bei GSM900 und GSM1800 (Vorlage nach [21])

Bandbezeichnung	Bereich	Uplink (MHz)	Downlink (MHz)	Kontinent
GSM850	GSM850	824,0-849,0	869,0-894,0	Amerika
P-GSM	GSM900	890,0-915,0	935,0-960,0	Afrika, Amerika, Asien, Australien, Europa
E-GSM	GSM900	880,0-915,0	925,0-960,0	Europa
R-GSM	GSM900	876,0-915,0	921,0-960,0	Asien, Europa
DCS1800	GSM1800	1710,0-1785,0	1805,0-1880,0	Afrika, Amerika, Asien, Australien, Europa
PCS1900	GSM1900	1850,0-1910,0	1930,0-1990,0	Amerika

Tabelle 2.2: Auswahl an wichtigen Frequenzbändern [5], [16], [26]

## 2.3 GSM-Systemarchitektur

Das GSM-Netz basiert auf dem Prinzip von zellulär angeordneten Funkzellen. Dies bedeutet, dass das Versorgungsgebiet auf mehrere Funkzellen aufgeteilt wird, wobei in der Theorie eine Funkzelle eine hexagonale Form und somit sechs Nachbarzellen besitzt. Als Funkzelle bezeichnet man den räumlichen Versorgungsbereich einer Basisstation. Die Basisstation befindet sich hierbei in der Mitte des Hexagons. Zur Vermeidung von Interferenzen dürfen benachbarte Basisstationen nicht auf der gleichen Frequenz arbeiten. Die gleiche Frequenz darf frühestens nach Einhaltung einer Schutzhinterferenz „D“ erneut verwendet werden, diese wird auch „frequency reuse distance“ genannt. D kann aus der geometrischen Form der Hexagone abgeleitet werden und ist von der Clustergröße k und dem Zellradius R abhängig:  $D = R\sqrt{3k}$ . Die Abbildung 2.2 veranschaulicht verschiedene Clustergrößen, wobei eine Gruppe von Zellen einen Cluster bilden, welche den gesamten Frequenzbereich abdecken. Überdies benutzen Zellen mit derselben Nummerierung die gleichen Frequenzen. Mit Hilfe von Messungen und Computersimulationen hat sich eine minimale Clustergröße von 7 als vorteilhaft herausgestellt. Der Abbildung 2.3 kann solch eine theoretische Anordnung der Hexagone im Vergleich

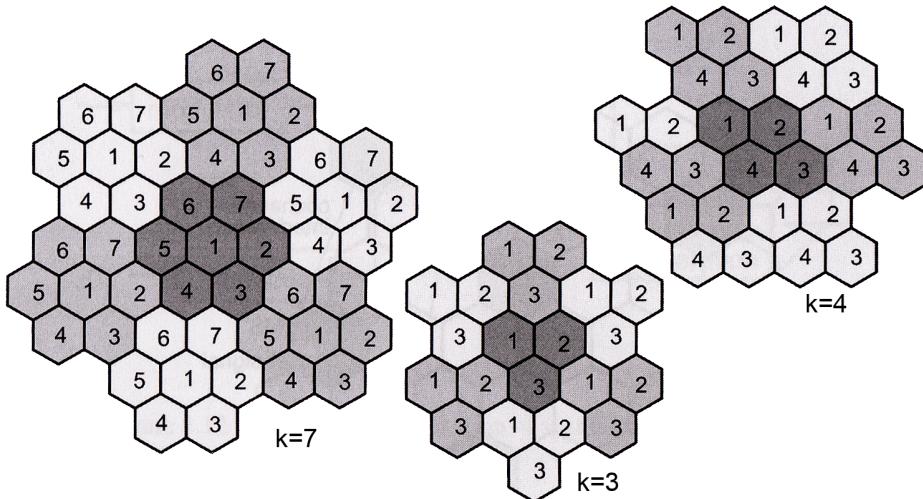


Abbildung 2.2: Frequency reuse innerhalb verschiedener Clustergrößen [13]

zu einer in der Praxis realistischen Anordnung der Funkzellen entnommen werden. Die in der Praxis abweichende Form der Zellen ergibt sich aus verschiedenen Faktoren. Zum einen blockieren Gebäude die Funkwellen oder brechen sie teilweise um. Zum anderen spielen weitere Faktoren wie beispielsweise die Bevölkerungsdichte, Landschaftsbeschaffenheit und technische Parameter eine Rolle. Bewegt sich ein mobiler Teilnehmer von einer Zelle zu einer anderen Zelle, muss gewährleistet werden, dass seine bestehende Verbindung von einer Basisstation zur anderen weitergegeben wird. Man spricht hierbei von einem sogenannten „Handover“. Die gesamte GSM-Infrastruktur ist hierarchisch aufgebaut und lässt sich in die vier Hauptbereiche Mobile Station, Base Station Subsystem, Network Subsystem und Operation Maintenance Center gliedern. Abbildung 2.4 verdeutlicht die Struktur, die im Folgenden näher erläutert wird.

### 2.3.1 Mobile Station

Die Mobile Station (MS) besteht aus dem Mobilfunktelefon und aus der SIM-Karte (Abbildung 2.5). Ohne eine betriebsbereite SIM-Karte kann das GSM-Netz seit dem 13.02.2009

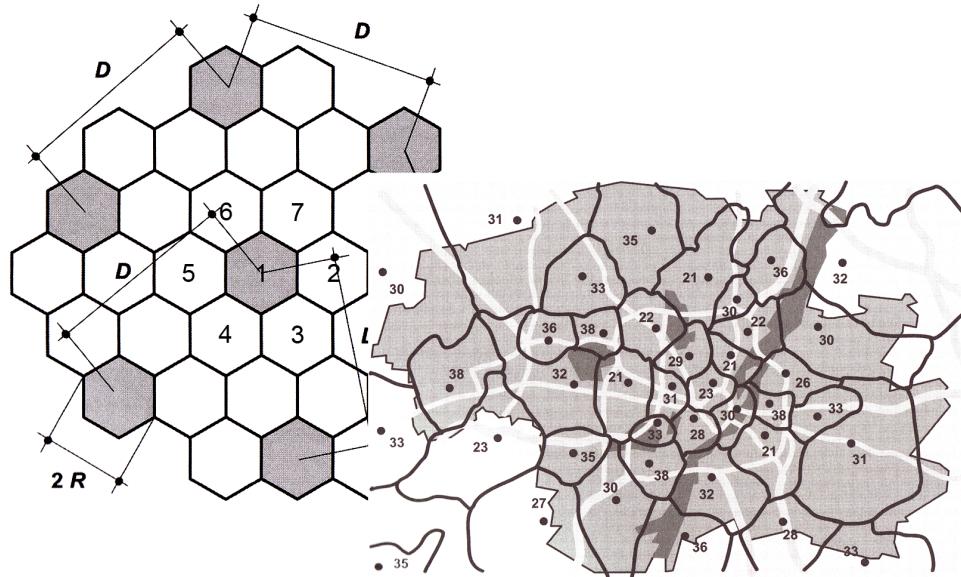


Abbildung 2.3: Vergleich der theoretischen und praktischen Zellenform [13]

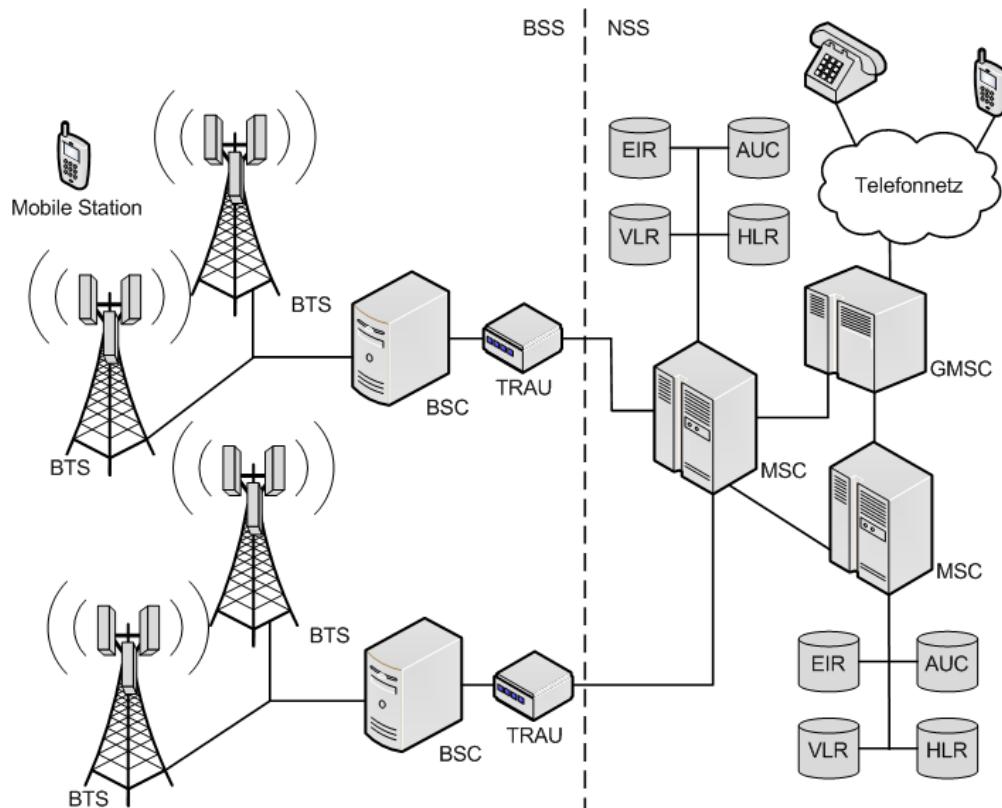


Abbildung 2.4: Struktur eines GSM-Netzes

nicht mehr genutzt werden.<sup>2</sup> Sie ermöglicht es, den Karteninhaber zu identifizieren und zu authentisieren. Die Mobile Station ist durch eine international eindeutige Seriennummer (IMEI, International Mobile Equipment Identity) gekennzeichnet. Die Identifizierung des Benutzers geschieht über die auf der SIM-Karte gespeicherte IMSI (siehe Abschnitt 2.3.3).



Abbildung 2.5: Mobile Station, bestehend aus Mobilfunktelefon und SIM-Karte

### Subscriber Identity Module

Das Subscriber Identity Module ist eine Chipkarte und stellt das Herzstück der Mobile Station dar. Zu den Aufgaben der SIM-Karte gehören die Identifikation des Teilnehmers, die Datenverschlüsselung und die Speicherung von diversen Daten. Die wichtigsten auf der SIM-Karte gespeicherten Daten sind:

- Seriennummer der SIM-Karte
- IMSI: Die International Mobile Subscriber Identity Nummer dient der eindeutigen Identifizierung des Teilnehmers. Sie wird im Home Location Register (Kapitel 2.3.3) des Mobilfunkanbieters gespeichert. Die 15-stellige Nummer setzt sich aus dem dreistelligen Ländercode, aus dem zwei- bis dreistelligen Anbieternetzcode sowie einer höchstens zehnstelligen Teilnehmeridentität zusammen.
- TMSI: Da die IMSI die eindeutige Identifizierung des Teilnehmers erlaubt, dies aber aus Sicherheitsgründen nicht bei jeder Teilnehmeraktion (eingehender oder ausgehender Rufaufbau und SMS) übertragen werden soll, wird dem Teilnehmer beim Einbuchen in das Netz eine temporäre IMSI (TMSI) zugewiesen.
- MSISDN: Die Mobile Subscriber ISDN Number ist die mobile Telefonnummer, über die der Teilnehmer erreichbar ist.
- PIN: Die PIN ist eine Geheimzahl, mit deren Hilfe der Teilnehmer die SIM-Karte freischaltet. Falls der Teilnehmer die Geheimzahl dreimal falsch eingibt, wird die SIM-Karte gesperrt.
- PUK: Falls die PIN dreimal falsch eingegeben wurde, kann mit Hilfe des Entsperrschlüssels PUK (PIN Unblocking Key) die SIM-Karte freigeschaltet werden.
- Liste der Trägerfrequenzen für die Zellenwahl bevorzugter Netze
- Liste der gesperrten Netze

<sup>2</sup>Bis zu diesem Zeitpunkt wurden Notrufe ohne SIM-Karte akzeptiert. Jedoch sind seit der Verordnung [6] Notrufe von einem Mobilfunktelefon nur noch mit einer betriebsbereiten SIM-Karte möglich.[3]

- Bereitgestellter Speicher für Benutzerdaten wie beispielsweise Telefon- und Notizbuch, SMS und zuletzt gewählte Telefonnummern

Für die Authentifizierung und Verschlüsselung – beschrieben in Abschnitt 2.3.3 – notwendige Daten sind:

- Geheimschlüssel Ki: Der private Geheimschlüssel Ki ist für den Authentifizierungsprozess wichtig und wird bei den Algorithmen A3 und A8 verwendet.
- Schlüssel Kc: Wird für die Verschlüsselung der Kommunikationsdaten benötigt
- Zufallszahl RAND: Eine vom Home Location Register generierte Zufallszahl
- Algorithmus A3: Dieser Algorithmus wird bei der Authentifizierung benötigt und verwendet als Eingangsparameter Ki und RAND. Sowohl das Netz als auch die MS generieren auf diese Weise eine Signed Response, die miteinander verglichen werden.
- Algorithmus A8: Generiert den Schlüssel Kc und verwendet als Eingangsparameter ebenfalls Ki und RAND.
- Algorithmus A5: Verschlüsselungsalgorithmus, der als Eingangsparameter Kc verwendet und den Datenstrom (Gesprächsdaten, Rufaufbau und Authentifizierung) verschlüsselt.

### 2.3.2 Base Station Subsystem

Das Base Station Subsystem (BSS) – auch Mobilfunksendesystem genannt – ist beim GSM-Netz für den mobilfunkbezogenen Teil verantwortlich. Bestandteile des BSS sind zum einen die Base Transceiver Stations und zum anderen die Base Station Controller. Diese Netzelemente sind für den Funkverbindungsaufbau zwischen der Mobile Station und dem Network Sub System verantwortlich. Die Qualität der Verbindungsdaten wird während der Verbindung überwacht und eventuelle Reaktionen wie ein Zellenwechsel (Handover) eingeleitet. Die Menge aller BSS wird als Radio Subsystem (RSS) bezeichnet.[26]

#### Base Station Controller

Der Base Station Controller (BSC) ist für die Steuerung und die Kontrolle der an ihm angeschlossenen Base Transceiver Station verantwortlich. An einem BSC können bis zu 100 Base Transceiver Stations angeschlossen und verwaltet werden. Des Weiteren ist er über die Transcoder and Rate Adaption Unit (TRAU, Transcodierungseinheit) mit dem Mobile Switching Center verbunden. Die TRAU übernimmt hierbei die Aufgabe, die mit einer niedrigen Datenrate arbeitenden Nutzkanäle der Funkschnittstelle (siehe Kapitel 2.4) in 64 kBit/s-Kanäle der festen Netzinfrastruktur und umgekehrt zu wandeln. Der BSC stellt für die Zelle eine Art „Datenbank“ dar und leitet die Informationen von den Zellen an das zuständige Mobile Switching Center weiter. Sobald ein Teilnehmer während eines aktiven Gesprächs das Versorgungsgebiet einer Base Transceiver Station verlässt und in das Versorgungsgebiet einer anderen Base Transceiver Station, einem BSC oder dem Mobile Swichting Center, eintritt, muss ein Zellwechsel veranlasst werden, um eine unterbrechungsfreie Übertragung zu gewährleisten. Da der BSC für die Funkressourcenverwaltung zuständig ist, entscheidet er, ob auf Grund der mitgeteilten Funkmessung ein Handover durchgeführt werden muss. Hierzu muss der BSC prüfen, ob in der neuen Zelle ein Funkkanal frei ist und somit der Handover stattfinden kann.<sup>3</sup> Die wichtigsten Aufgaben des BSC's sind:

<sup>3</sup>BSC – Base Station Controller, [http://www.umtslink.at/index.php?page\\_id=GSM\\_base\\_station\\_controller](http://www.umtslink.at/index.php?page_id=GSM_base_station_controller) [Online; letzter Aufruf 30.09.2009]

- Aufbau eines Signalisierungskanals für
  - Rufaufbau sowohl ausgehend als auch eingehend
  - SMS senden und empfangen
- Aufrechterhaltung einer Verbindung, falls notwendig Handover einleiten
- Leistungsregelung, zum Beispiel bei gutem Empfang die Sendeleistung des Mobilfunktelefons reduzieren. Dies ist besonders im Hinblick auf die Energieersparnis des Mobilfunktelefons von besonderer Bedeutung.

### Base Transceiver Station

Die Base Transceiver Station (BTS) – die sogenannte Basisstation – bestehend aus den Senden- und Empfangsanlagen (Transceiver), stellt die Funkverbindung über die Luftschnittstelle  $U_m$  (siehe Kapitel 2.4) zu den Mobile Stations her und überträgt die Daten an den BSC. Je nach Konfiguration werden die Steuerungs- und Überwachungsfunktionen entweder von der BTS oder überwiegend vom BSC wahrgenommen. Das abgedeckte Gebiet einer BTS wird normalerweise nicht durch eine, sondern durch zwei oder drei Transceiver-Einheiten versorgt. Somit spannen zwei  $180^\circ$  beziehungsweise drei  $120^\circ$  Sektorantennen (dargestellt in Abbildung 2.6) eine Funkzelle auf. Die Größe der Zellen kann bedingt durch äußere Einflüsse und der verwendeten Frequenz wenige Meter bis maximal 35 Kilometern betragen. In jeder Funkzelle werden mehrere Frequenzen für die Kommunikation genutzt, jedoch kann nicht das ganze Funkspektrum verwendet werden, da benachbarte Funkzellen sich gegenseitig stören würden.<sup>4</sup> Die Zuordnung eines Satzes bestimmter Frequenzen wird auch als Cell Allocation (CA) bezeichnet. Eine oder mehrere Funkzellen werden zu einer Location Area (LA) zusammengefasst. Zur Identifizierung der LA enthält diese einen eindeutigen Location Area Identifier (LAI), welcher auf dem Broadcast Channel (siehe Kapitel 2.4.2) gesendet wird. Somit kann jede Mobile Station mit Hilfe der LAI den aktuellen Aufenthaltsort feststellen. Im Gegensatz hierzu wird innerhalb einer LA eine einzelne Zelle mit einem eindeutigen Cell Identifier (CI) gekennzeichnet. Zu den wesentlichen Aufgaben einer BTS gehören:<sup>5</sup>

- Aktivierung und Deaktivierung der zugewiesenen Funkkanäle
- Verschlüsselung und Entschlüsselung
- Verbindungskontrolle
- Überwachung des Empfangspegels
- Überwachung der Empfangsqualität
- Einstellung der Sendeleistung
- Signalanpassung an die PCM-Schnittstelle,<sup>6</sup> über die die Verbindung an den BSC erfolgt

<sup>4</sup> Abhängig von der Clustergröße (siehe Abschnitt 2.3) werden die Frequenzen wieder verwendet, was auch als frequency reuse bezeichnet wird.

<sup>5</sup> Base Transceiver Station, [http://de.wikipedia.org/wiki/Base\\_Transceiver\\_Station](http://de.wikipedia.org/wiki/Base_Transceiver_Station) [Online; letzter Aufruf 30.09.2009]

<sup>6</sup> PCM = Puls-Code-Modulation: Beschreibt ein Pulsmodulationsverfahren, welches zeit- und wertkontinuierliche analoge Signale in zeit- und wertdiskrete digitale Signale umwandelt

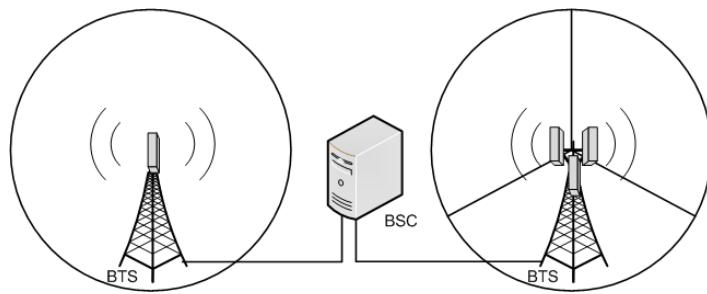


Abbildung 2.6: BTS mit einer 360° Sektorantenne (links) beziehungsweise drei 120° Sektorantennen (rechts)

### 2.3.3 Network Subsystem

Innerhalb des GSM-Netzes stellt das Network Subsystem (NSS) die Vermittlungsstelle und die Schnittstelle zwischen dem GSM-Funknetz und dem Festnetz dar. Gleichzeitig werden notwendige Funktionen wie beispielsweise die Registrierung, Authentifizierung, Ortsüberwachung und Rufmitteilung bereitgestellt. Diese Aufgaben werden vom Mobile Switching Center, Home Location Register, Visitor Location Register, Authentication Center und dem Equipment Identity Register wahrgenommen.

#### Mobile Switching Center

Das Mobile Switching Center (MSC) ist die eigentliche Vermittlungsstelle. Jedes MSC verwaltet eine Vielzahl an BSC's, welche wiederum jeweils mehrere BTS kontrollieren. Der Abbildung 2.7 kann die Relation der einzelnen Komponenten und ihre Verwaltung innerhalb eines Netzanbieters entnommen werden. Zu den wichtigsten Aufgaben des MSC's gehören zum einen die Verbindungskontrolle (Call Control) und zum anderen die Mobilitätskontrolle (Mobility Management). Hierbei wird bei der Verbindungskontrolle der Teilnehmer beim Einschalten seiner Mobile Station im Netz registriert, die Verbindung innerhalb des Funknetzes oder zum Festnetz über ein Gateway (GMSC) hergestellt sowie Kurzmitteilungen (SMS) weitergeleitet. Da der Teilnehmer nicht mit einer festen Leitung mit dem MSC verbunden ist und sich innerhalb des GSM-Netzes frei bewegt, muss die Mobilitätskontrolle einerseits dafür sorgen, dass der Teilnehmer authentifiziert wird und andererseits der Standort des Teilnehmers bekannt ist. Damit das MSC die Aufgaben wahrnehmen kann, muss es zu diesem Zweck eine Anbindung an die HLR, VLR und AUC Datenbanken besitzen. Die Aufgaben des MSC's im Gesamtüberblick: [23]

- Registrierung der Teilnehmer im VLR
- Vernetzung mit anderen MSC's
- Verbindung zu anderen Netzen (Festnetz und Mobilfunknetz)
- Mobilitätsmanagement für leitungsorientierte Dienste
- Umschalten zwischen verschiedenen BSC's bei Inter-BSC Handover
- Echounterdrückung bei Verbindungen zwischen Mobil- und Festnetz
- Verbindungs- und Signalisierungssteuerung
- Gebührenerfassung

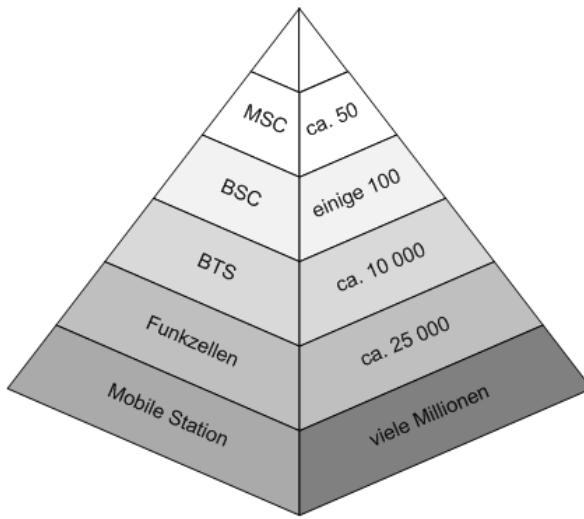


Abbildung 2.7: Mengenrelationen der GSM-Hardwarekomponenten eines Netzanbieters

### Home Location Register

Das Home Location Register (HLR) ist die zentrale Datenbank, die Informationen über alle Mobilfunkteilnehmer des Anbieters speichert. Im HLR sind sowohl alle permanenten Daten wie die eindeutige Kennung der SIM-Karte (IMSI), die Teilnehmerrufnummer (MSISDN), abonnierte Dienste als auch relevante temporäre Daten wie der momentane Aufenthaltsort (Location Area), Adresse des zuständigen VLR's und MSC's sowie Gebührendaten gespeichert. Der aktuelle Aufenthaltsort ist für die Zuordnung von eingehenden Anrufern oder SMS-Übertragungen von Bedeutung. Die abonnierten Dienste lassen sich in Basisdienste und zusätzliche Dienste einordnen. Basisdienste sind zum Beispiel Telefonie, SMS, Fax und Datendienste, wobei hier die zulässige Datenübertragungsgeschwindigkeit eingestellt werden kann. Zu den zusätzlichen Diensten gehören unter anderem das Anklopfen,<sup>7</sup> die Rufumleitung, Anzeige der Rufnummer des Anrufers oder auch Konferenzschaltungen.

### Visitor Location Register

Im Gegensatz zum HLR wird im Visitor Location Register (VLR) lediglich eine Auswahl der Teilnehmerdaten gespeichert, die für das Telefonieren oder andere Services notwendig ist. Dies ist vor allem im Hinblick auf die Sicherheit von Bedeutung, da hierdurch nur die tatsächlich benötigten Daten vorhanden und die Stammdaten nicht einsehbar sind. Pro Anbieter existiert lediglich eine (verteilte) HLR-Datenbank, wohingegen jedem MSC eine VLR-Datenbank zugeordnet wird. Da innerhalb der GSM-Infrastruktur mehrere MSC's (siehe Abbildung 2.7) beteiligt sind, existieren somit mehr VLR- als HLR-Datenbanken. Die VLR-Datenbank enthält Informationen über alle Teilnehmer, die sich im Versorgungsbereich des MSC's befinden. Die Informationen werden hierzu temporär aus dem HLR geladen und als Kopie im VLR gespeichert. Zusätzlich wird im VLR eine temporäre Benutzeridentifikationsnummer (TMSI) abgespeichert. Da bei jedem Verbindungsauftakt die Teilnehmerdaten überprüft werden, ist es die Hauptaufgabe des VLR's, den Signalisierungsaufwand zwischen dem MSC und dem HLR zu reduzieren. Die im VLR gespeicherten Daten werden sofort gelöscht, sobald ein Teilnehmer das Versorgungsgebiet des MSC's verlässt.

<sup>7</sup> Signalisiert während eines aktiven Gesprächs einen weiteren einkommenden Anruf

## Authentication Center

Das Authentication Center (AUC) ist eng mit dem HLR verbunden und ist im Wesentlichen für zwei Sicherheitsaufgaben verantwortlich. Die Aufgaben umfassen die Teilnehmerauthentifizierung und die Verschlüsselung der Daten. Das AUC schützt das GSM-System somit gegen unerlaubte Nutzung beziehungsweise vor dem Abhören oder Abfangen von Benutzerdaten oder Sprach- und Textnachrichten zwischen der  $U_m$  Luftschnittstelle und der Mobile Station.

## Authentifizierung

Die Benutzauthentifizierung findet jeweils beim Einbuchen der Mobile Station statt. Hierbei wird die Identität des Teilnehmers mit dem klassischen Challenge-Response-Verfahren überprüft, wobei unter anderem ein Geheimschlüssel  $Ki$  benötigt wird, der nur auf der SIM-Karte und dem AUC gespeichert ist. Das HLR ist nach erfolgreicher Authentifizierung in der Lage, die SIM- und Mobilfunkdienste zu verwalten. Jedoch muss bei jedem Verbindungsaufbau und bei jedem Location Update<sup>8</sup> die Identität vom AUC überprüft werden. Anhand der IMSI wird zunächst der zugehörige Geheimschlüssel  $Ki$  abgerufen und ein sogenanntes Authentication Triplet gebildet. Das Authentication Triplet besteht aus drei Werten und wird im zuständigen VLR gespeichert:[22]

- RAND: Eine 128-Bit Zufallszahl
- SRES: Die Signed Response (SRES) wird aus  $Ki$  und RAND mit dem Authentifizierungsalgorithmus A3 erzeugt und hat eine Länge von 32-Bit.
- Kc: Der Ciphering Key Kc wird aus  $Ki$  und RAND erzeugt und wird für die Verschlüsselung verwendet.

Wie in Abbildung 2.8 ersichtlich, werden mit Hilfe der Zufallszahl RAND, dem Geheimschlüssel  $Ki$  und dem Authentifizierungsalgorithmus A3 beziehungsweise dem Verschlüsselungsalgorithmus A8 die SRES und der Schlüssel Kc gebildet. Da das Authentication Triplet lediglich für eine Verbindung gültig ist, bei jedem Verbindungsaufbau und Location Update der Teilnehmer jedoch authentifiziert werden muss, werden vom AUC meistens mehrere Triplets

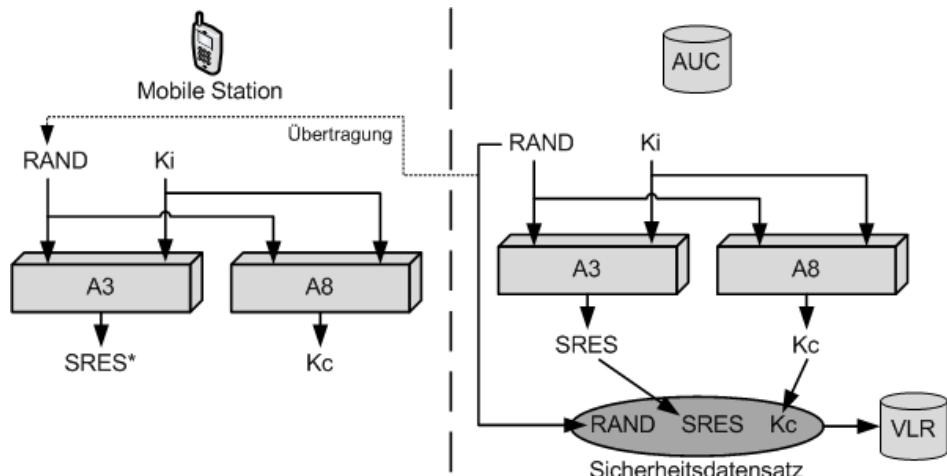


Abbildung 2.8: Generierung des Sicherheitsdatensatzes: RAND, SRES und Kc

<sup>8</sup>Sobald ein Teilnehmer das Versorgungsgebiet einer MSC verlässt und das Versorgungsgebiet einer anderen MSC eintritt, muss der neue Aufenthaltsort dem HLR mitgeteilt und gespeichert werden.

gebildet und im VLR gespeichert. Das MSC überträgt anschließend die Zufallszahl RAND an die Mobile Station. Die Mobile Station berechnet auf dieselbe Weise mit Hilfe des auf der SIM-Karte gespeicherten Geheimschlüssels Ki, des Algorithmus A3 und der übertragenen Zufallszahl eine Signed Response (SRES\*). Die Antwort SRES\* wird im Anschluss an das MSC übertragen, welches SRES und SRES\* miteinander vergleicht. Stimmen beide Werte überein, wurde die Mobile Station erfolgreich authentifiziert (Abbildung 2.9). Wie ebenfalls der Abbildung 2.9 entnommen werden kann, wird der Geheimschlüssel Ki aus Sicherheitsgründen zu keinem Zeitpunkt aus dem AUC oder der SIM-Karte übertragen. Im Gegensatz dazu werden RAND und SRES\* unverschlüsselt übertragen. Da diese Werte nur einmalig verwendet werden, stellt dies kein Sicherheitsrisiko dar. Ein Angreifer kann aus diesen Werten nicht den Geheimschlüssel Ki ermitteln.

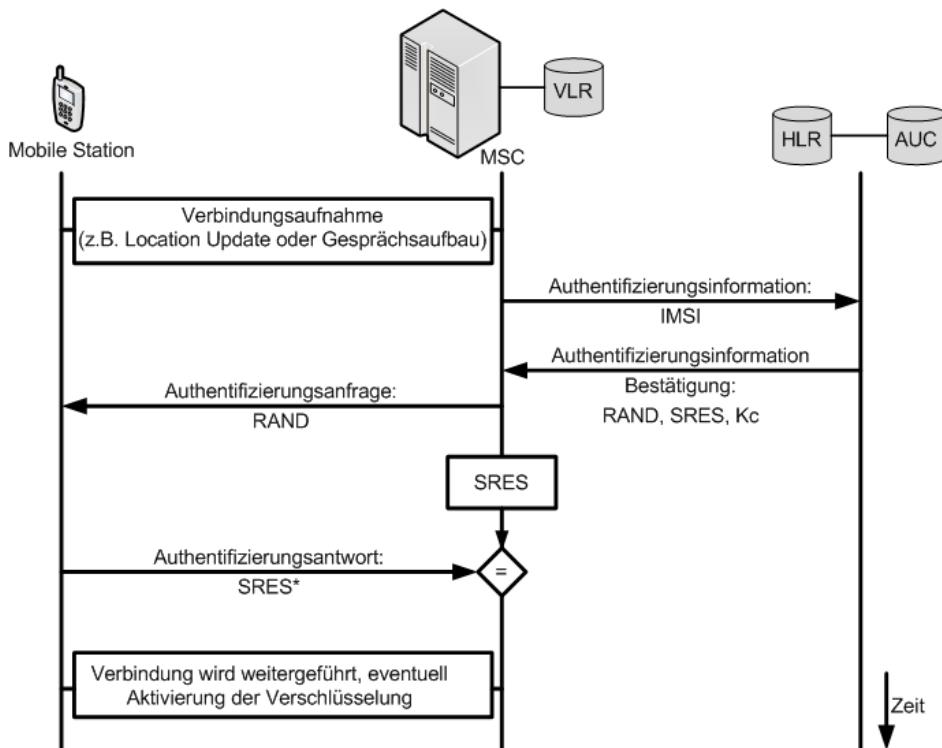


Abbildung 2.9: Nachrichtenfluss bei einer Authentifizierung (Vorlage nach [22])

## Verschlüsselung

Der generierte Schlüssel Kc wird für die Verschlüsselung der gesamten Kommunikation auf der  $U_m$  Luftschnittstelle zwischen der Mobile Station und dem GSM-Netz verwendet. Die Generierung dieses Schlüssels kann der Abbildung 2.8 entnommen werden. Wie in Abbildung 2.9 ersichtlich, wird wie bei der Authentifizierung weder SRES noch Kc vom Authentication Triplet an die Mobile Station übertragen. Da die Verschlüsselung jedoch ein symmetrisches Verfahren<sup>9</sup> ist, muss die Mobile Station über den gleichen Schlüssel Kc verfügen. Dies stellt jedoch kein Problem dar, da die Zufallszahl RAND an die Mobile Station übertragen wird und der Geheimschlüssel Ki sowie der Verschlüsselungsalgorithmus A8 in der SIM-Karte gespeichert sind. Daher kann die Mobile Station auf dieselbe Weise wie das AUC den Schlüssel Kc erstellen (Abbildung 2.8).

<sup>9</sup> Der Sender und der Empfänger verwenden jeweils den gleichen Schlüssel für die Ver- und Entschlüsselung.

Der Kc und die aktuelle TDMA-Rahmennummer (vgl. Kapitel 2.4 Luftschnittstelle) dienen als Eingabeparameter für den Verschlüsselungsalgorithmus A5. Die Abbildung 2.10 beschreibt den Verschlüsselungsprozess, wobei die aktuelle TDMA-Rahmennummer auf dem SCH (vgl. Kapitel 2.4.2 Logische Kanäle) übertragen wird und sich alle 4,615 ms ändert. Somit generiert der Verschlüsselungsalgorithmus A5 alle 4,615 ms eine 114-Bit breite Schlüssel-Bitsequenz. Durch eine XOR-Verknüpfung mit dieser Schlüssel-Bitsequenz wird der unverschlüsselte Datenstrom verschlüsselt. Der nun verschlüsselte Datenstrom wird über die Luftschnittstelle übertragen und mit derselben Schlüssel-Bitsequenz wieder entschlüsselt.

