ASMONIA

Attack analysis and Security concepts for MOBILE Network infrastructures, supported by collaborative Information exchange

Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

Contributors: ERNW Enno Rey Netzwerke GmbH
Fraunhofer Research Institution for Applied and Integrated Security (AISEC)
Nokia Siemens Networks GmbH & Co KG
RWTH Aachen

Editor: Peter Schneider (Nokia Siemens Networks)
About the ASMONIA project

Given their inherent complexity, protecting telecommunication networks from attacks requires the implementation of a multitude of technical and organizational controls. Furthermore, to be fully effective these measures call for the collaboration between different administrative domains such as network operators, manufacturers, service providers, government authorities, and users of the services.

ASMONIA is the acronym for the German name* of a research project that aims to improve the resilience, reliability and security of current and future mobile telecommunication networks. For this purpose the ASMONIA consortium made up of several partners from academia and industry performs a number of research tasks, based on the specific expertise of the individual partners. The project running from September 2011 till May 2013 receives funding from the German Federal Ministry of Education and Research (Bundesministerium für Bildung und Forschung, BMBF). Various associated partners further contribute on a voluntary basis.

* The full name is "Angriffsanalyse und Schutzkonzepte für MObilfunkbasierte Netzinfrastrukturen unterstützt durch kooperativen InformationsAustausch" (Attack analysis and security concepts for mobile network infrastructures, supported by collaborative information exchange).

Partners:
Cassidian Systems
ERNW Enno Rey Netzwerke GmbH
Fraunhofer Research Institution for Applied and Integrated Security (AISEC)
Hochschule Augsburg
Nokia Siemens Networks GmbH & Co KG
RWTH Aachen

Associated Partners:
Federal Agency for Digital Radio of Security Authorities and Organizations (BDBOS)
Federal Office for Information Security (BSI)
Deutsche Telecom AG (DTAG)

For more details about the project please visit www.asmonia.de.
Executive Summary

This paper documents the threat and risk analysis for mobile communication networks and mobile terminals that has been carried out by the ASMONIA consortium. The analysis covers on one side various types of mobile terminals and on the other side mobile communication networks according to the architecture specified by 3GPP, i.e. GSM, UMTS and in particular LTE/SAE networks.

For the purpose of this analysis generic threat categories have been defined. Using these threat categories, the different assets (i.e. components of the mobile network) have been assessed according to a common method, which is specified in this document. This assessment method comprises the estimation of the likelihood of attacks, the overall vulnerability of the assets, and the impact of successful attacks on the network in a qualitative way. From these factors, a risk value is calculated that allows to compare the significance of the different threats and to give a ranking of the different network elements according to the risk they are exposed to.

The assessments were performed by groups of experts, selected according to the required expertise on each asset. In some cases, the theoretical analysis was complemented by practical penetration testing, e.g. attacks on network elements were carried out in lab environments.

The results of the analysis show that there are significant differences in the risks associated to different network elements. The highest risks are related to base stations, to the short message service, to core network elements that aggregate large amounts of user plane traffic, to the IP multimedia subsystem and to servers used for operation and maintenance of the network. For mobile terminals, the analysis displays high risks for the devices and their users as well. Compromise of a mobile terminal, e.g. via user installed software, which turns out to be malicious, constitutes a quite substantial threat for the affected user. A compromise of a large set of mobile terminals, leading to the establishment of a mobile botnet, can easily endanger a whole mobile network.

It can be concluded that security concepts for mobile terminals and mobile communication networks, including improved attack analysis concepts, as explored in the ASMONIA project, are of vital importance to counter the various threats and reduce the security risks of future mobile network infrastructures.
# Table of Contents

1 Introduction 7  

2 Threat and Risk Analysis Method 11  

2.1 Related Work and Relevant Standards 11  
2.1.1 ISO 27000 Family 11  
2.1.1.1 ISO 27005 11  
2.1.1.2 ISO 27011 11  
2.1.1.3 ISO 27033-1 12  
2.1.2 3GPP Sourced Documents 12  
2.1.2.1 Technical Specification 3GPP TS 33.120 (Release 4) 12  
2.1.2.2 Technical Specification 3GPP TS 21.133 (Release 4) 12  
2.1.2.3 Technical Report 3GPP TR 33.821 (Release 9) 12  
2.1.3 ETSI TS 102 165-1 V4.2.1 13  
2.1.4 ITU X.805 13  
2.1.5 ITU E.408 "Telecommunication Networks Security Requirements 13  
2.1.6 NIST SP800-30 13  
2.2 Risk Assessment Approach Used within this Document 13  
2.2.1 Definitions 14  
2.2.2 Source(s) of Threats 15  
2.2.3 Factors Contributing to a Risk 15  
2.2.4 Method of Estimation 16  
2.2.5 Scale & Calculation Formula Used 17  
2.2.6 Reliability of the Assessments 18  

3 Threat Categories 19  

3.1 Focus of the Threat Analysis 19  
3.2 Threat Categories 20  
3.3 Application of the Threat Categories in Assessments 23  

4 Threat and Risk Analysis per Asset 24  

4.1 Terminals 24  
4.1.1 Baseband Part 25  
4.1.1.1 Hardware and Software Architecture (Modem) 27  
4.1.1.2 Subscriber Identification Module (SIM) 27  
4.1.1.3 Security Goals of GSM 28  
4.1.1.4 Attacks on GSM Security 31  
4.1.1.5 UMTS Security 39  
4.1.1.6 Attacks on UMTS 39  
4.1.1.7 Fuzzing Baseband Implementations 40  
4.1.2 Applications Part 43  
4.1.2.1 Machine-to-Machine 43  
4.1.2.2 Feature Phones or Simple Mobile Phones 44  
4.1.2.3 Mobile Operating Systems for Smart Phones 49  
4.1.2.4 Comparison of the Security Features of the Most Popular Operating Systems for Current Smart Phones: Android, iOS and Windows Phone 7 75  
4.1.2.5 PCs with 3G/4G Module 87
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

4.1.3 Threat Analysis

4.2 Access Network

4.2.1 2G Access Networks

4.2.1.1 Base Station Controller (BSC)

4.2.1.2 Base Transceiver Station (BTS)

4.2.2 3G Access Networks

4.2.2.1 Radio Network Controller (RNC)

4.2.2.2 NodeB

4.2.2.3 Home NodeB (HNB)

4.2.2.4 Other Components for the Support of HNBs

4.2.3 4G Access Networks

4.2.3.1 E-UTRAN Node B (eNB)

4.2.3.2 Relay Node (RN)

4.2.3.3 Home eNB (HeNB)

4.2.3.4 Home eNB Gateway (HeNB-GW)

4.2.3.5 Other Components for the Support of HeNB

4.3 Core Network

4.3.1 Evolved Packet Core (EPC)

4.3.1.1 SAE-GW

4.3.1.2 MME

4.3.1.3 SGSN

4.3.1.4 ePDG

4.3.1.5 3GPP AAA-Server or –Proxy

4.3.2 Subscriber Management

4.3.2.1 Home Subscriber Server (HSS)

4.3.2.2 Equipment Identity Register (EIR)

4.3.3 2G/3G PS Domain

4.3.3.1 2G/3G SGSN

4.3.3.2 GGSN

4.3.3.3 GPRS Tunneling Protocol for Control Plane (GTP-C)

4.3.4 CS Domain (2G/3G only)

4.3.5 IP Multimedia System

4.3.6 Charging

4.3.6.1 Charging Systems

4.3.6.2 Policy and Charging Rules Function (PCRF)

4.3.7 Security Gateway

4.3.8 Other Services and Functions

4.3.8.1 Location Services (LCS)

4.3.8.2 Short Message Service (SMS)

4.4 Application Servers

4.4.1 OAM Servers

4.4.2 Web Proxies

4.5 Network Infrastructure

4.5.1 Backbone Routers

4.5.2 DNS Servers

5 Mobile Botnets – the Next Large Scale Threat to the Mobile Business

5.1 Establishment and Operation of Mobile Botnets

5.2 Attacks Against End Users

Copyright © 2012 ASMONIA consortium. All rights reserved.
5.3 Attacks Against Mobile Networks
5.3.1 Protection of Mobile Networks Against Malicious Mobiles 170
5.3.2 Distributed Denial of Service (DDoS) Attacks 170
5.3.2.1 Scenario 1: DoS Against the Radio Interface 171
5.3.2.2 Scenario 2: DoS Against the HSS 172
5.3.3 Impact of Mobile Botnets on the MNO Business 173

6 Conclusion 174

References 178

Abbreviations 185

Revision History 189

Annex A: Threats to Mobile Networks Documented by 3GPP 192
1 Introduction

It is well known that IP based networks are exposed to various threats. Attacks on such networks are launched aiming at the theft of information, the distortion of information, destroying information or software on hosts or making information or services unavailable.

Mobile networks of the second and third generation, i.e. GSM and UMTS networks, used to be largely based on TDM/ATM transport networks in their wired part. However, such legacy transport techniques are more and more replaced by packet transport, e.g. Carrier Ethernet and IP/MPLS networks. The fourth generation of mobile communication networks, which is currently at the beginning of its deployment phase, makes full use of IP based transport networks. For example, the 3GPP specified 4G mobile network, called Evolved Packet System (EPS), applies IP based transport on all interfaces (except for the radio interface), and all user plane traffic, including voice, is based on IP, i.e. no circuit switched service is provided anymore.

Consequently, future mobile communication networks will be increasingly exposed to the various threats known for IP based networks, and there is no doubt that also new, currently unknown attacks will be mounted, targeted specifically against the network elements and services of mobile communication networks.

Similarly, terminals, i.e. mobile phones, have significantly changed during the last years. While earlier mobile phones supported voice and a limited set of data applications, today's smart phones are rather mobile computers than voice handsets. They are more and more dominated by data applications that require significant local computing resources as well as a constant interconnection to the Internet. Very similar to personal computers, they are increasingly threatened by all kinds of malware, like worms, viruses, or trojans. This is a threat not only to the users of such mobile terminals, but also to the mobile network operators (MNOs) and the community of users of the mobile network infrastructure, because powerful terminals can substantially endanger mobile communication networks, if a high number of them are abused for mounting attacks against the network. For example, a future botnet of smart phones may be able to execute a so called Denial of Service (DoS) attack, i.e. exploit some weaknesses in the implementation of a mobile communication network in a way that renders the mobile network or parts of it unavailable.

While it is rather obvious that future IP networks will be endangered by various threats, it is less clear, which threats will be the most significant ones and which network elements will be most endangered. As such knowledge is highly relevant for guiding the efforts to secure future mobile communication networks, both in research and in implementation and deployment of security measures, the ASMONIA project has taken up the task to perform a thorough threat and risk assessment for mobile communication networks and mobile terminals.

This threat and risk assessment aims at future 4th Generation (4G) networks and terminals. Officially, 4G networks are networks that comply with the requirements of the ITU-R's IMT-Advanced initiative. End of 2010, the ITU-R has decided that two technologies will be accorded the official designation of IMT-advanced:

- **LTE-advanced**, developed by 3GPP as 3GPP Release 10 and beyond
- **WirelessMAN-advanced**, developed by IEEE as Standard 802.16m

Of these two technologies, LTE-advanced is the one that is the natural evolution step for current mobile networks, including GSM and UMTS but also CDMA2000. WirelessMAN-advanced would be an evolution step for current WiMAX based networks. As
GSM/UMTS/CDMA2000 mobile networks are much more deployed around the globe, and are the predominant technology in nearly all countries, including Germany, this document focuses on LTE-advanced, or more generally, on the network architectures specified by 3GPP.

Strictly speaking, the official label 4G will only apply to 3GPP networks from Release 10. However, 3GPP Release 8 already introduced a new radio technology called LTE (Long Term Evolution) and a new system architecture called SAE (System Architecture Evolution). The LTE/SAE mobile network is called Evolved Packet System (EPS), and today, the EPS is often called a 3GPP 4G network, even if it is not yet a Release 10 EPS.

This document focuses on the EPS, but does not exclude GSM (2G) and UMTS (3G) networks. As the migration to a new radio access technology is a major effort, it must be assumed that 3G and even 2G access networks will be used still for many years, and 3GPP has specified how the Evolved Packet Core (EPC), i.e. the core network of the EPS, can be accessed via 2G/3G access networks. This document even briefly discusses 2G/3G core networks, as these will still be operational for a considerable amount of time.