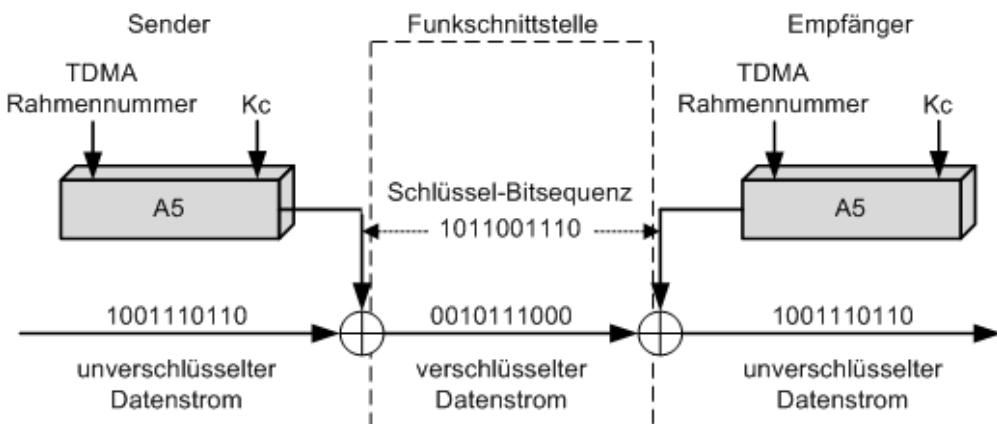


Abbildung 2.10: Symmetrische Verschlüsselung und Entschlüsselung mit einer Schlüssel-Bitsequenz

### Equipment Identity Register

Das Equipment Identity Register (EIR) stellt eine Möglichkeit dar, die IMEI Nummern (International Mobile Station Equipment Identity) der Mobile Stations in eine schwarze, graue oder weiße Liste einzutragen. Die schwarze beziehungsweise graue Liste bedeutet hierbei, dass die Mobile Station wegen Diebstahls oder anderer Gründe gesperrt wurde und somit der Zugang zum GSM-Netz verwehrt werden soll. Demgegenüber steht die weiße Liste, in der alle zugelassenen Geräte aufgeführt sind. Als Diebstahlschutz ist diese Methode jedoch nur bedingt geeignet, da sich die IMEI-Nummern der Geräte neu programmieren lassen. So ist es beispielsweise bei iPhones möglich, mit Hilfe des „Ziphone“<sup>10</sup> Tools oder bei anderen Mobilfunktelefonen mit der „VK Mobile Workshop“<sup>11</sup> Software der Firma Vygis die IMEI Nummer zu ändern.

### 2.3.4 Operation and Maintenance Center

Das Operation and Maintenance Center (OMC) ist die zentrale Betriebs- und Wartungszentrale innerhalb des GSM-Netzes. Zu den Aufgaben des OMC's gehören unter anderem die Verwaltung des BSC's, MSC's und der HLR- und VLR-Datenbanken sowie Fehler im Netz zu erkennen und auf jede Art von Ereignissen wie beispielsweise Überlast oder Ausfall zu reagieren.

<sup>10</sup> Neues Tool von iPhone Hacker Zibri veröffentlicht: Ziphone, <http://www.iphone-dev.de/index.php/2008/02/11/neues-tool-von-iphone-hacker-zibri-veröffentlicht-ziphone/> [Online; letzter Aufruf 30.09.2009]

<sup>11</sup> Vygis VK Mobile Workshop, <http://www.vygis.net/products/vkmobile/VKW.php> [Online; letzter Aufruf 12.09.2009]

## 2.4 Luftschnittstelle

Die Kommunikation zwischen der Mobile Station und der BTS läuft über die Luftschnittstelle ab, die auch als Air-Interface oder als  $U_m$ -Interface bezeichnet wird. Für die Übertragung der digitalen Daten wird eine Mischung aus den Frequenz- und Zeitmultiplexing-Verfahren genutzt. Die Kombination beider Verfahren ist der Abbildung 2.11 zu entnehmen.

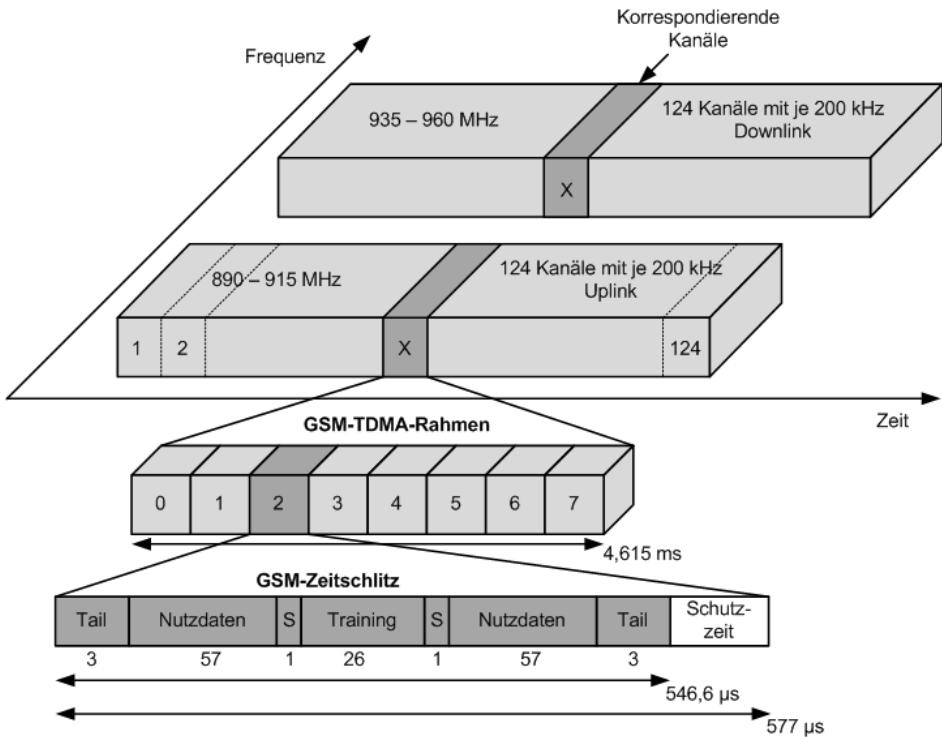


Abbildung 2.11: GSM900-Rahmenstruktur (Vorlage nach [26])

### Frequency Division Multiple Access

Beim GSM-Netz wird das Frequenzband zunächst mit Hilfe des Frequency Division Multiple Access (FDMA, Frequenzmultiplexing) Verfahrens in mehrere Kanäle unterteilt, die jeweils 200 kHz breit sind (Abbildung 2.12). Folglich sind beim GSM900-Netz im Bereich von 890 - 915 MHz 124 Kanäle für den Uplink zur Basisstation und im Bereich von 935 - 960 MHz 124 Kanäle für den Downlink vorgesehen. Da das GSM1800-Netz ein größeres Frequenzspektrum abdeckt, stehen jeweils 374 Kanäle für den Up- und Downlink zur Verfügung. Je ein Kanal der Breite 200 kHz dient für den Uplink und den Downlink als Schutzband. Die Kanäle für den Up- bzw. Downlink erhalten eine eindeutige Nummer. Der Uplink-Kanal bildet mit dem Downlink-Kanal mit derselben Nummer einen Duplex-Kanal, wobei beide Kanäle einen Frequenzabstand von 45 MHz bei GSM900 bzw. 95 MHz bei GSM1800 besitzen. Jedoch reicht das FDMA Verfahren für sich genommen nicht aus, da pro Kanal nur ein Teilnehmer versorgt werden kann. Folglich können bei einer Sektorisierung (vgl. Kapitel 2.3.2 Base Transceiver Station) der Funkzelle in drei Sektoren und einer Zuweisung von zwei Frequenzen pro Sektor insgesamt nur sechs Teilnehmer gleichzeitig versorgt werden. Auf Grund dessen wird das FDMA-Verfahren mit dem TDMA-Verfahren kombiniert.

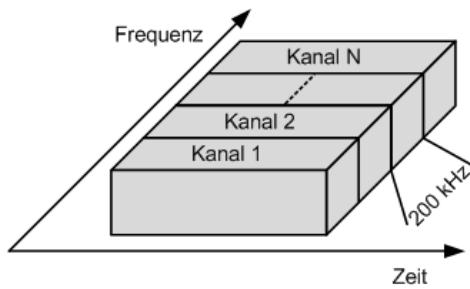


Abbildung 2.12: Frequency Division Multiple Access

### Time Division Multiple Access

Erst die Kombination aus dem FDMA-Verfahren und dem Time Division Multiple Access (TDMA) Verfahren ermöglicht eine ausreichende Menge an gleichzeitiger Kommunikation mehrerer Teilnehmer. Jeder FDMA Kanal wird in acht Zeitschlitzte mit einer Länge von jeweils  $577 \mu\text{s}$  unterteilt, wobei die Gesamtheit aller acht Zeitschlitzte als TDMA-Rahmen bezeichnet wird und die Rahmendauer 4,615 ms ( $8 \cdot 577 \mu\text{s}$ ) beträgt (Abbildung 2.11). In jedem Zeitschlitzt, auch Burst genannt, kann ein Teilnehmer Daten übertragen beziehungsweise empfangen. So mit können acht Teilnehmer gleichzeitig pro Kanal kommunizieren, wobei jedem Teilnehmer ein fester Zeitschlitzt zugewiesen wird. Der Teilnehmer nutzt für den Uplink und den Downlink den Zeitschlitzt mit der gleichen Nummer. Damit eine Antenne an der Mobile Station ausreicht, werden die Uplink und Downlink Zeitschlitzte um jeweils drei Zeitschlitzte versetzt (Delay) (Abbildung 2.13). Somit ist sichergestellt, dass nach einem Sende-Burst die Mobile Station genügend Zeit besitzt, auf die Empfangsfrequenz umzuschalten. Im Vergleich zum FDMA stehen bei gleicher Sektorierung der Funkzelle in drei Sektoren mit jeweils zwei Frequenzen nominell 16 Zeitschlitzte pro Sektor für die Kommunikation zur Verfügung. Von den 16 Zeitschlitzten werden jedoch vier oder mehr für den GPRS Datendienst und zwei für die Signalisierung verwendet. Hieraus ergibt sich eine Versorgung von 30 Teilnehmern innerhalb der Funkzelle. Es wird jedoch davon ausgegangen, dass innerhalb einer Stunde ein Teilnehmer nur eine Minute telefoniert. Grob geschätzt kann eine BTS 60 mal mehr passive als aktive Teilnehmer und demzufolge insgesamt 1800 Teilnehmer pro Funkzelle versorgen. [22]

#### 2.4.1 Burst

Für die Übertragung der Sprach- und Signalisierungsdaten werden verschiedene Bursts gebraucht. Es existieren insgesamt fünf verschiedene Burst-Typen: Normal Burst, Frequency Correction Burst, Synchronization Burst, Dummy Burst und Access Burst. Jeder Burst unterscheidet sich in seinem Aufbau, wobei alle eine Schutzzeit (Guard Time) besitzen, in der keine Daten gesendet oder empfangen werden. Dieser mindestens 8,25-Bit breite Bereich ist dafür zuständig, dass bei Überschneidungen mit anderen Bursts keine Übertragungsfehler entstehen. Solche Überschneidungen können aufgrund von Timing-Problemen auftreten. Ursachen hierfür sind unter anderem, dass sich der Teilnehmer von einer Basisstation entfernt, sich hinter Gebäuden oder Bäumen bewegt, die Funkwellen gebeugt oder durch Mehrwegausbreitung gespiegelt werden. Die Guard Time reicht alleine jedoch nicht aus, um Überschneidungen der Bursts zu verhindern. Da die Funkwellen bei größerer Entfernung eine längere Strecke zurücklegen müssen, wird anhand der Timing Advance Regelung der Sendezeitpunkt überwacht und angepasst. Ohne Timing Advance Regelung treffen Bursts von weiter entfernten Teilnehmern später ein und überschneiden sich mit dem Burst im nächsten Zeitschlitzt (Abbildung

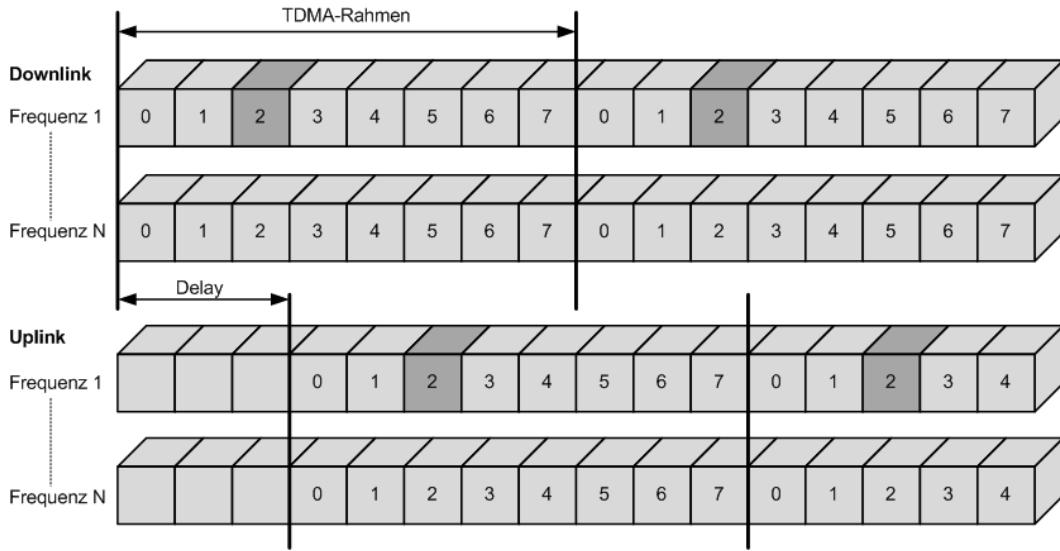


Abbildung 2.13: Zeitversetzte Kommunikation für den Down- und Uplink

2.14). Umso weiter ein Teilnehmer von der Basisstation entfernt ist, um so früher muss der Burst gesendet werden, damit dieser rechtzeitig bei der Basisstation eintrifft. Der Aufbau der einzelnen Bursts ist in Abbildung 2.16 dargestellt.

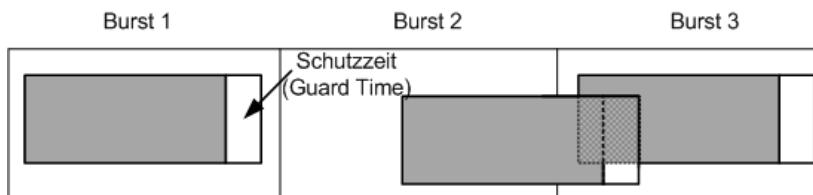


Abbildung 2.14: Zeitverschiebung eines Bursts ohne Timing Advance Regelung

- Normal Burst (Up- und Downlink): Der Normal Burst wird für die Übertragung von Nutzinformationen wie Daten oder Sprache beziehungsweise Signalisierungsdaten für den Up- und Downlink genutzt. Der genaue Aufbau kann der Abbildung 2.15 entnommen werden. Am Anfang und am Ende sind jeweils drei Tail Bits, die immer auf „0“ gesetzt sind und die die Zeit überbrücken, in der die Sendeleistung zu Beginn und am Ende hoch- beziehungsweise heruntergetastet wird und daher nicht für die Datenübertragung zur Verfügung steht. Für die Nutzdatenübertragung stehen dem Teilnehmer jeweils zwei 57-Bit breite Datenblöcke zur Verfügung. Diese Datenblöcke werden mit Hilfe der 26-Bit breiten Training Sequence vor Fehlern geschützt. Die Training Sequence beinhaltet ein vordefiniertes Bitmuster, anhand dessen die Intersymbolinterferenz<sup>12</sup> reduziert beziehungsweise eliminiert werden kann, welche bedingt durch Laufunterschiede bei der Mehrwegausbreitung auftreten. Die Stealing Flags S sind Signalisierungsbits, die angeben, ob der Burst Nutz- oder Signalisierungsdaten überträgt. Sie erlauben es, den Zeitschlitz vom Traffic Channel (mehr dazu in 2.4.2 Logische Kanäle) zu übernehmen, damit Signalisierungsdaten, die beispielsweise beim Handover wichtig sind, schnell

<sup>12</sup>Symbolübersprechung bei digital kodierten Übertragungstechniken zwischen zeitlich aufeinanderfolgenden Sendesymbolen

übertragen werden können. Die ursprünglich zu übertragenden Nutzdaten gehen hierbei verloren.

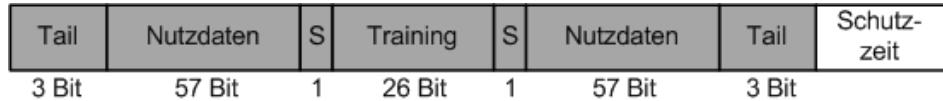


Abbildung 2.15: Aufbau eines Normal Burst

- Frequency Correction Burst (Downlink): Die Mobile Station hört auf dem Downlinkkanal den Frequency Correction Burst ab, um die Frequenz und den TDMA-Rahmen zu synchronisieren. Das wiederholte Senden des Frequency Correction Burst durch die Basisstation bildet den FCCH (Frequency Correction Channel, mehr dazu in 2.4.2 Logische Kanäle).
- Synchronization Burst (Downlink): Mit Hilfe des Synchronization Burst werden Informationen wie beispielsweise die laufende Nummer des TDMA-Rahmens übertragen. Die Mobile Station kann sich hierdurch mit der BTS im Groben synchronisieren.
- Dummy Burst (Downlink): Dieser Burst wird auf den Downlinkkanal gesendet, falls kein anderer Burst gesendet werden soll. Da die Mobile Station kontinuierlich Pegelmessungen durchführt, muss sichergestellt werden, dass vom System ständig ein Burst gesendet wird, selbst wenn keine Daten übertragen werden sollen.
- Access Burst (Uplink): Für die Verbindungsaufnahme der Mobile Station und der BTS wird auf dem RACH (Random Access Channel) der Access Burst benötigt. Im Gegensatz zu den anderen Bursts besitzt der Access Burst eine größere Schutzzeit, um die Wahrscheinlichkeit von Kollisionen auf dem Kanal durch nicht synchronisierte Mobile Stations zu verringern.

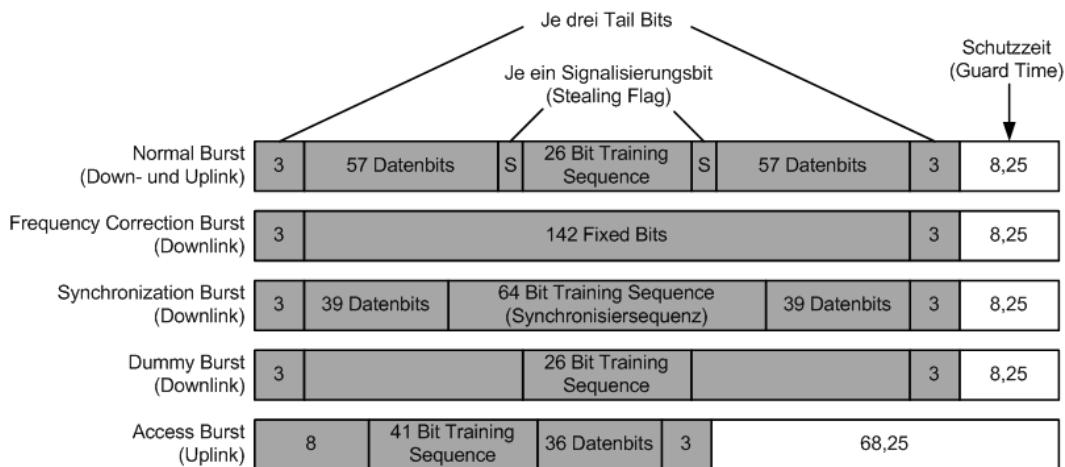


Abbildung 2.16: Übersicht der fünf Burst-Typen (Vorlage nach [13])

## 2.4.2 Logische Kanäle

Die acht Zeitschlüsse (0 bis 7) stellen somit die acht physikalischen Kanäle der Luftschnittstelle dar. Für die Übertragung der Nutz- beziehungsweise Signalisierungsdaten werden die

Zeitschlüsse in logische Kanäle eingeteilt. Zu den Aufgaben der logischen Kanäle zählen: Nutzdatenübertragung, Signalisierung, Broadcast für allgemeine Systeminformationen, Synchronisation und Kanalzuweisung. Die logischen Kanäle lassen sich in Verkehrskanäle und Signalisierungskanäle einteilen. Eine Übersicht und die Gruppierung der logischen Kanäle und ihre Kommunikationsrichtung ist in der Tabelle 2.3 dargestellt.

### Verkehrskanäle

Für die Übertragung der Nutzdaten ist der Traffic Channel (TCH) verantwortlich. Hierbei können entweder Sprachdaten oder leitungsvermittelnde Datendienste übertragen werden. Der TCH kann in zwei Kanalarten unterteilt werden:

- Fullrate (TCH/F): Die Sprachdaten werden mit einer Geschwindigkeit von 13 kbit/s beziehungsweise Daten mit einer Geschwindigkeit von 14,5, 12,6 oder 3,6 kbit/s übertragen.
- Halfrate (TCH/H): Innerhalb des Halfrate Kanals werden die Sprachdaten mit 5,6 kbit/s beziehungsweise Daten mit einer Geschwindigkeit von 6 oder 3,6 kbit/s übertragen.

Gruppe	Kanal	Funktion	Richtung
Traffic Channel	TCH	TCH/F TCH/H	Fullrate TCH Halfrate TCH
	Signaling Channels	BCCH	Broadcast control
		FCCH	Frequency correction
		SCH	Synchronization
	CCCH	RACH	Random access
		AGCH	Access grant
		PCH	Paging
		NCH	Notification
	DCCH	SDCCH	Stand-alone dedicated control
		SACCH	Slow associated control
		FACCH	Fast associated control

Tabelle 2.3: Übersicht und Klassifizierung der logischen Kanäle (Vorlage nach [13])

### Signalisierungskanäle

Die Signalisierungskanäle (CCH, Control Channels) werden in drei Hauptgruppen untergliedert: Broadcast Channel, Common Control Channel und Dedicated Control Channel.

#### Broadcast Channel

Der Broadcast Channel (BCH) kann von allen Teilnehmern abgehört werden. Die Teilnehmer können auf diesem Kanal keine Nutzdaten senden. Die Kommunikation ist somit undirektional. Auf dem BCH werden Systeminformationen der Zelle verbreitet.

- Broadcast Control Channel (BCCH): Wichtige Zellinformationen wie die Kanalkonfiguration (aktuelle und benachbarte Zellen), Synchronisationsinformationen (Frequenzen und Rahmennummerierung) sowie Registrierungskennzeichen (Location Area Identifier (LAI), Cell Identifier (CI)) werden an alle Mobile Stations übertragen.

- Frequency Correction Channel (FCCH): Der FCCH besteht aus dem wiederholten Senden des Frequency Correction Burst. Die Information wird von der Mobile Station für die Frequenzkorrektur benötigt.
- Synchronization Channel (SCH): Mit Hilfe des SCH kann eine BTS identifiziert werden. Darüber hinaus wird der Kanal für die Rahmensynchronisierung einer Mobile Station benötigt.

### Common Control Channel

Die Verbindungsaufnahme wird über den Common Control Channel (CCCH) erledigt. Die Kommunikation stellt eine point-to-multipoint Verbindung dar. Zu seinen Aufgaben zählen die Zuweisung der Kanäle oder das „paging“, womit eine Mobile Station lokalisiert werden kann.

- Paging Channel (PCH): Bei eingehenden Anrufen oder SMS-Nachrichten wird der Teilnehmer über den PCH gesucht, da innerhalb des HLR lediglich die Location Area, nicht aber der genaue Standort des Teilnehmers gespeichert ist. Hierzu wird in jeder Versorgungszelle innerhalb der Location Area eine Nachricht an die betreffende IMSI beziehungsweise TMSI gesendet.
- Random Access Channel (RACH): RACH ist der einzige Kanal, in dem die Kommunikation ausschließlich vom Endgerät Richtung Netz erfolgt. Für die Nutzung des Kanals wird das Slotted-Aloha Verfahren<sup>13</sup> verwendet. Das Endgerät sendet eine Channel Request Nachricht auf diesem Kanal, sobald ein Teilnehmer ein Gespräch beziehungsweise eine SMS verschicken möchte oder über den PCH eine Kontaktaufnahme signalisiert wird. Falls keine Kollision der Channel Request Nachricht auf dem RACH stattgefunden hat, wird dem Teilnehmer über dem AGCH ein dedizierter Kanal zugewiesen.
- Access Grant Channel (AGCH): Sobald ein Teilnehmer über den RACH erfolgreich eine Kanalanfrage durchgeführt hat, wird dem Teilnehmer ein SDCCH oder TCH mit einer Immediate Assignment Nachricht zugewiesen. Die Nachricht enthält wichtige Informationen über den SDCCH beziehungsweise TCH wie beispielsweise die Frequenz und die Rahmennummer.
- Notification Channel (NCH): Über den NCH werden Point-to-Multipoint Gruppeninformationen wie beispielsweise Verkehrsinformationen an die Mobile Station gesendet.

### Dedicated Control Channel

Die Kommunikation über den Dedicated Control Channel (DCCH) erfolgt über eine bidirektionale point-to-point Verbindung. Die Daten sind somit nur für einen einzelnen Nutzer bestimmt.

- Stand-alone Dedicated Control Channel (SDCCH): Der SDCCH ist ein reiner Signallierungskanal zwischen der Mobile Station und dem BSS und wird während des Gesprächsaufbaus, eines Location Updates, oder für das Senden und Empfangen von SMS

<sup>13</sup> Das Slotted-Aloha Protokoll ist ein Zugriffsverfahren und ermöglicht jedem Teilnehmer ohne feste Zuweisung des Kanals in diskreten Zeitabschnitten Daten auf den Kanal zu senden. Da jeder Teilnehmer innerhalb dieses Zeitabschnittes senden darf, kann es zu Kollisionen kommen, weshalb die Daten anschließend nach Abwarten einer zufälligen Zeit erneut gesendet werden müssen.

verwendet, falls der Teilnehmer keine aktive Verbindung (einen zugewiesenen TCH) besitzt. Der Kanal wird über den RACH angefordert und über den AGCH zugewiesen.

- Slow Associated Control Channel (SACCH): Der SACCH wird einem Teilnehmer bei einer aktiven TCH oder SDCCH Verbindung zugewiesen und dient dazu, kontinuierlich Messergebnisse der Signalpegelmessung der aktiven und benachbarten Zellen an den BSC zu senden. Auf Grund der Messergebnisse werden Leistungsregelung der Signalsendestärke der Mobile Station übermittelt oder gegebenenfalls ein Handover eingeleitet. Überdies erhält die Mobile Station Informationen für die Timing Advance Regelung.
- Fast Associated Control Channel (FACCH): Der FACCH wird innerhalb eines Normal Burst, also dem gleichen Timeslot wie der TCH übertragen. Hierzu werden die Stealing Flags (Abbildung 2.16) im Normal Burst gesetzt und die Nutzdaten des Teilnehmers entfernt. Dies dient dazu, dringende Signalisierungsnachrichten wie beispielsweise ein Handover-Kommando schnell übertragen zu können.

Die ersten beiden Zeitschlüsse – Zeitschlitz 0 und 1 – werden üblicherweise für die Signalisierungskanäle reserviert, die restlichen sechs stehen für die Verkehrskanäle zur Verfügung. Auf Grund der Vielzahl an logischen Signialisierungskanälen reichen die zwei physikalischen Kanäle (Zeitschlitz 0 und 1) nicht aus. Daher werden 51 TDMA-Rahmen zu einem sich ständig wiederholenden Multiframe mit einer Zeitspanne von ungefähr 235 ms ( $8 \cdot 577 \mu s \cdot 51$ ) zusammengefasst. In einem solchen Multiframe ist genau festgelegt, in welcher Reihenfolge welche logischen Kanäle übertragen werden. Für die Nutzdatenübertragung werden 26 TDMA-Rahmen mit einer Dauer von circa 120 ms ( $8 \cdot 577 \mu s \cdot 26$ ) zu einem Multiframe zusammengefasst. 51 der 26er Multiframes beziehungsweise 26 der 51er Multiframes werden zu einem Superframe zusammengefasst, während 2048 Superframes zu einem Hyperframe zusammengefasst werden, für dessen Übertragung annährend 3,5 Stunden benötigt werden. Da der A5 Verschlüsselungsalgorithmus (Kapitel 2.3.3 Authentication Center – Verschlüsselung) die TDMA-Rahmennummer als Eingangszahl verwendet, wird der Hyperframe benötigt. Die Eingangszahl kann daher einen Wert von bis zu 2715648 ( $2048 \cdot 51 \cdot 26$ ) erreichen. Die Abbildung 2.17 verdeutlicht die Struktur der Hyperframes, Superframes und der 26er beziehungsweise 51er Multiframes.

### Kombination der logischen Kanäle

Die logischen Kanäle werden nicht beliebig auf die 26er beziehungsweise 51er Multiframes verteilt. Die Kombinationen folgen einem bestimmten Muster, wobei ein physikalischer Kanal genau eine dieser Kombinationen enthält:

- K1: TCH/F + FACCH/F + SACCH/F
- K2: TCH/H + FACCH/H + SACCH/TH
- K3: TCH/H + TCH/H + FACCH/H + SACCH/TH
- K4: FCCH + SCH + BCCH + CCCH
- K5: FCCH + SCH + BCCH + CCCH + SDCCH/4 + SACCH/4
- K6: BCCH + CCCH
- K7: SDCCH/8 + SACCH/8

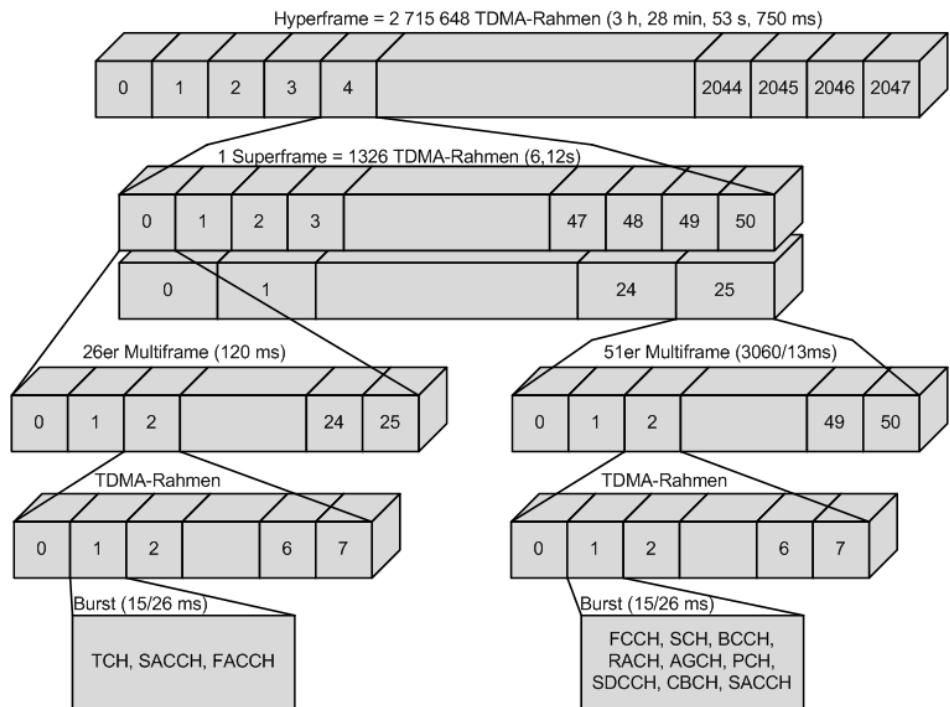


Abbildung 2.17: Strukturierung des Übertragungsmediums durch TDMA-Rahmen, Multiframes, Superframes und Hyperframes

Hierbei bedeutet die Zahl hinter dem logischen Kanal – wie im Fall des SDCCH/4 –, dass es auf einem physikalischen Kanal vier SDCCH Kanäle gibt. Die Kombination der logischen Kanäle bewirkt zwar, dass auf einem physikalischen Kanal mehrere logische Kanäle platziert werden können, dies bedeutet jedoch nicht, dass die Kanäle alle gleichzeitig genutzt werden können. Die Kanäle treten im Abstand von mindestens einer TDMA-Rahmenlänge nacheinander auf. In der Abbildung 2.18 sind auf dem ersten Zeitschlitz die Kanalkombination K4 und auf dem zweiten Zeitschlitz die Kanalkombination K7 für die 51 aufeinanderfolgenden TDMA-Rahmen beziehungsweise die Kanalkombination K1 in den Zeitschlitzten zwei bis sieben für zwei 26er Multiframes dargestellt. Da der FACCH die Informationen bei Bedarf innerhalb eines TCH's überträgt, wird ihm kein eigener Burst zugeteilt, und ist daher in Abbildung 2.18 nicht sichtbar.

## 2.5 ISO/OSI

Die internen Kommunikationsprotokolle zwischen den Komponenten bzw. zwischen der Mobile Station und der Base Station über die  $U_m$  Luftschnittstelle wurden formal so konzipiert, dass sich das GSM-Schichtenmodell am ISO/OSI-Schichtenmodell orientiert. Hierbei gilt zu beachten, dass in Anbetracht der speziellen Eigenschaften des GSM-Netzes weitere Protokollfunktionen zum Einsatz kommen und auf Grund von umfangreichen Aufgaben – wie beispielsweise der Bewertung und Zuteilung der notwendigen Kapazität auf dem Funkweg – nicht unbedingt nur eine einzige ISO/OSI-Schicht betroffen ist. Die Unterschiede zwischen dem ISO/OSI- und GSM-Schichtenmodell beschränken sich hierbei auf die untersten drei Schichten. Die Zuweisung der Aufgaben der verschiedenen Layer wurde größtenteils aus den Quellen [24] und [13] entnommen.