In general, it is assumed that future mobile networks may comprise a mix of equipment of different generations interconnected in complex ways. So threats endangering today’s 2G and 3G network elements will continue to be relevant also in future mobile networks.

*Figure 1* gives a schematic overview of the system that is discussed in this document. Note that not all components discussed in this document are shown in the picture. All components shown in orange (bright) color are in the focus of the document; external IP networks and non-3GPP access networks (shown in darker color) are not assessed here.
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(I)-1.0

Figure 1: Schematic View of a 4G Mobile Network

On the left side, we have the terminals. (In this document, the terms terminal, mobile terminal and User Equipment (UE) are used rather synonymously. As a rule, it is not intended to express different degrees of mobility by using one or another of these terms.) There are many different types of them, ranging from devices for machine-to-machine communication over simple feature phones to smart phones and personal computers equipped with UMTS/LTE modules. A particular attractive target for attacks may be the smart phones, as these are more and more used for various data applications handling sensitive data, like location data or data transmitted by online banking or mobile payment applications.

In the area of access networks, Figure 1 shows the various options supported by a 4G network. The GERAN (GSM EDGE Radio Access Network) is the 2G access network, the UTRAN (UMTS Radio Access Network) is the 3G access network. GERAN and UTRAN each consist of different types of network elements, namely the base stations (BTS/Node B) and the respective controllers (Base Station Controller/Radio Network Controller). A 4G RAN, i.e. an E-UTRAN (Enhanced UTRAN) has a flattened architecture, i.e. it is composed mainly of nodes of a single type only, the eNB (evolved Node B). For improving the radio coverage at specific locations, e.g. homes or public areas that cannot easily be covered from “towers”, two variants, the Home eNB and the Relay Node have been introduced.

GERAN, UTRAN and E-UTRAN have been specified by 3GPP. However, 3GPP also specifies how access to a 4G network can be done using access technologies not specified...
by 3GPP, so called non-3GPP access networks. Here, a differentiation is done between trusted and untrusted non-3GPP access networks. Whether an access network is trusted or untrusted is defined by the operator; it is not a property of the access network itself. E.g., an operator may use a well secured CDMA2000 network to provide services to CDMA2000 terminals, and consider this as a trusted access to the 4G core network. In addition, the operator may allow access to the services of the 4G network via public WLAN hotspots and the Internet – this will be considered as untrusted access.

In a pure 4G network, the core network is fully packet based, and is called Evolved Packet Core (EPC). Control and user plane are handled by the Mobility Management Entity (MME) and the SAE-Gateway (SAE-GW), respectively. To support 2G/3G access to the EPC, a Serving GPRS Support Node (SGSN) is needed. The SGSN is also one of the two central components of 2G/3G packet core networks. A 3G SGSN may be used "as is" in the EPC, but 3GPP has also specified an enhanced SGSN that supports native EPC interfaces.

A core network element of central importance is the Home Subscriber Server (HSS). This central database holds the subscriber information, including authentication credentials, profile information and location information. Network elements like MME or SGSN retrieve information from the HSS when a subscriber has to be authenticated. In case of access via non-3GPP access networks, it is the 3GPP AAA Server that retrieves subscriber information from the HSS in order to authenticate subscribers. In case of access via untrusted non-3GPP access networks, an evolved Packet Data Gateway (ePDG) is used that demarcates the border between the EPC and the untrusted access network.

Charging is an important function in mobile networks. Various core components interact with charging systems to perform this function. 3GPP has also specified a Policy and Charging Control architecture, with a Policy and Charging Rules Function (PCRF) as the central element, controlling so called Policy and Charging Enforcement Functions (PCEFs) inside user plane core network elements like the SAE-GW.

Radio access networks and the packet core allow terminals to access IP based service networks. Such networks may be part of the mobile communication network, or PLMN (Public Land Mobile Network). For example, an IP multimedia subsystem (IMS) may provide voice and multimedia services. In other cases, IP based services are external to the mobile network, e.g. corporate IP networks or the Internet.

The threat and risk assessment undertaken here focuses on terminals and the network elements of a PLMN. It is very comprehensive, but still it does not cover each and every possible network element. In some cases, it provides an aggregated view only, e.g. it does not distinguish between different types of SIP proxies and SIP application servers within the IMS, but assesses the IMS as a whole.

In some areas, we carried out a number of practical tests. For example, we reconstructed some known GSM attacks like the fake base station attack or the decryption of the GSM cipher algorithm A5/1 to verify the feasibility and evaluate the effort required for such attacks (see 4.1.1.4.1 and 4.1.1.4.4.2). Other practical testing was carried out on 4G access network components and interfaces. It is documented here along with the assessment of the respective components. Practical testing has proved to be very valuable for understanding the ways how the investigated network elements may be affected by attacks.
2 Threat and Risk Analysis Method

This document mainly aims to provide an extensive overview of threats (in particular a certain subset of the overall set of threats, that are attacks). However just looking at threats ("bad things that can happen") might not sufficiently enable the reader to take well-informed decisions based on the information laid out here. We therefore decided to contribute an estimation of the risks associated with the discussed threats as well. This may help to get an understanding of the relevance of the individual threats in the context of 4G mobile telecommunication networks.

2.1 Related Work and Relevant Standards

There are a number of standard documents from different standardization bodies elaborating on threats and risks in computer or telecommunication networks. This section gives an overview of these papers.

2.1.1 ISO 27000 Family

As of [ISO27000, p. v] the family of standards assembled in the ISO 27000 series¹ "is intended to assist organizations of all types and sizes to implement and operate an ISMS" (Information Security Management System). Given the importance of risk management within the overall ISMS approach as of ISO 27001 various references to threats and risks can be found in ISO 27000 standards.

2.1.1.1 ISO 27005

ISO 27005 "Information technology - Security techniques - Information security risk management" [ISO27005] "contains the description of the information security risk management process and its activities" [ISO 27005, p. 3] which include "context establishment [...], risk assessment [...], risk treatment [...], risk acceptance [...], risk communication [...], and risk monitoring and review" ([ISO27005, p. 4]. While not pre-describing an exact methodology for risk analysis, the terminology and concepts used in ISO 27005 are widely accepted and the document is regarded as the most prevalent source as for conducting a risk based information security approach. The methodology used in the present paper is largely in accordance with ISO 27005 (see also section 2.2). The document was published in June 2008.

2.1.1.2 ISO 27011

ISO 27011 "Information technology - Security techniques - Information security management guidelines for telecommunications organizations based on ISO/IEC 27002" [ISO27011], published in December 2008, "provides interpretation guidelines for the implementation and management of information security management [sic!] in telecommunications organizations based on ISO/IEC 27002" ([ISO27011, p. vi]). Section 4.2.2, titled "Security considerations in telecommunications", lists some "environmental and operational security incidents" to be considered. ISO 27011 does not contain a dedicated discussion of threats or risks in the telecommunication networks context. Still it can be regarded as a major source for implementation and control² guidance in telecommunications networks.

¹ See [ISO27000, p. 18] for an overview of the standards included, as of 2009.
² It should be noted that implementation and control are out of the scope of the present paper.
³ It should be noted that implementation and control are out of the scope of the present paper.
2.1.1.3 ISO 27033-1

ISO 27033-1 "Information technology - Security techniques - Network security - Part 1: Overview and concepts" ([ISO27033-1], whose "purpose [...] is to provide detailed guidance on the security aspects of the management, operation and use of information system networks, and their inter-connections" ([ISO27033-1, p. vi]), replaces ISO/IEC 18028-1:2006 which in turn was based on the X.805 framework covered in section 2.1.4 of the present paper.

The document was published in December 2009 and it contains a road map "of the ISO/IEC 27033 series of standards" ([ISO27033-1, p. 10]) stating that a future document ISO27033-2 will be published based on ISO 18028-2 which encompasses a chapter on security threats (in the context of networks). The ISO18028 series of standards has never gained wide acceptance though.

2.1.2 3GPP Sourced Documents

It seems that 3GPP has not followed the approach to perform and document extensive threat and risk analyses for all parts of the mobile network. However, some 3GPP documents exist that list threats to the mobile network. The most relevant of them are discussed in this section. The "Annex A: Threats to Mobile Networks Documented by 3GPP" (page 192) provides more detailed information.

2.1.2.1 Technical Specification 3GPP TS 33.120 (Release 4)

[3GPP_TS33120] provides "the objectives and principles of 3GPP security [where these] principles state what is to be provided by 3G security as compared to the security of second generation systems [while at the same time these] principles will also ensure that 3G security can secure the new services and new service environments offered by 3G systems" ([3GPP_TS33120, p. 5]). The document does not contain a discussion of threats or risks but features a dedicated section on "Weaknesses in Second Generation security" that lays out vulnerabilities in 2G systems to be addressed in 3G networks and beyond. The document was published in March 2001.

2.1.2.2 Technical Specification 3GPP TS 21.133 (Release 4)

[3GPP_TS21133] covers "Security threats and requirements" and "contains an evaluation of perceived threats to 3GPP and produces subsequently a list of security requirements to address these threats" [3GPP_TS21133, p.6]. In this context the document defines requirements for system designs, system architecture, billing, interworking, data types and roles in 3G telecommunication services. The requirements "shall be used as input for a choice of security features" [3GPP_TS21133, p. 6]. The document was published in 2001 which means that the security discussion taking place in it does not consider attack developments since then. The approach of a dedicated discussion of security threats and requirements within the 3GPP standards was discontinued after Release 4.

2.1.2.3 Technical Report 3GPP TR 33.821 (Release 9)

[3GPP_TR33821], published in June 2009, "collects the identified threats [in LTE networks] and proposed countermeasures, and includes the design choices and rationale for why proposed security mechanisms are accepted or rejected to record the history of the final security solution". It can be considered as the most comprehensive discussion of threats relevant for the approach of the present document but mainly focuses on protocols and
algorithms. It does not take the enhanced perspective including devices that is provided here.

2.1.3 ETSI TS 102 165-1 V4.2.1

[ETSI_TS1021651] was published in December 2006 and describes a Threat Vulnerability and Risk Analysis (TVRA) as a method to identify “the required security functionality to ensure that the objectives can be met without damage to the system” [ETSI_TS1021651, p. 12]. The method is based on key elements like the relationship of system design, requirements and objectives and countermeasures. The purpose is to determine how open to attacks systems and components are and to get a method to measure “attack potential” for telecommunication networks. While the methodology described can be regarded as a detailed and useful one the document does not provide specific threats but just a mere general approach of assessing attacks.

2.1.4 ITU X.805

The International Communication Union (ITU) is an agency of the United Nations and deals with technical aspects of telecommunication services. In [ITU_X805] the ITU defines a “security architecture for providing an end-to-end network security” [ITU_X805, p. 1] to “address the global security challenges of service providers, enterprises and consumers” [ITU_X805, p. 2]. In this architecture security dimensions like authentication, integrity and availability are discussed, but also security layers, planes and threats. The discussion of threats refers to another document (X.801) which in turn does not contain any useful threat discussion for telecommunication networks.

2.1.5 ITU E.408 "Telecommunication Networks Security Requirements"

In comparison to [ITU_X805], in [ITU_E408] the ITU provides security requirements to telecommunication networks. The requirements are defined in relation to the kind of actor roles: customer/subscriber, public community/authorities and network operator/service providers [ITU_E408, p. 2-6]. With regard to security requirements the document defines a security framework to identify security risks in general and to give guidance for planning countermeasures.

2.1.6 NIST SP800-30

The National Institute of Standards and Technology (NIST) is a federal agency of the United States of America which provides general technical standards. With [NIST_SP800-30] it delivers a „Risk Management Guide for Information Technology Systems“ which presents a guide for risk assessment, mitigation and evaluation. The risk assessment methodology is presented in nine steps beginning with system characterization and threat identification to likelihood determination and control recommendations. Overall the document outlines a process and does not provide detailed guidance as for a risk assessment approach.