Nummer	Zeitschlitz 0	Zeitschlitz 1	Nummer	Zeitschlitz 0	...	Zeitschlitz 7
0	FCCH	SDCCH / 0	0	TCH		TCH
1	SCH	SDCCH / 0	1	TCH		TCH
2	BCCH	SDCCH / 0	2	TCH		TCH
3	BCCH	SDCCH / 0	3	TCH		TCH
4	BCCH	SDCCH / 1	4	TCH		TCH
5	BCCH	SDCCH / 1	5	TCH		TCH
6	AGCH / PCH	SDCCH / 1	6	TCH		TCH
7	AGCH / PCH	SDCCH / 1	7	TCH		TCH
8	AGCH / PCH	SDCCH / 2	8	TCH		TCH
9	AGCH / PCH	SDCCH / 2	9	TCH		TCH
10	FCCH	SDCCH / 2	10	TCH		TCH
11	SCH	SDCCH / 2	11	TCH		TCH
12	AGCH / PCH	SDCCH / 3	12	SACCH		SACCH
13	AGCH / PCH	SDCCH / 3	13	TCH		TCH
14	AGCH / PCH	SDCCH / 3	14	TCH		TCH
15	AGCH / PCH	SDCCH / 3	15	TCH		TCH
16	AGCH / PCH	SDCCH / 4	16	TCH		TCH
17	AGCH / PCH	SDCCH / 4	17	TCH		TCH
18	AGCH / PCH	SDCCH / 4	18	TCH		TCH
19	AGCH / PCH	SDCCH / 4	19	TCH		TCH
20	FCCH	SDCCH / 5	20	TCH		TCH
21	SCH	SDCCH / 5	21	TCH		TCH
22	SDCCH / 0	SDCCH / 5	22	TCH		TCH
23	SDCCH / 0	SDCCH / 5	23	TCH		TCH
24	SDCCH / 0	SDCCH / 6	24	TCH		TCH
25	SDCCH / 0	SDCCH / 6	25	frei		frei
26	SDCCH / 1	SDCCH / 6	0	TCH	26er Multiframe	TCH
27	SDCCH / 1	SDCCH / 6	1	TCH		TCH
28	SDCCH / 1	SDCCH / 7	2	TCH		TCH
29	SDCCH / 1	SDCCH / 7	3	TCH		TCH
30	FCCH	SDCCH / 7	4	TCH		TCH
31	SCH	SDCCH / 7	5	TCH		TCH
32	SDCCH / 2	SACCH / 0	6	TCH		TCH
33	SDCCH / 2	SACCH / 0	7	TCH		TCH
34	SDCCH / 2	SACCH / 0	8	TCH		TCH
35	SDCCH / 2	SACCH / 0	9	TCH		TCH
36	SDCCH / 3	SACCH / 1	10	TCH		TCH
37	SDCCH / 3	SACCH / 1	11	TCH		TCH
38	SDCCH / 3	SACCH / 1	12	SACCH		SACCH
39	SDCCH / 3	SACCH / 1	13	TCH		TCH
40	FCCH	SACCH / 2	14	TCH		TCH
41	SCH	SACCH / 2	15	TCH		TCH
42	SACCH / 0	SACCH / 2	16	TCH		TCH
43	SACCH / 0	SACCH / 2	17	TCH		TCH
44	SACCH / 0	SACCH / 3	18	TCH		TCH
45	SACCH / 0	SACCH / 3	19	TCH		TCH
46	SACCH / 1	SACCH / 3	20	TCH		TCH
47	SACCH / 1	SACCH / 3	21	TCH		TCH
48	SACCH / 1	frei	22	TCH		TCH
49	SACCH / 1	frei	23	TCH		TCH
50	frei	frei	24	TCH		TCH
			25	frei		frei

51er Multiframe

Abbildung 2.18: Nutzung der physikalischen Kanäle im Downlink (Vorlage nach [22])

### 2.5.1 Layer 1 - Bitübertragung

Der Layer 1 ist für die Übertragung der Verkehrs- und Signalisierungskanäle zuständig. Hierzu werden die Bursts erstellt, in den TDMA-Rahmen gebündelt und über die dedizierten physikalischen Kanäle übertragen. Weitere Aufgaben des Layer 1 bestehen darin, Fehlererkennungs- und Fehlerkorrekturmechanismen bereitzustellen, um fehlerhafte Blöcke nicht an den Layer 2 weiterzuleiten, die Verschlüsselung des Datenstromes zu ermöglichen und die Synchronisation bei der Rahmenübertragung zu gewährleisten. Im Unterschied zum klassischen ISO/OSI-Schichtenmodell greift auch eine Instanz des Layer 3 direkt auf den Layer 1 zur Informationsgewinnung zu. Auf Basis dieser Informationen wie beispielsweise der Empfangsstärke werden Kanäle zugewiesen beziehungsweise ein Handover eingeleitet.

### 2.5.2 Layer 2 - Sicherung

Auf dem Layer 2 werden Protokolle für die Kommunikation zwischen der MS und der BTS, der BTS und dem BSC sowie dem BSC und dem MSC bereitgestellt. Für die Kommunikation zwischen der MS und der BTS wird das *LAPD<sub>m</sub>* Protokoll – basierend auf dem *ISDN LAPD Protokoll*<sup>14</sup> – verwendet, welches die Daten der Signalisierungsprotokolle auf der Luftschnittstelle sichert und Layer 3 bei unbehebbaren Fehlern benachrichtigt. Zu den weiteren Aufgaben gehören, Layer 3 Nachrichten eindeutig einem logischen Kanal zuzuordnen und Konkurrenzsituationen auf dem physikalischen Kanal nach erfolgtem Zufallszugriff auf dem RACH aufzulösen.

### 2.5.3 Layer 3 - Vermittlung

Mit Hilfe der Signalisierungsprotokolle des Layer 3 können Point-to-Point Verbindungen zwischen den Teilnehmern aufgebaut und bei Störungen aufrecht erhalten sowie eine Verbindung ordnungsgemäß abgebaut werden. Zusätzlich ermöglicht Layer 3 die Registrierung und Authentifizierung der Teilnehmer und weist diesen eine TMSI zu.

---

<sup>14</sup>Link Access Procedure for the D-Channel: Dient zur Sicherung der Datenübertragung im Signalisierungs-kanal (D-Kanal)

## 3 Hardware

Das von der Firma Ettus<sup>1</sup> hergestellte Universal Software Radio Peripheral (USRP) ermöglicht, die gesamte Signalverarbeitung mittels Software zu realisieren. Im Gegensatz zu einer Hardware, die speziell für einen bestimmten Einsatzgebiet konstruiert wird, kann mit dem Software-defined Radio, mit Hilfe anpassbarer Hardware, unterschiedliche Funkverfahren durch alleinige Änderung der Software verarbeitet werden. Hierzu werden verschiedene Hardwarekomponenten (siehe Abschnitt 3.3) angeboten, mit deren Hilfe Frequenzen von 1 MHz bis zu 5,9 GHz empfangen und anschließend am Computer verarbeitet werden können.<sup>2</sup> Somit ist eine größtmögliche Flexibilität gewährleistet, da die Implementierung verschiedener Protokolle möglich ist. In Abbildung 3.1 ist ein fertig zusammengebautes USRP1 dargestellt, welches im Folgenden näher betrachtet wird und die Grundlage für den Betrieb einer Basisstation und somit eines IMSI-Catchers bildet. Insgesamt werden zwei Versionen, das USRP1 und das USRP2 (Abbildung 3.1, rechts) von Ettus vertrieben. Da die Signalverarbeitung jedoch mittels Software erfolgt, kann das USRP2 aktuell auf Grund der Softwareumgebung, die für den USRP1 ausgelegt ist, noch nicht für den Betrieb eines IMSI-Catchers genutzt werden. Ist im Folgenden die Rede vom USRP, so ist daher stets das USRP1 gemeint.



Abbildung 3.1: Universal Software Radio Peripheral Version 1 (links) und Version 2 (rechts)

### 3.1 USRP

Das USRP ist ein Universal Software-Radio; dies bedeutet, dass mit verschiedenen Modulen verschiedene Frequenzen softwareseitig verarbeitet werden können. So kann damit beispielsweise Hörfunk empfangen und gesendet werden. Auch ist es möglich, in der WLAN, DECT oder DVB-T Frequenz zu arbeiten. Für den Betrieb einer Basisstation sind allerdings die Frequenzen von GSM900 beziehungsweise GSM1800 – dargestellt in der Tabelle 2.2 Kapitel 2.2 – von Bedeutung. Angesichts des breiten Einsatzgebietes des USRP's ist die Hauptplatine modular aufgebaut und kann mit verschiedenen Modulen ausgerüstet werden. So können beispielsweise zwei Transceiverboards eingebaut werden, mit denen das gleichzeitige Senden

<sup>1</sup> Ettus Research LLC, <http://www.ettus.com> [Online; letzter Aufruf 30.09.2009]

<sup>2</sup> Brochure for the entire USRP product family, [http://www.ettus.com/downloads/er\\_broch\\_trifold\\_v5b.pdf](http://www.ettus.com/downloads/er_broch_trifold_v5b.pdf) [Online; letzter Aufruf 30.09.2009]

und Empfangen möglich ist. Für den Betrieb des IMSI-Catchers wurden zwei solche RFX900 Transceiverboards, die auf dem GSM900-Frequenzband arbeiten, auf die USRP-Hauptplatine aufgesteckt. Der interne Aufbau und die notwendigen Komponenten sind in Abbildung 3.2 dargestellt. Als zentrale Steuerungseinheit wird ein Altera Cyclone EP1C12Q240C8 FPGA (field programmable gate array) verwendet.

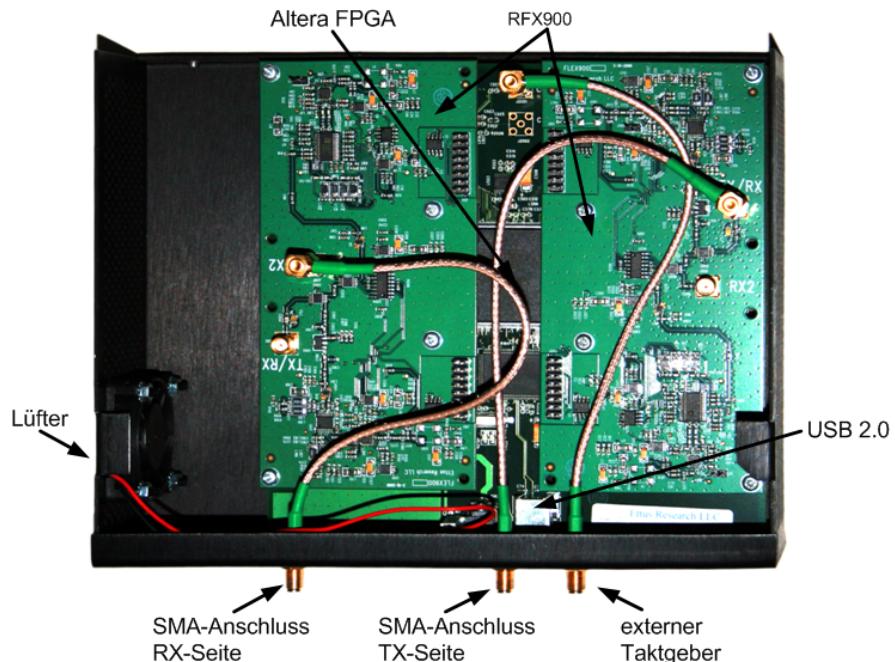


Abbildung 3.2: USRP Hauptplatine, ausgerüstet mit zwei RFX900 Transceiver Modulen

Die Hauptaufgabe des FPGA's besteht darin, die Rohdaten über einen USB2.0 Controller an den Computer weiterzuleiten, bei dem die Signalverarbeitung anschließend softwareseitig geschieht. Daher werden für den IMSI-Catcher verschiedene Softwarekomponenten benötigt, die in Kapitel 4 näher vorgestellt werden. Zur Synchronisation aller Komponenten verfügt die USRP-Hauptplatine über einen internen 64 MHz Taktgeber. Das USRP wurde allerdings nicht ausschließlich für den Betrieb einer Basisstation oder eines IMSI-Catchers entwickelt. Aus Kostengründen wurde ein interner Taktgeber verbaut, der für GSM nicht genau und stabil genug arbeitet. Aus diesem Grund müssen diverse Änderungen sowohl bei der Hardware als auch bei der Software vorgenommen werden, welche im Abschnitt 3.4 und in Kapitel 5.6 erläutert werden. Abschließend die wichtigsten Eckdaten<sup>3</sup> des USRP im Gesamtüberblick:

- Vier 64 Mega-Sample/s 12-Bit analog-digital Konverter
- Vier 128 Mega-Sample/s 14-Bit digital-analog Konverter
- Vier Erweiterungssteckplätze für zwei bis vier Module
- Altera Cyclone EP1C12Q240C8 FPGA
- High-speed USB 2.0 Interface (480 Mbit/s)
- Interner Ecliptek EC2620ETTS64.000M Taktgeber

<sup>3</sup> USRP motherboard datasheet, [http://www.ettus.com/downloads/er\\_ds\\_usrp\\_v5b.pdf](http://www.ettus.com/downloads/er_ds_usrp_v5b.pdf) [Online; letzter Aufruf 30.09.2009]

## 3.2 USRP2

Das USRP2 (Abbildung 3.1) ist die Weiterentwicklung des USRP1 und stellt das neueste Modell der USRP-Produktfamilie dar. Insgesamt ist das USRP2 leistungsfähiger und bietet eine höhere Signalbandbreite. Da dementsprechend eine größere Datenlast anfallen kann, wird zur Kommunikation mit dem Computer ein Anschluss mit einer hohen Bandbreite benötigt. Aus diesem Grund kann das USRP2 nicht mehr über USB mit dem Computer verbunden werden. Zur Kommunikation wird daher zwingend ein Gigabit Ethernet Anschluss benötigt. Neu hinzugekommen ist beim USRP2 ein integrierter SD-Karten Leser, der es ermöglicht, die auf einer SD-Karte gespeicherten Firmware-, Konfigurations- oder sonstigen Daten dem USRP2 zur Verfügung zu stellen, damit dieser als Stand-alone Gerät betrieben werden kann. Im Vergleich mit dem USRP1 ist das USRP2 ebenfalls modular aufgebaut und kann mit denselben Modulen ausgerüstet werden. Im Gegensatz zum USRP1 kann allerdings nur ein Modul verwendet werden. Da die Softwareumgebung jedoch für den Betrieb mit zwei Transceiverboards entwickelt wurde, um das Übersprechen der Signale auf einem Transceiverboard zu verhindern, kann das USRP2 aktuell nicht für den Einsatz als Basisstation genutzt werden. Dessen ungeachtet findet das USRP2 dennoch Verwendung in vielen Einsatzgebieten. So wird es beispielsweise für die Frequenzanalyse in Kapitel 5.4.5 benötigt. Des Weiteren wird das USRP2 als Störsender für den IMSI-Catcher – ebenfalls beschrieben in Kapitel 5.4.5 – verwendet. Die im Vergleich zum USRP1 verwendete Komponenten:<sup>4</sup>

- Zwei 100 Mega-Sample/s 14-Bit analog-digital Konverter
- Zwei 400 Mega-Sample/s 16-Bit digital-analog Konverter
- Erweiterungssteckplatz für ein Modul
- Xilinx Spartan 3-200 FPGA
- Gigabit Ethernet Anschluss
- SD-Karten Leser

## 3.3 Hardwarekomponenten

Als Zubehör stehen für das USRP1 und USRP2 eine Vielzahl an kompatiblen Komponenten zur Verfügung, die über den Online Shop von Ettus<sup>5</sup> bestellt werden können.<sup>6</sup> Die Empfangs- und Sendemodule unterscheiden sich vor allem durch die verwendeten Frequenzen und ihre jeweilige Sendestärke, aufgelistet in Tabelle 3.1. Die vier Steckplätze auf der USRP-Hauptplatine gliedern sich in eine „A“ und eine „B“ Seite, wobei die RFX-Transceiverboards jeweils zwei Steckplätze benötigen. Das OpenBTS Softwareprojekt – als Basis für den IMSI-Catcher – benötigt entweder zwei RFX900 oder zwei RFX1800 Transceiverboards. Da OpenBTS auf dem Transceiverboard der Seite „A“ sendet, muss die Antenne an den „TX/RX“ Connector und dementsprechend die Antenne an den „RX2“ Connector auf dem Transceiver der Seite „B“ angeschlossen werden.<sup>7</sup> Der Breitband-Receiver DBSRX eignet sich vor allem für die Spektrumsanalyse mit dem USRP2, da sowohl das GSM900 als auch das GSM1800 analysiert werden kann. Alle verwendeten Hardwarekomponenten, die direkt oder indirekt für den Betrieb des IMSI-Catchers notwendig sind, werden in Abbildung 3.3 dargestellt.

<sup>4</sup> USRP2 motherboard datasheet, [http://www.ettus.com/downloads/ettus\\_ds\\_usrp2\\_v2.pdf](http://www.ettus.com/downloads/ettus_ds_usrp2_v2.pdf) [Online; letzter Aufruf 30.09.2009]

<sup>5</sup> Ettus Onlineshop, <http://www.ettus.com/order> [Online; letzter Aufruf 30.09.2009]

<sup>6</sup> So ist es möglich verschiedene Empfangs- und Sendemodule, Kabel, Antennen, Netzteile und Montagematerialien zu erwerben.

<sup>7</sup> Desktop Testing of OpenBTS, <http://sourceforge.net/apps/trac/openbts/wiki/OpenBTS/DesktopTestingKit> [Online; letzter Aufruf 30.09.2009]

Bezeichnung	Beschreibung	Frequenz	Sendestärke
RFX900	Transceiver	800 bis 1000 MHz	200 mW
RFX1800	Transceiver	1,5 bis 2,1 GHz	100 mW
DBSRX	Receiver	800 MHz bis 2,4 GHz	
VERT900	Rundantenne	824 bis 960 MHz	
LP0926	Breitbandantenne	1710 bis 1990 MHz 900 MHz bis 2,6 GHz	

Tabelle 3.1: Übersicht der Verwendeten Hardwarekomponenten und deren Sendestärke

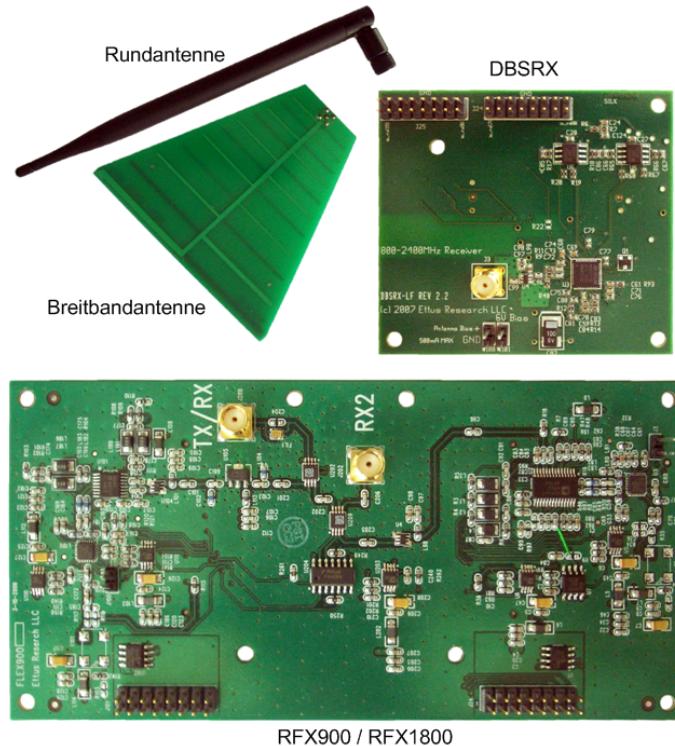


Abbildung 3.3: Rund- und Breitbandantenne sowie DBSRX Receiver und RFX900 / RFX1800 Transceiver

## 3.4 Hardwaremodifikationen

Für den Betrieb des IMSI-Catchers mit dem USRP waren verschiedene Modifikationen an der Hardware notwendig. Zunächst musste ein 900 MHz Filter entfernt werden, damit auf dem GSM900 Frequenzbereich gesendet werden kann (Abschnitt 3.4.1). Anschließend musste auf Grund von Taktungenaugkeiten ein externer Taktgeber zusammengebaut (Abschnitt 3.4.2) sowie der interne Taktgeber deaktiviert werden (Abschnitt 3.4.3).

### 3.4.1 RFX900 ISM-Band Filter

Das RFX900 Board wird ab Werk mit einem aktivierten 902 - 928 MHz ISM-Band Filter ausgeliefert. ISM-Bänder sind Frequenzbänder für lizenzzfreie Übertragungen, bei denen lediglich die Sendeleistung und die Störung für benachbarte Frequenzbereiche geregelt sind. Das RFX900 Board besitzt diesen Filter, da in den USA innerhalb dieses Bereiches ein ISM-Band verfügbar ist auf dem frei gesendet werden darf. In Europa befindet sich auf dieser Frequenz

Teile des GSM-Bandes. Ein entsprechendes ISM-Band ist in Europa bei 2,4 GHz (2,4000 GHz - 2,4835 GHz) vorhanden, das unter anderem von Mikrowellenherden, Bewegungsmeldern, WLANs, Bluetooth, WirelessUSB und Richtfunk genutzt werden.<sup>8</sup> Damit das RFX900 Board im vollen Frequenzbereich von 800 - 1000 MHz eingesetzt werden kann, muss der Filter deaktiviert werden. Hierzu wird die Leiterbahn zum Filter „FIL.1“ unterbrochen und ein 100 pF Kondensator parallel zum Filter (an Position C204) angelötet (Abbildung 3.4).<sup>9</sup> Es gilt jedoch zu beachten, dass bei fehlerhafter Modifikation der Verstärker zerstört wird. Alternativ kann

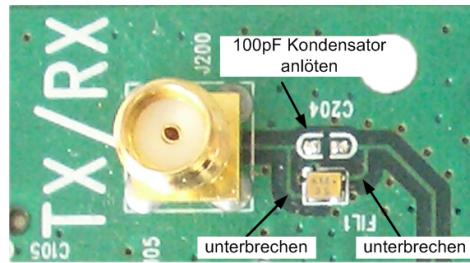


Abbildung 3.4: ISM-Band Filter entfernen

bei fehlendem Werkzeug oder fehlender Lötkenntnis ein RFX1800 in ein RFX900 Board umprogrammiert werden. Hierzu muss das RFX1800 Board in das USRP eingebaut und in den `gnuradio-3.2.2/usrp/host/apps/` Ordner gewechselt werden. Anschließend wird mit folgendem Kommando das EEPROM des RFX1800 umprogrammiert:

```
./burn-db-eeprom -A --force -t rfx900_mimo_b
```

Auf gleiche Weise kann das zu einem RFX900 umprogrammierte RFX1800 wieder in ein RFX1800 Board zurückversetzt werden. Jedoch gilt zu beachten, dass dies nur mit einem ehemals umprogrammierten RFX1800 Board möglich ist. Ein unmodifiziertes RFX900 Board kann logischerweise auf Grund des ISM-Band Filters nicht zu einem RFX1800 Board transformiert werden.<sup>10</sup>

```
./burn-db-eeprom -A --force -t rfx1800_mimo_b
```

Der Tabelle 3.2 kann entnommen werden, welche Frequenzen mit und ohne Modifikationen genutzt werden können.[14]

### 3.4.2 Taktgeber

Das USRP wird intern mit einem 64 MHz Taktgeber der Firma Ecliptek Modell „EC2620ETTS-64.000M“ synchronisiert, der eine Abweichung von  $\pm 20$  ppm (parts per million) aufweist. Somit ergibt sich bei 64 MHz eine maximale Abweichung von 1,28 kHz. Eine spezifikationskonforme BTS-Zelle wird in der Regel mit einem „Stratum-3 OCXO“ (eine per Thermostat beheizte „Oven Controlled Crystal Oscillator“) betrieben, der eine Abweichung von  $\pm 20$  ppb (parts per billion) aufweist. Die erlaubte Abweichung einer BTS ist laut Spezifikation GSM 05.10 [9]  $\pm 0,05$  ppm (= 3,2 Hz) beziehungsweise bei einer PicoBTS Zelle  $\pm 0,1$  ppm (= 6,4 Hz). Da der interne Taktgeber des USPR's somit nicht mehr mit der Spezifikation konform ist

<sup>8</sup> ISM-Frequenzen, <http://www.bundesnetzagentur.de/enid/a2.html> [Online; letzter Aufruf 30.09.2009]  
ISM-Band, <http://de.wikipedia.org/wiki/ISM-Band> [Online; letzter Aufruf 30.09.2009]

<sup>9</sup> USRP Daugherboard: RFX900, <http://gnuradio.org/trac/wiki/UsrpDBoardRFX900> [Online; letzter Aufruf 30.09.2009]

<sup>10</sup> Daughterboards Questions, <http://gnuradio.org/trac/wiki/UsrpFAQ/DBoards> [Online; letzter Aufruf 30.09.2009]

RFX900 mit Filter	
TX/RX	RX2
902 bis 928 MHz	800 bis 1000 MHz

RFX900 ohne Filter	
TX/RX	RX2
800 bis 1000 MHz	

RFX900 als RFX1800	
TX/RX	RX2
1.5 bis 2.1 GHz	

Tabelle 3.2: Nutzbare Frequenzen des RFX900 Transceiverboards mit und ohne Filter, sowie als RFX1800 Transceiverboard umprogrammiert

und sich schnell herausgestellt hat, dass die Frequenzfehler zu Erkennungsproblemen der Basisstation durch die getesteten Mobilfunktelefonen führen, war es notwendig einen externen Taktgeber zu verwenden. Hierfür eignet sich der Bausatz FA-SY1, der von Funkamateuren genutzt wird und für ca. 50 Euro über den Onlineshop der Fachzeitschrift „Funkamateur“ bezogen werden kann (Abbildung 3.5).<sup>11</sup> Der Zusammenbau des Bausatzes kann im Anhang A eingesehen werden. Da der Taktgeber extern an das USRP angeschlossen wird, wurde der Bausatz in ein Aluminium-Gehäuse eingebaut (Abbildung 3.5). Das Aluminium Gehäuse gewährleistet gleichzeitig eine Abschirmung gegenüber äußerer Einflüssen und besitzt einen SMA-Anschluss, über den der hochfrequente Takt ausgegeben wird. Der Taktgeber lässt sich über USB auf eine Frequenz zwischen 10 bis 160 MHz einstellen und weist eine Abweichung von  $\pm 20$  ppm auf. Für eine temperaturunabhängige Frequenzstabilität besitzt der Taktgeber einen Heiztransistor, der über einen Temperaturfühler geregelt wird.

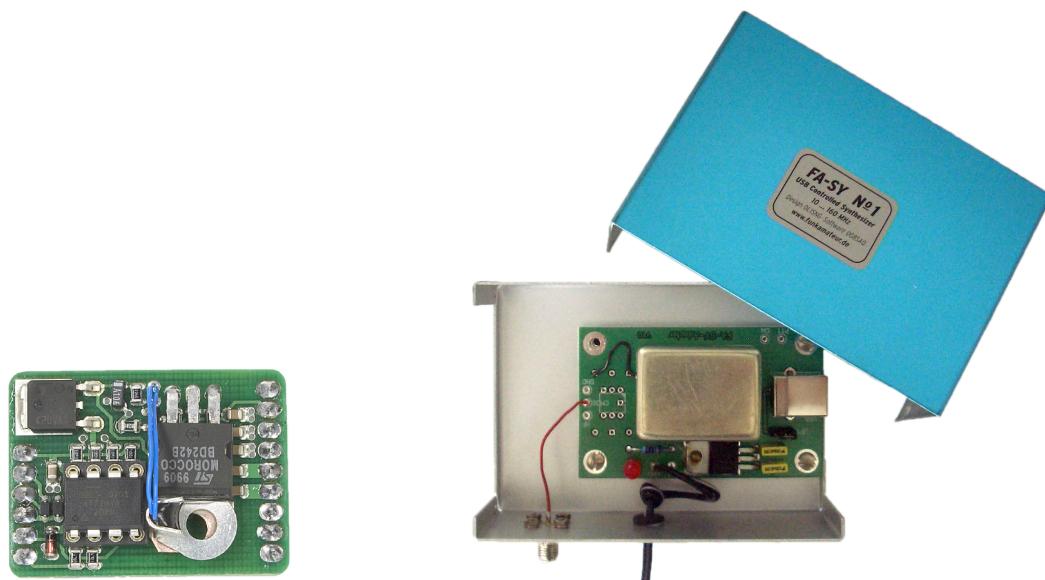


Abbildung 3.5: Externer Taktgeber FA-SY1: links der Taktgeber, rechts zusammengebaut und mit Heizkörper versehen

<sup>11</sup> Box 73 Amateurfunkservice GmbH, <http://www.box73.de/catalog/> [Online; letzter Aufruf 30.09.2009]

## Kalibrierung

Die initiale Ungenauigkeit kann mit einer Kalibrierung minimiert werden. Hierzu wird zum einen das unter Windows lauffähige Programm *USB\_Synth.exe* für die Kalibrierung und zum anderen ein Frequenzmessgerät, wie beispielsweise ein „Hewlett Packard 53131A 225 MHz Universal Counter“, benötigt. Nach einer Aufwärmzeit von mindestens zehn Minuten kann ausgehend von der 15 MHz vorgegebenen Herstellerfrequenz die Kalibrierung gestartet werden. Die Herstellerfrequenz muss hierzu in das Kalibrierungsprogramm als Startfrequenz eingetragen werden (siehe Abbildung 3.6). Anschließend muss die tatsächliche Frequenz mit dem Frequenzmessgerät gemessen und als „Factory Startup Frequency“ eingegeben werden. Anstelle von 15,0000000 MHz wurde ein Wert von 14,9999217 MHz gemessen, was einer Ungenauigkeit von 78,3 Hz entspricht. Nach erfolgreicher Kalibrierung kann der Taktgeber auf 64 MHz eingestellt werden.

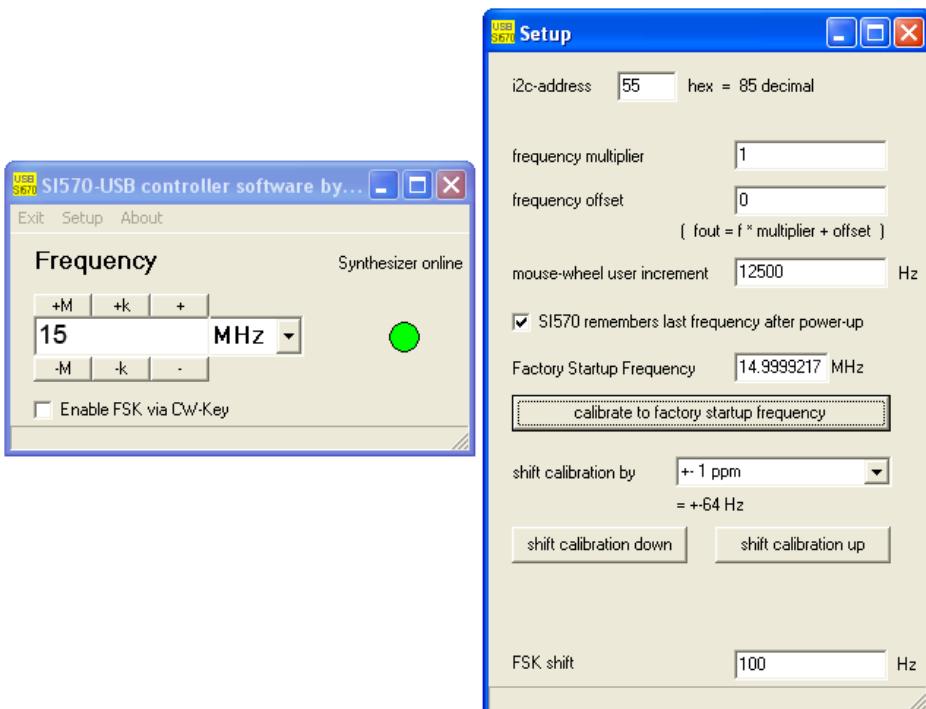


Abbildung 3.6: Kalibrierung des FA-SY1 Taktgebers

### 3.4.3 USRP

Damit der externe Taktgeber mit dem USRP betrieben werden kann, muss der interne Taktgeber deaktiviert werden. Hierfür muss an der Hauptplatine die Brücke (0 Ohm) von der Position R2029 an die Position R2030 umgelötet werden. Anschließend muss der Transistor an der Position C925 zur Position C926 umgelötet und der Transistor an Position C924 entfernt werden.<sup>12</sup>

<sup>12</sup> USRP Clocking Notes, <http://gnuradio.org/trac/wiki/USRPClockingNotes> [Online; letzter Aufruf 30.09.2009]

## 3.5 Testumgebung

### 3.5.1 Computer

Da die gesamte Signalverarbeitung in Software realisiert wurde und die Rechenlast mit steigender Anzahl von Teilnehmern, die verwaltet werden müssen und Gespräche führen, anwächst, werden entsprechende Anforderungen an den Rechner gestellt. Dies ist vor allem in Hin-

	Testrechner 1	Testrechner 2
Modell	HP Compaq 6910p	Samsung NC10
CPU	Intel Core2Duo T7300 @ 2 GHz	Intel Atom @ 1,6 GHz
RAM	2 GB	2 GB
Ethernet	1 Gigabit	100 Mbit
Anschlüsse	USB 2.0	USB 2.0

Tabelle 3.3: Verwendete Hardware

blick darauf, dass der IMSI-Catcher zeitweise gleichzeitig den USRP1 und auch den USRP2 steuert, verständlich. Der Betrieb mit dem USRP2 benötigt überdies eine Gigabit Netzwerk-schnittstelle. Als Entwicklungsumgebung wurde ein HP Laptop (Tabelle 3.3) der Universität Freiburg verwendet. Mit diesem Modell ist der gleichzeitige Betrieb des IMSI-Catchers mit dem USRP1 und dem USRP2 ohne Weiteres möglich. Als Minimalkonfiguration ist für den IMSI-Catcher auch die Verwendung eines Netbook's möglich, dann aber nur mit dem USRP1. In Abbildung 3.7 ist die Testumgebung, bestehend aus dem USRP1 und dem Samsung NC10 dargestellt. Der IMSI-Catcher konnte hierbei mit vier aktiven Gesprächen betrieben werden (unterer Abschnitt).

### 3.5.2 Mobilfunktelefone

Folgende Mobilfunktelefone (Abbildung 3.8) wurden während der Entwicklung des IMSI-Catchers verwendet. Funktioniert haben alle, wobei sich herausgestellt hat, dass bei verschiedenen OpenBTS Versionen die Nokia Mobilfunktelefone immer am stabilsten gearbeitet haben. Bermerkbar hat sich das vor allem bei der Verbindungsstabilität und der Netzsuche bzw. dem Regitrierungsvorgang gemacht. Falls mit OpenBTS ein eigenes Netz mit einem eigenem Shortname<sup>13</sup> wie beispielsweise „OpenBTS“ verwendet wird, kann dies lediglich vom Nokia N80 korrekt angezeigt werden. Alle anderen Modelle zeigen sowohl bei der Netzsuche als auch später im Display eine Kombination aus dem verwendeten Länder- und Netzanbietercode an. Falls beispielsweise OpenBTS den Ländercode 262 (für Deutschland) und ein Netzanbietercode 09 verwendet, wird als Netzname „262 09“ oder „CC 262 NC 09“ bzw. „NOR 09“ angezeigt.<sup>14</sup> Die Darstellung ist von Hersteller zu Hersteller verschieden. Da der IMSI-Catcher allerdings reale Funknetze simuliert, stellt dies kein Problem dar. Dieser Umstand ist darauf zurück-zuführen, dass die SIM-Karte die Kennzeichnungen der originalen Netze gespeichert hat und diese korrekt anzeigt. Als SIM-Karten wurden Prepaid-Karten aller deutschen Netzbetreiber, überwiegend allerdings Vodafone Karten, verwendet. Da ein breites Spektrum an Mobilfunktelefonen und SIM-Karten aller Netzanbieter im Einsatz waren, deutet alles darauf hin, dass die überwiegende Anzahl der sich im Umlauf befindenden Geräte für den IMSI-Catcher an-fällig sind. Die Präsenz eines IMSI-Catchers wird den Teilnehmern allerdings nicht von ihrem Mobilfunktelefon signalisiert. Normalerweise ist die Verbindung während eines Gesprächs im

<sup>13</sup> Die Bezeichnung der GSM-Zelle

<sup>14</sup> Eine Erklärung der einzustellenden Parameter erfolgt in den Kapiteln 4.2.3 und 5.7.3.

### 3 Hardware

---

GSM-Netz verschlüsselt (siehe Kapitel 5.5). Beim IMSI-Catcher ist die Verbindung nicht verschlüsselt. Dies wird jedoch lediglich vom Siemens S55 während des Gesprächs im Display in Form eines Ausrufezeichens angezeigt. Alle anderen Modelle geben keinen Hinweis darauf.



Abbildung 3.7: IMSI-Catcher in Betrieb mit dem USRP1 und dem Samsung NC10 Netbook



Abbildung 3.8: Verwendete Mobilfunktelefone

## 4 Software

Die Abbildung 2.4 auf Seite 7 verdeutlicht die Vielzahl an Hardwarekomponenten, die für den Betrieb eines GSM-Netzes notwendig sind. Da der IMSI-Catcher eine reale Funkzelle vortäuscht und die GSM-Infrastruktur nachbildet, werden entsprechende Hard- sowie Softwarekomponenten benötigt. Insgesamt sind drei frei verfügbare Softwarekomponenten erforderlich, die im Weiteren näher erläutert werden:

- GNU Radio
- OpenBTS
- Asterisk

Die Abbildung 4.1 gibt einen Überblick, wie die einzelnen Hard- und Softwarekomponenten zusammenwirken. Als Hardwarebasis dient das Universal Software Radio Peripheral, welches in Kapitel 3 näher beschrieben wurde und in der Analogie der GSM-Infrastruktur zusammen mit den Softwarekomponenten die BTS darstellt. Das USRP ist die einzige Hardwarekomponente, alle anderen Komponenten werden softwareseitig implementiert. Hierbei gilt zu beachten, dass die Netzinfrastruktur nicht exakt in Software umgesetzt wurde. Ausgestattet ist das USRP mit zwei RFX900 Transceiverboards, die es ermöglichen, auf dem GSM900-Frequenzband zu senden. Die Signalverarbeitung findet jedoch nicht auf dem USRP selbst sondern softwareseitig statt. Hierfür wird das GNU Radio benötigt, welches die vom USRP generierten Daten verarbeitet. Die GNU Radio Software stellt lediglich eine allgemeine Schnittstelle zum USRP dar. Erst mit Hilfe der OpenBTS Software ist es möglich, eine GSM-Basisstation zu betreiben. OpenBTS verfügt hierzu über eine Art integrierte VLR, in der die TMSI's verwaltet werden. Als Grundvoraussetzung benötigt OpenBTS einen Asterisk-Server, mit dem es möglich ist, die Teilnehmer zu identifizieren und authentifizieren (HLR). Des Weiteren ermöglicht der Asterisk-Server den Teilnehmern, Telefongespräche innerhalb von OpenBTS bzw. in das allgemeine Telefonnetz zu führen. Die verwendeten Softwarekomponenten können der Tabelle 4.1 entnommen werden.

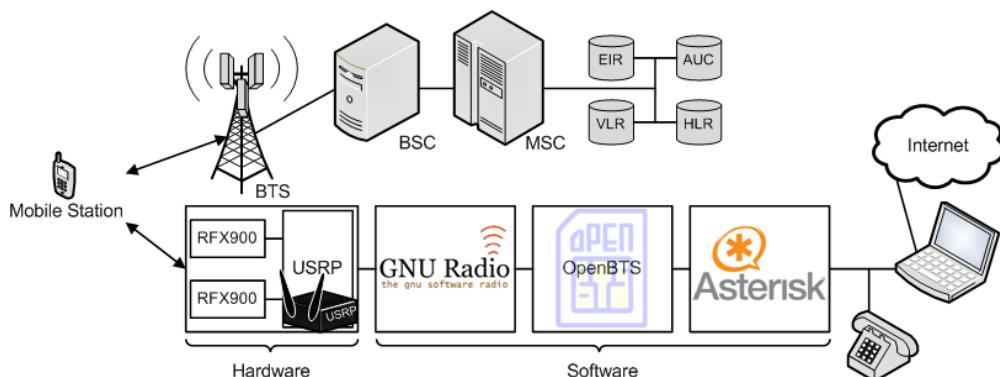


Abbildung 4.1: Systemüberblick der benötigten Hard- und Softwarekomponenten (Vorlage nach [15])

Für die Installation der Softwarekomponenten können diverse Betriebssysteme gewählt werden. Zum einen ist es möglich, verschiedene Linux-Distributionen wie beispielsweise Ubuntu, Fedora oder Debian zu wählen, zum anderen ist es ebenfalls möglich, Windows, MacOS oder NetBSD zu nutzen. Allerdings wurde OpenBTS für MacOS und Linux entwickelt.<sup>1</sup> Aus diesem Grund ist die Verwendung von Windows und Cygwin nicht zu empfehlen. Für das Testsystem wurden zunächst die zu diesem Zeitpunkt aktuell verfügbaren Linux-Distributionen Ubuntu 9.04 und Fedora 10 verwendet. Da die Installation und Verwendung von Fedora 10 für Linux-Anfänger auf Grund von diversen Problemen wie beispielsweise fehlender Zugriffsberechtigungen nicht trivial ist, fiel die Entscheidung auf Ubuntu 9.04. Die Installation unter Ubuntu 9.04 ist ohne tiefergreifende Kenntnisse möglich.

Software	Version
Ubuntu	9.04 - 32-bit mit Kernelversion 2.6.28-13
GNU Radio	3.2.2
OpenBTS	2.4.1
Asterisk	1.6

Tabelle 4.1: Verwendete Software

## 4.1 GNU Radio

Entwickelt wurde die GNU Radio Software [20] ursprünglich als „Software Defined Radio“ mit dem Ziel, die Signalverarbeitung eines Hochfrequenz-Senders oder -Empfängers mit Hilfe anpassbarer Hardware in Software abzubilden. Das GNU Radio stellt somit im eigentlichen Sinne lediglich die Laufzeitumgebung für die Ansteuerung des USRP's und die anschließende Signalverarbeitung dar. Für die Ansteuerung der USRP-Hardware können eigene sog. Module mit den Programmiersprachen Python sowie C++ geschrieben werden. Von GNU Radio werden zur Unterstützung einige Beispielprogramme sowie nützliche Tools bereitgestellt, die überwiegend in der Programmiersprache Python geschrieben sind. Die eigentliche Signalverarbeitung wurde von den Entwicklern auf Grund der notwendigen Effizienz und Geschwindigkeit in C++ geschrieben. Ein wichtiges Beispielprogramm stellt das *usrp\_fft.py* dar, mit dessen Hilfe eine Spektrumsanalyse auf einer einstellbaren Frequenz durchgeführt werden kann und für spätere Zwecke in Kapitel 5.4.5 benötigt wird. Eines der wichtigsten Tools ist überdies, *GNU Radio Companion*<sup>2</sup> (GRC), das mit dem Kommando *grc* gestartet wird. Mit GRC können ohne Programmierkenntnisse solche Module erstellt werden. Hierzu stellt GRC verschiedene Komponenten wie beispielsweise Signalquellen bereit, bei denen lediglich die Frequenz, die Amplitude und weitere Einstellungen vorgenommen werden müssen und die anschließend mit anderen Komponenten verbunden werden (Abbildung 4.2). Das „zusammengeklickte“ Modul kann anschließend als Pythonscript exportiert werden.

In Anhang B.1 wird eine ausführliche GNU Radio 3.2.2 Installation für Ubuntu 9.04 beschrieben. Nach erfolgreicher Installation befinden sich die erwähnten Beispielprogramme in */usr/local/share/gnuradio/examples/*.

<sup>1</sup> Desktop Testing of OpenBTS, <http://sourceforge.net/apps/trac/openbts/wiki/OpenBTS/DesktopTestingKit> [Online; letzter Aufruf 26.10.2009]

<sup>2</sup> GNU Radio Companion, <http://www.gnuradio.org/trac/wiki/GNURadioCompanion> [Online; letzter Aufruf 30.09.2009]

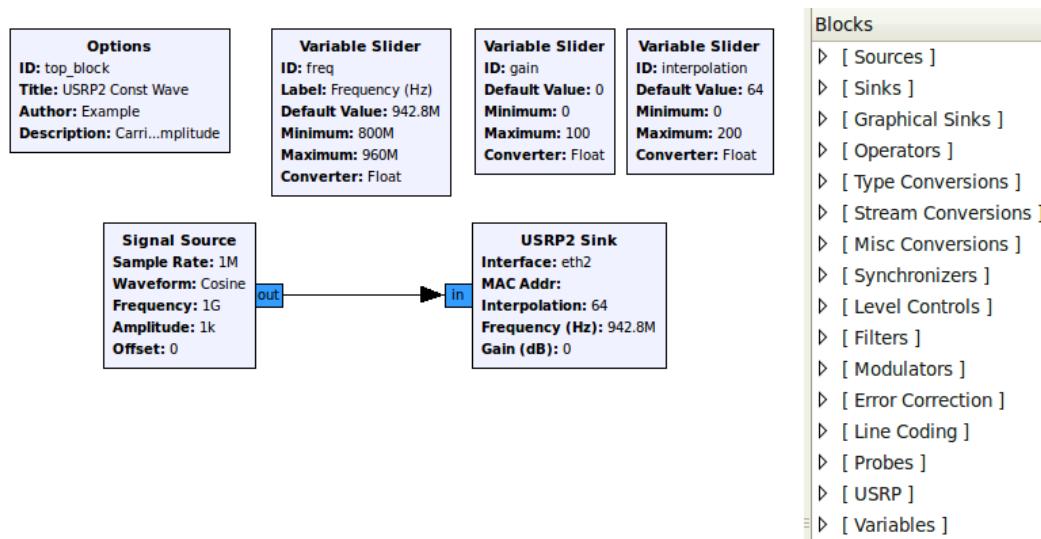


Abbildung 4.2: GNU Radio Companion

## 4.2 OpenBTS

Das OpenBTS Projekt wurde gegründet, um den gesamten GSM-Stack in Software zu implementieren und frei anzubieten, damit in Gebieten mit geringer Netzardeckung beziehungsweise in Entwicklungsländern kostengünstig GSM-Netze aufgebaut werden können. Im Gegensatz zu einem konventionellen GSM-Netz kann OpenBTS eher als Acces Point angesehen werden, welches das gesamte NSS und den BSC ersetzt. Nicht notwendige Netzelemente wie z.B. die verschiedenen Register, die TRAU oder das GMSC wurden hingegen nicht adäquat implementiert bzw. werden vom Asterisk-Server bereitgestellt. So bietet Asterisk eine Benutzerverwaltung und leitet die Anrufe weiter. Die Kommunikation mit den Mobile Stations erfolgt über die  $U_m$  Luftschnittstelle des USRP's.

### 4.2.1 Installation

Die Installationsanleitung für OpenBTS kann ebenfalls dem Anhang B.2 entnommen werden. Anschließend kann OpenBTS mit Hilfe der *OpenBTS.conf* im Ordner *osic/apps* konfiguriert und mit dem Kommando *sudo ./OpenBTS* gestartet werden. Die Konfigurationsdetails werden in Abschnitt 4.2.3 näher erläutert.

Die Hauptordner und Funktionen von OpenBTS gliedern sich wie folgt:

- *Apps*: In diesem Ordner befindet sich das eigentliche OpenBTS Programm und die Konfigurationsdatei *OpenBTS.conf*, mit der verschiedene Netz- und Kommunikationsparameter eingestellt werden.
- *CLI*: Die CLI Schnittstelle stellt eine Vielzahl an Funktionen, wie beispielsweise das Anzeigen der eingebuchten IMSI's, bereit und ermöglicht das Steuern von OpenBTS.
- *CommonLibs*: Dieser Ordner enthält allgemeine Klassen, die nicht GSM spezifisch sind. Hierzu gehören unter anderem C++ Wrapper Klassen für pthreads<sup>3</sup> und Unix sockets.

<sup>3</sup>Pthreads ist ein POSIX Standard für Threads in Systemen wie Linux, Mac OSX und Solaris und ist für die Programmiersprache C definiert.

- *Control*: Die meisten Kontrollmechanismen für den GSM-Layer3 und VoIP-Nachrichten werden hier von verschiedenen Klassen bereitgestellt.
- *GSM*: Der eigentliche GSM-Stack wurde hier, verteilt auf verschiedene GSM spezifische Klassen für die Layer L1 bis L3, implementiert.
- *GUI*: Dieser Ordner wurde für den Open Source IMSI-Catcher erstellt und beinhaltet eine grafische Oberfläche für OpenBTS. OpenBTS wird daher nicht mehr über den Apps Ordner gestartet, sondern mit dem Kommando `sudo ./GUI`.
- *SIP*: Die SIP spezifische Komponenten, die von der Kontrollsicht benötigt werden, ermöglichen SIP-Nachrichten zu erstellen und an den Asterisk-Server weiterzuleiten bzw. die empfangenen SIP-Nachrichten zu interpretieren.
- *SMS*: Der noch nicht vollständig entwickelte SMS-Stack
- *Transceiver*: Die Transceiver Software, die mittels UDP Interface über bestimmte Ports mit den OpenBTS Anwendungen kommuniziert und in der *OpenBTS.conf* eingestellt werden kann:
  - Port 5060 für den Asterisk-Server
  - Port 5062 für das OpenBTS SIP-Interface
  - Portbereich 5700 für das OpenBTS Transceiver Interface
- *TRXManager*: Ein Kontrollinterface für den Transceiver zwischen dem GSM-Stack und dem GNU Radio.

#### 4.2.2 Patch

Da OpenBTS 2.4.1 auf GNU Radio 3.1 basiert, mussten Änderungen innerhalb der *USRDevice.h* und *USRDevice.cpp* Dateien im Ordner *Transceiver* und *Transceiver52M* vorgenommen werden, damit GNU Radio 3.2.2 verwendet werden kann. Dies war notwendig, da erst ab der neusten Version das GRC-Tool enthalten ist, welches im späteren Verlauf benötigt wird. Die vorgenommenen Änderungen wurden in einem Patch zusammengefasst und können dem Anhang B.3 bzw. der CD im Ordner *Patch/* entnommen werden. Sofern die modifizierte OpenBTS Version für den IMSI-Catcher genutzt wird, müssen die Patches nicht installiert werden, da diese bereit integriert sind.

#### 4.2.3 Konfiguration

Innerhalb der *OpenBTS.conf* können wichtige Einstellungen wie etwa die verwendeten Ports für die Kommunikation, GSM-Netzeinstellungen, SMS-Nachrichten beim Einbuchen in das GSM-Netz sowie die Ausgabe von Logdateien und die Tiefe des Loglevels vorgenommen werden. Die wichtigsten GSM-Netzeinstellungen im Überblick:

```
GSM.Band 900          # GSM-Frequenzband
GSM.ARFCN 29          # Frequenzkanal
GSM.PowerAttenDB 10    # Sendestärke in dBm
GSM.NCC 0              # Network Color Code
GSM.BCC 0              # Basestation Color Code
GSM.MCC 262            # Mobile Country Code
GSM.MNC 09             # Mobile Network Code
GSM.LAC 793            # Location Area Code
```

---

```
GSM.CI 1022          # Cell ID
GSM.ShortName OpenBTS # Name der Basisstation
```

Erlaubte Werte für GSM.Band sind 850, 900, 1800 und 1900, welches dem Frequenzband von GSM850, GSM900, GSM1800 oder GSM1900 entspricht. Anschließend kann mit dem GSM.ARFCN der Kanal und somit die entsprechende Frequenz, auf der gesendet werden soll, bestimmt werden. Eine Tabelle der ARFCN Nummern und der jeweiligen Frequenz befindet sich im Anhang D für das GSM900-Netz beziehungsweise im Anhang E für GSM1800. Die Sendestärke kann mit der Variable *GSM.PowerAttenDB* in dBm<sup>4</sup> eingestellt werden. Hierbei ist der Maximalwert 23 dBm, welcher 200 mW entspricht. Der Mobile Country Code (GSM.MCC), Mobile Network Code (GSM.MNC), Location Area Code (GSM.LAC) und die Cell ID (GSM.CI) bilden die Information für eine aktuelle Zelle. Für die Simulation eines deutschen GSM-Netzes muss der Ländercode 262 verwendet werden. Komplette Konfigurationsdateien, die benötigt werden, um ein konventionelles GSM-Netz eines Providers vorzutäuschen, können auf der CD im Ordner *osic/GUI/asterisk/* eingesehen werden. Überdies gibt die Tabelle 5.1 in Kapitel 5.3.1 einen Einblick über die benötigten Werte.

### 4.3 Asterisk PBX

Asterisk<sup>5</sup> ist eine Open Source Software-Telefonanlage. Sie unterstützt Voice-over-IP (VoIP) mit unterschiedlichen Protokollen und ermöglicht Gespräche zwischen den Teilnehmern sowie Gespräche über VoIP in das Fest- und Mobilfunknetz. Die Installation erfolgt über das Ubuntu Repository:

```
sudo apt-get install asterisk
```

Die Konfigurationsdateien liegen nach erfolgreicher Installation in */etc/asterisk/*. Wichtig hierbei sind die Konfigurationsdateien *sip.conf* und *extensions.conf*. In der *sip.conf* werden allgemeine Einstellungen wie der Port, über den der Asterisk-Server an das Internet angebunden ist, eingestellt und Benutzerkonten verwaltet. Die Benutzer können einem Kontext – eine Zuweisung in Berechtigungsklassen – zugeteilt werden. Der Kontext wird in der *extensions.conf* näher spezifiziert und beschreibt, welche Aktionen ein Benutzer tätigen darf. Hierzu gehören die Wahlregeln, die bestimmen, welche Telefonnummern oder Telefonnummernklassen angerufen werden dürfen. Außerdem können noch weitere Features, wie z.B. ein Anrufbeantworter dem Benutzer bereitgestellt werden. Die gesamte Kommunikation im GSM-Netz wie Anrufe innerhalb des OpenBTS Netzes oder auch Anrufe, die in das Fest- oder Mobilfunknetz weitergeleitet werden, geschieht über den Asterisk-Server. Aus diesem Grund muss jede MS ein eigenes Benutzerkonto besitzen, über das die MS sich registriert und der Benutzer somit authentifiziert werden kann (siehe Kapitel 5.4). Somit fungiert der Asterisk-Server als eine Kombination aus HLR, AUC und TRAU. Für die Authentifizierung wird die IMSI der MS als Benutzernamen genutzt. Denkbar wäre aber auch die IMEI. Folgendes Beispiel einer *sip.conf* stellt die Minimalkonfiguration für ein Benutzerkonto mit der IMSI „262026789012345“ dar:

```
[262026789012345]
context=default
canreinvite=no
type=friend
host=dynamic
allow=all
```

---

<sup>4</sup> Leistungspegel mit der Bezugsgröße 1 mW.

<sup>5</sup> Asterisk, <http://www.asterisk.org> [Online; letzter Aufruf 30.09.2009]

Hierbei wird der Benutzer in den Kontext „default“ eingeordnet, das folgende Wahlregeln beinhaltet und in der *extensions.conf* gespeichert ist:

```
[default]
exten => 2000,1,Dial(SIP/262026789012345)
exten => _0.,1,Dial(SIP/10${EXTEN}@comsys02)
exten => 12345,1,SayDigits(${CALLERID(num)})
```

Die erste Wahlregel ordnet dem Benutzer die Telefonnummer „2000“ zu, unter der er erreichbar ist. Die zweite Wahlregel erlaubt es dem Benutzer, externe Anrufe zu tatigen. Hierbei bedeutet „\_0.“, dass die Rufnummer mit einer 0 anfangen muss. Die gewahlte Rufnummer wird dem Benutzer comsys02 zugeordnet, der ein Gateway zum Telefonnetz besitzt. Die letzte Wahlregel ermoglicht es dem Benutzer, sich seine IMSI-Nummer vorlesen zu lassen, sobald dieser die Nummer „12345“ wahlt.

## 5 IMSI-Catcher

Die primäre Aufgabe eines IMSI-Catchers ist es, alle IMSI-Nummern der Mobile Stations im Empfangsbereich zu ermitteln. Die IMSI ist eine weltweit einmalige Kennzahl und wird auf der SIM-Karte der Mobile Station gespeichert. Der erste IMSI-Catcher mit dem Namen GA 090 (Abbildung 5.1) wurde von der deutschen Firma Rohde & Schwarz<sup>1</sup> 1996 in München vorgestellt.<sup>2</sup> Er wurde ursprünglich als Test- und Messsystem konstruiert und später zur Bestimmung der Endgerätekennung von Mobile Stations weiterentwickelt. Der IMSI-Catcher GA 090 hat die Größe eines Personal Computers (PC) und besteht aus Steuerungsrechner, Antenne und Messgeräten. Mittels des GA 090 kann die IMSI und die IMEI eines Mobilfunkteilnehmers ermittelt werden. Überdies ist die Lokalisierung der Mobile Station anhand einer bekannten IMSI möglich. Der weiterentwickelte GA 900 IMSI-Catcher ermöglicht sogar die Aufnahme eines ausgehenden Gesprächs. Das Bundeskriminalamt (BKA) und der Bundesgrenzschutz (BGS) setzen seit 1998 „in außergewöhnlichen Polizeilagen und Fällen mit hohem Gefährdungsgrad zur Bekämpfung besonders schwerer Kriminalität“ [10] den IMSI-Catcher ein. Der Grund hierfür besteht zum einen darin, dass keine Abhörmaßnahmen getätigt werden können, sobald eine observierte Person keine Kommunikation über das herkömmliche Festnetz tätigt, und zum anderen der Aufenthaltsort bei mobiler Nutzung nicht bekannt ist. Um dennoch eine Abhörmaßnahme starten zu können, wird ein IMSI-Catcher eingesetzt. Hierzu sei jedoch angemerkt, dass das BKA oder der BGS wohl eher lediglich die IMSI einer Person ermittelt und anschließend die Gespräche über den Netzanbieter abhören lässt. Außerdem dürfen nur diese beiden Behörden den IMSI-Catcher nutzen. Das Landeskriminalamt (LKA) hingegen



Abbildung 5.1: IMSI-Catcher GA090

<sup>1</sup> Rohde & Schwarz, <http://www.rohde-schwarz.de/> [Online; letzter Aufruf 30.09.2009]

<sup>2</sup> IMSI-Catcher – Wanzen für Handys, [http://www.iwi.uni-hannover.de/1v/ucc\\_ws04\\_05/riemer/literatur/imsi-catcher.htm](http://www.iwi.uni-hannover.de/1v/ucc_ws04_05/riemer/literatur/imsi-catcher.htm) [Online; letzter Aufruf 30.09.2009]

gen darf einen solchen beispielsweise nicht besitzen. Der Erwerb solch eines IMSI-Catchers ist nur für Behörden bzw. Geheimdienste bestimmt, wobei die Kosten sich auf ungefähr 200.000 Euro<sup>3</sup> belaufen.

Auf Basis der USRP Hardware wurde ein IMSI-Catcher Prototyp entwickelt, für den zunächst geklärt wird, welche Hard- und Softwarekomponenten benötigt werden. Anschließend werden die bereitgestellten Sicherheitsmechanismen im GSM-Netz aufgezeigt und der IMSI-Catcher mit einem Standard IMSI-Catcher verglichen. Darauf aufbauend werden die Funktionsweise, daraus resultierende Probleme und deren Lösungen aufgezeigt. Als Letztes wird die entwickelte Oberfläche sowie die Bedienung des IMSI-Catchers näher erläutert.

## 5.1 Vorbedingung

Für den Betrieb des IMSI-Catchers werden bestimmte Hardware- sowie Softwarekomponenten benötigt. Der Aufbau und die Inbetriebnahme der Komponenten wird in Abschnitt 5.7.3 näher beschrieben. Detailliertere Angaben zu den Hardwarekomponenten befinden sich in Kapitel 3, und das Zusammenspiel der Komponenten wird in Kapitel 4 erläutert.

### 5.1.1 Hardware

Als Hardware kommt das USRP1 zum Einsatz, welches für den GSM900 Bereich mit zwei RFX900 Transceiverboards ausgestattet ist. Falls im GSM1800 Bereich gesendet werden soll, müssen die RFX1800 Transceiverboards eingebaut werden. An das USRP1 werden der externe Taktgeber und die notwendigen Antennen angeschlossen.

### 5.1.2 Software

Softwareseitig muss das GNU Radio und die modifizierte OpenBTS Version für den IMSI-Catcher installiert sein. Ebenfalls notwendig ist ein eigener Asterisk-Server. Hiermit wird jedem Teilnehmer ein SIP-Account zugewiesen, wodurch eine Authentifizierung des Benutzers und ausgehende Gespräche sowie die Aufzeichnung dieser Gespräche ermöglicht werden. Die benötigten Konfigurationsdateien *sip.conf* und *extensions.conf* befinden sich im Ordner *osic/GUI/asterisk* und müssen in den Installationspfad des Asterisk-Server kopiert sowie die darin enthaltenen Pfade für die Gesprächsaufzeichnung angepasst werden. Die vorgenommenen Änderungen der OpenBTS Version werden in Abschnitt 5.6 genauer beschrieben.

## 5.2 GSM-Sicherheitsaspekte

### 5.2.1 IMSI - TMSI

Die IMSI (vgl. Kapitel 2.3.1) hat immer 15 Zeichen und setzt sich aus einem dreistelligen Ländercode (MCC), einem zwei- bis dreistelligen Netzanbietercode (MNC) sowie einer höchstens zehnstelligen Teilnehmeridentität (MSIN) zusammen. Beispielsweise:

$\overbrace{262}^{\text{MCC}} \overbrace{02}^{\text{MNC}} \overbrace{7033983293}^{\text{MSIN}}$

Hierbei steht 262 für Deutschland und 02 für den Netzanbieter Vodafone. Da die IMSI einem Teilnehmer eindeutig zugeordnet werden kann, soll sie aus Sicherheitsgründen so selten wie

---

<sup>3</sup> IMSI-Catcher, <http://de.wikipedia.org/wiki/IMSI-Catcher> [Online; letzter Aufruf 30.09.2009]

möglich übertragen werden. Daher wird dem Teilnehmer beim Einbuchen in das Netz temporär eine TMSI innerhalb einer LA zugewiesen. Bei jeder Teilnehmeraktion wird anstelle der IMSI die TMSI übertragen, um das Erstellen von Bewegungsprofilen zu erschweren und die Anonymität des Teilnehmers zu gewährleisten. Somit muss der Teilnehmer nur in wenigen Fällen mit seiner IMSI identifiziert werden. Beim IMSI-Catcher wird die IMSI für die Registrierung und Identifizierung innerhalb des Asterisk-Servers genutzt. Nur mit einem Asterisk-Account kann der Teilnehmer Gespräche führen, die wiederum im IMSI-Catcher angezeigt werden.

### 5.2.2 Verschlüsselung der Kommunikation

Zwischen der MS und dem BTS wird die aktive Kommunikation auf der  $U_m$  Schnittstelle verschlüsselt übertragen (siehe Kapitel 2.3.3). Die Angriffe auf die Verschlüsselung [4] lassen sich in drei Hauptgruppen gliedern:

- Umgehen der Verschlüsselung mit Hilfe eines IMSI-Catchers
- Nachträgliche Entschlüsselung eines mitgeschnittenen verschlüsselten Datenstroms<sup>4</sup>
- Den eventuell unverschlüsselten Datenstrom nach der  $U_m$  Schnittstelle wie beispielsweise auf dem Richtfunk zwischen dem BSC und dem MSC mitschneiden. Ebenfalls möglich, ist ein Angriff auf die physikalischen Telefonleitungen, um dort das Gespräch abzuhören.

Da auf der Richtfunkstrecke sehr viele Kommunikationsverbindungen übertragen werden, kann mit Hilfe der IMSI bzw. TMSI nach einer bestimmten Kommunikation gefiltert werden.

## 5.3 Funktionsweise

Die Abbildung 5.2 veranschaulicht die Funktionsweise eines Standard IMSI-Catchers. Der IMSI-Catcher simuliert die Basisstation einer regulären Funkzelle eines Netzanbieters.<sup>5</sup> Die Mobile Station bucht sich in den IMSI-Catcher ein und authentifiziert sich wie bei einer regulären Basisstation (siehe Kapitel 2.3.3). Da sich die MS gegenüber dem Netz, das Netz jedoch nicht gegenüber der MS, authentifizieren muss, ist die Simulation einer Basisstation und somit der Betrieb eines IMSI-Catchers möglich. Diese Sicherheitslücke wurde im UMTS-Netz behoben, da sich hier das Netz ebenfalls authentifizieren muss. Der Teilnehmer kann während der Authentifizierung und im Stand-by-Betrieb<sup>6</sup> nicht feststellen, dass die MS nicht in einem regulären Netz eingebucht ist. Sobald ein Teilnehmer ein Gespräch führt, leitet der IMSI-Catcher das Gespräch über eine an ihn angeschlossene MS in das reguläre Funknetz weiter. Dies entspricht einem klassischen Man-in-the-Middle Angriff. Hierbei gilt zu beachten, dass die MS des IMSI-Catchers nicht die Identität des Teilnehmers vortäuscht, sondern die eigene Identität und Verschlüsselung des Netzes nutzt. Der genaue Vorgang kann der Abbildung 5.5 entnommen werden, wobei der Registrierungsvorgang im Abschnitt 5.4 näher erläutert wird. Der IMSI-Catcher schaltet lediglich die Verschlüsselung des Teilnehmers aus, wodurch es ihm möglich ist die Gespräche, die er anschließend selbst verschlüsselt, abzuhören.

Die Funktionsweise des Open Source IMSI-Catchers (siehe Abbildung 5.3) unterscheidet sich vom Standard-IMSI-Catcher dahingehend, dass für die Weiterleitung der Daten keine an den IMSI-Catcher angeschlossene MS verwendet werden kann. Diese Beschränkung ergibt sich auf

<sup>4</sup>Open-Source-Projekt geht GSM an den Kragen, <http://www.heise.de/newsticker/Open-Source-Projekt-geht-GSM-an-den-Kragen--/meldung/144332> [Online; letzter Aufruf 30.09.2009]

<sup>5</sup>Die Simulation einer Basisstation wird auch als Masquerade-Attack bezeichnet.

<sup>6</sup>Eine MS befindet sich im Stand-by-Betrieb, falls sie in einem Netz eingebucht ist, aber keine aktive Verbindung wie zum Beispiel bei einem Gespräch besitzt.

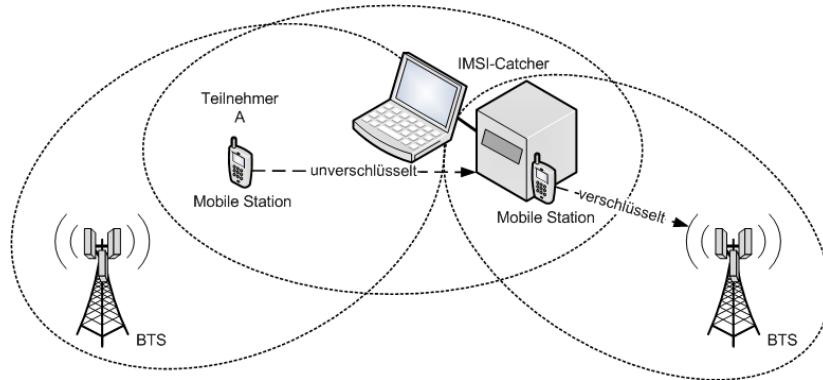


Abbildung 5.2: Standard IMSI-Catcher

Grund der verwendeten Hard- und Softwarekomponenten. Daher ist der Aufbau des IMSI-Catchers und die Authentifizierung (siehe Abschnitt 5.4) gegenüber dem Standard IMSI-Catcher leicht verändert. Der unverschlüsselte Datenverkehr wird nicht mehr über eine MS sondern über den Computer und darin installiertem Asterisk-Server über das Internet an den Asterisk-Server der Universität Freiburg weitergeleitet. Dieser Server besitzt einen SIP-Account, der es ermöglicht, Festnetzgespräche zu führen. In der Realität ist es aber ohne Weiteres möglich, den Asterisk-Server auf dem Computer mit einem SIP-Account einzurichten, mit dem sowohl in das Festnetz als auch in das Mobilfunknetz telefoniert werden kann. Wie beim Standard IMSI-Catcher wird jedoch nicht die Identität des Teilnehmers vorgetäuscht. Aus diesem Grund muss die Rufnummerübermittlung deaktiviert werden. Der angerufene Teilnehmer bekommt somit einen Anruf von einem „unbekannten Teilnehmer“. Dieser Umstand führt jedoch selten zu einem Problem, da sich die Teilnehmer höchstens darüber wundern, dass die Rufnummer nicht angezeigt wird. Diese „Fehlfunktion“ würden sie allerdings auf etwaige Fehlkonfigurationen oder Probleme im Netz zurückführen.

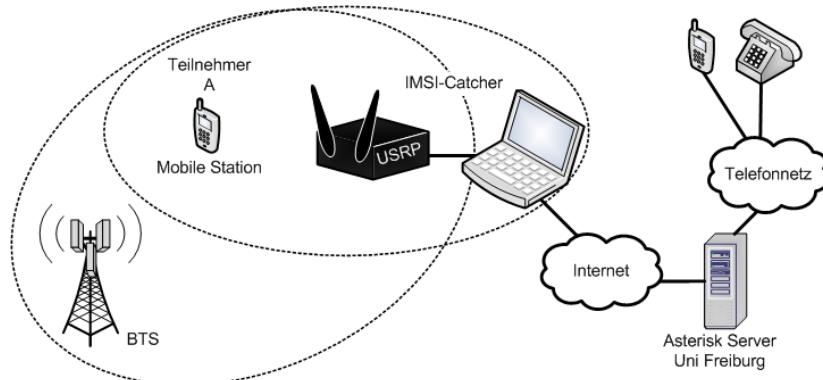


Abbildung 5.3: Open Source IMSI-Catcher mit USRP und Asterisk Server

### 5.3.1 Reale Funknetze simulieren

Damit die MS das Netz, welches vom IMSI-Catcher simuliert wird, nicht von einem originalen Netz unterscheiden kann, muss der IMSI-Catcher ein reales Netz eines Netzanbieters vortäuschen. Hierfür müssen diverse Einstellungen in der *OpenBTS.conf* vorgenommen werden. Wichtige Parameter für das Simulieren einer Funkzelle können der Tabelle 5.1 entnommen werden. Diese Werte haben jedoch nur einen Beispielcharakter, da sich die Frequenzen

(ARFCN), der LAC und die CID von Standort zu Standort unterscheiden. Wichtig für das Simulieren sind lediglich der richtige Country Code (MCC), der vorzutäuschenende Netzanbieter Code (MNC), die Frequenz und der Name der Funkzelle (Shortname), der identisch dem Netzanbieter sein muss. Allerdings darf die Frequenz nicht die selbe sein, falls die Basisstation zu stark sendet, da sich die Frequenzen überlagern würden und nicht vorherzusagen ist, was genau passiert würde. Daher müssen Informationen über aktuelle Basisstationen mit Hilfe der Frequenzanalyse aus Abschnitt 5.4.5 abgerufen werden. Alle anderen Parameter sind irrelevant, da diese Informationen lediglich der MS dazu dienen, ihren Standort (LAC und CID) zu bestimmen. Weitere Parameter, die für den Betrieb einer Funkzelle notwendig sind, werden in Abschnitt 5.7.3 näher betrachtet.