2.2 Risk Assessment Approach Used within this Document

In the documents introduced above, there are a number of different definitions of the term risk, without a consistent meaning and use throughout the different documents. In the following we will rely on the definitions furnished by the standard documents ISO 31000 Risk
management — Principles and guidelines (providing a widely recognized paradigm for risk management practitioners from different backgrounds and industry sectors) and ISO/IEC 27005 Information technology — Security techniques — Information security risk management (with a dedicated focus on the information security context).

ISO 31000 defines risk simply as

"effect of uncertainty on objectives"

where uncertainty is "the state, even partial, of deficiency of information related to, understanding or knowledge of an event, its consequence, or likelihood" [ISO31000, p. 2]. Given the nature of the present document (examining threats and their associated risks in telecommunication networks based on an upcoming technology/architecture that is 4G) we have to face the fact that incomplete information is available as for some of the factors contributing to an overall risk of a given threat. Accepting uncertainty as being the main constituent of risk is a fundamental prerequisite for our approach outlined below. It must be well understood that first a certain degree of uncertainty is intrinsic to dealing with risks and second that there's always a trade-off between the – given resource constraints and human bounded rationality – necessary reduction of complexity and (presumed) accuracy during an exercise of risk assessment.

[ISO31000, p. 8] emphasizes that "the success of risk management will depend on the effectiveness of the [...] framework" and [ISO31010, p. 18] concludes that "a simple method, well done, may provide better results than a more sophisticated procedure poorly done."

So, going with a simple method and thereby preserving the ability to perform exercises in a time-efficient manner, while accepting some fuzziness, might provide better results than striving for complex scenarios and calculations requiring more time to evaluate and perform.

These considerations lead to the decision not to adopt one of the methods which are publicly available or even described by standardization organizations (like e.g. the one described in [ETSI_TS1021651]). Rather, based on experiences with different standardized or proprietary approaches, the most suitable elements of these approaches were combined to form the method described in the following subsections.

2.2.1 Definitions

According to [ISO13335-4, p.4] we define a threat as "a potential cause of an incident that may result in harm to a system or organization" and a vulnerability as "a weakness of an asset or group of assets that can be exploited by one or more threats". To describe an

---

3 Based on AS/NZS 4360 which in turn is regarded as a major contribution to the mainstream concept of risk in the 20th century.

The definition of the term "risk" within ISO 31000 is taken from ISO GUIDE 73:2009 and it can be expected that future versions of ISO 27005 will incorporate this definition (and the underlying idea) as well.

4 It should be noted that the terms (and concepts) of "risk" and "uncertainty" might dispose of some duality on their own (see [COFTA07, p. 54ff.] for a detailed discussion on this). Still, we strictly follow the ISO 73 approach here.

5 Where "assessing them" is one step in "dealing with risks".

6 [COFTA07, p.29] employs the concept of a "transactional horizon" to express the inherent limitations.

7 ISO 31010] gives an overview of risk assessment techniques.
asset's overall state with regard to vulnerabilities, in the following we use the term vulnerability factor.

2.2.2 Source(s) of Threats

In general two main possible approaches can be identified here:

- Use of a well-defined threat catalogue (usually one and the same at different points of execution) which might be provided by an industry association, a standards body or a government agency regulating a certain industry sector. While this may serve the common advantages of a standards based approach (accelerated setup of overall procedure, easy acceptance within peer community etc.), [ISO31010, p. 31] lists some major drawbacks of this course of action, laying out that check-lists\(^8\)
  - tend to inhibit imagination in the identification of risks;
  - address the 'known knowns', not the 'known unknowns' or the 'unknown unknown'.
  - encourage 'tick the box' type behavior;
  - tend to be observation based, so miss problems that are not readily seen.

- Adoption of (mostly) individual threats for individual risk assessment performances, depending on the amount of available resources, the context and "the question to be answered by means of the exercise". This certainly requires more creativity and most notably experience on the participating contributors' side, but will generally produce better and more holistic results.

Within the ASMONIA project it was therefore agreed upon to follow the latter approach (individual threats, depending on context), with the stated goal of performing the identification of relevant threats "as precisely as possible and affordable". The threats used in the present document are described in chapter 5.

2.2.3 Factors Contributing to a Risk

ISO 27005 (currently, that is as of 2008) defines information security risk as the

"potential that a given threat will exploit vulnerabilities of an asset [...] and thereby cause harm to the organization".

Following this, three main factors contribute to the risk associated with a given threat:

- the threat's potential
- the vulnerabilities to-be-exploited
- the harm caused once the threat successfully materializes.

\(^8\) Looking at the description of check-lists in ISO 31010, it becomes clear that they designate what is called threat catalogues in other contexts.
Within the present document, a given threat's potential will be expressed by (the more common term) likelihood.

Furthermore there's a vast consensus amongst risk assessment practitioners that it makes sense to work with an explicit "vulnerability factor" (see also section 2.2.1) expressing how vulnerable an asset is in case a threat shows up, for two main reasons:

- When thinking about threats, this allows to differentiate between "external phenomena" (attacks happen, malware is around, hardware fails occasionally, humans make errors) and "internal conditions" ("in this environment there's protection against certain attacks", "our malware controls might be insufficient", "we don't have clustering of some important servers", "our change control procedures are circumvented too often").

- This differentiation allows for governance and steering in the phase of risk treatment ("one can't change [the badness of] the world, but one can mitigate the vulnerability [conditions]"; which then is expressed by a diminished vulnerability factor and subsequently reduced overall risk). In case of the present document this means that the reader will be enabled to modify the vulnerability factor for a particular (e.g. the reader's own) environment and still be able to get meaningful results.

In the ASMONIA context this furthermore facilitates looking at some asset's (e.g. a sample mobile telecommunications network or a sample mobile phone architecture) intrinsic properties (leading to vulnerabilities) without knowing too many details about the environment the asset is operated in.

### 2.2.4 Method of Estimation

Again, two main approaches exist:\(^9\):

- **Qualitative estimation** which uses a scale of qualifying attributes (e.g. *Low, Medium, High*) to describe the magnitude of each of the contributing factors listed above. [ISO 27005, p. 14] states that qualitative estimation may be used
  
  - As an initial screening activity to identify risks that require more detailed analysis.
  - Where this kind of analysis is appropriate for decisions.
  - Where the numerical data or resources are inadequate for a quantitative estimation.

- **Quantitative estimation** which uses a scale with numerical values (rather than the descriptive scales used in qualitative estimation) for impact and likelihood\(^10\), using data from a variety of sources. [ISO 27005, p. 14] states that "quantitative estimation in most cases uses historical incident data, providing the advantage that it can be related directly to the information security objectives and concerns of the organization. A disadvantage is the lack of such data on new risks or information security weaknesses. A disadvantage of the quantitative approach may occur where

---

\(^9\)[ISO 27000], section 8.2.2.1 provides a good overview.

\(^10\) Models with quantitative estimation don’t use a "vulnerability factor" (as this one usually can't be expressed in a quantitative way).
factual, auditable data is not available thus creating an illusion of worth and accuracy of the risk assessment."

Given that historical incident data is not yet available for 4G telecommunication networks further stresses the need to use a qualitative approach in the present document.

2.2.5 Scale & Calculation Formula Used

Each of the contributing factors (that are likelihood, vulnerability [factor] and impact) will be rated on a scale from 1 ("very low") to 5 ("very high"). Experience shows that other scales either are not granular enough (as is the case for the scale "1–3") or lead to endless discussions if too granular (as is the case for the scale "1–10").

Most (qualitative) approaches use a "1–5" scale\textsuperscript{11}.

To reflect the inherent fuzziness, within the ASMONIA project (and the present document) ranges of values can be used for each contributing factor (e.g. "1–3" or "3–4" or sth.) as well.

It should be noted that the values for likelihood and the vulnerability factor are mapped to concrete definitions. For the purpose of this document and the assessments undertaken here these are the following:

**Likelihood**

1: < once in 5 yrs  
2: < once a year  
3: < once a month  
4: < once a week  
5: > once a week

**Vulnerability Factor**

1: Extensive controls, threat can only materialize if multiple failures coincide.  
2: Multiple Controls, but highly skilled+motivated attacker might overcome those.  
3: Some control(s) in place, but highly skilled+motivated attackers will overcome those. Overall exposure might play a role.  
4: Controls in place but they have limitations. High exposure given and/or medium skilled attacker required.  
5: Maybe controls, but with limitations if at all. High Exposure and/or low skills required.

\textsuperscript{11} And so does the example 2 (section E.2.2) of [ISO27005] which can be compared to the methodology described in this document.
Given the obvious nature of the term "impact" and the absence of a specific environment with specific (e.g. financial) impact values, no scale for the impact was used. However there was a joint understanding of the experts involved as for a high impact vs. a low impact in the course of the assessments. Furthermore the "impact value" is not split into "subvalues" for different security objectives (like individual values for "impact on availability", "impact on confidentiality" and so on) in order to preserve the efficiency of the overall approach.

To get the resulting risk, all values (or their respective ranges) will be multiplied (which is the most common way of calculating risks).

The values themselves usually have been discussed by a group of experts (appropriate to the asset-to-be-evaluated). Where possible some lines of reasoning are given for each value assigned for documentation and future use purposes.

2.2.6 Reliability of the Assessments

Inherently, the assessments reflect the best knowledge of the expert groups by whom they have been done. There is no way to proof the correctness of the assessments. However, we are strongly convinced that the overall results are highly meaningful. If different expert groups would assess the same risks, we believe the results would be very similar, at least on the level of the overall risk associated to the different assets and the different threat categories.
3 Threat Categories
As none of the existing threat classifications was identified to be useful for the purpose of the present document (see above, section 2.2.2), a definition of generic threats for the assets within 4G networks was done. This section describes the threats identified and is used as the baseline throughout the remainder of the present document.

3.1 Focus of the Threat Analysis
The focus of the threat analysis is on deliberate attacks, in particular attacks carried out via networks.

- External attacks via networks comprise attacks
  - from the Internet or another connected packet data network (PDN), e.g. a corporate IP network;
  - from the GPRS roaming exchange (GRX) or other network interconnecting PLMNs;
  - from a connected PLMN;
  - from an external transport network, e.g. for mobile backhauling or interconnection of core sites;
  - from an external shared RAN;
  - from an external non-3GPP access network;
  - from interconnected equipment operated by a LEA (Law Enforcement Agency) or the networks interconnecting PLMN and LEA.

Note that the external attacker may be a user only, but may also be an administrator of an interconnected network.

- External attacks that involve physical access to network entities comprise
  - attacks on the radio interface;
  - tampering with easily accessible devices (in particular in RANs, e.g. the (Home)(e)NB);
  - unauthorized physical access to network ports (e.g. at switches in buildings hosting network equipment).

- Attacks from mobiles (against the mobile network or other mobiles)
- Insider attacks (abuse of administrator rights by malicious MNO staff)

The following picture illustrates these threats for an EPS.
Focusing on the above threats means that natural or man-made disasters are out of scope of this document. Moreover, failure of systems in absence of an attack condition is also out of scope, e.g. failure due to a memory fault. Note however that software failures typically are the consequence of some programming error that may also be exploited by a deliberate attack.

3.2 Threat Categories

There are various ways how the threats to a system can be categorized. In the following, a top down approach is given. It starts with the often quoted three main security objectives: availability – confidentiality – integrity. Loss of availability or confidentiality has an easily understandable direct "real world" (e.g. financial) impact. Loss of integrity may impact a system and its owner or user in a more complicated way – e.g. maliciously changing data may lead to all kind of negative consequences (called "unwanted actions" below), depending on the meaning and use of the data. Such consequences comprise loss of availability and loss of confidentiality, meaning that the separation between the three main security objectives becomes somewhat blurred.

The list below comprises three additional threat categories in a group called "loss of control". They relate to compromise of network elements by external attackers or abuse of network elements by insiders. Threats in these categories can affect all three main security objectives. They have been explicitly listed to take account of their specific significance.

An important threat related to a mobile network is theft of service. This is mostly facilitated by some break of integrity (e.g. of charging data) or confidentiality (e.g. of data that allow to impersonate another user).
Loss of availability:

T1 Flooding an interface
   Attacker causes DoS in a network element by sending a flood of packets (overload).