T-Mobile		Vodafone	
Ländercode (MCC)	262	Ländercode (MCC)	262
Netzcode (MNC)	01	Netzcode (MNC)	02
Location Area (LAC)	29191	Location Area (LAC)	793
Kanal (ARFCN)	102	Kanal (ARFCN)	64
ZellenID (CID)	29242	ZellenID (CID)	6913

E-Plus		$O_2$	
Ländercode (MCC)	262	Ländercode (MCC)	262
Netzcode (MNC)	03	Netzcode (MNC)	07
Location Area (LAC)	588	Location Area (LAC)	50945
Kanal (ARFCN)	984	Kanal (ARFCN)	711
ZellenID (CID)	11768	ZellenID (CID)	29790

Tabelle 5.1: Beispielparameter zum Vortäuschen einer realen Funkzelle der Netzanbieter T-Mobile, Vodafone, E-Plus und  $O_2$

## 5.4 Registrierung

Die Registrierung der MS in den IMSI-Catcher ist von zentraler Bedeutung. Aus diesem Grund wird die Registrierung des IMSI-Catchers mit der Spezifikation für die Registrierung innerhalb des GSM-Netzes miteinander verglichen. Zur Abgrenzung und Verdeutlichung der Funktionsweise wird überdies ein Vergleich mit einem Standard IMSI-Catcher und dem Open Source IMSI-Catcher aufgezeigt. Als Grundvoraussetzung bei allen drei Methoden muss die MS zunächst die Frequenz einer Basisstation finden, bevor sie sich registrieren kann. Hierzu kann sie die Frequenz der zuletzt eingebuchten Basisstation speichern. Auf dieser Frequenz versucht die MS als erstes eine Basisstation zu finden. Falls dort eine entsprechende Basisstation existiert, liest die MS die BCCH Daten wie beispielsweise Synchronisationsinformationen aus und versucht sich zu registrieren. Falls allerdings keine Basisstation auf der zuletzt gespeicherten Frequenz existiert oder die MS keine Liste von bekannten Basisstationen besitzt, wird die „normal cell selection“ ausgeführt. Hierbei scannt die MS verschiedene Frequenzkanäle und versucht BCCH Informationen einer aktiven Basisstation zu empfangen. [8]

### 5.4.1 Registrierung – GSM

Der GSM-Registrierungsprozess, dargestellt in Abbildung 5.4, ist folgendermaßen gegliedert: Die MS startet ein Location Update Request und überträgt ihre IMSI und ihren aktuellen

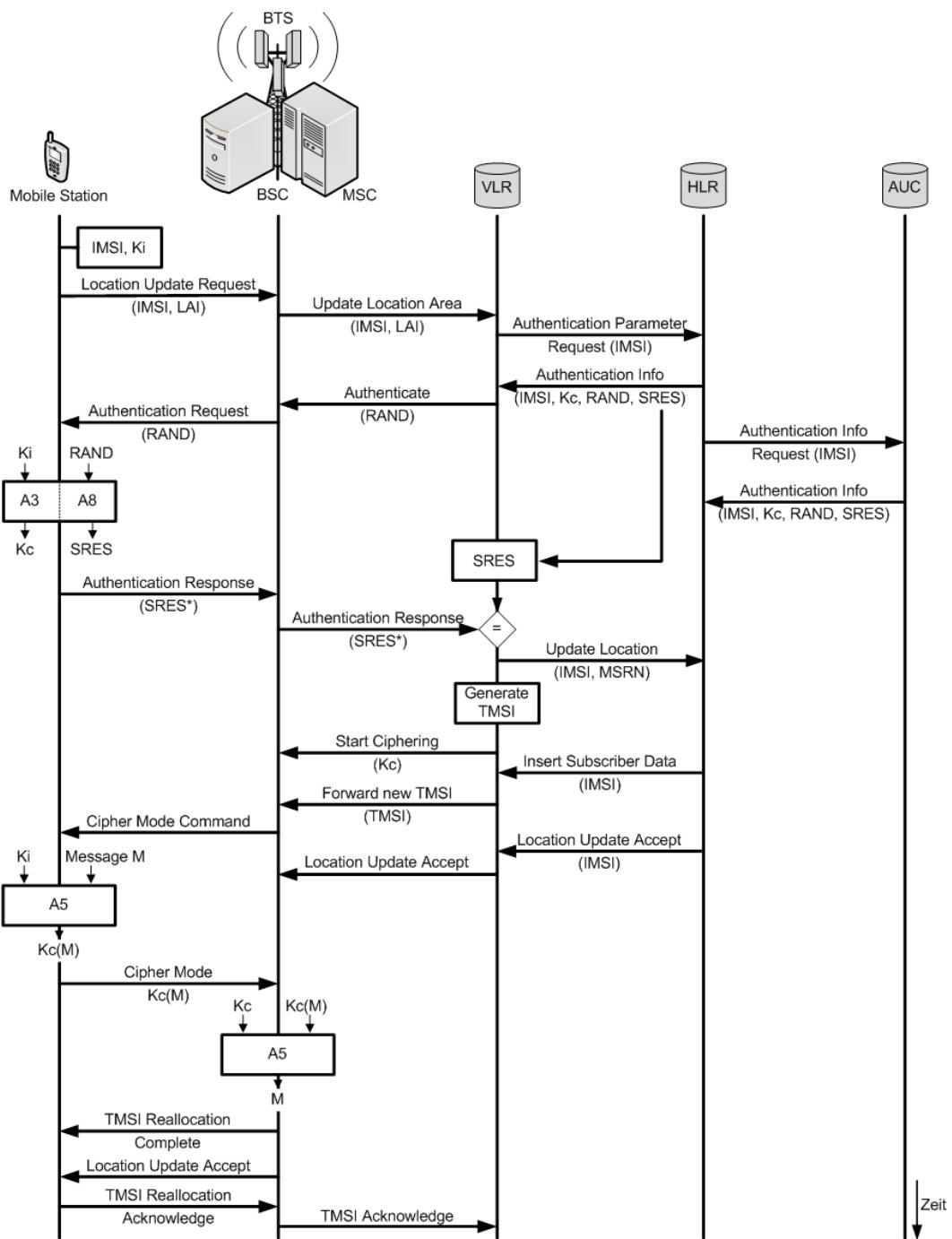


Abbildung 5.4: Registrierungsprozess bei GSM (Vorlage nach [13])

Aufenthaltsort (LAI). Das MSC aktualisiert den Aufenthaltsort in seiner VLR-Datenbank. Anschließend wird ein vorhandenes Authentication Triplet für die IMSI aus dem HLR an das VLR übertragen. Die Zufallszahl RAND wird der MS mitgeteilt und somit das Challenge-Response-Verfahren zur Authentifizierung gestartet. Die MS generiert eine SRES\* (siehe Kapitel 2.3.3), welche mit der SRES im VLR verglichen wird. Bei erfolgreicher Authentifizierung wird der aktuelle Aufenthaltsort im HLR gespeichert und der Verschlüsselungsmechanismus gestartet. Das VLR erstellt anschließend eine TMSI und verschickt diese an die MS. Die MS bestätigt den Location Update sowie die erhaltene TMSI.

#### 5.4.2 Registrierung – Standard IMSI-Catcher

Die Registrierung beim Standard IMSI-Catcher läuft im Wesentlichen wie nach der GSM-Spezifikation ab. Wichtige Unterschiede sind in der Abbildung 5.5 ersichtlich. Beim Authentifizierungsprozess sendet der IMSI-Catcher der MS die RAND zur Authentifizierung, die SRES\* wird jedoch vom IMSI-Catcher anschließend nicht beachtet. Der IMSI-Catcher weist der MS eine TMSI zu und schaltet die Verschlüsselung aus. Anschließend registriert sich die IMSI-Catcher MS ganz normal beim GSM-Netz. Der Kommunikationsstrom des Teilnehmers wird an die IMSI-Catcher MS weitergeleitet, die die Verschlüsselung der Daten wie bei einer normalen Kommunikation zum GSM-Netz vornimmt.

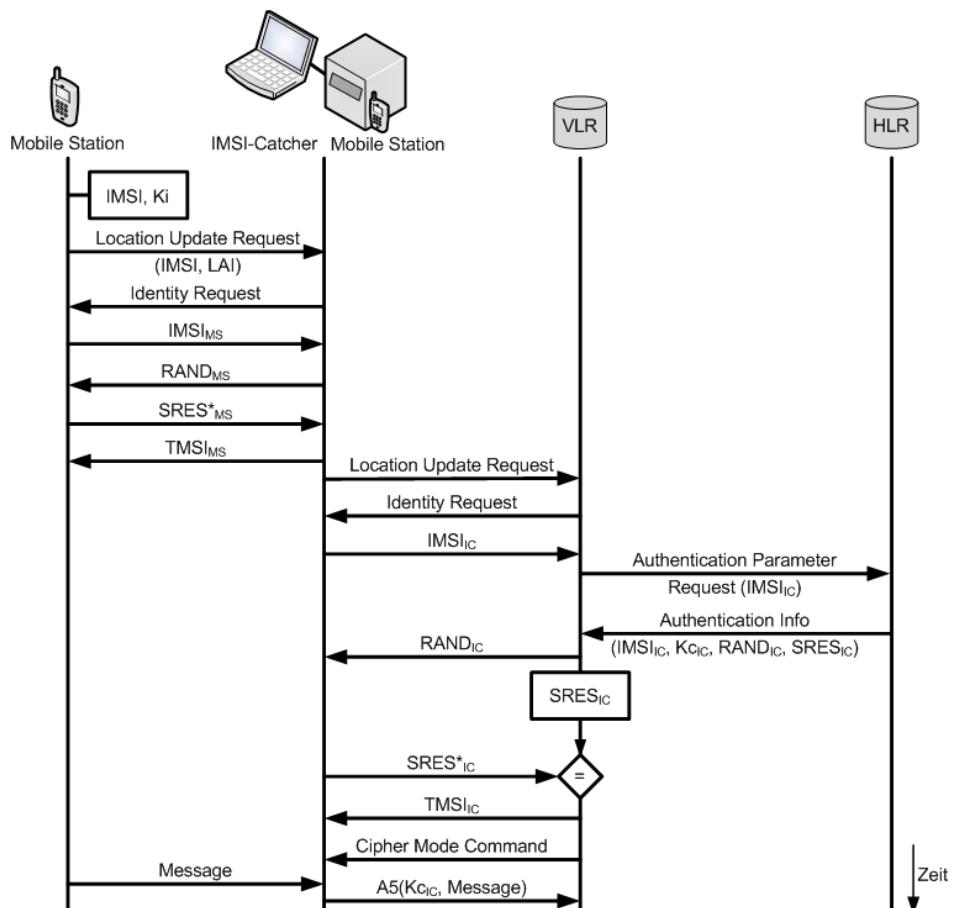


Abbildung 5.5: Standard IMSI-Catcher Registrierung und Kommunikation

### 5.4.3 Registrierung – Open Source IMSI-Catcher

Der Registrierungsprozess (Abbildung 5.6) beim IMSI-Catcher wurde nach den GSM-Spezifikationen weitestgehend übernommen. Unterschiede gibt es jedoch bei der Authentifizierung und bei der Verschlüsselung. Im Gegensatz zum GSM-Registrierungsprozess geschieht die Authentifizierung mit Hilfe des Asterisk-Servers. Hierzu sendet der IMSI-Catcher eine Registrierungsanfrage in Form einer SIP-Message mit der IMSI als Benutzername. Falls der Teilnehmer einen Benutzer-Account besitzt, ist die Authentifizierung erfolgreich und der Asterisk-Server sendet eine „200 OK“ SIP-Message zurück. Anschließend wird die TMSI generiert und unverschlüsselt an die MS übertragen, die diese bestätigt. Falls der Teilnehmer keinen Benutzer-Account besitzt, schlägt die Authentifizierung fehl und der Asterisk-Server überträgt daraufhin eine „404 Not Found“ SIP-Message. Der MS wird anschließend eine „IMSI not in VLR“ Nachricht geschickt und der Übertragungskanal geschlossen. Der IMSI-Catcher bietet allerdings die Option der „OpenRegistration“, die es allen Teilnehmern ermöglicht sich ohne Asterisk-Account anzumelden. Der Ablauf der Registrierung kann den Logfiles *manuelle\_registrierung\_nokia3310.DEBUG.out* und *manuelle\_registrierung\_nokia3310.INFO.out* auf der CD im Ordner *Logfiles* entnommen werden. Hierbei handelt es sich um Logfiles mit dem Loglevel „DEBUG“ und „INFO“. Im DEBUG Loglevel ist der Registrierungsvorgang detaillierter dokumentiert.

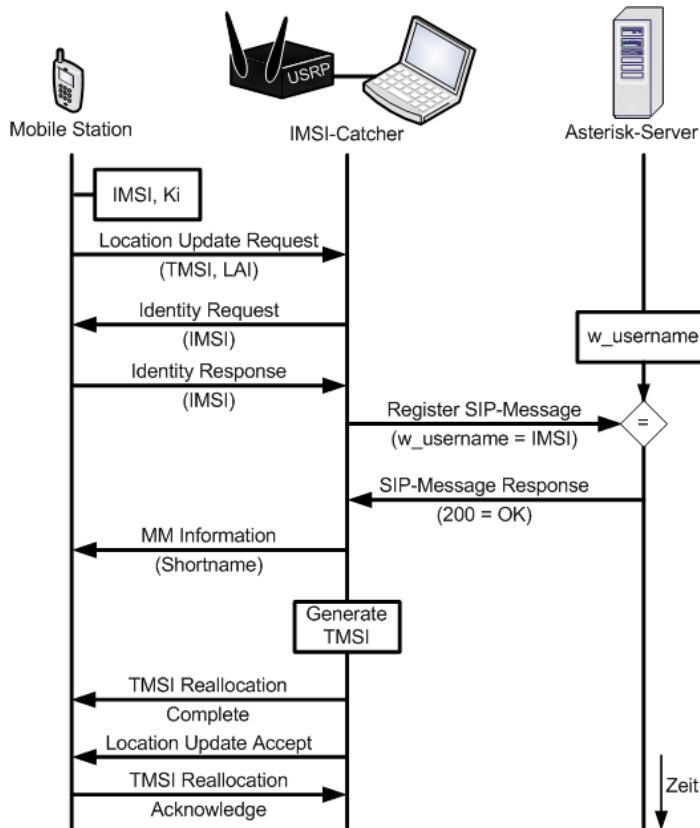


Abbildung 5.6: Registrierungsprozess beim IMSI-Catcher

Insgesamt existieren drei Möglichkeiten, wie sich eine MS in ein vorgetäusches Netz des IMSI-Catchers einbuchen kann. Diese drei Möglichkeiten lassen sich in zwei Hauptkategorien gliedern: Zum einen falls das vorzutäuschenende Netz eines Netzanbieters nicht vorhanden ist (zum Beispiel in schlecht versorgten Gebieten oder Räumlichkeiten), zum anderen falls

das Zielnetz vorhanden ist. Diese Unterscheidung der Fälle ist sinnvoll, um abhängig davon verschiedene Maßnahmen einzuleiten, damit die MS sich in das Netz des IMSI-Catchers einbucht. Wie sich durch diverse Versuche herausgestellt hat, ist es dabei unerheblich, ob der Teilnehmer eine manuelle oder automatische Netzauswahl eingestellt hat. Da die MS das Netz des IMSI-Catchers nicht von einem Originalnetz seines Netzanbieters unterscheiden kann, ist dieses Verhalten leicht zu erklären.

### **1. Fall - Zielnetz nicht vorhanden**

Die MS besitzt keine Konnektivität und befindet sich im Suchmodus (normal cell selection), wodurch nacheinander verschiedene Frequenzen nach einer aktiven BTS gescannt werden. In diesem Fall muss der IMSI-Catcher lediglich eine beliebige Frequenz, den Ländercode 262 als auch den benötigten Netzanbietercode (siehe Tabelle 5.1) und Shortname zum Vortäuschen einer realen Funkzelle einstellen. Sobald die MS diese Frequenz scannt, versucht sie sich einzubuchen. Dieses Verhalten konnte durch verschiedene Versuche nachgewiesen werden und bildet die Basis für den Fall 2.2.

### **2. Fall - Zielnetz vorhanden**

Problematischer ist der Fall, insofern ein aktives Netz vorhanden und die MS in diesem eingebucht ist. Hier existieren zwei Möglichkeiten, wie eine MS dazu gebracht werden kann, sich in den IMSI-Catcher einzubuchen. Eine Möglichkeit besteht darin, einen manuellen Zellwechsel anzuregen und eine weitere, die Frequenz, auf der die MS im Netz des Anbieters eingebucht ist, zu stören, damit die MS sich in den IMSI-Catcher einbucht.

#### **1. Erzwungener Zellwechsel**

Die Grundidee bei diesem Szenario ist, auf Grund der Nachbarschaftsliste einen Zellwechsel der MS in den IMSI-Catcher zu erzwingen. In Abschnitt 5.4.4 wird detaillierter auf dieses Szenario eingegangen.

#### **2. Jammer**

Ein GSM-Jammer ist ein Störsender, dessen Zweck darin besteht, die Frequenz, in der die MS eingebucht ist, zu stören. Mit Hilfe des Jammers verliert die MS die Verbindung zu der aktuellen BTS und wechselt in den Frequenzsuchmodus. Ziel ist es, dass der IMSI-Catcher anschließend als Netz ausgewählt wird. Abschnitt 5.4.5 erläutert die Funktionsweise und die notwendigen Komponenten näher.

#### **5.4.4 Erzwungener Zellwechsel**

Beim Einbuchen der MS in eine BTS teilt die BTS unverschlüsselt der MS eine Nachbarschaftsliste mit. Diese Nachbarschaftsliste beinhaltet die Frequenzen bekannter Basisstationen in der näheren Umgebung. Auf diesen Frequenzen empfängt die MS den unverschlüsselten Broadcast der Nachbarn und führt Empfangsmessungen zu diesen Basisstationen durch. Auf dem Broadcast der Nachbarn werden ebenfalls deren Nachbarschaftslisten übertragen, auf denen die MS gleichermaßen versucht den Broadcast abzuhören. Sobald der IMSI-Catcher eine eigene Funkzelle erstellt und auf einer bestimmten Frequenz sendet, würde die MS die neue Funkzelle nicht bemerken. Aus diesem Grund wird versucht, einen Zellwechsel anzuregen, damit sich die MS in den IMSI-Catcher einbucht. Es wird der Umstand ausgenutzt, dass die MS Empfangsmessungen zu ihrer Nachbarschaft durchführt. Mit Hilfe des Nokia Netzmonitors (siehe Abschnitt 5.4.5) können die einzelnen Empfangswerte zu den benachbarten Basisstationen ausgelesen werden. Hierzu muss das Nokia 3310 mit einer SIM-Karte

des Zielnetzes ausgerüstet werden, sich in das Netz einbuchen und die Nachbarschaftsliste anzeigen. Der IMSI-Catcher muss anschließend auf der Frequenz der Basisstation mit dem schlechtesten Empfangswert senden. Die Abbildung 5.7 verdeutlicht die Grundidee. Die MS ist mit der Basisstation BTS1 verbunden, welche ihr die benachbarten Basisstationen BTS2, BTS3 und BTS4 mitteilt. Im weiteren Verlauf misst sie periodisch die Empfangswerte zu den Basisstationen. Sobald der IMSI-Catcher auf der Frequenz der BTS4 sendet, verbessert sich der Empfangswert der MS zu der Frequenz der BTS4. Die MS nimmt also an, dass sich der Empfangswert zur BTS4 verbessert hat. Da der Empfangswert von der BTS4 nun besser ist als mit der zur Zeit verbundenen BTS1, führt die MS einen Zellwechsel durch. Sie wird sich jedoch nie in die BTS4 einbuchen, sondern in den IMSI-Catcher. Dieses Verfahren funktioniert allerdings nur, falls sich die MS im Stand-by-Betrieb befindet und somit keine aktive Kommunikation führt. Für den Fall, dass sie eine aktive Verbindung hergestellt hat, wird bei diesem Verfahren versucht, netzseitig ein Handover zu erzwingen. Auf Grund fehlender Implementierung kann kein Handover in OpenBTS stattfinden. Um Mobile Stations, die eine aktive Kommunikation führen, dennoch zum Wechsel in den IMSI-Catcher zu bewegen wird ein Störsender benötigt, der die Frequenz, auf der die Kommunikation stattfindet, stört und somit die Verbindung abbricht. Anschließend führt der erzwungene Zellwechsel erneut zum Erfolg.

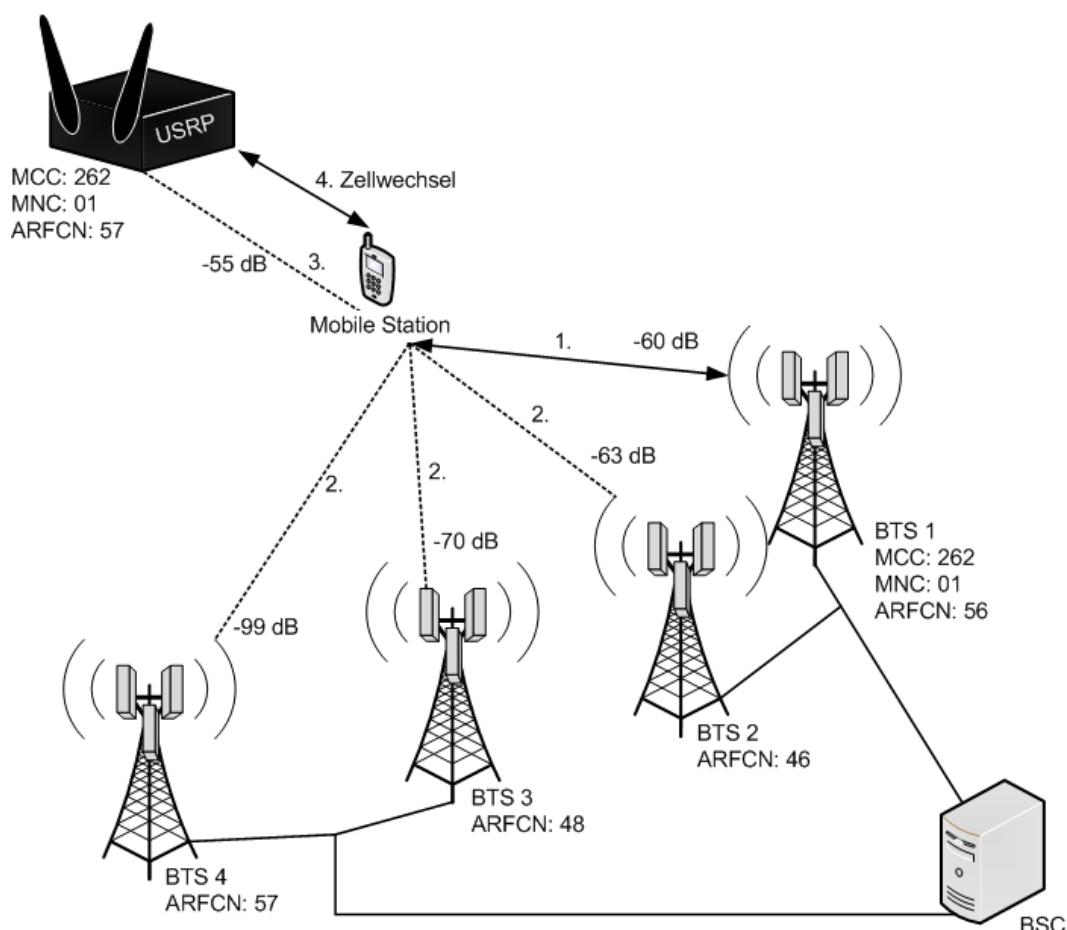


Abbildung 5.7: Erzwungener Zellwechsel von der BTS1 zum IMSI-Catcher

### 5.4.5 Jammer

Falls der IMSI-Catcher auf einer bestimmten Frequenz arbeiten soll, diese aber nicht mit der Nachbarschaftsliste (siehe Abschnitt 5.4.4) der MS übereinstimmt, wird die Frequenz des IMSI-Catchers nicht gescannt. Aus diesem Grund kann die MS sich im Stand-by-Betrieb nicht in den IMSI-Catcher einbuchen. Hinsichtlich der fehlenden Handover-Implementierung von OpenBTS besteht ein weiteres Problem darin, dass die MS bei einer aktiven Kommunikation sich ebenfalls nicht in den IMSI-Catcher einbucht. Mittels eines GSM-Jammers können allerdings gezielt die Frequenzen der Basisstationen gestört werden. Sofern der Jammer stärker sendet als die MS ihrer Basisstation empfängt, verliert die MS die Verbindung. Die Abbildung 5.8 verdeutlicht die Funktionsweise, wobei der Jammer (grüne Linie) stärker sendet als die Basisstation (blaue Linie). Die MS sucht anschließend nach bekannten Basisstationen bzw. führt die „*normal cell selection*“ (siehe Abschnitt 5.4) aus. Im Idealfall registriert sich die MS beim IMSI-Catcher. Es kann aber durchaus vorkommen, dass die MS eine andere Basisstation aus ihrer Liste wählt und sich nicht in den IMSI-Catcher einbucht. Da ein Gebiet von meh-

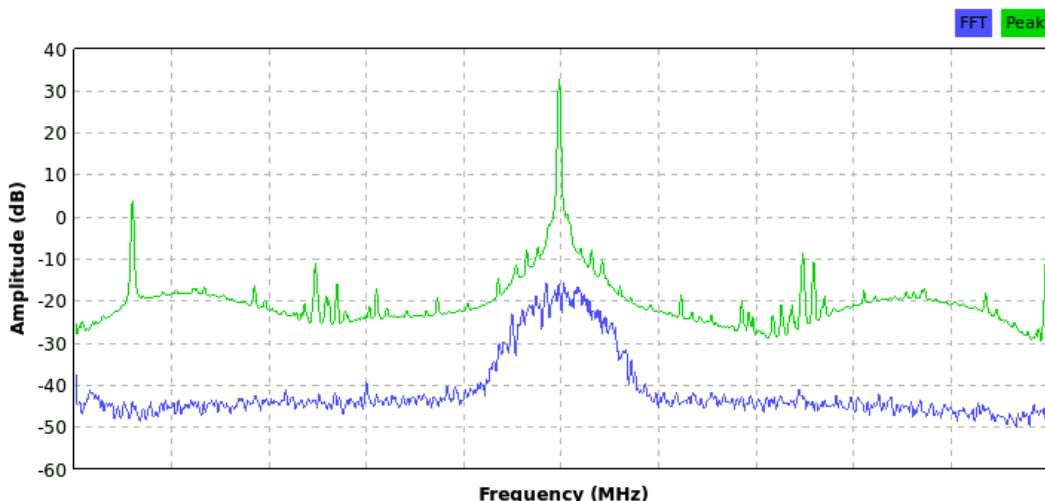


Abbildung 5.8: GSM-Jammer im Einsatz. Die blaue Linie stellt eine Basisstation und die grüne Linie den Jammer dar.

reren Basisstationen versorgt und innerhalb einer Zelle verschiedene Frequenzen verwendet werden, kann der IMSI-Catcher auf Grund seiner Sendeleistung nur einen bestimmten Bereich stören, in denen die Teilnehmer anschließend keinen Empfang mehr haben. Daher muss der Jammer nacheinander verschiedenen Frequenzen oder ein breiteres Spektrum stören. Hierzu müssen zunächst die Frequenzen der Basisstationen bekannt sein. Im Folgenden werden zwei Frequenzanalyse-Methoden, basierend auf dem Nokia Netzmonitor und dem USRP2, vorgestellt.

### Frequenzanalyse

Um die Basisstationen in näherer Umgebung stören zu können, müssen dem Jammer deren Frequenzen bekannt sein. Nur dann können die Mobile Stations dazu gebracht werden, sich in den IMSI-Catcher einzubuchen. Hierfür gibt es mehrere Methoden. Zum einen ist es die Netzanalyse mit Hilfe des Nokia Netzmonitors, der gleichzeitig Informationen über die in der aktuell eingebuchten Basisstation als auch über benachbarte Basisstationen liefert. Diese Informationen werden zum Vortäuschen eines Originalnetzes verwendet. Und zum anderen gibt es die Spektrumsanalyse mit Hilfe des USRP2. Eine ausführliche Beschreibung und weitere

Methoden der GSM-Analyse erfolgen überdies in der Masterarbeit von Holger Bertsch.[2]

Einige Nokia Modelle sind in der Lage, mittels Software einen Netzmonitor (NetMonitor) zu aktivieren. Dieser Netzmonitor ist ein verstecktes Menü, welches eigentlich für Techniker und Mobilfunkbetreiber zu Testzwecken verwendet wird und eine Vielzahl an Informationen anzeigt. Eine Anleitung für die Aktivierung von DCT-3 bzw. DCT-4 Mobilfunkgeräten kann der Nokiaport<sup>7</sup> Seite entnommen werden. Für die GSM-Analyse wurde daher ein Nokia 3310 mit aktiviertem Netzmonitor<sup>8</sup> verwendet. Die für den IMSI-Catcher wichtigen Nachbarschaftsinformationen befinden sich auf dem Display 03 und 04 (Abbildung 5.9). Hierbei bezeichnet die erste Spalte die ARFCN der Nachbarn bzw. die ersten Zeile die ARFCN der aktuell eingebuchten Basisstation und die dritten Spalte den Empfangswert. Die Basistation mit dem schlechtesten Empfangswert auf dem Display 04 oder 05 stellt beim erzwungenen Zellwechsel (Kapitel 5.4.4) ein guter Kandidat zum Vortäuschen dar. In Abbildung 5.9 entspricht dies der Basisstation, die auf dem Kanal 71 sendet und einen Empfangswert von  $-102\text{ dBm}$  hat. Das Display 11 enthält weitere Informationen, die zum Vortäuschen eines Netzes wichtig sind, wie beispielsweise den LAC. Die ARFCN und die Netzparameter müssen lediglich in die *OpenBTS.conf* eingetragen werden. Für das gezielte Stören mehrerer Basisstationen benötigt der Jammer allerdings deren Frequenzen. Deshalb muss das Nokia 3310 mit der SIM-Karte des Zielnetzes ausgerüstet und die ARFCN der Nachbarschaft ausgelesen werden. Anschließend müssen die ARFCN auf eine Frequenz zurückgerechnet oder den Tabellen in Anhang D für GSM900 bzw. Anhang E für GSM1800 entnommen werden.



Abbildung 5.9: Nokia 3310 Netzmonitor

Die Trägerfrequenzen der Up- und Downlinkrichtung können anhand der ARFCN mit Hilfe von zwei Formeln<sup>9</sup> zurückgerechnet werden:

$$f_{downlink} = \text{Startfrequenz} + (\text{ARFCN} - \text{Offset}) \cdot 0,2\text{ MHz}$$

$$f_{uplink} = f_{downlink} - \text{Abstand}$$

Die für die Berechnung notwendigen Parameter können der Tabelle 5.2 entnommen werden, wobei E-GSM900 für den Frequenzbereich von 925 bis 934,8 MHz (Downlink) bzw. 880 bis 890 MHz (Uplink) verwendet werden muss. Der Frequenzabstand ergibt sich aus der Duplexkanaleigenschaft (vgl. Kapitel 2.4). Zur Verdeutlichung eine Beispielrechnung für die ARFCN 71 aus dem Netzmontior:

$$f_{downlink} = 935\text{ MHz} + 71 \cdot 0,2\text{ MHz} = 949,2\text{ MHz}$$

$$f_{uplink} = 949,2\text{ MHz} - 45\text{ MHz} = 904,2\text{ MHz}$$

<sup>7</sup> Nokia Net-Monitor, <http://nokiaport.de/index.php?mid=2&pid=netmon> [Online; letzter Aufruf 02.10.2009]

<sup>8</sup> Eine detaillierte Erklärung aller Displays liefert die Seite [http://www.mwiacek.com/gsm/netmon/faq\\_net2.htm#12](http://www.mwiacek.com/gsm/netmon/faq_net2.htm#12) [Online; letzter Aufruf 02.10.2009]

<sup>9</sup> ARFCN, <http://de.wikipedia.org/wiki/ARFCN> [Online; letzter Aufruf 03.10.2009]

Name	ARFCNs	Startfrequenz	Offset	Abstand
GSM900	1 - 124	935 MHz	0	45 MHz
E-GSM900	975 - 1023	935 MHz	1024	45 MHz
GSM1800	512 - 885	1805,2 MHz	512	95 MHz

Tabelle 5.2: Werte für die Berechnung der GSM-Frequenzen

Eine weitere Methode, die GSM-Frequenzen zu untersuchen, stellt das *FFT* Tool für das USRP2 dar, welches bei der Installation von GNU Radio mitgeliefert wird. Dieses Tool ermöglicht die Frequenzen anhand ihrer Empfangsstärke in Echtzeit grafisch darzustellen. Ziel ist es, im GSM-Band nach Basisstationen zu suchen, um den Jammer auf diese Frequenz einzustellen. Verwendet wird hierzu das USRP2, welches mit einem Transceiverboard oder mit dem DBRSX Empfänger ausgerüstet ist, um sowohl im GSM900 als auch im GSM1800 Bereich arbeiten zu können. Da das USRP2 allerdings gleichzeitig als Jammer verwendet wird, sollte zwingend ein RFX900 bzw. RFX1800 Transceiverboard verwendet werden. Gestartet wird das Pythonscript mit dem Kommando `sudo usrp2_fft.py`. Im anschließenden Fenster (Abbildung 5.10) sollte die Option „Average“ aktiviert werden, welche eine bessere Erkennung der Basisstationen ermöglicht. Wie deutlich zu erkennen, gibt es drei Ausschläge – auch Peaks genannt – im Bereich um 928 MHz. Diese drei Peaks sind 200 kHz breite Kanäle, die von verschiedenen bzw. einer Basisstation übertragen werden.

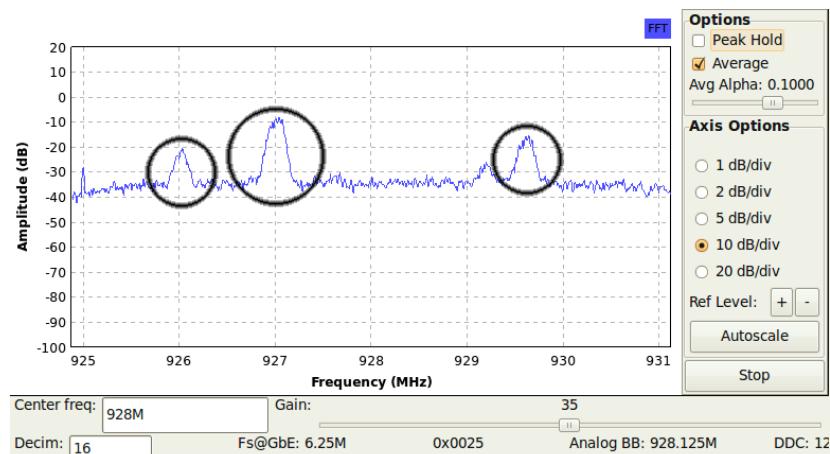


Abbildung 5.10: Frequenzanalyse

Die mit Hilfe der Spektrumsanalyse gefundenen Basisstationen können anschließend nacheinander gestört werden. Falls der IMSI-Catcher ein Vodafone-Netz vortäuscht, wäre es nicht notwendig, die Basisstationen von T-Mobile zu stören. Aus der Spektrumsanalyse ist allerdings nicht ersichtlich, um welche Basisstation es sich im konkreten Fall handelt. Um jedoch nicht alle gefundenen Basisstationen stören zu müssen, kann der Frequenznutzungsplan aus der Tabelle 5.3 verwendet werden.<sup>10</sup> Die Tabelle veranschaulicht, welche Anbieter in welchem Frequenzbereich senden und gibt einen Hinweis, in welchem Bereich gesucht und gestört werden muss. So darf beispielsweise Vodafone von 935,2 bis 937,4, 945,0 bis 951,0 und von 955,6 bis 959,2 MHz senden. Alle anderen Bereiche sind irrelevant und müssen weder betrachtet

<sup>10</sup> GSM 900 (mit GSM-R) Frequenznutzungsplan Provider DEUTSCHLAND. Sortiert nach Frequenz, <http://www.aaronia.de/Frequenzplan-GSM900.htm> [Online; letzter Aufruf 05.10.2009]

GSM 1800 Frequenznutzungsplan DEUTSCHLAND. Sortiert nach Frequenz, <http://www.aaronia.de/Frequenzplan-GSM1800.htm> [Online; letzter Aufruf 05.10.2009]

noch gestört werden. Eine andere Möglichkeit stellt das *Airprobe*<sup>11</sup> Projekt dar, welches auf einer gegebenen Frequenz die Daten mitschneidet, diese anschließend analysiert und einen Rückschluss auf den Netzanbieter ermöglicht.<sup>12</sup>

Name	von MHz	bis MHz	Name	von MHz	bis MHz
E-Plus	925,1	930,1	T-Mobile	1820,2	1825,0
O <sub>2</sub>	930,1	935,1	O <sub>2</sub>	1825,0	1847,4
Vodafone	935,2	937,4	Vodafone	1847,8	1853,0
T-Mobile	937,6	944,8	E-Plus	1853,2	1875,4
Vodafone	945,0	951,0			
T-Mobile	951,2	955,4			
Vodafone	955,6	959,2			
T-Mobile	959,4	959,8			

Tabelle 5.3: GSM900 und GSM1800 Frequenznutzungsplan, sortiert nach Frequenz

## Entwicklung

Ein Prototyp eines GSM-Jammers wurde mittels GRC (siehe Kapitel 4.1) für den USRP2 entwickelt, da das USRP1 für den Betrieb des IMSI-Catchers notwendig ist. Abbildung 5.11 verdeutlicht die Struktur des GSM-Jammers. Die „*Signal Source*“ liefert als Eingangsstream eine Cosinus-Welle. Als Input erwartet das USRP2 eine „*Sample Rate*“ von 100 Mega-Samples/Interpolation. Die Sample Rate oder auch Abtastrate genannt, gibt an, mit welcher Häufigkeit ein Signal abgetastet und in ein zeitdiskretes Signal umgewandelt wird. Für eine vollständige Abbildung eines Signals wird eine mindestens doppelt so hohe Abtastrate benötigt.<sup>13</sup> Da die maximale Frequenz („*var\_freq3*“) für das Störsignal 25 MHz beträgt, ist eine Interpolation von „2“ und somit eine Abtastrate von 50 MHz ausreichend. Als weiteren Input-Parameter erwartet das USRP2 eine Amplitude mit einem Wert zwischen 0 und 1.0. Die verschiedenen Parameter wurden auf experimentelle Weise ermittelt. Zur Steuerung des Jammers werden zwei „Slider“ bereitgestellt, mit dessen Hilfe die untere und obere Frequenz eingetragen werden kann, zwischen denen der Jammer stören soll. Sowohl das Pythonscript als auch die GRC-Datei sind auf der CD im Ordner *osic/GUI/jammer* enthalten.

## Verwendung des Jammers

Zunächst müssen mit Hilfe der Frequenzanalyse die zu störenden Frequenzen ermittelt werden. Da sich innerhalb des Frequenzbereiches mehrere Basisstationen befinden, müssen diese nacheinander, oder breitbandig gestört werden. Hierzu muss der IMSI-Catcher in Betrieb genommen und anschließend der Jammer innerhalb der GUI (siehe Abbildung 5.16) oder mit dem Kommando *sudo ./jammer.py* im Ordner *osic/GUI/jammer* gestartet werden. Innerhalb des Jammers (Abbildung 5.12) muss lediglich der zu störende Frequenzbereich eingetragen werden. Die Mobile Stations, die sich im Empfangsbereich des IMSI-Catchers befinden werden anschließend die Verbindung zu ihrer Basisstation verlieren und sich im Idealfall in den IMSI-Catcher einbuchen. Durch mehrere Versuche hat sich herausgestellt, dass die MS nach kurzer Zeit, ungefähr 30 Sekunden, die Verbindung verliert, weshalb eine Jammer-Betriebsdauer von

<sup>11</sup> Airprobe, <https://svn.berlin.ccc.de/projects/airprobe> [Online; letzter Aufruf 05.10.2009]

<sup>12</sup> Airprobe wird in der Masterarbeit von Holger Bertsch [2] näher erläutert.

<sup>13</sup> Abtastrate, <http://de.wikipedia.org/wiki/Abtastrate> [Online, letzter Aufruf 22.10.2009]

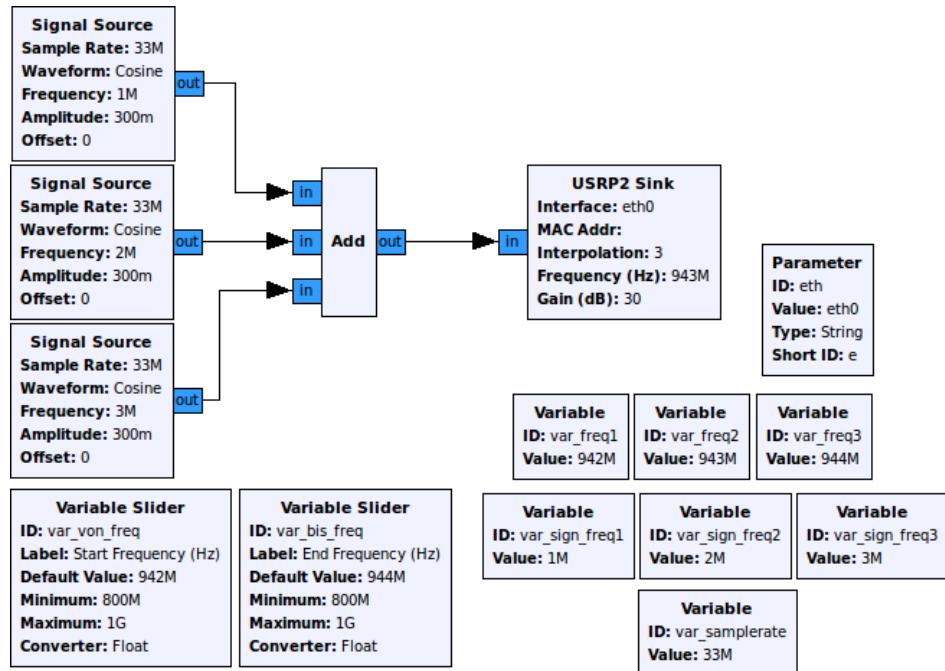


Abbildung 5.11: GSM-Jammer, entwickelt mit GRC

wenigen Minuten pro Basisstation ausreichend ist. Da die MS sich allerdings von einer Basisstation zur anderen registrieren kann und neue Mobile Stations hinzukommen, sollten die Basisstationen periodisch bzw. dauerhaft gestört werden. Aus diesem Grund muss der IMSI-Catcher auf einer Frequenz, die weit genug entfernt ist, senden.

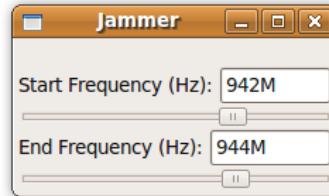


Abbildung 5.12: Jammer

## 5.5 Rufaufbau

Da sich die MS beim Einbuchen in den IMSI-Catcher im Originalnetz ausbucht, ist sie nicht über ihre Telefonnummer erreichbar. Den Mobile Stations, die im IMSI-Catcher eingebucht sind, ist es daher nur möglich, ein ausgehendes Gespräch zu führen. Eingehende Gespräche wären nur möglich, wenn der IMSI-Catcher sich im Originalnetz einbuchen würde und die Identität der MS vortäuschen würde. Da auf Grund der Authentifizierung mit Hilfe des privaten Schlüssels Ki dies nicht möglich ist, ist ein eingehender Anruf ausgeschlossen. Der Rufaufbau eines Standard IMSI-Catchers unterscheidet sich nicht von der GSM-Spezifikation, da er die Daten der MS ausschließlich weiterleitet (dargestellt in Abbildung 5.5). Deshalb wird lediglich ein Vergleich zwischen dem Rufaufbau im GSM-Netz und beim IMSI-Catcher aufgezeigt.

### 5.5.1 GSM-Rufaufbau

Beim ausgehenden Rufaufbau (Abbildung 5.13) – auch Mobile Originated Call Establishment genannt – signalisiert die MS dem MSC den gewünschten Rufaufbau mittels *CM-Service Request*. Das MSC leitet die Anfrage weiter, dass die MS mit der gegebenen TMSI innerhalb der LAI telefonieren möchte und die Authentifizierung gestartet werden soll. Die Authentifizierung und Verschlüsselung geschieht wie bei der Registrierung (siehe Abschnitt 5.4). Die BTS überträgt anschließend ein *CM-Service Accept*. Bisher wurde die Telefonnummer noch nicht übertragen, die angerufen werden soll. Erst nach erfolgreicher Verschlüsselung wird der Zielteilnehmer mitgeteilt (*Setup*). Das MSC teilt der MS den Rufaufbau mit und reserviert einen Kanal (*Assign command*), welchen die MS bestätigt. Der gewünschte Rufaufbau wird durch eine IAM-Nachricht innerhalb des Netzes weitergeleitet und, falls der Teilnehmer erreichbar ist, mittels ACM-Nachricht beantwortet. In diesem Fall wird der MS der erfolgreiche Rufaufbau signalisiert. Sobald der gewünschte Teilnehmer abnimmt, wird die Verbindung aufgebaut.

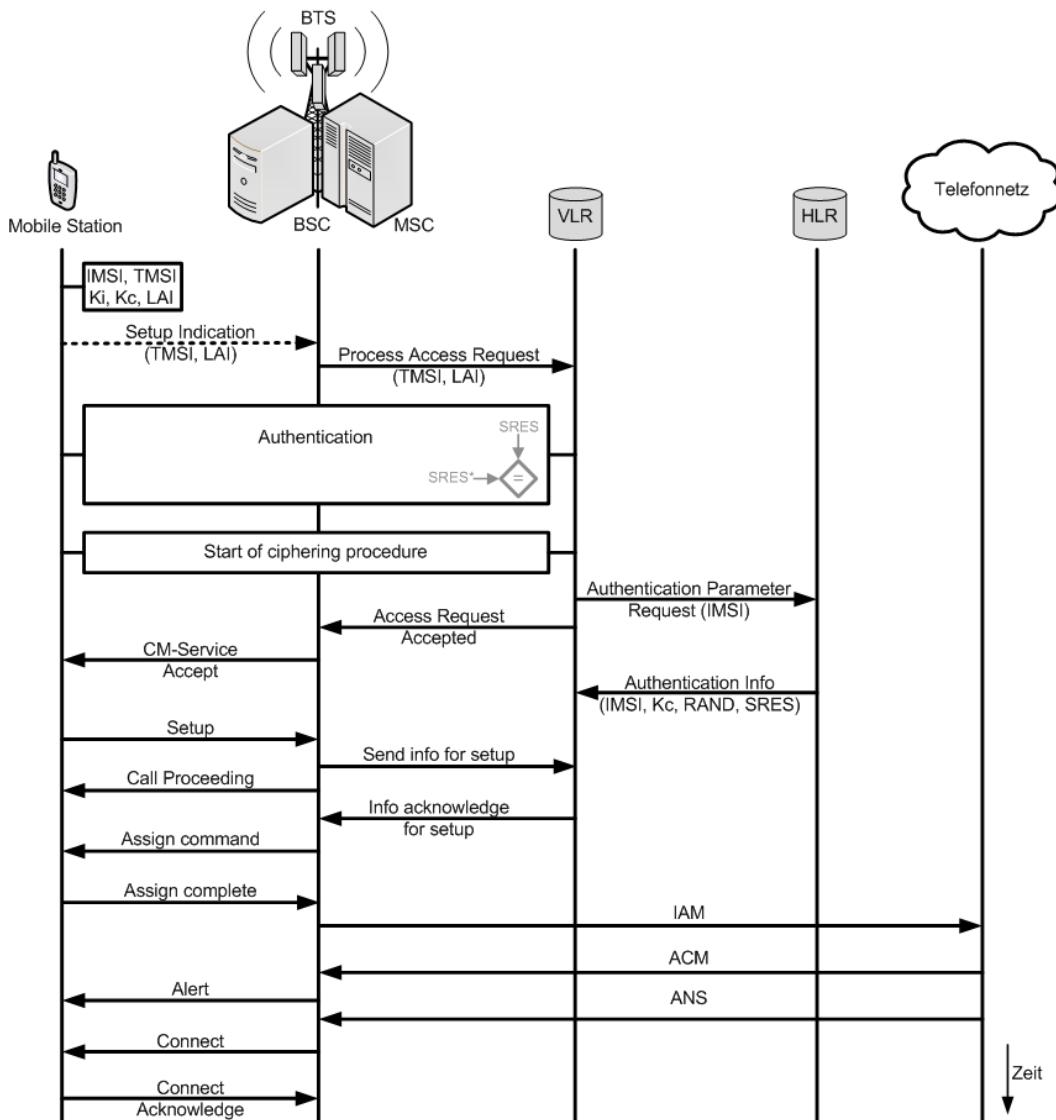


Abbildung 5.13: Ablauf ausgehender Rufaufbau bei GSM (Vorlage nach [13])

### 5.5.2 IMSI-Catcher-Rufaufbau

Der Rufaufbau beim IMSI-Catcher läuft im Wesentlichen nach GSM-Spezifikation ab, mit dem Unterschied der Authentifizierung und der fehlenden Verschlüsselung (siehe Abbildung 5.14). Zunächst wird ebenfalls der gewünschte Rufaufbau signalisiert. Falls die „*Very Early Assignment*“ Option in der *OpenBTS.conf* eingeschalten ist, kann der MS schon hier ein fester Kanal zugewiesen werden. Die Kommunikation verschiebt sich daher vom „langsamem“ SDCCH zum TCH/FACCH. Nachteilig hierbei ist, dass ein TCH belegt wird, bevor der Verbindungsaufbauversuch erfolgreich ist, und daher der Funkkanal unnötigerweise belegt wird. Die von der MS übertragene Setup Anweisung bewirkt einen regulären SIP-Ablauf. Zunächst wird eine „*INVITE*“ SIP-Message an den Asterisk-Server gesendet. Der MS wird der Rufaufbau mitgeteilt und falls noch nicht erfolgt ein Kanal zugewiesen. Bei erfolgreichem Rufaufbau wird eine „*180 Ringing*“ SIP-Message generiert und der MS signalisiert. Sobald der Teilnehmer den Anruf annimmt, wird eine „*200 OK*“ SIP-Message erstellt und die Verbindung mit der MS hergestellt, die den Aufbau bestätigt.

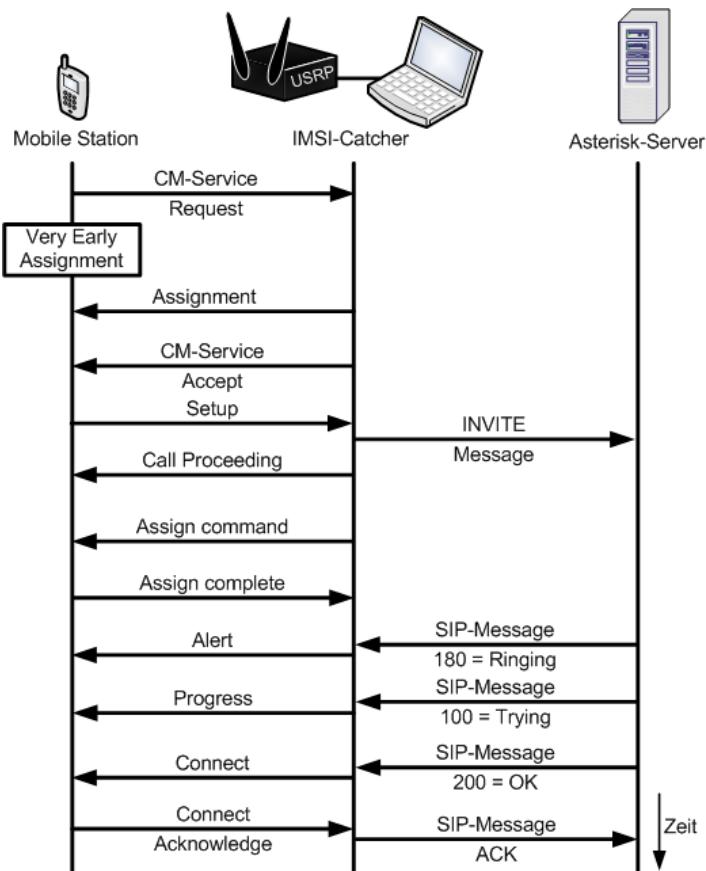


Abbildung 5.14: Ablauf ausgehender Rufaufbau beim IMSI-Catcher

## 5.6 Änderungen

Der IMSI-Catcher basiert auf dem Code von OpenBTS, der sich noch in der Entwicklung befindet. Auf Grund der anderen Zielsetzung von OpenBTS werden nicht alle Funktionen bereitgestellt, die für den Betrieb des IMSI-Catchers notwendig sind. Da der IMSI-Catcher jedoch auch eine Basisstation simuliert, können mit Änderungen am OpenBTS Code die not-

wendigen Funktionen zur Verfügung gestellt werden. In diesem Abschnitt werden lediglich die Änderungen am bestehenden Code und ihre Notwendigkeit aufgezeigt. Der selbst entwickelte Code für den IMSI-Catcher und der grafischen Oberfläche kann auf der CD im Ordner *osic/GUI* eingesehen werden.

### 5.6.1 SIP-Account erstellen

Bei OpenBTS wird beim Einbuchungsvorgang zunächst geprüft, ob der Teilnehmer mit der IMSI als Benutzername sich beim Asterisk-Server anmelden kann. Wenn dies nicht möglich ist, entweder weil kein Account vorhanden ist oder weil Asterisk in einer gegebenen Zeitspanne nicht erreichbar ist und somit der SIPTimeout Fehler geworfen wird, wird geprüft, ob die Option „OpenRegistration“ aktiviert ist. Ist dies nicht der Fall, wird die Mobile Station abgewiesen. Ist die Option aktiviert, wird die Mobile Station in OpenBTS registriert. Der Benutzer besitzt im Asterisk-Server jedoch keinen eigenen Account und kann daher dort nicht eindeutig zugeordnet oder bestimmten Wahlregeln zugewiesen werden. Da eine Zuordnung und somit eine Klassifizierung des Benutzers anhand seiner IMSI zwingend notwendig für den IMSI-Catcher zwecks gerichteter Kommunikation ist, muss eine weitere Abfrage in die *MobilityManagement.cpp* im Ordner *osic/Control/* eingebaut werden. Für eine leichtere Ein- und Abschaltung dieser Funktionalität wurde in der *OpenBTS.conf* eine neue Variable hinzugefügt:

```
# Create a SIP-Account if there is no SIP-Account for the user
Control.CreateSIP 1
```

Falls die Erstellung eines SIP-Accounts für den unbekannten Benutzer nicht gewünscht ist, muss die Variable auf 0 gesetzt werden. Der dahingehend veränderte Registrierungsprozess innerhalb der *MobilityManagement.cpp*:

Listing 5.1: SIP-Account erstellen, innerhalb der MobilityManagement.cpp

```
1 void Control::LocationUpdatingController(const L3LocationUpgradingRequest* lur,
                                           SDCCHLogicalChannel* SDCCH)
2 {
3     :
4     // Create a SIP account if necessary
5     bool createSIPaccount = false;
6     if(gConfig.defines ("Control.CreateSIP")){
7         createSIPaccount = gConfig.getNum ("Control.CreateSIP");
8     }
9
10    // Failed to register IMSI and openRegistration is on
11    // => create SIP Account and register again
12    if(!success && openRegistration){
13        try {
14            SIPEngine engine;
15            if(createSIPaccount){
16                LOG(INFO) << "Create a SIP Account for IMSI:" << mobID.digits();
17                createSIP(mobID.digits());
18            }
19            engine.User(mobID.digits());
20            LOG(DEBUG) << "LocationUpdatingController waiting for registration";
21            success = engine.Register();
22        }
23        catch(SIPTimeout) {
24            LOG(ALARM) "SIP registration timed out. Is Asterisk running?";
25            // Reject with a "network failure" cause code, 0x11.
26            SDCCH->send(L3LocationUpgradingReject(0x11));
27            // Release the channel and return.
```

```

28     SDCCH->send( L3ChannelRelease() );
29     return;
30 }
31 }
32 :
33 }
```

Die Abbildung 5.15 veranschaulicht den veränderten Registrierungsprozess. Es wird zunächst geprüft, ob eine vorangegangene Registrierung auf Grund eines nicht vorhandenen SIP-Accounts fehlgeschlagen ist und ob „Open Registration“ aktiviert ist. Ist dies der Fall, wird mit Hilfe der *createSIP(mobID.digits())* Funktion ein SIP-Account in der *sip.conf* angelegt und die *sip.conf* mit dem Kommando *sip reload* in den Asterisk-Server geladen. Das Kommando lädt jedoch nur neue Accounts in den Asterisk-Server und stört nicht den laufenden Betrieb. Anschließend wird mit *engine.User(mobID.digits())* der Registrierungsprozess erneut eingeleitet. Die Implementierung der *createSIP*-Funktion kann in Anhang C.1 eingesehen werden.

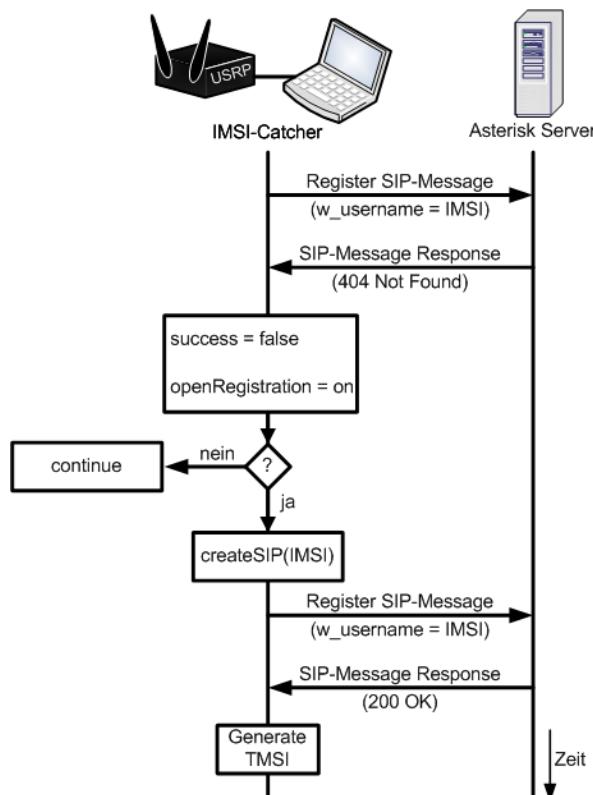


Abbildung 5.15: Geänderter Registrierungsablauf im Vergleich zu Abbildung 5.6

### 5.6.2 Selektive Registrierung

Die automatische Erstellung eines SIP-Accounts für einen unbekannten Benutzer und die von OpenBTS bereitgestellte „OpenRegistration“ löst das Problem der Authentifizierung und der direkten Kommunikation. Für den korrekten Betrieb eines IMSI-Catchers ist es aber unerlässlich, dass sich ausschließlich MS's in das vorgetäuschte Netz einbuchen können, die die erforderliche Netzberechtigung besitzen. Dies bedeutet, falls der IMSI-Catcher beispielsweise ein D1 Netz vortäuscht, dass sich lediglich die MS's einbuchen dürfen, die eine SIM-Karte

dieses Netzanbieters besitzen. Bisher ist es für jede MS möglich, sich manuell oder bei fehlender Verbindung zum Heimnetz auf Grund der offenen Registrierung automatisch in den IMSI-Catcher einzubuchen, obwohl ein fremder Netzanbieter vorgetäuscht wird. Der MS wird somit ein falsches Netz angezeigt, weshalb der IMSI-Catcher leicht zu identifizieren ist. Für die korrekte Zuordnung der MS zu ihrem Heimnetz kann die IMSI herangezogen werden. Die Ziffern vier und fünf (siehe Abschnitt 5.2.1) der IMSI geben deren Netzanbieter an. Aus diesem Grund müssen diese Ziffern überprüft und daraufhin entschieden werden, ob die IMSI sich in das Netz einbuchen darf. Um diese Funktionalität bereitzustellen, bedarf es wiederum Änderungen innerhalb der *MobilityManagement.cpp* und der Definition einer neuen Variable in der *OpenBTS.conf*:

```
# Accept only IMSI's that belong to a particular provider (GSM.MNC)
Control.AcceptOnlyMNC 1
```

Listing 5.2: Selective Registrierung, innerhalb der MobilityManagement.cpp

```
1 void Control::LocationUpdatingController(const L3LocationUpgradingRequest* lur,
                                           SDCCHLogicalChannel* SDCCH)
2 {
3     :
4     // Accept only IMSI's that belong to a particular provider
5     bool acceptOnlyMNC = false;
6     if (gConfig.defines ("Control.AcceptOnlyMNC")) {
7         acceptOnlyMNC = gConfig.getNum("Control.AcceptOnlyMNC");
8     }
9     :
10    :
11    // Failed to register IMSI and openRegistration is on
12    // and accept only IMSI's that belong to a particular provider
13    // => create SIP Account and register again
14    if (!success && openRegistration && acceptOnlyMNC) {
15        if(acceptIMSI(mobID.digits()))
16        {
17            :
18        }
19        else{
20            // Reject with a "network failure" cause code, 0x11.
21            SDCCH->send(L3LocationUpgradingReject(0x11));
22            // Release the channel and return.
23            SDCCH->send(L3ChannelRelease());
24            return;
25        }
26    }
27    :
28 }
```

Zunächst wird überprüft, ob die Variable *Control.AcceptOnlyMNC* definiert wurde und welchen Wert sie besitzt. Anschließend wird innerhalb der IF-Bedingung geprüft, ob lediglich IMSI's akzeptiert werden sollen, die gleich dem vorgetäuschten MNC sind. Die Funktion *acceptIMSI(mobID.digits())* ermittelt anschließend, ob die IMSI gleich dem MNC ist. Ist dem nicht so, wird der MS ein „network failure“ Code gesendet und der Kanal geschlossen. Im Erfolgsfall wird die Registrierung wie in Abschnitt 5.6.1 fortgeführt. Die Implementierung der *acceptIMSI* Funktion mit der IMSI als Übergabeparameter kann dem Anhang C.2 entnommen werden.

### 5.6.3 IMEI auslesen

Von Interesse ist auch die IMEI-Nummer, mit der die MS bei einem SIM-Kartenwechsel einem Teilnehmer eindeutig zugeordnet werden kann. Auch ist es damit möglich, Teilnehmer die einen Dual-SIM<sup>14</sup> Adapter oder fähige MS besitzen zu identifizieren. Bei OpenBTS ist die Abfrage der IMEI-Nummer ebenfalls nicht vorgesehen. Aus diesem Grund, musste der Code dementsprechend angepasst werden. Die Abfrage wird während dem Registrierungsprozess, nachdem die IMSI angefragt wurde, an die MS übertragen. Dies erfolgt wie bei der Erstellung eines SIP-Accounts innerhalb der *MobilityManagement.cpp*:

Listing 5.3: Start der IMEI-Abfrage, innerhalb der MobilityManagement.cpp

```

1 void Control::LocationUpdatingController(const L3LocationUpdatingRequest* lur,
                                           SDCCHLogicalChannel* SDCCH)
2 {
3     :
4     // Resolve an IMSI and see if there's a pre-existing IMSI-TMSI mapping.
5     L3MobileIdentity mobID = lur->mobileIdentity();
6     bool sameLAI = (lur->LAI() == gBTS.LAI());
7     // Ask for IMSI
8     unsigned assignedTMSI = resolveIMSI(sameLAI, mobID, SDCCH);
9     // Ask for IMEI
10    L3MobileIdentity IMEI = resolveIMEI(mobID, SDCCH);
11
12    :
13    if (assignedTMSI) SDCCH->send(L3LocationUpdatingAccept(gBTS.LAI()));
14    else {SDCCH->send(L3LocationUpdatingAccept(gBTS.LAI(), gTMSITable.assign(mobID.
15        digits(), IMEI.digits())));}
16 }
```

Die Abfrage der IMEI erfolgt in Zeile 10. Hierfür wurde innerhalb der *ControlCommon.cpp* eine *resolveIMEI(L3MobileIdentity, Channel)*-Funktion implementiert, dessen Code im Anhang C.3 eingesehen werden kann. Die Antwort wird in der *IMEI*-Variable abgelegt, die im späteren Verlauf der IMSI zugeordnet und innerhalb einer TMSITable<sup>15</sup> gespeichert wird (Zeile 14). Für die Speicherung der IMEI war es ebenfalls notwendig innerhalb der TMSITable-Klasse eine neue TMSI/IMEI Tabelle („mIMEImap“) zu erstellen. Die Zuweisung erfolgt über die Funktion TMSITable::assign(const char\* IMSI), die hierfür überladen werden musste.

Listing 5.4: Auszug der TMSITable-Klasse innerhalb der ControlCommon.h

```

1 class TMSITable {
2 private:
3     TMSIMap mIMEImap;           ///< TMSI/IMEI mapping
4 public:
5     // Overloaded function. This assigns the IMSI and IMEI and returns the TMSI
6     unsigned assign(const char* IMSI, const char* IMEI);
7     const char* findIMEI(unsigned TMSI) const;
8     TMSIMap::const_iterator ibegin() const { return mIMEImap.begin(); }
9     TMSIMap::const_iterator iend() const { return mIMEImap.end(); }
10 }
```

<sup>14</sup> Mit einem Dual-SIM Adapter könnten zwei SIM-Karten mit einer Sende- und Empfangseinheit betrieben werden. Die SIM-Karten können allerdings nicht gleichzeitig genutzt werden. Bei Dual-SIM fähigen Mobile Stations können die SIM-Karten simultan genutzt werden, da dies zwei Sende- und Empfangseinrichtungen besitzen.

<sup>15</sup> Die TMSITable ist eine Tabelle, in der mit Hilfe von *std::map* die IMSI einer eindeutigen TMSI zugeordnet wird.

Listing 5.5: Zuordnung TMSI - IMEI und die *findIMEI*-Hilfsfunktion, innerhalb der ControlCommon.cpp

```

1 // Create TMSI and assign IMSI and IMEI to the same TMSI
2 unsigned TMSITable::assign(const char* IMSI, const char* IMEI)
3 {
4     // delete last Entry, if TMSITable is full
5     purge();
6     mLock.lock();
7     unsigned TMSI = mCounter++;
8     // TMSI - IMSI pair
9     mMap[TMSI] = IMSI;
10    // TMSI - IMEI pair
11    mIMEImap[TMSI] = IMEI;
12    mLock.unlock();
13    return TMSI;
14 }
15
16 // Return IMEI-Number
17 const char* TMSITable::findIMEI(unsigned TMSI) const
18 {
19     mLock.lock();
20     TMSIMap::const_iterator iter = mIMEImap.find(TMSI);
21     mLock.unlock();
22     if (iter==mIMEImap.end()) return NULL;
23     return iter->second.c_str();
24 }
```

Mit der *assign*-Funktion werden sowohl die IMSI, als auch die IMEI in ihre jeweilige Tabelle (mMap bzw. mIMEImap) abgespeichert. Hierbei werden die Daten in einen standard C++-Container („`std::map`“) abgelegt. Dieser besteht aus jeweils zwei Elementen, einen eindeutigen Schlüssel (TMSI) und ihrem zugeordneten Wert (IMSI). Aus diesem Grund war es nicht möglich die IMEI in den gleichen Container zu speichern. Entsprechend wurde ein eigener TMSI/IMEI-Container erstellt, der die selbe TMSI enthält. Hiermit wird die Zuordnung der IMEI zu einer IMSI ermöglicht. Um für eine gegebene IMSI die IMEI auszulesen muss die *findIMEI(unsigned TMSI)*-Funktion genutzt werden, die als Eingabeparameter die TMSI benötigt und die IMEI als Antwort liefert. Für die Speicherung der IMEI innerhalb der *sip.conf* musste die *createSIP*-Funktion ebenfalls verändert werden. Sie benötigt nun zwei Parameter, die IMSI und die IMEI. Dahingehend wurde der Aufruf innerhalb der *MobilityManagement.cpp* von *createSIP(mobID.digits())* zu *createSIP(mobID.digits(), IMEI.digits())* geändert. Die modifizierte Variante kann ebenfalls dem Anhang C.1 entnommen werden.

## 5.7 OpenBTS als IMSI-Catcher

Als Basis für den IMSI-Catcher dient der Code von OpenBTS, das über ein CLI-Interface<sup>16</sup> bedient wird. Dieser Code wurde für den IMSI-Catcher angepasst und um notwendige sowie zusätzliche Funktionen ergänzt. Insgesamt können auf Grund der Softwarearchitektur von OpenBTS maximal sieben Teilnehmer gleichzeitig ein Gespräch führen. Die Beschränkung ergibt sich, da OpenBTS lediglich einen Kanal (Frequenz) nutzt und von den acht verfügbaren Zeitschlitten einer für den Broadcast (vgl. Logische Kanäle, Kapitel 2.4.2) benötigt wird. Es ist allerdings geplant, dass in einer späteren Version mehrere Frequenzen genutzt werden können. Für eine komfortable Bedienung des IMSI-Catchers wurde überdies eine grafische Benutzeroberfläche entwickelt, dargestellt in Abbildung 5.16. Der Code für den IMSI-Catcher und den in Abschnitt 5.4.5 beschriebenen Jammer befindet sich im Ordner *osic/GUI*.

<sup>16</sup> Das Programm wird innerhalb einer Konsole über ein Command-line Interface (Kommandozeile) gesteuert.