T2 Crashing a network element via a protocol or application implementation flaw
   Attacker crashes a network element by sending maliciously crafted packets (exploit of a flaw in the protocol implementation) or by exploiting a weakness in a user application on the network element. The consequence is DoS with respect to the functions of the network element.

Loss of confidentiality:

T3 Eavesdropping
   Attacker eavesdrops sensitive data during transit (traffic). This includes gaining information by analyzing encrypted traffic (e.g. monitoring timing, packet length, traffic volume etc.) without being able to decrypt it.

T4 Unauthorized access to sensitive data on a network element via leakage
   Attacker gets access to sensitive data on the network element by triggering leakage of such information (e.g. by abusing network protocols or exploiting a flaw in a user application on the network element).

Loss of integrity:

T5 Traffic modification
   Attacker modifies/fakes information during transit and by this causes some unwanted action (e.g. of the network element receiving the faked information).

T6 Data modification on a network element
   Attacker modifies information on a network element or inserts faked information on a network element (e.g. configuration data) and by this causes some unwanted action (triggered when the faked information is used somehow).
   May be done by abusing network protocols (e.g. unauthenticated TFTP upload) or by exploiting a flaw in the user application on the network element.

Loss of control (compromise/abuse of network elements)

T7 Compromise of a network element via a protocol or application implementation flaw
   Attacker exploits a flaw in protocol stacks or applications of a network element to gain full control over the network element and abuses the network element to perform unwanted actions subsequently.

T8 Compromise of a network element via a management interface
   Attacker exploits a weakness in the configuration of the management interfaces (e.g. weak/default passwords, access to OAM applications via publicly accessible interfaces) to gain full control over the network element and abuses the network element to perform unwanted actions subsequently.
T9 Malicious insider

Attacker abuses access rights/privileges on a network element assigned to him/her as part of his/her job function and performs unwanted actions

Theft of service

T10 Theft of service

Attacker exploits a flaw (e.g. in the authentication and authorization mechanisms or within the charging procedures) to use services without being charged.

Concerning loss of confidentiality or integrity (T3 – T6), it may be differentiated between different classes of data, in particular user plane data versus signaling/control and management data. The impact of a successful attack is expected to depend on which class of data is affected.

Modification of control or management data may result in loss of availability (e.g. a network element shuts down) or loss of confidentiality (e.g. a network element reveals sensitive data).

Modification of user plane data may cause unwanted "real world" actions that affect users heavily, like fraudulent bank transactions. However, it is assumed that highly sensitive user data do not rely solely on the security provided by the mobile network. For example, electronic banking may use TLS between the terminal and the banking server.

Many network elements of a mobile network handle user plane traffic, but typically do not store it for a significant amount of time. So user plane traffic is mainly at risk during transit. Therefore, for network elements handling user plane traffic, T3 and T5 are subdivided into control plane (T3.1/T5.1) and user plane (T3.2/T5.2), allowing to address the two cases separately.

Compromise/abuse of network elements (T7-T9) means that the attacker may perform various unwanted actions, including DoS against the compromised network element, access to confidential information on the network element, modification of data on the network element, or attacking other network elements, in particular those that have a trust relationship to the compromised/abused network element.

The malicious insider threat (T9) as stated above focuses on malicious persons operating the asset. The likelihood of a malicious insider is assumed to be relatively low, but may depend on (hard to influence factors) like the social and cultural context.

Abuse of network elements by a malicious insider cannot be prevented solely by technical means. It may be made somewhat more difficult however by enforcement of secure OAM procedures, including logging (so the attacker may fear to be detected afterwards). But despite such means, the vulnerability against the malicious insider threat mainly depends on organizational processes within the operator organization, and on operational practices with respect to the network operation. Such organizational aspects are not the main focus of this document, however.

There can also be malicious insiders within the supply chain, e.g. within product development departments of the equipment manufacturer, or in service companies installing the software on network elements etc. Such insiders would be able to insert malicious functions into the systems, like backdoors that allow unauthorized access and control of network elements in the field. The vulnerability against this threat is generally very high, as effective technical means against sophisticated backdoors are not known. For example, there is no reliable way
known to detect such backdoors inside complex network elements as used in mobile networks.

Concerning likelihood and impact of such malicious functions, we assume similar values as for the operative malicious insider threat (T9). As a rule, in this document for each asset only T9 is assessed explicitly – the risk caused by malicious functions inserted into the system by a malicious insider in the supply chain may be derived from the T9 assessment by setting the vulnerability to 5, i.e. calculating this risk as:

\[ \text{likelihood\_as\_assessed} \times 5 \times \text{impact\_as\_assessed} \]

### 3.3 Application of the Threat Categories in Assessments

In the assessments of the risks of the mobile network elements in the following chapter, for each asset each threat category is assessed according to the method described in chapter 2.2 as detailed in the following:

“Likelihood” is estimated as the probability that any specimen of the respective asset type is attacked, e.g. the likelihood that any of the SAE-GW within the network is attacked.

“Vulnerability” is estimated as the average vulnerability of a specimen of the respective asset type, where the average is meant over all deployments (including different manufacturers, different operators, different configurations). This average may need to capture a considerable variety, e.g. an SAE-GW may or may not

- provide an inter-PLMN interface
- support non-3GPP access
- terminate IPsec from the backhauling link
- use encrypted management protocols only
- …

“Impact” is estimated primarily as the impact on the network from the operator’s point of view, which is in many cases also the point of view of the community of network users. An example for this is the availability of the network, which is essential for the operator as well as the users. A counterexample is when attackers gain access to services free of charge, which may heavily impact the operator but doesn’t matter to the user community.

For the threats addressing specifically the user-plane (T3.2 and T5.2) the direct impact is on users, not on the network. For a single user, the impact can be high, although we assume that highly sensitive user plane data are protected independently (see reasoning in the closing paragraphs of section 3.2). However, loss of confidentiality or integrity of user plane traffic has also an impact on the operator, which we estimate – as a rule – to be in the range from 1 to 3 according our scale described in section 2.2.5.

Note further that the list of threat categories given above is meant to be a baseline for the risk assessments for the various assets covered in this document. Where appropriate, these generic threats are enhanced by asset specific threats in the respective sections of the document.

In version (I) of this document, also the terminals were assessed applying the above scheme. However, this has not turned out to be very useful during the further work and therefore has been abandoned in this version of the document.
4 Threat and Risk Analysis per Asset

[3GPP_TS21133] defines categories of threats against a 3GPP mobile network and discusses how they apply to the following three different parts of a mobile network:

- Terminals and UICC/USIM;
- Radio interface;
- Other part of the system.

Following this method, the present document divides the 4G mobile network into several parts on the basis of physical network entities and structures the threat analysis accordingly. The following major parts are distinguished:

- Terminals
- Access network: Radio interface and 3GPP specified entities in radio access networks like the eNB
- Core network: Core network interfaces and 3GPP specified core network entities (in the MNO domain)
- Application servers: Non standardized application servers for OAM, billing and end user applications
- Network infrastructure: Non standardized network elements operating on layer 3 and below like IP routers or MPLS switches, as well as respective network control servers (e.g. DNS servers).

The subsections of this chapter follow this structure.

4.1 Terminals

Terminals in a 4G mobile network are a very heterogeneous mass of devices. Even though individual devices always consist of two core components:

- the **baseband** part (called "baseband" in the following) and
- an application that uses the baseband.
Figure 3 shows four common terminal architectures and how these two components are usually combined. On the left side two common shared CPU architectures are shown. These can be normally found in cheap mobile phones. The first one (a) is typically for old feature phones, in which the application part is mainly a GUI to handle calls and SMS and is integrated into the baseband software stack—usually without any memory protection, which leads to a high vulnerability at this point. The second architecture (b) is used for example in the Motorola Evoke QA4 which uses an L4 microkernel with a paravirtualized BREW\textsuperscript{12} baseband stack and a Linux based application stack. On the right side two common multi-CPU architectures are shown. The first one (c) is a common architecture for smart phones and 3G USB modems connected to laptops via a serial line or shared memory. The second architecture (d) is also common for smart phones. In this shared RAM architecture the baseband CPU is usually the master and the application CPU is the slave which means that the baseband software stack may have full access to sensitive data of the smart phone operating system. This leads to a higher risk, as the impact of a successful attack on the baseband part is higher.

4.1.1 Baseband Part

Today’s cellular baseband software stacks are usually backward compatible to GSM and GPRS and have accordingly historically grown since the 1990s. Therefore, these standards need to be considered as well in this part. Figure 4 shows the development of the number of

\textsuperscript{12} http://brew.qualcomm.com
mobile subscriptions subdivided to 2G and 3G parts to illustrate their relevance for mobile devices today. Although GSM was developed in the 1980s, several security concerns were taken into account that was not as obvious as they would be today. However, the GSM specification is not adequate for today’s security requirements, because most algorithms are proprietary and lack public scrutiny, and there is no authentication of the GSM network by the GSM user.

To give an overview of expected basebands in a mobile network, Table 1 shows the current market shares of common baseband suppliers by shipments in Q3 2009. However, the market shares of the organizations who have implemented the software stacks are not publicly available and implementation details are only known to a very small group of engineers. Therefore this section and the corresponding analysis focus mainly on conceptual aspects. Besides this, these market shares are relevant when it comes to an assessment of buffer overflow vulnerability exploitation.
Another important issue which should be taken into account, is that the baseband software stack in many devices is not updated regularly, if ever. This leads to an increased vulnerability. A more concrete estimation of the code quality and therefore the vulnerability of these implementations will be part of the next version of this document.

### 4.1.1.1 Hardware and Software Architecture (Modem)

The baseband hardware in terminals consists of the following parts:

- **RF Frontend**
- **Analog Baseband**
- **Digital Baseband**

For this risk analysis only the digital baseband is considered. The only relevant threat for the other parts is radio jamming, which cannot be prevented by measures on usual low cost terminals.

Besides a DSP for signal processing the main component of the digital baseband usually is an ARM SoC running a small real-time operating system which is responsible for parts of layer 1 and everything above:

- **Layer 1**: hardware specific physical layer
- **Layer 2/3**: hardware independent, nested implementations of all 2G/3G/4G layers

### 4.1.1.2 Subscriber Identification Module (SIM)

Another essential part of the baseband system is the **Subscriber Identity Module (SIM)** which is a cryptographic smartcard and issued by the provider. The SIM holds in its ROM, besides an operating system, the security algorithms for authentication and key generation. In its EEPROM it holds data for providing anonymity, namely the **International Mobile Subscriber Identity (IMSI)** and **Temporary Mobile Subscriber Identity (TMSI)** and a secret $K_i$, which is shared with the provider. Extracting the data, including the secret key, may allow an attacker to create a cloned SIM card. This was a significant threat in the 1990s (see section 4.1.1.4.2 for details).

The mobile device or rather the software running on it can be restricted to work only with SIM cards that fulfill specific requirements. This is generally called a **SIM Lock.** For example a

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Texas Instruments</td>
<td>26%</td>
</tr>
<tr>
<td>Mediatek</td>
<td>24%</td>
</tr>
<tr>
<td>Qualcomm</td>
<td>19%</td>
</tr>
<tr>
<td>Infineon/Intel</td>
<td>11%</td>
</tr>
<tr>
<td>ST-Ericsson</td>
<td>10%</td>
</tr>
<tr>
<td>Broadcom</td>
<td>3%</td>
</tr>
<tr>
<td>Freescale</td>
<td>2%</td>
</tr>
<tr>
<td>Others</td>
<td>5%</td>
</tr>
</tbody>
</table>

*Table 1: Cellular Baseband Shipments According to Strategy Analytics*
mobile phone could be limited to the use of cards belonging to a specific network, which is called Net Lock, or to a specific country. Most restrictive is a lock to an individual SIM card which is delivered together with the phone. Due to the great interest of end users in removing these restrictions, we have a high likelihood value for such attacks.

Another aspect is that the communication between the SIM card and the mobile device is not encrypted. This allows an attacker with physical access to the device to implement a man-in-the-middle attack, e.g. to abuse SIM Application Toolkit functions such as SMS sending.