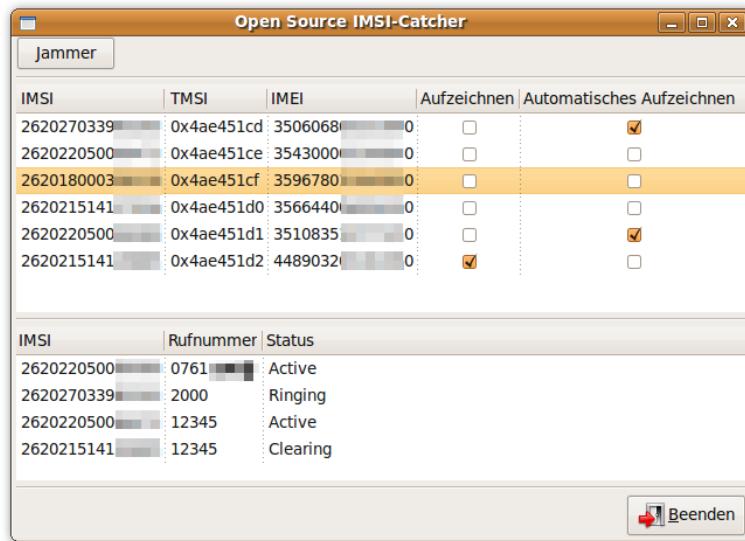


Abbildung 5.16: Grafische Oberfläche des IMSI-Catchers

### 5.7.1 Funktionsumfang

Die grafische Oberfläche wurde bewusst übersichtlich und funktional gehalten, um eine einfache Bedienung zu ermöglichen. Bei Bedarf können erweiterte Funktionen, wie eine Unterscheidung zwischen aktiv eingebuchten und inaktiven Teilnehmern, nachträglich hinzugefügt werden. Die eingebuchten Mobile Stations werden in tabellarischer Form, sortiert nach Einbuchungszeitpunkt, aufgelistet. Da jeder Teilnehmer einen Account für den Asterisk-Server besitzen muss, werden alle bekannten IMSI's in der *sip.conf* protokolliert und müssen nicht separat gespeichert werden. Dies trifft auch auf die angefragte IMEI zu, die unterhalb des Benutzernamens als Kommentar in die *sip.conf* abgespeichert wird. Die IMEI-Nummer ist ebenfalls 15-stellig, wobei die letzte Ziffer eine Prüfzahl ist, die nur in der MS gespeichert und nicht über das Netz übertragen wird.<sup>17</sup> Aus diesem Grund wird beim IMSI-Catcher jeweils eine „0“ als letzte Zahl angezeigt. Ebenfalls aufgelistet wird die während des Registrierungsprozesses zugewiesene TMSI. Sofern eine MS ein ausgehendes Gespräch führt, kann das Gespräch manuell aufgezeichnet werden. Hierzu muss lediglich der Haken zum Aufzeichnen der gewünschten IMSI gesetzt werden. Zum Beenden der Aufzeichnung muss der Haken wieder entfernt werden. Der Start der Aufzeichnung funktioniert allerdings nur, falls tatsächlich ein Gespräch stattfindet. Sollte kein Gespräch stattfinden, erscheint ein Fenster mit einer Fehlermeldung, dass die Aufzeichnung nicht gestartet werden konnte. Somit muss der richtige Zeitpunkt zum Aufzeichnen des Gespräches gewählt werden. Aus diesem Grund werden zwei Funktionen bereitgestellt. So wird eine Gesprächsübersicht zur Verfügung gestellt, die eine Zuordnung der IMSI und der Gespräche ermöglicht, sowie die Zielrufnummer und den aktuellen Status anzeigt. Hierdurch kann anschließend ein aktives Gespräch gezielt aufgezeichnet werden. Eine andere Möglichkeit ist eine erweiterte Aufzeichnungsfunktion, die – sofern gewünscht – die Gespräche automatisch von Beginn an aufzeichnet. Diese Option wird ebenfalls für jede IMSI separat eingestellt, indem der Haken gesetzt wird. Die Aufzeichnungen befinden sich anschließend im Wave-Format im *osic/GUI/asterisk* Ordner und werden im Format „IMSI\_Datum\_Zeit.wav“ benannt. So steht beispielsweise der Dateiname 123456789012345\_30.10.2009\_14:15:20.wav dafür, dass die MS mit der IMSI „123456789012345“ am 30.10.2009 um 14:15 Uhr (und 20

<sup>17</sup> International Mobile Equipment Identity, [http://de.wikipedia.org/wiki/International\\_Mobile\\_Equipment\\_Identity](http://de.wikipedia.org/wiki/International_Mobile_Equipment_Identity) [Online; letzter Aufruf 25.10.2009]

Sekunden) ein Gespräch gestartet hat. Der Ablauf des Gesprächsaufbaus, wurde in Abschnitt 5.5.2 beschrieben. Die MS merkt nicht, dass das Gespräch über den IMSI-Catcher geführt und aufgezeichnet wird. Dieses Szenario entspricht dem klassischen Men-in-the-middle Angriff, bei dem weder Teilnehmer A noch B Kenntnis vom IMSI-Catcher hat.<sup>18</sup> Die Abbildung 5.17 veranschaulicht dieses Szenario.

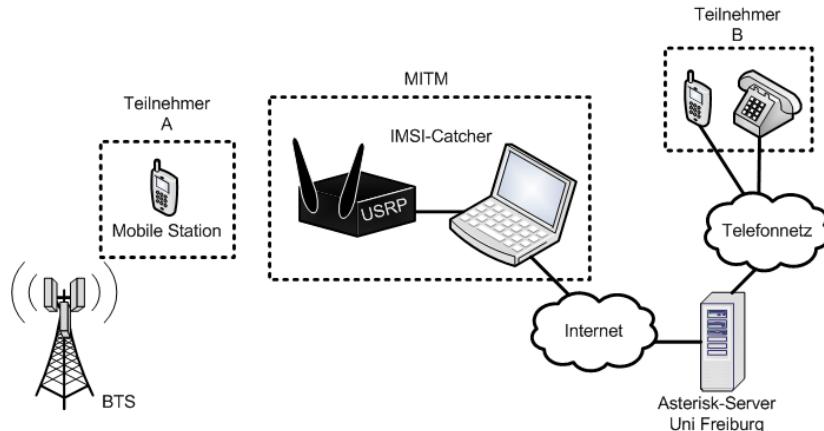


Abbildung 5.17: Typisches MITM Szenario

### 5.7.2 Aufnahme

Die Aufzeichnung der Gespräche erfolgt über den Asterisk-Server. Für die manuelle Aufzeichnung müssen zum Starten der Aufzeichnung und zum Beenden zwei Kommandos an den Asterisk-Server geschickt werden.

```
1 asterisk -r -x "mixmonitor" start "SIP/IMSI /PATH/osic/GUI/asterisk/IMSI_getTime().wav"
2 asterisk -r -x "mixmonitor" stop "SIP/IMSI"
3 chmod 744 /PATH/osic/GUI/asterisk/IMSI*
```

Der erste Befehl (Zeile 1) bewirkt, dass die Aufzeichnung im *osic/GUI/asterisk* Ordner gespeichert wird. Sobald die Aufzeichnung beendet werden soll, sorgt der zweite und dritte Befehl dafür, dass die Aufzeichnung beendet wird und die Zugriffsrechte der Datei dahingehend geändert werden, dass der Benutzer die notwendigen Leserechte bekommt. Dies ist notwendig, da der Prozess vom Asterisk-Server gesteuert und die Datei unter der Gruppe:Benutzer „*asterisk:asterisk*“ angelegt wird. Aus diesem Grund muss der Asterisk-Server Zugriffsrechte auf den *osic/GUI/asterisk* Ordner besitzen.<sup>19</sup>

Im Gegensatz zur manuellen Aufzeichnung müssen dem Asterisk-Server bei der automatischen Aufzeichnung keine Kommandos vom IMSI-Catcher geschickt werden. Die automatische Aufzeichnung geschieht über einen definierten Kontext. Hierfür wurde in der *extensions.conf* ein Kontext „*monitor*“ erstellt, der alle Gespräche automatisch von Beginn an aufzeichnet. Alle anderen Teilnehmer-Accounts befinden sich im Kontext „*default*“, in dem keine Aufzeichnung der Gespräche erfolgt. Beim Erstellen der Teilnehmer-Accounts (siehe Abschnitt 5.6.1) werden alle Benutzer dem „*default*“ Kontext zugeordnet. Falls die automatische Aufzeichnung

<sup>18</sup> Bei einem MITM-Angriff steht ein Angreifer zwischen zwei Kommunikationspartnern und hat dabei die vollständige Kontrolle über die gesendeten Daten. Der Angreifer täuscht dem jeweiligen Kommunikationspartner die Identität des anderen vor, so dass beide keine Kenntnis über den Angreifer haben.

<sup>19</sup> Falls erforderlich sollte für die notwendigen Rechte ein *chmod 766* auf den Ordner ausgeführt werden.

für einen Teilnehmer aktiviert wird, wird die IMSI vom „*default*“ in den „*monitor*“ Kontext verschoben. Alle Kontextänderungen der Teilnehmer werden persistent gespeichert. Sollte eine bekannte MS sich in den IMSI-Catcher einbuchen, wird ihr Kontext ausgelesen, und falls vorher aktiviert, die Gespräche automatisch aufgezeichnet. Listing 5.6 verdeutlicht die zwei Kontexte und die zugehörigen Wahl- sowie Aufnahmeregeln.

Listing 5.6: Der *default* und *monitor* Kontext in der extensions.conf

```

1 [ default ]
2 exten => _0.,1,Dial(SIP/10${EXTEN}@comsys02)
3 exten => 12345,1,Answer(1500)
4 exten => 12345,2,SayDigits(${CALLERID(num)})
5 exten => 12345,3,Hangup
6
7 [ monitor ]
8 exten => _0.,1,MixMonitor(${CALLERID(num)} ${STRFTIME(${EPOCH},Europe/Berlin,%d.%m
    .%Y_%H:%M:%S)}.wav,,chmod 755 /var/spool/asterisk/monitor/* && mv /var/spool/
    asterisk/monitor/* /PATH/osic/GUI/asterisk/.)
9 exten => _0.,2,Dial(SIP/10${EXTEN}@comsys02)
10 exten => 12345,1,Answer(1500)
11 exten => 12345,2,SayDigits(${CALLERID(num)})
12 exten => 12345,3,Hangup

```

Die Zeile 8 veranlasst den Asterisk-Server, sofern der Teilnehmer anrufen möchte, als ersten Schritt die Aufzeichnung zu starten und dann den Anruf unter dem Benutzer *comsys02*, der ein Gateway in das Telefonnetz besitzt, weiterzuleiten (Zeile 9). Hierbei muss die gewählte Rufnummer mit einer „0“ anfangen. Die Aufzeichnung wird unter demselben Format wie bei der manuellen Registrierung (IMSI\_Datum\_Zeit.wav) gespeichert. Die IMSI wird aus der *CALLERID* ausgelesen und das Datum und die Zeit mit Hilfe der *STRFTIME*-Funktion abgerufen und entsprechend formatiert. Die Datei befindet sich anschließend im Ordner */var/spool/asterisk/monitor/* und muss nach Beendigung der Aufzeichnung verschoben werden. Für den Zugriff auf die Datei müssen, wie bei der manuellen Aufzeichnung, die Zugriffsrechte ebenfalls geändert werden.

### 5.7.3 Inbetriebnahme

Zunächst müssen die korrekten Einstellungen zum Vortäuschen eines GSM-Netzes in der *OpenBTS.conf* vorgenommen werden. Wichtige Einstellungen sind hierbei spezifische GSM- und Kontroll-Parameter. Alle anderen Einstellungsparameter wie beispielsweise Kommunikationsports und Loglevel-Tiefe bzw. Pfad müssen nur bei Bedarf geändert werden. Die Werte für den GSM.MCC, GSM.MNC, GSM.LAC und GSM.CI können in Abschnitt 5.3.1 eingesehen bzw. mit Hilfe der Netzanalyse in Abschnitt 5.4.5 bestimmt werden. Falls zur Registrierung die Methode des erzwungenen Zellwechsels (Abschnitt 5.4.4) genutzt wird, muss ein GSM.ARFCN Wert eingestellt werden, dessen Kanal sich in der Nachbarschafstliste befindet. Die Sendestärke des IMSI-Catchers kann über den GSM.PowerAttenDB geregelt werden. Hierbei ist der Maximalwert „23“, was 200 mW entspricht. Für eine korrekte Funktionsweise des IMSI-Catchers müssen die drei Kontrollparameter Control.OpenRegistration, Control.AcceptOnlyMNC und Control.CreateSIP wie in Abschnitt 5.6 beschrieben, aktiviert werden. Mit folgenden Werten kann der IMSI-Catcher innerhalb des GSM900 Frequenzband betrieben und z.B. ein Vodafone Netz vorgetäuscht werden:

```

Control.OpenRegistration 1
Control.AcceptOnlyMNC 1
Control.CreateSIP 1

```

#D2 – Vodafone

GSM.MCC 262  
GSM.MNC 02  
GSM.LAC 793

GSM.CI 666  
GSM.ARFCN 69  
GSM.ShortName Vodafone

GSM.Neighbors 69  
GSM.PowerAttenDB 20  
GSM.T3212 6

Falls als LAC der selbe Code verwendet wird, der auch von den umliegenden Basisstationen genutzt wird und die MS durch den erzwungenen Zellwechsel oder den Jammer dazu gebracht wird sich in den IMSI-Catcher einzubuchen, ergibt sich daraus ein Problem. Der T3212 Wert gibt an, in welchen Vielfachen von sechs Minuten ein Location Update erfolgen muss. Das heißt, dass die MS jede sechs Minuten prüft, ob sie ein Location Update machen muss. Da sich die MS bei beiden Szenarien lediglich in eine andere Zelle innerhalb der gleichen Location Area einbucht, führt sie kein Location Update aus. Der T3212 Wert ist von Provider zu Provider unterschiedlich. So ist beispielsweise im Vodafone Netz ein T3212 Wert von zehn eingestellt, was bedeutet, dass alle 60 Minuten ein Location Update ausgeführt werden muss. Beim IMSI-Catcher ist ein Wert von sechs Minuten eingestellt. Der interne Timer der MS läuft jedoch beim Zellwechsel weiter. Dies bedeutet: Falls drei Minuten seit dem letzten Überprüfen vergangen sind und sich die MS in den IMSI-Catcher einbucht, überprüft die MS nach weiteren drei Minuten, ob sie ein Location Update durchführen muss. Auf Grund der getroffenen Einstellungen im IMSI-Catcher muss die MS ein Location Update ausführen. Das Problem ist allerdings, dass der IMSI-Catcher die MS nur erkennt, wenn die MS ein Location Update macht. Bis zu diesem Zeitpunkt ist dem IMSI-Catcher nicht bekannt, dass die MS eingebucht ist. Maximal muss somit eine Dauer von sechs Minuten gewartet werden, bis die MS erkannt wird. Um dieses Problem zu umgehen, kann ein LAC Wert gewählt werden, der sich vom aktuellen Ort unterscheidet. Laut der GSM-Spezifikation muss bei einem Zellwechsel mit einer anderen Location Area sofort ein Location Update ausgeführt werden.[8] Ebenfalls beachtet werden muss, dass nicht die selbe CellID (GSM.CI) einer Basisstation in näherer Umgebung verwendet werden darf, da die Mobile Station die Basisstationen in diesem Fall nicht unterscheiden kann.

Nachdem die richtigen Konfigurationsparameter eingestellt sind, muss der IMSI-Catcher mit dem Kommando `sudo ./GUI` im Ordner `osic/GUI` gestartet werden. Der Jammer befindet sich im Unterordner `osic/GUI/jammer` und kann über die GUI oder bei Bedarf mit dem Kommando `sudo ./jammer` gestartet werden.

## 6 Fazit und Ausblick

Ziel dieser Arbeit war es, die exakte Funktionsweise eines IMSI-Catchers zu demonstrieren. Diese Geräte existieren eigentlich schon länger, waren aber für die Allgemeinheit nicht verfügbar. Wie diese Arbeit allerdings zeigt, ist der Bau eines eigenen IMSI-Catchers mit geringem finanziellen Aufwand möglich. Im Vordergrund stand, mit Hilfe des Universal Software Radios einen prototypischen IMSI-Catcher zu entwickeln, anhand dessen die Verwundbarkeit von GSM demonstriert werden sollte. So konnte veranschaulicht werden, dass der Teilnehmer nicht bemerkt, dass er sich in den IMSI-Catcher einbucht. Lediglich das Siemens S55 und die modifizierten Nokia 3310 und 3210 sind in der Lage, während einer aktiven Kommunikation darzustellen, dass die Verbindung nicht verschlüsselt ist. Teil der Entwicklung war es, den OpenBTS Code für die Verwendung des IMSI-Catchers zu modifizieren und ein komfortables Frontend bereitzustellen. Für den zentralen Aspekt der Registrierung der MS in den IMSI-Catcher wurden anhand unterschiedlicher Szenarien verschiedene Vorgehensweisen vorgestellt. So können mit Hilfe des Nokia 3310 wichtige Daten zum Vortäuschen einer Basisstation gewonnen und die Mobile Stations, die sich im Stand-by-Betrieb befinden, mit einem sog. erzwungenen Zellwechsel in den IMSI-Catcher überführt werden.

Wie in Abschnitt 5.4.4 erwähnt, bietet eine aktive Kommunikation Schutz vor dem Open Source IMSI-Catcher, da es aktuell nicht möglich ist, eine MS, die ein Gespräch führt, mit Hilfe eines Handovers in den IMSI-Catcher zu überführen. Dieses Problem besteht auf Grund der fehlenden Implementierung des Handoverprozesses bei OpenBTS, der jedoch für die Zukunft geplant ist. Anschließend müsste die Idee, dargestellt in Abbildung 6.1, dazu führen, dass netzseitig ein Handoverprozess angeregt wird und sich die MS in den IMSI-Catcher einbucht. Dieser Prozess verläuft ähnlich des erzwungenen Zellwechsels ab. Der IMSI-Catcher täuscht eine BTS innerhalb der Nachbarschaftsliste vor. Die MS misst während des Gesprächs die Empfangsstärke und teilt der BTS1 mit, dass der Empfang zur BTS4 besser ist. Die BTS1 veranlasst anschließend ein Handover. Die aktive Kommunikation würde dabei allerdings abbrechen, da die BTS1 einen Kanal in der BTS4 reserviert. Die MS bucht sich dagegen in den IMSI-Catcher ein und verlässt somit das Netz. Mit Hilfe des Nokia 3310 Netzmonitors kann demonstriert werden, dass die Basisstation das Handover-Kommando an die Mobile Station überträgt (ersichtlich durch ein „H“ im Modus 03). Die Mobile Station versucht den Handover durchzuführen, worauf OpenBTS hingegen nicht reagiert. Wann und wie der Handoverprozess implementiert wird, steht allerdings noch nicht fest. Daher bleibt der Jammer aktuell die einzige Möglichkeit, eine aktive Kommunikation zu beenden, damit sich die MS in den IMSI-Catcher einbucht.

Schutz vor dem Open Source IMSI-Catcher bietet ebenfalls das UMTS-Netz. Der IMSI-Catcher kann in diesem Netz nicht ohne Weiteres eine Basisstation vortäuschen, da sich die Basisstation ebenfalls bei der MS authentifizieren muss. Bevor die MS sich komplett in das Netz einbucht, überprüft sie das „Authentication Token“ der Basisstation. Nur wenn das „Authentication Token“ gültig ist, bucht sich die MS in die Basisstation ein.<sup>1</sup> UMTS ist allerdings nicht flächendecken verfügbar und wird aus Kostengründen auf lange Sicht nicht vollständig ausgebaut, da UMTS auf einem höheren Frequenzband arbeitet und somit nicht großflächig

<sup>1</sup> AUTN – Authentication Token, <http://www.umtslink.at/index.php?pageid=autn> [Online; letzter Aufruf 08.10.2009]

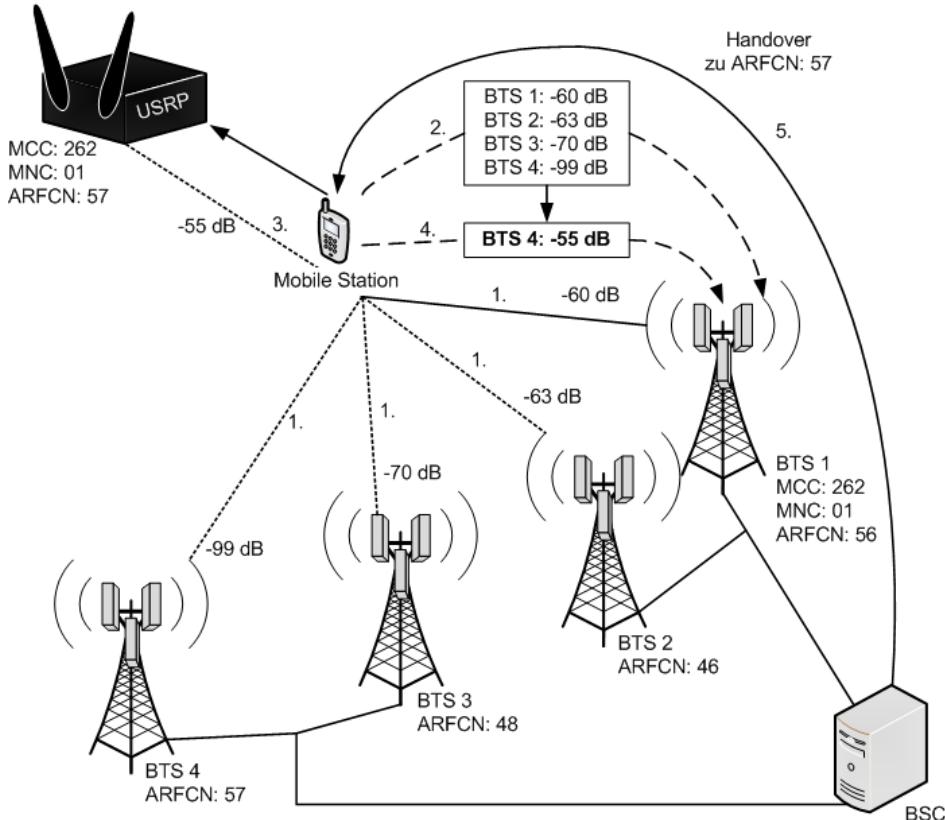


Abbildung 6.1: Erzwungener Handover

betrieben werden kann. Deshalb werden UMTS und GSM parallel betrieben. Infolgedessen wurde eine „fallback“-Funktion implementiert, die der MS die Nutzung des GSM-Netzes ermöglicht, falls kein UMTS-Netz verfügbar ist. Diese Funktion erlaubt es dem IMSI-Catcher auch die Mobile Stations, die UMTS nutzen, „zu fangen“, indem mit Hilfe des Jammers die UMTS-Frequenz gestört wird und die MS sich anschließend über GSM in den IMSI-Catcher einbucht.

Wird allerdings ausschließlich das UMTS-Netz verwendet, muss lediglich ein anderer IMSI-Catcher entwickelt werden, der sich an dem Standard IMSI-Catcher orientiert und Daten an das Originalnetz weiterleitet. Die Funktionsweise eines 3G-IMSI-Catchers [18] ist relativ ähnlich und läuft in drei Phasen ab:

1. Die IMSI bzw. TMSI der MS muss beim initialen Registrierungsprozess gespeichert werden.
2. Der IMSI-Catcher überträgt die gespeicherte IMSI zum Originalnetz und bekommt die Zufallszahl RAND und das „Authentication Token“ (AUTN) als Antwort. Der IMSI-Catcher trennt dann die Verbindung und speichert RAND und AUTN
3. Der IMSI-Catcher überträgt anschließend die RAND und das AUTN an die MS, die die Korrektheit von AUTN anerkennt, da das Token aktuell ist. Die MS bucht sich anschließend in den IMSI-Catcher ein, der wiederum die Verschlüsselung ausschaltet.

Die Funktionsweise wird im Flussdiagramm in der Abbildung 6.2 skizziert. Mit OpenBTS ist dieser Vorgang hingegen nicht möglich, da kein WCDMA-Verfahren unterstützt wird. Sollte

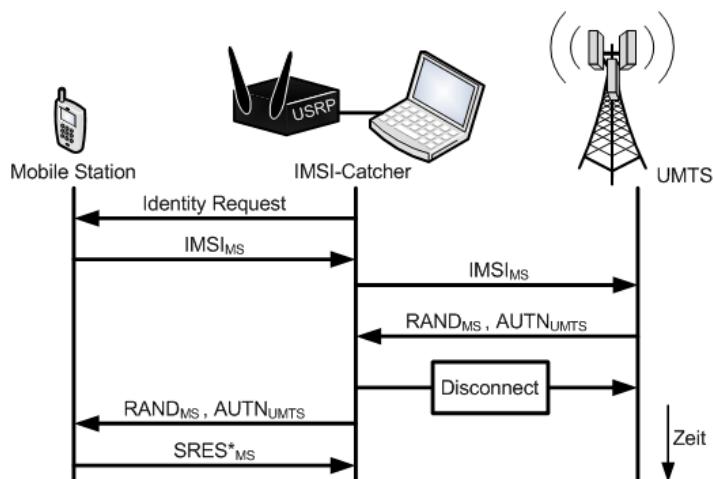


Abbildung 6.2: Funktionsweise eines UMTS-IMSI-Catchers

OpenBTS andererseits um UMTS erweitert werden, wäre die Realisierung solch eins IMSI-Catcher möglich.

Ein komplexeren Aspekt betrifft den Versand von Konfigurations-SMS an eingebuchte Mobile Stations mit Hilfe des IMSI-Catchers. Diese Technologie wird Over-The-Air – kurz OTA – genannt und ermöglicht, Programme (Java Applets) auf die MS zu laden oder Einstellungen zu verändern. Gedacht ist die Technologie ursprünglich für Serviceprovider die für ihre Kunden SMS, GPRS, MMS, E-MAIL, WAP und Gateways für Internet oder Instant Messenger Systemen sowie SIP VoIP Client Konfigurationen automatisch per SMS-Nachricht einstellen. Diese Einstellungen können komplett „geheim“ (sog. Silent-SMS), also ohne dass der Teilnehmer dies bemerkt (abhängig vom Mobilfunktelefon), vorgenommen werden. Diese Konfigurations-SMS werden in einem speziellen XML-Format kodiert und müssen als binär-kodierte SMS (keine Text-SMS) verschickt werden. Normalerweise sollten die Netzbetreiber solche SMS filtern. [1] Mit Hilfe des IMSI-Catchers wäre es aber möglich – nach geeigneten Änderungen am OpenBTS Code – solche Konfigurationsnachrichten zu übertragen. Genauer zu untersuchen wäre hierbei, wie die Einstellungen vorgenommen werden können und welche Mobilfunktelefone besonders anfällig sind. Vor allem im Hinblick auf sicherheitsspezifische Aspekte bilden OTA-Nachrichten ein weiteres Forschungsgebiet, da das Mobilfunktelefon als Bezahlungsmethode oder für Online-Banking immer populärer wird.

Gegenwärtig bietet GSM keinen ausreichenden Schutz vor dem IMSI-Catcher, weshalb solchen Angriffen durchführbar sind. Insgesamt sind die Anwendungs- und Erweiterungsmöglichkeiten vielfältig. Selbst das neuste Mobilfunknetz UMTS bietet nicht den notwendigen Schutz, da mit dem GSM-Jammer die Frequenzen von Basisstationen gestört werden können. Aus diesem Grund gibt es netzseitig für den Teilnehmer keine Möglichkeit den IMSI-Catcher zu erkennen oder sich davor zu schützen. Lediglich einige Mobilfunktelefone zeigen bei einer aktiven Kommunikation im Display an, dass das Gespräch nicht verschlüsselt übertragen wird. Einzig eine externe Verschlüsselung der Kommunikation wie sie unter anderem die Firma Rohde & Schwarz anbietet, verhindert das Abhören eines Gesprächs. Hierbei übernimmt das Mobilfunktelefon die Verschlüsselung der Daten durch ein Schlüsselpaar, das der Kommunikationspartner ebenfalls besitzen muss, um die Daten zu entschlüsseln. So hat jüngst die Bundesregierung mehrere Millionen Euro in verschlüsselungsfähige Mobilfunktelefone der Firma Secusmart investiert. Dies war die Reaktion auf eine Sicherheitswarnung des Bundesamt

## *6 Fazit und Ausblick*

---

für Sicherheit in der Informationstechnik.<sup>2</sup>

„Im Juli diesen Jahres publizierte das Bundesamt ein Papier zur Sicherheit von Mobiltelefonen nach GSM-Standard und kam darin unter anderem zu folgender Bewertung: Die Kommunikation mit GSM-Mobiltelefonen ist ohne hinreichende Sicherheitsmaßnahmen als unsicher anzusehen.“

Schutz vor dem IMSI-Catcher und dem Auslesen der sensiblen Daten wie die IMSI oder die IMEI bietet diese Methode allerdings ebenfalls nicht.

Als Argument könnte dienen, dass GSM schon relativ alt ist und sein Nachfolger, UMTS, bereits verwendet wird und die Sicherheitslücken daher bald der Vergangenheit angehören. UMTS hat sich indes noch nicht komplett durchgesetzt. Aus diesem Grund wird die GSM-Technik noch lange Verwendung finden und keinen Schutz vor einem IMSI-Catcher bieten.

---

<sup>2</sup> Bund schafft tausende verschlüsselte Handys an, <http://winfuture.de/news,50958.html> [Online; letzter Aufruf 27.10.2009]

# Literaturverzeichnis

- [1] AIRPROBE: *Over-the-Air (OTA) attack*. WWW-Dokument, <https://svn.berlin.ccc.de/projects/airprobe/wiki/OTA>. [Online; letzter Aufruf 09.10.2009].
- [2] BERTSCH, HOLGER: *Open Source GSM BTS Setup und Analyse für Demo-Zwecke*. PDF-Dokument, Masterarbeit am LfKs Universität Freiburg, Oktober 2009.
- [3] BRIEGLB, VOLKER: *Kein Notruf ohne SIM-Karte*. WWW-Dokument, <http://www.heise.de/newsticker/Kein-Notruf-ohne-SIM-Karte--/meldung/132539>. [Online; letzter Aufruf 30.09.2009].
- [4] BUNDESAMT FÜR SICHERHEIT IN DER INFORMATIONSTECHNIK: *GSM-Mobilfunk – Gefährdungen und Sicherheitsmaßnahmen*. PDF-Dokument, [https://www.bsi.bund.de/cae/servlet/contentblob/475756/publicationFile/30778/gsm\\_pdf.pdf](https://www.bsi.bund.de/cae/servlet/contentblob/475756/publicationFile/30778/gsm_pdf.pdf), 2003.
- [5] BUNDESNETZAGENTUR: *Frequenznutzungsplan*. PDF-Dokument, <http://www.bundesnetzagentur.de/media/archive/17259.pdf>, April 2008.
- [6] BUNDESRAT: *Verordnung über Notrufverbindungen (NotrufV)*. PDF-Dokument, [http://www.bundesrat.de/cln\\_090/SharedDocs/Drucksachen/2008/0901-1000/967-08 TEMPLATEID=raw,property=publicationFile.pdf/967-08.pdf](http://www.bundesrat.de/cln_090/SharedDocs/Drucksachen/2008/0901-1000/967-08 TEMPLATEID=raw,property=publicationFile.pdf/967-08.pdf), 17.12.2008.
- [7] EMF INSTITUT: *Die Bedeutung des Frequenzunterschieds zwischen GSM-900 und GSM-1800*. PDF-Dokument, [http://www.attendorn.de/mobilfunkattendorn/dokumente/unterschied\\_GSM900\\_GSM1800.pdf](http://www.attendorn.de/mobilfunkattendorn/dokumente/unterschied_GSM900_GSM1800.pdf), Juni 2009.
- [8] ETSI EUROPEAN TELECOMMUNICATIONS STANDARDS INSTITUTE: *European digital cellular telecommunications system (Phase 2); Functions related to Mobile Station (MS) in idle mode (GSM 03.22)*. PDF-Dokument, <http://www.etsi.com>, Februar 1995.
- [9] ETSI EUROPEAN TELECOMMUNICATIONS STANDARDS INSTITUTE: *Digital cellular telecommunications system (phase 2+); radio subsystem synchronization (gsm 05.10 version 8.4.0 release 1999) Technical Report ETSI TS 100 912 V8.4.0 (2000-08)*. PDF-Dokument, <http://www.etsi.com>, August 2000.
- [10] FOX, DIRK: *Der IMSI-Catcher*. PDF-Dokument, <http://www.datenschutz-und-datensicherheit.de/jhrg26/imsicatcher-fox-2002.pdf>, September 2009.
- [11] GRAUBNER, NORBERT: *Baumappe zum FA-Synthesizer FA-SY*. PDF-Dokument, <http://www.box73.de/catalog/pdf/BX-026.pdf>, 2008.
- [12] GSM ASSOCIATION: *GSM Coverage Maps*. WWW-Dokument, <http://www.gsmworld.com/roaming/gsminfo/index.shtml>. [Online; letzter Aufruf 30.09.2009].
- [13] JÖRG EBERSPÄCHER, HAND-JÖRG VÖGEL, CHRISTIAN BETTSTETTER CHRISTIAN HARTMANN: *GSM – Architecture, Protocols and Services*). Wiley, 3. Auflage, 2009.

- [14] LACKEY, JOSHUA: *On turning the RFX900 to an RFX1800 and back again.* WWW-Dokument, <http://www.nabble.com/on-turning-the-RFX900-to-an-RFX1800-and-back-again-td6399750.html>. [Online; letzter Aufruf 30.09.2009].
- [15] LOULA, ALEXANDER: *OpenBTS – Installation and Configuration Guide.* PDF-Dokument, Version 0.1, [https://81.56.142.154/Cour/These/OpenBTS/OpenBTS\\_Guide\\_En\\_v0.1.pdf](https://81.56.142.154/Cour/These/OpenBTS/OpenBTS_Guide_En_v0.1.pdf), Mai 2009.
- [16] LÖSER, FREDERIK: *Position System mit GSM.* PDF-Dokument, Studienarbeit am LfKs Universität Freiburg, <http://www.ks.uni-freiburg.de/download/studienarbeit/WS05/03-06-GSMposition-Loeser/03-06-positionsysteGSM-FLoeser.pdf>, März 2006.
- [17] MEYER, DIPL.-ING. INGO: *Baumappe zum FA-SY-Adapter.* PDF-Dokument, <http://www.box73.de/catalog/pdf/BX-029.pdf>, 2008.
- [18] MEYER, ULRIKE und SUSANNE WETZEL: *A man-in-the-middle attack on UMTS.* In: *WiSe '04: Proceedings of the 3rd ACM workshop on Wireless security*, Seiten 90–97, New York, NY, USA, 2004. ACM.
- [19] NEUMANN, DIRK THOMAS: *Roamingpartner – Telefonieren im Ausland mit den deutschen Netzbetreibern.* PDF-Dokument, <http://www.roaminginfo.de/Roamingpartner.pdf>, 23.09.2009.
- [20] RADIO, GNU: *The GNU Software Radio.* WWW-Dokument, <http://www.gnuradio.org>. [Online; letzter Aufruf 30.09.2009].
- [21] RUNDFUNK UND TELEKOM REGULIERUNGS GMBH: *Spektrum 900/1800 MHz – Übersicht GSM-Bereich.* WWW-Dokument, [http://www.rtr.at/de/tk/Spectrum\\_900\\_1800\\_0v](http://www.rtr.at/de/tk/Spectrum_900_1800_0v). [Online; letzter Aufruf 30.09.2009].
- [22] SAUTER, MARTIN: *Grundkurs Mobile Kommunikationssysteme: Von UMTS und HSDPA, GSM und GPRS zu Wireless LAN und Bluetooth Piconetzen (Broschiert).* vieweg, 3. Auflage, 2008.
- [23] UMTSLINK: *Mobile Switching Center – MSC.* WWW-Dokument, [http://www.umtslink.at/index.php?pageid=GSM\\_msc](http://www.umtslink.at/index.php?pageid=GSM_msc). [Online; letzter Aufruf 30.09.2009].
- [24] WALKE, BERNHARD: *Mobilfunknetze und ihre Protokolle 1 – Grundlagen, GSM, UMTS und andere zellulare Mobilfunknetze.* Teubner, 3. Auflage, 2001.
- [25] WEB CORP CC: *The History Of GSM: 1982 to 2001.* WWW-Dokument, <http://www.cellular.co.za/gsmhistory.htm>. [Online; letzter Aufruf 30.09.2009].
- [26] WIKIPEDIA: *Global System for Mobile Communications.* WWW-Dokument, [http://de.wikipedia.org/wiki/Global\\_System\\_for\\_Mobile\\_Communications](http://de.wikipedia.org/wiki/Global_System_for_Mobile_Communications). [Online; letzter Aufruf 30.09.2009].

# Abkürzungsverzeichnis

<b>Abkürzung</b>	<b>Beschreibung</b>
AGCH	Access Grant Channel
ARFCN	Absolute Radio Frequency Channel Number
AUC	Authentication Center
AUTN	Authentication Token
BCC	Basestation Color Code
BCCH	Broadcast Control Channel
BCH	Broadcast Channel
BSC	Base Station Controller
BSS	Base Station Subsystem
BTS	Base Transceiver Station
CCCH	Common Control Channel
CCH	Control Channels
CID	Cell ID
DCCH	Dedicated Control Channel
DCS	Digital Cellular System
DECT	Digital Enhanced Cordless Telecommunications
EIR	Equipment Identity Register
FACCH	Fast Associated Control Channel
FCCH	Frequency Correction Channel
FDMA	Frequency Division Multiple Access
FPGA	Field Programmable Gate Array
GPRS	General Packet Radio Service
GSM	Global System for Mobile Communications
HLR	Home Location Register
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
LAC	Location Area Code
LAI	Location Area Identifier
LAPD	Link Access Procedure D-Channel
MCC	Mobile Country Code
MNC	Mobile Network Code

## *Abkürzungsverzeichnis*

---

<b>Abkürzung</b>	<b>Beschreibung</b>
MS	Mobile Station
MSC	Mobile Switching Center
MSISDN	Mobile Subscriber ISDN Number
NCC	Network Color Code
NCH	Notification Channel
NSS	Network Subsystem
OMC	Operation and Maintenance Center
OSIC	Open Source IMSI-Catcher
PCH	Paging Channel
PIN	Personal Identification Number
PUK	Personal Unblocking Key
RACH	Random Access Channel
RSS	Radio Subsystem
SACCH	Slow Associated Control Channel
SCH	Synchronization Channel
SDCCH	Stand-alone Dedicated Control Channel
SIM	Subscriber Identity Module
SMS	Short Message Service
SRES	Signed Response
TCH	Traffic Channel
TCH/F	Traffic Channel Fullrate
TCH/H	Traffic Channel Halfrate
TDMA	Time Division Multiple Access
TMSI	Temporary Mobile Subscriber Identity
TRAU	Transcoder and Rate Adaptation Unit
UMTS	Universal Mobile Telecommunications System
USRP	Universal Software Radio Peripheral
VLR	Visitor Location Register
WAP	Wireless Application Protocol

# Abbildungsverzeichnis

2.1	Frequenzbänder bei GSM900 und GSM1800 (Vorlage nach [21]) . . . . .	5
2.2	Frequency reuse innerhalb verschiedener Clustergrößen [13] . . . . .	6
2.3	Vergleich der theoretischen und praktischen Zellenform [13] . . . . .	7
2.4	Struktur eines GSM-Netzes . . . . .	7
2.5	Mobile Station, bestehend aus Mobilfunktelefon und SIM-Karte . . . . .	8
2.6	BTS mit einer 360° Sektorantenne (links) beziehungsweise drei 120° Sektorantennen (rechts) . . . . .	11
2.7	Mengenrelationen der GSM-Hardwarekomponenten eines Netzanbieters . . . . .	12
2.8	Generierung des Sicherheitsdatensatzes: RAND, SRES und Kc . . . . .	13
2.9	Nachrichtenfluss bei einer Authentifikation (Vorlage nach [22]) . . . . .	14
2.10	Symmetrische Verschlüsselung und Entschlüsselung mit einer Schlüssel-Bitsequenz	15
2.11	GSM900-Rahmenstruktur (Vorlage nach [26]) . . . . .	16
2.12	Frequency Division Multiple Access . . . . .	17
2.13	Zeitversetzte Kommunikation für den Down- und Uplink . . . . .	18
2.14	Zeitverschiebung eines Bursts ohne Timing Advance Regelung . . . . .	18
2.15	Aufbau eines Normal Burst . . . . .	19
2.16	Übersicht der fünf Burst-Typen (Vorlage nach [13]) . . . . .	19
2.17	Strukturierung des Übertragungsmediums durch TDMA-Rahmen, Multiframes, Superframes und Hyperframes . . . . .	23
2.18	Nutzung der physikalischen Kanäle im Downlink (Vorlage nach [22]) . . . . .	24
3.1	Universal Software Radio Peripheral Version 1 (links) und Version 2 (rechts) .	26
3.2	USRP Hauptplatine, ausgerüstet mit zwei RFX900 Transceiver Modulen . .	27
3.3	Rund- und Breitbandantenne sowie DBSRX Receiver und RFX900 / RFX1800 Transceiver . . . . .	29
3.4	ISM-Band Filter entfernen . . . . .	30
3.5	Externer Taktgeber FA-SY1: links der Taktgeber, rechts zusammengebaut und mit Heizkörper versehen . . . . .	31
3.6	Kalibrierung des FA-SY1 Taktgebers . . . . .	32
3.7	IMSI-Catcher in Betrieb mit dem USRP1 und dem Samsung NC10 Netbook .	34
3.8	Verwendete Mobilfunktelefone . . . . .	34
4.1	Systemüberblick der benötigten Hard- und Softwarekomponenten (Vorlage nach [15]) . . . . .	35
4.2	GNU Radio Companion . . . . .	37
5.1	IMSI-Catcher GA090 . . . . .	41
5.2	Standard IMSI-Catcher . . . . .	44
5.3	Open Source IMSI-Catcher mit USRP und Asterisk Server . . . . .	44
5.4	Registrierungsprozess bei GSM (Vorlage nach [13]) . . . . .	46
5.5	Standard IMSI-Catcher Registrierung und Kommunikation . . . . .	47
5.6	Registrierungsprozess beim IMSI-Catcher . . . . .	48
5.7	Erzwungener Zellwechsel von der BTS1 zum IMSI-Catcher . . . . .	50

## *Abbildungsverzeichnis*

---

5.8	GSM-Jammer im Einsatz. Die blaue Linie stellt eine Basisstation und die grüne Linie den Jammer dar. . . . .	51
5.9	Nokia 3310 Netzmonitor . . . . .	52
5.10	Frequenzanalyse . . . . .	53
5.11	GSM-Jammer, entwickelt mit GRC . . . . .	55
5.12	Jammer . . . . .	55
5.13	Ablauf ausgehender Rufaufbau bei GSM (Vorlage nach [13]) . . . . .	56
5.14	Ablauf ausgehender Rufaufbau beim IMSI-Catcher . . . . .	57
5.15	Geänderter Registrierungsablauf im Vergleich zu Abbildung 5.6 . . . . .	59
5.16	Grafische Oberfläche des IMSI-Catchers . . . . .	63
5.17	Typisches MITM Szenario . . . . .	64
6.1	Erzwungener Handover . . . . .	68
6.2	Funktionsweise eines UMTS-IMSI-Catchers . . . . .	69
A.1	Schaltplan FA-SY1 [17] . . . . .	77

## A Zusammenbau des externen Taktgebers

Der Bausatz muss zunächst eigenständig zusammengebaut<sup>1</sup> werden. Sein Aufbau kann dem Schaltplan in Abbildung A.1 entnommen werden. Eine detaillierte Anleitung für den Zusammenbau ist in den Baumappen zum FS-AY-Adapter enthalten.[17] [11] Da der Taktgeber extern an das USRP angeschlossen wird, wurde der Bausatz in ein Aluminium-Gehäuse eingebaut (Abbildung 3.5). Dieses Gehäuse gewährleistet gleichzeitig eine Abschirmung gegenüber äußeren Einflüssen und besitzt einen SMA-Anschluss, über den der hochfrequente Takt ausgegeben wird.

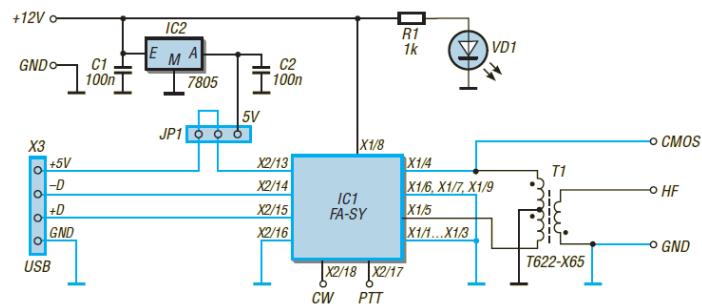


Abbildung A.1: Schaltplan FA-SY1 [17]

### Spannungsversorgung

Für die Heizung ergibt sich ein Spannungsbedarf von 12 V und für den Taktgeber (FA-SY) von 5 V. Die für den Taktgeber benötigten 5 V werden über einen Spannungswandler (IC2) auf der Adapterplatine aus den 12 V erzeugt. Der Taktgeber kann auch vollständig den Strom über USB beziehen. Hierfür muss der Jumper JP1 auf die zwei Stifte, die zum FA-SY zeigen, gesetzt werden. Es gilt jedoch zu beachten, dass die interne Heizung mangels Strom nicht mehr funktioniert und somit zu einer temperaturbedingten Ungenauigkeit in der Taktfrequenz führt.

<sup>1</sup> In diesem Zusammenhang, gilt Konrad Meier besonderen Dank, der den Bausatz zusammengebaut hat.

## B Installationsanleitungen

### B.1 GNU Radio

1. Für das GNU Radio müssen für Ubuntu 9.04 (Jaunty) folgende Pakete installiert werden:

```
sudo apt-get -y install swig g++ automake1.9 libtool python2.5-dev \
fftw3-dev libcppunit-dev libboost1.35-dev sdcc-nf libusb-dev \
libsdl1.2-dev python-wxgtk2.8 subversion guile-1.8-dev \
libqt4-dev python-numpy ccache python-opengl libgs10-dev \
python-cheetah python-lxml doxygen qt4-dev-tools \
libqwt5-qt4-dev libqwtplot3d-qt4-dev pyqt4-dev-tools
```

2. Für optionale Pakete:

```
sudo apt-get -y install gkrellm wx-common libwxgtk2.8-dev alsa-base \
autoconf xorg-dev g77 gawk bison openssh-server emacs cvs usbview octave
```

3. Anschließend muss die aktuellste Source aus dem SVN Repository mit dem folgenden Befehl in das aktuelle Verzeichnis heruntergeladen werden:

```
svn co http://gnuradio.org/svn/gnuradio/branches/releases/3.2 gnuradio
```

4. Nun ist es nötigt, dass die Version kompiliert und installiert wird. Hierzu muss zunächst in das Hauptverzeichnis von GNU Radio gewechselt werden, und anschließend müssen folgende Befehle eingegeben werden:

```
./bootstrap
./configure
make
sudo make install
```

Folgende Ausgabe nach ./configure sollte erscheinen:

```
The following GNU Radio components have been successfully configured:
```

config	gr-usrp2	gr-trellis
gruel	gr-audio-alsa	gr-video-sdl
omnithread	gr-audio-oss	gr-wxgui
gnuradio-core	gr-cvsd-vocoder	gr-qtgui
pmt	gr-gpio	gr-sounder
mblock	gr-gsm-fr-vocoder	gr-utils
usrp	gr-pager	gnuradio-examples
usrp2	gr-radar-mono	grc

gr-usrp	gr-radio-astronomy	docs
---------	--------------------	------

You may now run the make command to build these components.

\*\*\*\*\*

The following components were skipped either because you asked not to build them or they didn't pass configuration checks:

gcell	gr-audio-osx
gr-gcell	gr-audio-portaudio
gr-audio-jack	gr-audio-windows

These components will not be built.

5. Da Ubuntu „udev“ für hotplug Geräte nutzt und daher keinen „nicht-root“ Zugriff zum USRP Device erlaubt, muss eine eigene USRP-Gruppe hinzugefügt und der eigene Benutzername zu dieser Gruppe eingeteilt werden.

```
sudo addgroup usrp
sudo addgroup <YOUR_USERNAME> usrp
echo 'ACTION=="add", BUS=="usb", SYSFS{idVendor}=="ffffe",
      SYSFS{idProduct}=="0002", GROUP=="usrp", MODE=="0660"' > tmpfile
sudo chown root.root tmpfile
sudo mv tmpfile /etc/udev/rules.d/10-usrp.rules
```

6. Ubuntu weiß nun, wie es sich verhalten muss, sobald es das USRP Device über USB erkennt. Damit die Regeln aktiv werden, muss entweder „udev“ neu gestartet werden, oder, falls das nicht ausreicht, der PC neu gestartet werden.

```
sudo /etc/init.d/udev stop
sudo /etc/init.d/udev start
```

7. Mit folgendem Kommando kann überprüft werden, ob Ubuntu das USRP Device richtig erkennt.

```
ls -lR /dev/bus/usb | grep usrp
```

Ein erfolgreicher Eintrag sollte folgendermaßen aussehen:

```
crw-rw---- 1 root usrp 189, 514 Jun 24 09:46 003
```

8. Standardgemäß installiert GNU Radio Pythonscripte in den Pfad */usr/local/share/gnu-radio/examples*. Im USRP Unterordner befinden sich Scripte, mit denen geprüft werden kann, ob alles ordnungsgemäß installiert wurde und was ermöglicht, den maximalen Datendurchsatz (abhängig vom USB) zu messen. Hierzu muss lediglich folgendes Script gestartet werden:

```
cd /usr/local/share/gnuradio/examples/usrp
sudo ./usrp_benchmark_usb.py
```

Die Ausgabe sollte anschließend folgendermaßen aussehen:

Testing 2MB/sec...	Testing 4MB/sec...	Testing 8MB/sec...
usb_throughput = 2M	usb_throughput = 4M	usb_throughput = 8M
ntotal = 1000000	ntotal = 2000000	ntotal = 4000000
nright = 998435	nright = 1998041	nright = 3999272
runlength = 998435	runlength = 1998041	runlength = 3999272
delta = 1565 OK	delta = 1959 OK	delta = 728 OK
Testing 16MB/sec...	Testing 32MB/sec...	
usb_throughput = 16M	usb_throughput = 32M	
ntotal = 8000000	ntotal = 16000000	
nright = 7992153	nright = 15986239	
runlength = 7992153	runlength = 15986239	
delta = 7847 OK	delta = 13761 OK	
Max USB/USRP throughput = 32MB/sec		

Es ist ersichtlich, dass der maximale Datendurchsatz zwischen dem PC und dem USRP 32 MB/sec. beträgt. Sollte es beim Ausführen der Pythonscripte zu einem Import Error der Form „*Import Error: libgnuradio-core.so.0: cannot open shared object file: No such file or directory*“ kommen, kann folgender Befehl für Abhilfe sorgen:

```
export PYTHONPATH=/usr/local/lib/python2.5/site-packages  
sudo ldconfig
```

## B.2 OpenBTS / Open Source IMSI-Catcher

Die Installation orientiert sich größtenteils an dem Installationsguide von GNU Radio Wiki.<sup>1</sup>

1. Bevor OpenBTS installiert werden kann, müssen die fehlenden Bibliotheken für SIP „OSIP2“ und RTP „ORTP“ installiert werden:

```
sudo apt-get install libosip2-3deb libosip2-dev libortp7 libortp7-dev
```

2. Für die GUI des entwickelten IMSI-Catchers werden überdies folgende Pakete benötigt:

```
sudo apt-get install libglademm-2.4-dev
```

Der Vollständigkeit halber sei erwähnt, dass hierdurch folgende Pakete mitinstalliert werden:

```
build-essential debhelper dpkg-dev gettext html2text intltool-debian  
libatk1.0-dev libcairo2-dev libcairomm-1.0-dev libglade2-dev  
libglademm-2.4-1c2a libglademm-2.4-dev libglibmm-2.4-dev  
libgtk2.0-dev libgkmm-2.4-dev libmail-sendmail-perl libpango1.0-dev  
libpangomm-1.4-dev libpixman-1-dev libsigc++-2.0-dev  
libsys-hostname-long-perl libxcb-render-util0-dev libxcb-render0-dev  
libxcomposite-dev libxdamage-dev libxml2-dev patch po-debconf  
x11proto-composite-dev x11proto-damage-dev
```

<sup>1</sup> Building and Running OpenBTS, <http://www.gnuradio.org/trac/wiki/OpenBTS/BuildingAndRunning> [Online; letzter Aufruf 30.09.2009]

Auf dem Testsystem wurden folgende Versionen installiert:

Bibliothek	Version
libosip2-3deb	3.1.0-1
libosip2-dev	3.1.0-1
libortp7	2.1.1-1ubuntu2
libortp7-dev	2.1.1-1ubuntu2
libglademm	2.4
libglademm-2.4-dev	2.4

Tabelle B.1: Installierte Versionen

3. Anschließend kann mit der Installation von OpenBTS fortgefahrene werden. Hierzu müssen folgende Befehle ausgeführt werden:

```
tar xvfz osic.tar.gz
cd osic
./configure
make
sudo make install
```

4. Für die korrekte Funktionsweise des Open Source IMSI-Catchers müssen die Asterisk Konfigurationsdateien *sip.conf* und *extensions.conf* vom Ordner */osic/GUI/asterisk* nach */etc/asterisk/* kopiert und den Aufnahmepfad dementsprechend angepasst werden.

### B.3 OpenBTS-Patch

Die folgenden Änderungen mussten für die Dateien *USRPDevice.h* und *USRPDevice.cpp* vorgenommen werden, um OpenBTS 2.4.1 mit GNU Radio 3.2 betreiben zu können. Die angegebenen Patches müssen sowohl für die Dateien im *Transceiver* als auch im *Transceiver52M* Ordner verwendet werden. Sollte das tar-Archiv von der CD genutzt werden, müssen die Patches nicht installiert werden, da diese bereits integriert sind. Die Patches befinden sich auf der CD im Ordner *Patch/* und können mit folgendem Kommando angewendet werden:

```
patch -p1 < name_des_Patch.diff
```

Listing B.1: USRPDevice.h.diff Patch

```
1 --- /openbts-2.4.1.Kinder-org/Transceiver/USRPDevice.h 2009-10-18 23:59:42.000000000
      +0200
2 +++ /openbts-2.4.1.Kinder/Transceiver/USRPDevice.h 2009-10-24 16:07:59.000000000
      +0200
3 @@ -42,8 +42,8 @@ class USRPDevice {
4     private:
5
6     double desiredSampleRate; ///< the desired sampling rate
7 -     usrp_standard_rx* m_uRx; ///< the USRP receiver
8 -     usrp_standard_tx* m_uTx; ///< the USRP transmitter
9 +     usrp_standard_rx_sptr m_uRx; ///< the USRP receiver
10 +    usrp_standard_tx_sptr m_uTx; ///< the USRP transmitter
11
12    double actualSampleRate; ///< the actual USRP sampling rate
13    unsigned int decimRate; ///< the USRP decimation rate
```

Listing B.2: USRPDevice.cpp.diff Patch

```

1 --- /openbts-2.4.1.Kinder-org/Transceiver/USRPDevice.cpp 2009-10-18
2 23:59:42.000000000 +0200
2 +++ /openbts-2.4.1.Kinder/Transceiver/USRPDevice.cpp 2009-10-27 01:26:28.000000000
2 +0100
3 @@ -142,8 +142,8 @@ USRPDevice::USRPDevice (double _desiredS
4     double masterClockRate = (double) 64.0e6;
5     decimRate = (unsigned int) round(masterClockRate/_desiredSampleRate);
6     actualSampleRate = masterClockRate/decimRate;
7 -   m_uRx = NULL;
8 -   m_uTx = NULL;
9 +   m_uRx.reset();
10 +  m_uTx.reset();
11
12 #ifdef SWLOOPBACK
13   samplePeriod = 1.0e6/actualSampleRate;
14 @@ -161,7 +161,7 @@ bool USRPDevice::make(bool wSkipRx)
15 #ifndef SWLOOPBACK
16   string rbf = "std_inband.rbf";
17 // string rbf = "inband_1rxhb_1tx.rbf";
18 -   m_uRx = NULL;
19 +  m_uRx.reset();
20   if (!skipRx) {
21     try {
22       m_uRx = (usrp_standard_rx::make(0, decimRate, 1, -1,
23 @@ -171,9 +171,12 @@ bool USRPDevice::make(bool wSkipRx)
24
25   catch (...) {
26     LOG(ERROR) << "make failed on Rx";
27 -   delete m_uRx;
28   return false;
29 }
30 + if (!m_uRx) {
31 +   LOG(ERROR) << "make failed on Rx";
32 +   return false;
33 }
34 }
35
36 try {
37 @@ -183,9 +186,12 @@ bool USRPDevice::make(bool wSkipRx)
38
39 catch (...) {
40   LOG(ERROR) << "make failed on Tx";
41 -   delete m_uTx;
42   return false;
43 }
44 + if (!m_uTx) {
45 +   LOG(ERROR) << "make failed on Tx";
46 +   return false;
47 }
48
49 if (!skipRx) m_uRx->stop();
50 m_uTx->stop();

```

# C Code

## C.1 createSIP - Funktion

Erstellt für die als Eingabeparameter übergebene IMSI einen SIP-Account innerhalb der *sip.conf* (in */etc/asterisk*) und lädt diesen im laufenden Betrieb in den Asterisk-Server.

Listing C.1: *createSIP*-Funktion, innerhalb der MobilityManagement.cpp

```
1 void createSIP(const char* imsi){
2     std::fstream infile;
3     infile.open("/etc/asterisk/sip.conf", std::ios_base::out | std::ios_base::app);
4     const char* context_buffer = "context=default\n";
5     const char* reinvite_buffer = "canreinvite=no\n";
6     const char* type_buffer = "type=friend\n";
7     const char* host_buffer = "host=dynamic\n";
8     const char* allow_buffer = "allow=all\n";
9
10    //write account into sip.conf:
11    // [IMSI]
12    // context=default
13    // canreinvite=no
14    // type=friend
15    // host=dynamic
16    // allow=all
17    if(infile.is_open()){
18        infile.write("\n",1);
19        infile.write("[",1);
20        infile.write(imsi_buffer, strlen(imsi_buffer));
21        infile.write("]\n",2);
22        infile.write(context_buffer, strlen(context_buffer));
23        infile.write(reinvite_buffer, strlen(reinvite_buffer));
24        infile.write(type_buffer, strlen(type_buffer));
25        infile.write(host_buffer, strlen(host_buffer));
26        infile.write(allow_buffer, strlen(allow_buffer));
27        infile.write("\n",1);
28        infile.close();
29        //reload the sip.conf. this is neccesary to activate the new account
30        system("asterisk -r -x \"sip reload\"");
31    }
32    else{
33        std::cerr << "can't open sip.conf" << std::endl;
34    }
35 }
```

Da im weiteren Verlauf die IMEI ebenfalls in die *sip.conf* gespeichert wird, musste die Funktion dementsprechend geändert werden. Es werden nun die IMSI und die IMEI als Eingabeparameter erwartet. Der Ablauf hat sich nicht geändert, jedoch wird die IMEI unter den Benutzernamen als Kommentar gespeichert.

Listing C.2: Geänderte *createSIP*-Funktion, innerhalb der MobilityManagement.cpp

```
1 void createSIP(const char* imsi, const char* imei){
2     :
3     //write account into sip.conf:
```

```

4 // [imsi]
5 // ;IMEI=imei
6 // context=default
7 // canreinvite=no
8 // type=friend
9 // host=dynamic
10 // allow=all
11 if(infile.is_open()){
12     infile.write("\n",1);
13     infile.write("[",1);
14     infile.write(imsi,strlen(imsi));
15     infile.write("]\n",2);
16     infile.write(";IMEI=",6);
17     infile.write(imei,strlen(imei));
18     infile.write("\n",1);
19 }
20 }
21 }
```

## C.2 acceptIMSI - Funktion

Prüft, ob die IMSI die notwendige Netzberechtigung besitzt, sich in das vorgetäuschte Netz einzubuchen. Hierzu werden die Ziffern 3 und 4 mit dem vorgetäuschten Netzanbietercode verglichen. Falls dieser Übereinstimmt, wird *true* ausgegeben und die IMSI darf sich einbuchen.

Listing C.3: *acceptIMSI*-Funktion innerhalb der MobilityManagement.cpp

```

1 bool acceptIMSI(const char* imsi){
2     LOG(INFO) << "acceptIMSI: checking to accept IMSI=" << imsi;
3     std::string gsmMNC = "00";
4     std::string imsiMNC = imsi;
5     if(gConfig.defines("GSM.MNC")){
6         gsmMNC = gConfig.getStr("GSM.MNC");
7         if(imsiMNC.substr(3,2) == gsmMNC){
8             // Falls IMSI dem MNC entspricht, akzeptiere die IMSI
9             LOG(INFO) << "acceptIMSI: IMSI=" << imsi << " accepted";
10            return true;
11        }
12    } else{
13        // ... ansonsten lass die MS mit der IMSI nicht einbuchen
14        LOG(INFO) << "acceptIMSI: IMSI=" << imsi << " rejected";
15        return false;
16    }
17 }
18 // Falls MNC nicht definiert, lasse alle IMSI's zu
19 LOG(INFO) << "acceptIMSI: no GSM.MNC found => IMSI=" << imsi << " accepted";
20 return true;
21 }
```

## C.3 resolveIMEI - Funktion

Die Identität des Teilnehmers, sowie der Kommunikationskanal wird der *resolveIMEI*-Funktion übergeben, die auf diesem Kanal die IMEI anfragt. Als Antwort wird wiederum eine Identität ausgegeben, mit deren Hilfe die Nummer anschließend in die *mIMEImap*-Tabelle gespeichert wird.

Listing C.4: IMEI-Abfrage, innerhalb der ControlCommon.cpp

```
1 GSM::L3MobileIdentity Control::resolveIMEI(L3MobileIdentity& mobID, LogicalChannel*
2   LCH)
3 {
4   assert(LCH);
5   LCH->send(L3IdentityRequest(IMEIType));
6   // FIXME — This request times out on T3260, 12 sec. See GSM 04.08 Table 11.2.
7   L3Message* msg = getMessage(LCH);
8   L3IdentityResponse *resp = dynamic_cast<L3IdentityResponse*>(msg);
9   if (!resp) {
10     if (msg) delete msg;
11     throw UnexpectedMessage();
12   }
13   delete msg;
14   return resp->mobileID();
```

## D GSM900 Downlink ARFCN - Frequenz Tabelle

ARFCN	Frequenz	ARFCN	Frequenz	ARFCN	Frequenz	ARFCN	Frequenz
1	935.2	32	941.4	63	947.6	94	953.8
2	935.4	33	941.6	64	947.8	95	954.0
3	935.6	34	941.8	65	948.0	96	954.2
4	935.8	35	942.0	66	948.2	97	954.4
5	936.0	36	942.2	67	948.4	98	954.6
6	936.2	37	942.4	68	948.6	99	954.8
7	936.4	38	942.6	69	948.8	100	955.0
8	936.6	39	942.8	70	949.0	101	955.2
9	936.8	40	943.0	71	949.2	102	955.4
10	937.0	41	943.2	72	949.4	103	955.6
11	937.2	42	943.4	73	949.6	104	955.8
12	937.4	43	943.6	74	949.8	105	956.0
13	937.6	44	943.8	75	950.0	106	956.2
14	937.8	45	944.0	76	950.2	107	956.4
15	938.0	46	944.2	77	950.4	108	956.6
16	938.2	47	944.4	78	950.6	109	956.8
17	938.4	48	944.6	79	950.8	110	957.0
18	938.6	49	944.8	80	951.0	111	957.2
19	938.8	50	945.0	81	951.2	112	957.4
20	939.0	51	945.2	82	951.4	113	957.6
21	939.2	52	945.4	83	951.6	114	957.8
22	939.4	53	945.6	84	951.8	115	958.0
23	939.6	54	945.8	85	952.0	116	958.2
24	939.8	55	946.0	86	952.2	117	958.4
25	940.0	56	946.2	87	952.4	118	958.6
26	940.2	57	946.4	88	952.6	119	958.8
27	940.4	58	946.6	89	952.8	120	959.0
28	940.6	59	946.8	90	953.0	121	959.2
29	940.8	60	947.0	91	953.2	122	959.4
30	941.0	61	947.2	92	953.4	123	959.6
31	941.2	62	947.4	93	953.6	124	959.8
975	925.2	988	927.8	1000	930.2	1012	932.6
976	925.4	989	928.0	1001	930.4	1013	932.8
977	925.6	990	928.2	1002	930.6	1014	933.0
978	925.8	991	928.4	1003	930.8	1015	933.2
979	926.0	992	928.6	1004	931.0	1016	933.4
980	926.2	993	928.8	1005	931.2	1017	933.6
981	926.4	994	929.0	1006	931.4	1018	933.8
982	926.6	995	929.2	1007	931.6	1019	934.0
983	926.8	996	929.4	1008	931.8	1020	934.2
984	927.0	997	929.6	1009	932.0	1021	934.4
985	927.2	998	929.8	1010	932.2	1022	934.6
986	927.4	999	930.0	1011	932.4	1023	934.8
987	927.6						

## E GSM1800 Downlink ARFCN - Frequenz Tabelle

ARFCN	Frequenz	ARFCN	Frequenz	ARFCN	Frequenz	ARFCN	Frequenz
512	1805.2	606	1824.0	700	1842.8	793	1861.4
513	1805.4	607	1824.2	701	1843.0	794	1861.6
514	1805.6	608	1824.4	702	1843.2	795	1861.8
515	1805.8	609	1824.6	703	1843.4	796	1862.0
516	1806.0	610	1824.8	704	1843.6	797	1862.2
517	1806.2	611	1825.0	705	1843.8	798	1862.4
518	1806.4	612	1825.2	706	1844.0	799	1862.6
519	1806.6	613	1825.4	707	1844.2	800	1862.8
520	1806.8	614	1825.6	708	1844.4	801	1863.0
521	1807.0	615	1825.8	709	1844.6	802	1863.2
522	1807.2	616	1826.0	710	1844.8	803	1863.4
523	1807.4	617	1826.2	711	1845.0	804	1863.6
524	1807.6	618	1826.4	712	1845.2	805	1863.8
525	1807.8	619	1826.6	713	1845.4	806	1864.0
526	1808.0	620	1826.8	714	1845.6	807	1864.2
527	1808.2	621	1827.0	715	1845.8	808	1864.4
528	1808.4	622	1827.2	716	1846.0	809	1864.6
529	1808.6	623	1827.4	717	1846.2	810	1864.8
530	1808.8	624	1827.6	718	1846.4	811	1865.0
531	1809.0	625	1827.8	719	1846.6	812	1865.2
532	1809.2	626	1828.0	720	1846.8	813	1865.4
533	1809.4	627	1828.2	721	1847.0	814	1865.6
534	1809.6	628	1828.4	722	1847.2	815	1865.8
535	1809.8	629	1828.6	723	1847.4	816	1866.0
536	1810.0	630	1828.8	724	1847.6	817	1866.2
537	1810.2	631	1829.0	725	1847.8	818	1866.4
538	1810.4	632	1829.2	726	1848.0	819	1866.6
539	1810.6	633	1829.4	727	1848.2	820	1866.8
540	1810.8	634	1829.6	728	1848.4	821	1867.0
541	1811.0	635	1829.8	729	1848.6	822	1867.2
542	1811.2	636	1830.0	730	1848.8	823	1867.4
543	1811.4	637	1830.2	731	1849.0	824	1867.6
544	1811.6	638	1830.4	732	1849.2	825	1867.8
545	1811.8	639	1830.6	733	1849.4	826	1868.0
546	1812.0	640	1830.8	734	1849.6	827	1868.2
547	1812.2	641	1831.0	735	1849.8	828	1868.4
548	1812.4	642	1831.2	736	1850.0	829	1868.6
549	1812.6	643	1831.4	737	1850.2	830	1868.8
550	1812.8	644	1831.6	738	1850.4	831	1869.0
551	1813.0	645	1831.8	739	1850.6	832	1869.2
552	1813.2	646	1832.0	740	1850.8	833	1869.4
553	1813.4	647	1832.2	741	1851.0	834	1869.6
554	1813.6	648	1832.4	742	1851.2	835	1869.8
555	1813.8	649	1832.6	743	1851.4	836	1870.0
556	1814.0	650	1832.8	744	1851.6	837	1870.2

E GSM1800 Downlink ARFCN - Frequenz Tabelle

---

557	1814.2	651	1833.0	745	1851.8	838	1870.4
558	1814.4	652	1833.2	746	1852.0	839	1870.6
559	1814.6	653	1833.4	747	1852.2	840	1870.8
560	1814.8	654	1833.6	748	1852.4	841	1871.0
561	1815.0	655	1833.8	749	1852.6	842	1871.2
562	1815.2	656	1834.0	750	1852.8	843	1871.4
563	1815.4	657	1834.2	751	1853.0	844	1871.6
564	1815.6	658	1834.4	752	1853.2	845	1871.8
565	1815.8	659	1834.6	753	1853.4	846	1872.0
566	1816.0	660	1834.8	754	1853.6	847	1872.2
567	1816.2	661	1835.0	755	1853.8	848	1872.4
568	1816.4	662	1835.2	756	1854.0	849	1872.6
569	1816.6	663	1835.4	757	1854.2	850	1872.8
570	1816.8	664	1835.6	758	1854.4	851	1873.0
571	1817.0	665	1835.8	759	1854.6	852	1873.2
572	1817.2	666	1836.0	760	1854.8	853	1873.4
573	1817.4	667	1836.2	761	1855.0	854	1873.6
574	1817.6	668	1836.4	762	1855.2	855	1873.8
575	1817.8	669	1836.6	763	1855.4	856	1874.0
576	1818.0	670	1836.8	764	1855.6	857	1874.2
577	1818.2	671	1837.0	765	1855.8	858	1874.4
578	1818.4	672	1837.2	766	1856.0	859	1874.6
579	1818.6	673	1837.4	767	1856.2	860	1874.8
580	1818.8	674	1837.6	768	1856.4	861	1875.0
581	1819.0	675	1837.8	769	1856.6	862	1875.2
582	1819.2	676	1838.0	770	1856.8	863	1875.4
583	1819.4	677	1838.2	771	1857.0	864	1875.6
584	1819.6	678	1838.4	772	1857.2	865	1875.8
585	1819.8	679	1838.6	773	1857.4	866	1876.0
586	1820.0	680	1838.8	774	1857.6	867	1876.2
587	1820.2	681	1839.0	775	1857.8	868	1876.4
588	1820.4	682	1839.2	776	1858.0	869	1876.6
589	1820.6	683	1839.4	777	1858.2	870	1876.8
590	1820.8	684	1839.6	778	1858.4	871	1877.0
591	1821.0	685	1839.8	779	1858.6	872	1877.2
592	1821.2	686	1840.0	780	1858.8	873	1877.4
593	1821.4	687	1840.2	781	1859.0	874	1877.6
594	1821.6	688	1840.4	782	1859.2	875	1877.8
595	1821.8	689	1840.6	783	1859.4	876	1878.0
596	1822.0	690	1840.8	784	1859.6	877	1878.2
597	1822.2	691	1841.0	785	1859.8	878	1878.4
598	1822.4	692	1841.2	786	1860.0	879	1878.6
599	1822.6	693	1841.4	787	1860.2	880	1878.8
600	1822.8	694	1841.6	788	1860.4	881	1879.0
601	1823.0	695	1841.8	789	1860.6	882	1879.2
602	1823.2	696	1842.0	790	1860.8	883	1879.4
603	1823.4	697	1842.2	791	1861.0	884	1879.6
604	1823.6	698	1842.4	792	1861.2	885	1879.8
605	1823.8	699	1842.6				

## F CD-Inhalt

Die Struktur der beiliegenden CD ist wie folgt gegliedert:

- 1 Logfiles
- 2 Patch
- 3 Masterarbeit
- 4 OSIC

Hierbei befinden sich im Ordner *Logfiles* verschiedene Logdateien, die von OpenBTS während dem Registrierungsvorgang angelegt und für Analysezwecke verwendet wurden. Der Ordner *Patch* enthält die für OpenBTS wichtigen Änderungen, um GNU Radio 3.2 nutzen zu können. Die Masterarbeit und die Folien für die Endpräsentation befinden sich im Ordner *Masterarbeit*. Der auf Basis von OpenBTS entwickelte Open Source IMSI-Catcher und der prototypische Jammer befinden sich im Ordner *OSIC* (Open Source IMSI-Catcher).