4.1.1.3 Security Goals of GSM

In this subsection we explain the security services of GSM in more detail. Further information to the security goals can be found in [Pagliusi2002]. Security services provided are:

- Authentication
- Confidentiality
- Anonymity

4.1.1.3.1 Authentication

In GSM the network authenticates the MS, but the MS does not authenticate the network. A message flow for the authentication is given in Figure 2. The idea is that the network sends a 128 Bit random number to the MS. The MS is asked to send the response SRES of the A3 algorithm with parameters RAND and Ki back to the network. SRES might be the first 32 bits of the COMP128 algorithm, which is executed on the SIM.

![Figure 5: Authentication Process of a MS to the Provider Network](image)

After the MSC received the response of the MS it checks the result and, if it is correct, the MS is considered to be successfully authenticated. So the security of the authentication is based on the secret Ki and that it is impossible to derive Ki from one or many RAND, SRES pairs, because at this point there is no encryption provided and an attacker is able to eavesdrop the communication.
GSM was defined to be highly flexible and a worldwide standard and thus it is possible, that a MS needs to authenticate to an unknown PLMN. This provider doesn’t know the shared secret $K_i$ and is therefore unable to authenticate the SIM. To overcome the limitation of just being able to use the own providers network, GSM defines triplets as a mechanism to enable any network to authenticate any SIM. This happens via a three-tuple (triplet). A valid triplet consists of the challenge $RAND$, the signed response $SRES$ and a session key $K_c$. All network providers support each other with triplets if they are requested, thus making it unnecessary to transmit at anytime the shared secret $K_i$.

### 4.1.1.3.2 Confidentiality

Because data is transferred over the air between MS and BTS it is crucial for confidentiality to encrypt the data. GSM defines several algorithms to deal with confidentiality. At first a non-encryption mode A5/0 is specified, serving as a fall back mode if otherwise no communication could be established. The algorithm A5/1 is for actual encryption. A5/2 is a weaker algorithm than A5/1 to meet encryption limitations of several countries.

![](image.png)

**Figure 6: The A5/1 Encryption Algorithm. Source [Biryukov2003]**

A5/1 and A5/2 are executed on the ME and both are stream ciphers taking a 64 bit key and a 22 bit frame number as input. Both algorithms are built with LFSR, namely A5/1 (see Figure 3) has three LFSR and A5/2 has four of them. For a detailed description the reader is referred to [Person1999, Yousef2004, Biryukov2003, Barkan2008]. To avoid that the shared secret $K_i$ needs to be used for these algorithms, there is - after authentication - a session key $K_c$ created. In the GSM specification the algorithm to derive $K_c$ is referred as A8. Like A3 it is executed on the SIM and is proprietary for each operator. But it turned out, that just like for authentication the COMP128 algorithm was used in some cases [Briceño1998]. Whether this algorithm is still used, can’t be told for sure, because providers may have switched to other algorithms and haven’t published that [Nohl2010].
Before a phone call, the MSC sends via the BTS its supported cipher algorithms to the MS and the MS is free to choose one of them. For every phone call a new $Kc$ should be created, but a $Kc$ doesn't lose its validity if the MS enters a new VLR.

The strengths of A5/1 and A5/2 algorithms are that encryption happens on layer 1, resulting in encryption of every bit. Furthermore it is possible to simultaneously encrypt and decrypt, thus enabling full-duplex communication.

A5/1 and A5/2 are the algorithms that are supported by the majority of today's terminals. 3GPP has also specified A5/3, based on the Kasumi method. No attacks of any practical importance are known against A5/3. Although A5/3 is implemented on most network equipment shipped today, it is hardly used yet. As long as A5/1 and A5/2 remain valid options, active attackers (e.g. attackers faking a base station, see 4.1.1.4.1) will always have the possibility to enforce the usage of one of them, so the existence of the stronger A5/3 does not help against such attacks.

4.1.1.3.3 Anonymity

It is remarkable, that the GSM specification deals with anonymity, because at 1990 anonymity was not topic of discussion. The goal is to prevent an attacker to create profiles of GSM users by observing the resources being used, tracing a user's location and matching user and data transmitted. For this goal, usually not the world-wide unique IMSI is used to identify a subscriber, but a temporary id called TMSI. The IMSI is fixed for the whole lifetime of the SIM card and serves for the first communication to the provider and whenever the TMSI doesn't work. For the first communication to the provider, the MS sends its IMSI (in plain-text) to the provider and after authentication the provider sends an encrypted TMSI back.

The MS stores the TMSI and sends it to the MSC in three cases. First if the MS was turned off and needs to be registered again, second if the MS moves into the area of another MSC, and the last case is on explicit request of the MSC. In every case the last TMSI, respectively the IMSI, needs to be sent in plain-text to the BTS and a new TMSI is sent after authentication and thus encrypted. Since a new TMSI is a random number and is always sent encrypted, there is no way of learning a new TMSI from the old one. More information about anonymity can be found in [Vedder1998].
4.1.1.4 Attacks on GSM Security

There are a lot of possible attacks on GSM security and it is common sense, that GSM doesn’t meet high security requirements. Most attacks arise out of weak algorithms being used and several architecture flaws. Most attacks that have been described are just theoretical, showing that the promised security goals can be compromised. Reasons for that are on the one hand law restrictions forbidding such attacks, and on the other hand that the described attacks are still very complex. But recently several attacks have been published, that were deployed in a laboratory environment and show that the former theoretical work was correct. In this subsection we give an overview about the different attacks on GSM security.

4.1.1.4.1 Faking a BTS

GSM security suffers from a lot of bad architecture decisions. As already mentioned, the algorithms A3, A5 and A8 are proprietary, and operators did not publish the algorithms they used. It is common sense, that any attempt to gain security through obscurity almost never ends up in a real plus of security. Quite the contrary is true, because keeping an algorithm obscured disables the security community from having a detailed look on the algorithm and thereby identifying and reporting possible flaws.

The most important flaw in the GSM architecture regarding security is the absence of authentication of the network by the MS. This absence can be explained by the simple fact, that in 1990 nobody imagined, that an attacker would be able to impersonate a network by setting up a faked base station (e.g. no one imagined that prices for an actual BTS would drop so low, that nearly everybody could afford it and that software to run that hardware would be also available). This false estimation is in fact the biggest burden on mobile communication security nowadays. This problem gets sharpened by the policy that any GSM cell phone connects always to the BTS with the most intense signal. An attacker having a BTS to its disposal is able to force any GSM device to connect to it by simply providing the strongest signal. Furthermore, the attacker may be able to configure his BTS in a way that it connects to a genuine BTS (i.e. a legal mobile network) and thereby perform a man-in-the-middle attack. At this point several attacks are possible. The biggest threat might be the possibility of an attacker to receive all data sent by the MS in plain-text [Pagliusi2002]. To achieve this, the idea is to exploit the fact, that a BTS is free to propose any cipher mode possible and the MS mostly just follows by – so the attacker configures the faked BTS to propose A5/0, which means no encryption.

Furthermore with a faked BTS it is possible to perform a DoS attack, by simply not reacting on a MS request to start a call. Further attacks regarding a faked BTS are discussed in section 4.1.1.4.3 and section 4.1.1.4.5.
In order to get an understanding of the required resources for the mentioned attacks and the effects for this risk analysis, a private GSM network has been setup and the attacks have been reproduced. The hardware used is a device called nanoBTS, a small BTS which is produced in different versions by ip.access and sells for about 3300 Euro. The nanoBTS has an Ethernet port as power source (Power over Ethernet – PoE) as well for configuration and communication with the Base Station Controller (BSC) which is done via the A-bis over IP protocol. Instead of a commercial BSC a PC running the open source software OpenBSC\textsuperscript{13} is used to operate the nanoBTS. OpenBSC offers a lot of options, for example the configuration of the network parameters, the channel layout and whether encryption is used or not. With these the faking of a BTS belonging to an arbitrary network provider is easily possible. In the test just being the strongest BTS in range was sufficient to have many mobile phones trying to connect and in doing so revealing their IMSIs. This together with the deactivation of encryption, which is indicated to the user only by very few mobile phones and is thus typically not realized by the user, is the first step to a man-in-the-middle attack and a working DoS attack like described above. The know-how needed to build up such a setup is limited to some basic knowledge regarding Linux and its network configuration. Additionally the GPRS support can be enabled relatively easy by the use of OpenGGSN\textsuperscript{14}. With this not only the circuit switched (voice) traffic, but also the GPRS traffic (i.e. data traffic) going over the nanoBTS can be analyzed.

In summary it can be said that an attack by faking a BTS can already be realized with very limited financial and intellectual resources and can have a big impact on the security of the

\textsuperscript{13} http://openbsc.osmocom.org/
\textsuperscript{14} http://sourceforge.net/projects/ggsn/
GSM network, especially for small networks. In fact, these attacks were already successfully utilized in areas of political crisis.

**4.1.1.4.2 Attacks on COMP128**

As described in section 4.1.1.3.2 the COMP128 algorithm was or might still be used for both authentication and session key generation. So any attack on COMP128 would immediately compromise these GSM security features. Furthermore COMP128 was never published so there was no review by the cryptography community as with other security related algorithms. Goldberg and Briceno found in [Goldberg1998] that COMP128 suffers from a lack of diffusion and were thus able to determine the secret key $K_i$ by sending 160,000 random numbers via a SIM card reader to the SIM and observing the $SRES$ returned by the SIM. The number of 160,000 challenge responses seem huge, but they were able to mount the attack in around 8 hours. This amount of time is for several scenarios reasonable, like for cell phone repair shops. An implementation of this attack can be found on http://ftp.ccc.de/software/gsm/.

The impact of this attack [Goldberg1998] gets even higher with the result in [Goldberg1998a]. Goldberg and Briceno found, that the same attack can also be performed over the air with a faked BTS. This means, that an attacker wouldn’t even need to have physical access to the SIM card to perform the attack. The over the air attack would of course take more time, but the 160,000 needed challenge responses need not necessarily be controlled in one session, but can be collected over a longer period. Measurements showed, that over the air attack needs about 13 hours of constant $SRES$ requesting. This is reasonable, because the $K_i$ never changes.

Because the SIM is nothing different than a smart card any possible attacks in this field gets an impact on GSM security. In [Rao2002] the authors describe how through side channel attacks on several COMP128 implementations the secret key $K_i$ can be revealed, by exploiting vulnerabilities in the execution of COMP128 table lookups. The number of needed requests for a response of the SIM depends on how the requests are chosen. In the simplest case 1,000 arbitrary $RAND$ are sent to the SIM. This amount of requests can be reduced to 255 if the numbers sent to the SIM are chosen appropriately. The least number of requests are needed if the next random number is adaptively chosen depending on the results of previous request. In that way the amount of requests needed can be reduced down to 8. The attacker needs of course physical access to the SIM card, but with a duration of the attack of nearly a minute and with the result of $K_i$ it’s one of the practical attacks on GSM security.

The impact of the described attacks is enormous, because the secret key $K_i$ is revealed and with that every security precaution fails. It is even possible to clone the SIM thus impersonate the respective subscriber, i.e. to do calls on that subscriber's bill or to eavesdrop the communication of this subscriber. However, today COMP128 is probably not widespread any more.

**4.1.1.4.3 Attacks on Anonymity**

In [Yousef2004] the author describes how IMSI and current TMSI of an arbitrary MS can be revealed, by faking a BTS. This happens by a man-in-the-middle attack. They exploit, at the side from MS to false BTS, that an MSC is allowed to ask for the IMSI at any time. Thereby the false BTS learns the IMSI. At the side false BTS to genuine BTS, they exploit that a MS is in general free to reject the cipher algorithms A5/1 and A5/2 and thus forcing the communication to be not encrypted. This results in the BTS sending the TMSI in a not
encrypted message and thereby the false BTS learns that either. For more details see [Yousef2004].

### 4.1.1.4.4 Attacks on Confidentiality

A lot of work was published on attacking the A5/1 and A5/2 algorithms. Especially for A5/1 because A5/2 is a very weak encryption algorithm and it was even possible to break it with cipher-text only attacks.

The confidentiality of GSM is protected by the session key $K_c$, which is a 64 bit key. This means that the complexity of a brute force attack is $2^{64}$. As it turned out, the upper 10 bits of the algorithm input are set to zero, resulting in a brute force complexity of $2^{54}$. So every attack needs to be below this to be reasonable.

#### 4.1.1.4.4.1 Theoretical Work on Attacking A5/1

The first attempt was done by Anderson in [Anderson1994]. He found out, that on guessing the lower half of every register in A5/1 the second half of the third register can be calculated by 64 bit of available key stream for a known plaintext. This results in a worst case complexity of $2^{52}$ but is still faster than the brute force attack.

In [Golic1997] Golic published a divide-and-conquer attack on A5/1. If 64 successive bits of the key stream are known, parts of the registers in A5/1 can be guessed and the overall complexity of breaking is $2^{40.16}$ steps. It needs to be mentioned, that every step includes solving big linear equation systems and thus this attack is not as feasible as it sounds.

Biryukov, Shamir and Wagner published in [Biryukov2003] a time-memory trade-off attack on A5/1. The speed of this attack depends on the amount of data to be precomputed and the known key stream. If 292 GB precomputed data and two seconds of known key stream is available, there is a 60% chance, that the algorithm is able to determine the internal states of the registers and thus get $K_c$ in a couple of minutes. This attack is a fine work of crypto analytic work, but it’s not reasonable in practice, because using a false BTS is much easier. Still this attack is the only attack confirmed by the GSM, to be done on a correct implementation of A5/1 [Briceno1998].

Instead of a time-memory trade-off Ekdahl uses in [Ekdahl2003] correlation attacks to avoid having an exponentially complexity in the register length. The runtime of the proposed attack depends on how many clocks the encryption algorithm made before producing the first output. However, a needed plain-text length of 2 to 5 minutes makes this attack in practice very difficult.

A ciphertext-only attack on A5/1 is provided in [Barkan2008]. The authors exploit, that GSM does some error correction before the actual encryption. With that it is possible by observing the cipher stream to reveal some linear combinations of certain values. However, this attack needs 3 minutes on a given cipher stream to precompute 50 TB of data and results in a 60% chance to reveal $K_c$.

In the end, these attacks are not reasonable in practice, because using a false BTS is much easier.

#### 4.1.1.4.4.2 Implemented Attacks on A5/1

Keller and Seitz provide an attack that was actually run on hardware in [Keller2001]. They basically implemented the idea of [Anderson1994] on a XC4062 FPGA. With 7 instances of
the algorithm they were able to break the decryption in 236 days. They speeded the algorithm by cutting-off the search tree, but in [Gendrullis2008] it was shown, that the success rate of this attack is only 18%.

Gendrullis, Novotný and Rupp implementing in [Gendrullis2008] on a special hardware (COPACOBANA) an attack, that needs in average 6 hours and 12 hours at maximum. The described attack is a known-plaintext attack and needs 64 bit key stream, which is feasible because there are several status messages encrypted [Nohl2010], thus knowing the plaintext.

Another real world attack was published by Nohl [Nohl2010]. The described attack is basically a highly optimized time-memory trade-off attack. To overcome the typical bottleneck of such an attack, namely the storage lookups on the hard disk, they are using distinguish points. Furthermore they are using rainbow tables to avoid having too much redundancy in the precalculated data. To further improve the calculation time GPUs instead of CPUs or FPGAs were used, because GPUs offer high parallelization out of the box. Another speedup was found by analyzing the internal states of A5/1 and discovering, that the next internal state is of course not completely independent from the current state. On precomputation side they did some effort to minimize the needed memory and thus they were able to hold an index of the complete database in 2 GB RAM. The project software and several trade-off settings for deploying the attack are offered at the project web page. With that software and two encrypted known-plaintext messages, they were able to reveal the encryption key $K_c$ on 2 GPUs with about 90% probability. The amount of precalculated data was about 2 TB, which was calculated in one month.

15http://reflextor.com/trac/a51/
The attack by Nohl [Nohl2010] has been reproduced for this risk analysis in order to analyze the financial and technical resources needed nowadays to eavesdrop a GSM phone call or short message with open source software. The hardware used for this purpose was a Standard PC with an Ubuntu Linux Operating System and an external 2 TB HDD attached via eSATA holding the rainbow tables. To intercept the GSM radio signals a *Universal Software Radio Peripheral 2* (USRP2) by Ettus Research was used, which can be purchased for about 1400$. The USRP can be equipped with different daughterboard's which serve as RF-Frontend. In this case a DBSRX daughterboard supporting the reception of 800 MHz to 2.4 GHz signals was used which is sold separately for about 150$. Together with the PC and the HDD the total costs are below 2500$ which can be assumed to be affordable even by semi-professional attackers.

### Decrypting GSM Data

The first step is to gain access to the rainbow tables, which have a size of about 1.6 terabytes. The tables are available as download via BitTorrent. At the time the tables were downloaded for the test setup, they were highly available with download rates over 20 megabytes per second. Karsten Nohl published[^16] a tool called “Kraken” which uses the

[^16]: http://srlabs.de/research/decrypting_gsm/
rainbow tables to realize his attack ([Nohl2010]). Together with the tool itself he delivers miscellaneous helper scripts and programs. One of these scripts does the initial setup by converting the downloaded tables to a usable format and copying them on one or more HDDs. The distribution of the tables on the HDDs has to be configured via a configuration file. The HDDs are accessed as block devices and therefore only partitions without file system must be prepared on them. The script also generates index files which are loaded to memory when “Kraken” starts and allow faster searching for the needed chain. The program takes 114 bit key stream and in the case of success yields states of the A5/1 linear feedback shift registers producing the key stream (see also 4.1.1.3.2). The states can then be back clocked with another tool to get the session key Kc. Key stream is obtained by a bitwise exclusive-OR operation on the known-plaintext and the corresponding cipher text. It is important to realize that the 90 percent success probability given in [Nohl2010] is calculated with the assumption that two messages are available as plain text – cipher text pair. Since one message consists of four bursts with a length of 114 bits this equates 912 bits. Assuming the correctness of the given probability, the success probability for cracking one single burst is somewhat above 10 percent. These numbers conform to the tests made with the test setup. Additionally the cracking is hindered by bit-errors emerging during transmission. These occur in the plain text, which can be dealt with, as well as in the cipher text, where an erroneous burst results in an erroneous key stream and therefore yields a wrong key or no key at all. On our test setup described above cracking one burst takes about 115 seconds. Eight bursts must be cracked to reach the 90 percent success probability mentioned earlier, which results in about 15 minutes to get one session key in the worst case. Bit-errors in the cipher text cannot be recognized before decryption so erroneous bursts will be used for cracking although they cannot yield correct results. These are not included in the calculation above and cost extra time. The test setup used is a very cheap one and the time for cracking one burst can be strongly reduced for example by using a raid of fast SSD-HDDs and the computation support of an ATI graphics card.

**Intercepting GSM Data**

The other part of the attack is about getting the data which is then to be cracked. Therefore some different programs and the USRP2 are used. The first one called Airprobe consists of different scripts and programs, is open source and publicly available. Airprobe uses GNU Radio to record a whole ARFCN downlink, which means all timeslots on one frequency. Therefore given that it is a non hopping ARFCN all communication of their active users like voice calls or SMS can be recorded at once. The recording step results in a raw data file which is the input to another part of the Airprobe toolset. This program deals with the demodulation, protocol parsing and decoding and it is possible to use an up-to-date version of the widespread network analyzing tool Wireshark with the GSMTAP dissector to visualize all packets that were decoded successfully. All the bursts which are not decoded – probably because they are encrypted – are written out to a text file together with their frame number and other information. The unencrypted information can be used to find for example the call of interest. Then corresponding plain text – cipher text pairs have to be found manually. This is the most complex step because it is not fully automated and it requires some knowledge about the GSM protocol and the logical channel layout of the ARFCN recorded. With these pairs key stream material can be produced as described above and be fed into the “Kraken” program to get the session key. With the key the decoding part of
Airprobe is being rerun. Now the program decrypts the bursts encrypted with the given key and in case of a traffic channel with voice communication an audio file is written to disk.

Summarizing and in spite of the mentioned problems one can say that the described attack is applicable in real-world scenarios, especially if more money is spent on a faster setup which is able to crack one burst in some seconds. The know-how needed to realize the attack is manageable if one has some skill at programming and Linux and is willing and able to put time in understanding some details of the GSM standard. There have been recent developments in using a cheap feature phone like the Motorola C123 with the custom firmware OsmocomBB\(^{20}\) as a replacement for the expensive USRP2 for intercepting the GSM data. Such phones can be purchased already for about 10 EUR and the overall cost of the setup reduces accordingly. Another advantage besides the low price is that these phones are able to change radio frequencies very fast, which can make it possible to listen to up- and downlink of a channel at the same time or follow the channel hopping respectively.

On the other hand this attack is only feasible because of the known plain text of some data. Implementing some randomness in the network side at this point (e.g. for padding, see [3GPP_TS44006]) or just not sending this data again encrypted would be a very simple countermeasure against this attack.

### 4.1.1.4.4.3 Theoretical Work on Attacking A5/2

The A5/2 algorithm is much weaker than A5/1. This results in easier and more powerful attacks. The first step was done by Goldberg, Wagner and Green in [Goldberg1999]. With access to the used key stream of two frames, that are exactly \(2^{11}\) frames apart, they were able to find the key \(K_c\) in 10\(\text{ms}\).

In [Barkan2008] a known-plaintext attack and a ciphertext-only attack on A5/2 were published. The latter doesn't allow real-time decryption but still is feasible. On a modern PC of the year 2003 the precomputation part can be done in 5.5 hours and on a 500 MB hard disk. The encryption key \(K_c\) can then determined in less than a second.

### 4.1.1.4.5 Other Attacks

Like for the possibility to fake a BTS as described in Section 4.1.1.4.1, there are several other security flaws in GSM, which have conceptual reasons.

At first, as described in Section 4.1.1.3, the communication between MS and BTS might be encrypted. But the communication between BTS and BSC might not be encrypted. Most people involved in the specification process assumed, that this communication would happen through a cable which would be lain underground. As we know now it is more common, that this communication happens via point-to-point radio systems, which can easily be eavesdropped. By doing that, an attacker would get access to the communication of all MSs using the respective BTS in plaintext.

As described in Section 4.1.1.3, GSM specifies so called triplets to enable a world-wide usability of GSM. These triplets do never lose their validity, which means that a MS consistently accepts the same triplet. If an attacker possesses such a triplet he could break \(K_c\) or even \(K_i\) with a brute force attack. However, triplets are never sent to MSs, so it would be rather hard to acquire one.

4.1.1.5 UMTS Security

UMTS was designed to be compatible with the existing GSM and GPRS architecture. Still UMTS provides the following enhancements to avoid attacks like on the GSM infrastructure [Blanchard2000]:

- Mutual authentication
- Assurance that authentication information and keys are not being re-used (key freshness)
- Integrity protection of signaling messages, specifically the secured encryption algorithm negotiation process
- Use of stronger encryption by using longer keys and better algorithm design
- Termination of the encryption further into the core network to encompass microwave links

To deliver appropriate network access security the following mechanism are defined [Xenakis2006]: user identity confidentiality, authentication and key agreement and data confidentiality. All of them rely on different notation and algorithms then in the GSM architecture. Additionally UMTS defines integrity protection of signaling messages to avoid several security risks like the ones described in Section 4.1.1.4.1.

4.1.1.6 Attacks on UMTS

Attacks on network access security are so far not known, and no software has been published so far for real attacks on the UMTS security infrastructure. In [Kambourakis2010] Kambourakis et al. explain several possible DoS by exploiting unencrypted and unauthenticated signaling messages. For their attacks they assume that an attacker holds a false BTS or a modified MS. An attacker is able to intercept a valid session of others, sniff and replay packets, analyze traffic and spoof the data of UMTS frames. For more details the reader is referred to [Kambourakis2010].

A more general discussion of the weaknesses of UMTS and proposals to overcome them is given in [Xenakis2004]. For example the authors argue that if a TMSI cannot be allocated, the VLR is free to request the IMSI, which leads in a possible threat to user identity confidentiality. To overcome this problem, the authors suggest a second temporary identity of the user, which can be requested in case the TMSI allocation doesn’t work. Furthermore it is stated, that safety measures for the backbone network, like firewalls, aren’t applicable to assure for insider threats.

In [Dunkelman2010] Dunkelman, Keller and Shamir present an attack to reduce the $2^{128}$ complexity of the Kasumi algorithm. They are able to derive the whole session key by using 4 related keys, $2^{26}$ data, $2^{30}$ memory and $2^{32}$ time, by a technique they call sandwich attack. The authors also state, that their attack doesn’t work with the previous version of the Kasumi cipher Misty.

However, looking at practical attacks, there are no relevant conceptual flaws known at the time of writing, except the backward compatibility to GSM. In particular, the possibility to catch mobile 3G devices with a faked (2G) BTS as described in section 4.1.1.4.1 is relevant, as these devices are typically configured in a way that they are not restricted to 3G, but can use also 2G access.
4.1.1.7 Fuzzing Baseband Implementations

Today there are only a few different implementations of GSM protocol stacks, which are used in almost all mobile devices. In contrast to other popular communication technologies like wireless LAN, which received a lot of scrutiny, these implementations of the GSM protocol stack still lack a thorough and independent analysis. This is due to the fact that the required GSM hardware was very expensive in the past, and due to the lack of open source implementations of the GSM protocol stack. But with the recent availability of open source projects like OpenBSC (for a description of OpenBSC see 4.1.1.4.1) and OpenBTS\(^{21}\) it became possible to analyze the quality and reliability of these implementations.

In the context of the ASMONIA project some practical tests regarding the fuzzing of baseband implementations were made. Fuzzing means injecting various malformed protocol messages in order to test an implementation for its robustness with respect to such malformed messages. The fuzzing framework that was developed and used for this purpose is depicted in Figure 10.

\(^{21}\) http://openbts.sourceforge.net/
There are three main components, which serve different purposes. The first one is the combination of OpenBSC and the nanoBTS. Those are needed to run the GSM cell. They provide the infrastructure to establish a connection to the targeted mobile devices. The nanoBTS is physically connected to a computer running OpenBSC via an Ethernet connection.

The second big part also consists of two elements. On the one hand Sulley\(^{22}\), a framework which can be used to develop fuzzers, is used for generating and sending the data that should be tested with the phones. On the other hand there is the ipaccess-proxy which makes it possible to inject the data generated by Sulley into the GSM network in order to send it to the target. The ipaccess-proxy is located between the nanoBTS and OpenBSC.

The third component is the monitoring and logging facility. Again they consist of more than one part. First the Gammu library is used to monitor the status of the mobile devices during a fuzzing session. This information is combined with the output of Sulley, which makes it better possible to relate the crash of a phone to the certain message, which most likely caused the crash.

\(^{22}\)http://code.google.com/p/sulley/
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

Crash. Similar to that is the tcpdumpserver, which monitors and logs the traffic on the Abis interface. This offers a way to record all data exchanged between the GSM network and the mobile device and a way to draw conclusions about the status of the phone by interpreting its reactions.

The steps which are performed during a fuzzing session are described in the following: First it is necessary to establish a GSM cell and make the mobile station to be fuzzed attach to it. After that, the tcpdumpserver has to be started which then waits for commands. Then the session is started using a Sulley script for a specific test case. Sulley prepares the messages needed for the session. Then, recording on the Abis interface is started by sending a command to the tcpdumpserver. The Sulley script automatically opens a logical channel (usually a SDCCH) to the target by the use of the OpenBSC telnet interface. If the phone is supported by Gammu, the state of the phone can be determined. After the logical channel is successfully established and the mobile device is in the correct state the next step is to open a connection to the MS/Phone on the data link layer. This is achieved by injecting a prepared Establish Request message towards the nanoBTS which then takes care of opening and maintaining this DLL connection. Now the setup is ready for the actual messages that should be sent to the target.

The general approach of fuzzing is to inject more or less random data into software through a given interface, and watch for instabilities, anomalies and other unwanted or unexpected behavior. With the described setup it is possible to inject messages on the network layer of GSM and not on lower layers of the protocol stack, as they are managed by the BTS and are therefore not available for fuzzing. This means that only L3 and higher protocol layers can be tested. Most important is the fact that nearly all information is exchanged in IEs that are either of the form Length Value (LV) or Type Length Value (TLV). In both cases the length fields are the most promising parts when it comes to detecting bugs in the implementations of GSM software stacks and therefore these fields were the main target for the fuzzing tests.

After having tested 13 mobile devices, it became apparent that the most successful way to attack a phone using the GSM protocol stack is the Short Message Service (SMS). Six of the phones, which were tested, showed errors when processing fuzzed short messages. Of the five different types of short messages which were tested, two types triggered no errors at all. The rest of the messages triggered errors like the following (amongst others):

- SMS applications crashed when trying to read certain received messages.
- Whole phones crashed when trying to read certain messages. Sometimes these short messages could not be deleted anymore.
- Mobile devices crashed and rebooted. After the reboot they were unresponsive and unstable until all messages were deleted.
- Phones crashed just after having received certain messages.
- One phone even crashed on receiving a SMS and was broken afterwards.

Most crashes were triggered using concatenated SMSs, so it seems probable that a combination of incorrect length fields and different coding schemes lead to memory corruptions while reassembling messages parts. Another interesting result is that despite the detailed GSM specification some implementations behave differently regarding aspects like silent calls, sending of acknowledgements in the SMS delivery and Paging Requests.
4.1.2 Applications Part

The application part can be divided into four classes: Machine-To-Machine, simple mobile phones, smart phones and PCs with 3G/4G module. The first one refers to devices that operate without user interaction. Simple mobile phones or feature phones are still the most widely spread application in mobile communication networks, although smart phones as third application class are gaining more and more importance and market share (19.3 % in 2010 regarding to Gartner). Smart phones offer more expandability through third party apps and full internet capability. With increasing achievable internet bandwidth via 3G networks and dropping prices, PCs with 3G/4G module become more important as well.

4.1.2.1 Machine-to-Machine

The generic term Machine-to-Machine refers to communication between machines, systems, devices and, in some cases, even people, usually without user interaction at one or more communication endpoints. In most use cases endpoints are devices equipped with embedded GSM modules that are monitoring some kind of state that is being reported in regular intervals or on-demand to a centralized information system, which then processes the acquired data and provides a human interface. GSM is used especially in cases when it is not possible or economically reasonable to provide other types of networking infrastructure. Communication mostly takes place via SMS or GPRS. This sort of application is not yet as widespread as mobile phones, but its importance is expected to grow in the future. While the major part of applications is nowadays settled in the industrial or professional sector, it is also expected that applications in private homes are on the rise. Depending on the machine type and functionality very different values for likelihood of attacks, vulnerability of the device and impact of a successful attack are possible.

There is a wide variety of different use cases for this kind of communication involving devices diverging very much from one another. [3GPP_TR22868] refers to the following classes:

- **Security** This refers to automated alarm and access control systems available for cars and buildings that allow notification or control via the GSM network.

- **Tracking & Tracing** of objects or persons. One major application already in use is fleet management of transport and car rental companies in order to monitor vehicle movement with the help of GPS. Furthermore a similar approach has been chosen by certain insurance companies by monitoring their customer’s car usage. The collected data is then being used to calculate charges.

- **Payment** applications via the mobile phone, but also point of sales terminals for credit or debit card transactions and ATMs.

- **Health** Transmission of monitored vital signs for remote diagnosis. This is also known as eHealth.

- **Remote Maintenance & Control** of objects such as sensors, vending machines and vehicle diagnostics as well as lighting, pumps and valves in industrial applications.

- **Metering** Smart metering (e.g. power, gas, and water) for accounting purposes or building management in the energy industry.

Not every application is easy to classify, such as the EBuLa (Elektronischer Buchfahrplan und Langsamfahrstellen) system of the Deutsche Bahn, which provides a terminal in the trains with the necessary information for the locomotive driver on the current track.

Depending on the nature of data transmitted security topics have to be taken into account. Particularly in health applications sensitive data has to be asserted not to be eavesdropped,
while it is very crucial to avoid remote control applications from being tampered with. In addition devices used in machine-to-machine applications tend to have a long life-cycle, while not being easily physically accessible. These circumstances can develop into a risk when security measures that have been taken become outdated but cannot be replaced or altered remotely.

One application of the automotive industry would be the European eCall\textsuperscript{23} system. This is a car emergency call system that automatically sends location information in case of an accident to the international emergency number 112 in order to increase road safety. An EU directive aims to have every new car to be equipped with this by 2014. Apart from that there exist more sophisticated telematics applications (e.g. BMW ConnectedDrive\textsuperscript{24}, Volvo On Call\textsuperscript{25}). Typical features are assistance calls, SMS based traffic information services or vehicle tracking in case of theft. In terms of security there is a more accurate examination necessary of possibilities offered that are actively intervening with the cars functionality. Volvos On Call system allows remotely unlocking a car's doors or completely disabling driving capability once the car comes to a halt. In order to invoke this kind of functionality the customer has to authenticate against the car manufacturer's call center which then executes the commands on the car by SMS messaging. It has to be ensured that these functions cannot be executed by anyone else.

GM's telematics system OnStar\textsuperscript{26}, currently just available in the U.S., even hands remote control to the user enabling him to lock/unlock and ignite the car using a smart phone application, which is imaginable as a future attack vector.

\subsection{Feature Phones or Simple Mobile Phones}

The term \textit{feature phone} or \textit{simple mobile phone} in this risk analysis refers to a simple low-cost mobile phone which in contrast to smart phones offers a rather small set of fixed features concentrating on use cases like voice and SMS. In most cases their hardware architecture comprises one shared CPU acting as baseband processor and application processor, which is described in detail at the beginning of this section (see page 25). They usually neither offer the possibility to install third party apps nor the ability to connect to the internet except for highly restricted subsets like WAP. Despite the increasing spread and therefore also increasing importance of smart phones, feature phones still are most widely used and sold.

Devices of this class usually have further interfaces, which offer additional potential attack vectors:

- **Baseband** (see Section 4.1.1)
- **Audio**: In feature phones (and most smart phones) the routing of the audio interfaces like earpiece, microphone and speaker can be controlled by the baseband processor. This can be abused for example with a firmware which was unrecognized modified by an attacker. The phone could then for example be advised remotely via SMS to turn on the microphone to eavesdrop a conversation. Flaws in the GSM protocol or its implementation on the baseband processor are starting points in order to realize such an attack.

\textsuperscript{23} http://ec.europa.eu/ecall
\textsuperscript{24} http://www.bmw.com/connecteddrive
\textsuperscript{25} http://www.volvocars.com/uk/campaigns/misc/oncall/Pages/Overview.aspx
\textsuperscript{26} http://www.onstarmobiledemo.com/
USB: Many phones today have a USB port (or serial port in the past) over which they can be connected to a PC and be used as a terminal adapter to for example establish a data connection or remote control the phone in another way. This is done mostly via the AT commands set. Another common functionality via the USB port is the synchronization of calendar, contacts and media files. This can be exploited to realize attacks in both directions. As depicted in [Wang2010] the capability of connecting a mobile phone via USB turns into vulnerability due to the reason that there is no authentication involved. Security is just being assumed, since physical access to the device is necessary.

Two classes of attacks from a compromised mobile phone to a computer are being proposed. The first class is the mobile terminal acting as a HID (Human Interface Device) and therefore being recognized as a USB keyboard or mouse device. This can be used to send predefined input commands and thereby harming the connected computer as if the user had typed the malicious commands by himself. The second class makes use of the possibility to connect the phone as a mass-storage device providing malicious content, e.g. PDF or JPG files exploiting vulnerabilities in system software. Hereby a mobile phone poses a higher threat than a usual USB stick for its computational power. By looking at the USB Requesting Block ID (URB) in the USB packets the attacking terminal can identify the computer’s operating system, which allows more well-directed attacks resulting in a higher success rate.

There is a wide variety of possible attacks from computers to mobile terminals, as the USB interface often provides powerful possibilities. In general it is very easy for an adversary to gain information of all kind from the phone when having managed to gain access, for there is no authentication required. Furthermore a possible threat is imposed by a malicious firmware that may be flashed onto the phones memory.

JTAG: Some phones expose a JTAG port (Joint Test Action Group) which is standardized in IEEE 1149.1. The JTAG port provides low-level access to the hardware for example for debugging or boundary scan testing and is therefore usually not meant to be used by the end user. Consequentially in most phones the port is not accessible at all or only accessible by means of opening the phones body or modifying the hardware in other ways. Though if an attacker succeeds in doing so, he has a powerful tool at his disposal which can possibly be used to figure out vulnerabilities.

The security threat the JTAG interface is prone to depends on the implementation that has been chosen by the device developer. The more access is being intended for development purposes the bigger is the threat that results from leaving the interface intact in the final product. If unsecured it offers possibilities for a thorough analysis and intrusion into the system. Since there is often no documentation provided on possibly available JTAG interfaces on a circuit board [Breeuwsma2006] describes an approach to finding test pads by measuring pins on the board that might be leftover from JTAG interfaces only intended for use in the development process. Moreover there exist tools, e.g. JTAGEnum27 or JTAGFinder28, allowing a similar approach using low-budget hardware. After having reverse engineered the interface it can be possible to forensically extract the content held by flash memory connected to the probed component as it is. On one hand this might enable reconstruction of data that has been deleted from the phones file system as long as it is not yet overwritten. On

27 http://deadhacker.com/tools/
28 http://www.elinux.org/JTAG_Finder
the other hand an exact image of the memory state allows the analysis of the contained software in order to find vulnerabilities such as a buffer overflow or modify firmware (cp. [Park2010]).

As a consequence of the direct hardware access the interface also enables the possibility of side channel attacks. If a dedicated cryptographic component can be accessed via JTAG, registers connected to the scan chain will be readable. Although secret keys contained in the component are usually not accessible in this way, the authors of [Yang2004] and [Yang2006] show attacks on DES and AES hardware implementations. These make use of the extensive insight the JTAG boundary scan gives during the cryptographic process on known input opening up the possibility of reconstructing the encryption key via crypto analysis.

- **SD Card**: The SD card is a common way to enlarge the memory space of a phone e.g. for multimedia content like photos or music, but also for contact data and other stuff. The card is mounted as external storage device when the phone is connected to a PC. This fact can be exploited in a Phone-To-Host attack like explained in [Wang2010].

- **Bluetooth**: Many feature phones offer a Bluetooth interface for wireless communication with other devices, for example with a headset or a hands-free car set. The NIST therefore released a “Guide to Bluetooth Security” which gives a good technical overview ([NISTbluetooth2008]), while [Dunning2010] provides a broad survey of threats posed by the Bluetooth interface.

  First of all, since Bluetooth is a wireless interface, its radio waves can be monitored and therefore Bluetooth traffic can be sniffed. Aside from that aspect it can be used to track devices within signal range. This is especially the case, when the terminal is in discoverable mode. If a device can be associated with a certain owner this is a considerable threat against privacy. Another threat adherent to Bluetooth’s technical principle lies in the possibility of extending the wireless signal range. Through the use of a directional antenna attacks have been performed over a distance of more than one kilometer, which is more than 10 times the maximum specification envisions. Due to Bluetooth’s technique of frequency hopping, which is mainly a mechanism designed to oppose interference, costly professional Bluetooth analyzers are required to monitor all radio transmissions necessary.

  Since every Bluetooth capable device is equipped with a supposedly unique address, which is also depicting the device’s class, it is possible to spoof the identities of devices when their address is known beforehand. Bluetooth communication has a strict packet formatting standard. By deviating from that standard implementation flaws might be used to launch attacks that can have unforeseen consequences ranging from crashing a device to introducing malicious code due to a buffer overflow. In [Shaked2005] it is demonstrated that a short Bluetooth PIN can efficiently be cracked when having monitored the pairing process of two devices. This is realized by testing PINs via brute force, which can by verified cryptographically using the information from the pairing process. This can be done using the freely available BTCrack Tool\(^{29}\).

  There also exist concerns over the strength of the stream cipher encryption E0 that is being used for Bluetooth connections. A theoretical known-plaintext attack described

in [Lu2005] using conditional correlation allows recovering the encryption key in $2^{38}$ computations instead of $2^{128}$ necessary for a brute force attack.

Since late 2003 there have been collected several significant attacks by the trifinite group\(^{30}\) around Bluetooth security. There is also a certain amount of demonstration-software available.

- **BlueBug** A severe security flaw, empowering an attacker to issue AT commands in some older mobile terminals. Therefore the exploitation of this issue may be executed without notice of the user, while enabling a lot of possibilities to the attacker like gaining access to data on the phone or placing phone calls and many more activities depending on the features offered by the phone.

- **BlueSnarf** denotes the popular exploit of a falsely implemented OBEX Push Profile in some older devices allowing retrieving phone book or calendar files, when their location on the file system is known or guessed correctly. A similar attack on phones with an OBEX FTP server even allows read/write access on the file system.

- **BlueSmack** is a DoS attack performed by sending an echo request with a packet size too big for certain phones to handle making them crash.

- **BlueBump** When deleting the link-key of an authenticated device which is still connected, this device is able to request generation of a new key and thereby maintaining it in the list of authenticated devices.

- **BlueChop** This attack allows disrupting an established piconet (i.e. a small wireless personal area network) from outside by spoofing an arbitrary slave and contacting the master device of the net. This leads to confusion in the internal state of the master disabling the connection.

- **BlueDump** By spoofing one of a paired set of Bluetooth devices, while connecting to the other one, some devices discard their link key and change into pairing mode. This creates a possibility of key-exchange sniffing.

- **CarWhisperer** is a tool allowing injecting or eavesdropping audio signals from a headset or hands-free unit. It abuses the common implementation flaw of using a fixed PIN for the pairing process.

- **BlueStab** is the name of a DoS attack making use of some phones’ incapability of handling control characters in device names, resulting in crashing the mobile phone.

In addition the term bluejacking describes the act of arbitrarily sending content to mobile phones in discoverable mode. This is not exploitation of vulnerability per se, but flooding devices has a high chance of resulting in success, if a subject accepts the request without concern or by mistake and afterwards executes the malicious program.

- **WLAN**: Some feature phones (and especially smart phones) have a wireless LAN interface, which can be used to connect to a WLAN network, for example to gain access to the internet. This brings up some security concerns as denoted in [Welch2003]. Aside from a denial of service attack by jamming the WLAN signal the simplest form of an attack would be eavesdropping an unencrypted or WEP

\(^{30}\) http://trifinite.org/trifinite_stuff.html
encrypted connection made by a mobile device with an access point (AP), simply by monitoring the radio signal. This kind of connection could be made due to a badly configured AP. In [Fluhrer2001] the authors show how to efficiently exploit a weakness in the operation mode of the RC4 stream cipher used in the WEP encryption standard making it possible to derive the encryption key. Therefore a more sophisticated encryption is necessary. Apart from passively listening to such traffic an adversary may also inject modified traffic. Since the connection is a matter of radio signals their reach can once again be extended by the use of a directional antenna.

A more complex attack is setting up a so called rogue AP, which is an access point impersonating a nearby known wireless network infrastructure. By default many devices connect to the AP with the highest signal strength, without the AP having to authenticate against the device. This makes it easy to trick someone into associating with the wrong one. For once the rogue AP might have access to the desired network and therefore launch a man-in-the-middle attack against clients which have falsely connected to it, for it might provide greater signal strength. If the rogue AP cannot get access to the network, it cannot become man-in-the-middle, but it still can attack the victim device via the connection between rogue AP and device.

[Welch2003] also describes two attacks that can abuse a WLAN connected device opening up the possibility to gain access to a protected wireless network. One is session hijacking. Hereby the attacker masquerades himself as the authenticated target device, while forcing it to discontinue the session in order to overtake it. The other one is recording an authentication process and replaying the device’s messages to create a new session.

- **NFC**: Some newer feature phones contain a Near Field Communication (NFC) interface for exchanging data with other devices over small distances actively or passively like a smart card. It is an RFID technology and primary use cases are for example mobile payment, out-of-band device pairing for other connections such as Bluetooth and information gathering from smart posters. Several NFC related attacks are listed below.

In [Haselsteiner2006] the authors highlight several threats that are inherent to NFC technology. As communication takes place over radio frequency waves these can ordinarily be eavesdropped with an antenna from within a distance of 1 to 10 meters depending on the communication mode of the phone and environmental circumstances. Another threat would be jamming the radio signal in order to corrupt the data being transferred, which would result in a typical DoS attack. Physical data modification on transferred bits is theoretically possible according to the signal transmission encoding that is being used. Moreover the possibility of data insertion is proposed. This may be done with the condition that the inserted answer to an NFC message has already finished being transmitted, since an overlap with the real answer would lead to corrupt data.

Further threats are allocated in [Madlmayr2008]. Firstly a DoS attack can be launched since every touch of a mobile terminal with an NFC tag causes a reaction, so that even with an empty tag that just provokes an error message the phone may be occupied. In [Mulliner2009] it has also been achieved to crash a phone reading a tag that provided fuzzed information.

Secondly it is being pointed out that the standard is not secure against relay attacks that cannot be recognized by the devices involved. Therefore immediate proximity of these devices is not guaranteed. In [Francis2010] such an attack is demonstrated
using two ordinary phones connected via Bluetooth that serve as proxies relaying an NFC connection between another two phones.

The ability of an NFC capable device to act as a smart card implies that an index of the stored applications is readable, which may be a privacy issue. In addition due to the ISO14443 standard the device has a unique ID number, which is used for anti-collision purposes during the reading of transponders. Since this UID is being sent unencrypted and can therefore easily be accessed it can be used to spoof someone’s identity in applications using this UID for identification.

A major threat on the NFC interface is being addressed in [Mulliner2009]: Phishing attacks using modified or replaced NFC tags. By replacing the NFC tag e.g. that of an advertisement that is equipped with such a tag in order to distribute a URL conveniently, unaware subjects might be tricked into visiting a phishing website instead of the real website that is being promoted. This attack benefits from the usage of URI handling in a mobile phone, which may allow hiding the modified URL different from the one that might be advertised by showing the real one in the title field appended with blank lines while directing to the modified one in the URL field. It’s also a certain advantage for the attacker that in mobile browsers the URL is often not being displayed. The attack works in the same way when phone numbers are being distributed for charging services as it is in use with vending machines.

In [Mulliner2009] there has also been found a possibility to abuse the standardized NFC Java API by registering a generic URI type, which allows an application to be launched every time a URI NDEF (NFC Data Exchange Format), as being used in the above attack proposition, is read. A proof-of-concept worm has been written, that spreads by writing the URL of a copy of itself to a tag, which infects the next phone to read this tag. The worm is hiding itself by using the already mentioned phishing-technique and redirecting already infected phones to the original destination of the infected tag.

Today’s feature phones offer various interfaces and use a complex software stack. Both points ensure an increased vulnerability compared to older phones. On the other hand the software quality of older phones may be worse. However, a more attractive target for an attacker seems to be a smart phone where typically more sensitive information is stored and which uses an even more complex software stack.

4.1.2.3 Mobile Operating Systems for Smart Phones

In this chapter we will introduce the different operating systems of smart phones that are currently available. While analyzing their security characteristics will give insightful information for the threat analysis, we will keep the analysis on a general level. This will enable to use the results of the threat analysis to assess future and upcoming operating systems concepts for smart phones.

In the following we chose the currently most relevant smart phone operating systems based on their market share. These are Apple iOS, Symbian, Android, Windows Mobile 7, Linux and RIM systems.

The current market shares of the different platforms can be seen in the figure below.
### Figure 11: Current Market Share according to Gartner

<table>
<thead>
<tr>
<th>Operating System</th>
<th>3Q11 Units</th>
<th>3Q11 Market Share (%)</th>
<th>3Q10 Units</th>
<th>3Q10 Market Share (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Android</td>
<td>60,490.4</td>
<td>52.5</td>
<td>20,544.0</td>
<td>25.3</td>
</tr>
<tr>
<td>Symbian</td>
<td>19,500.1</td>
<td>16.9</td>
<td>29,480.1</td>
<td>36.3</td>
</tr>
<tr>
<td>iOS</td>
<td>17,295.3</td>
<td>15.0</td>
<td>13,484.4</td>
<td>16.6</td>
</tr>
<tr>
<td>Research In Motion</td>
<td>12,701.1</td>
<td>11.0</td>
<td>12,508.3</td>
<td>15.4</td>
</tr>
<tr>
<td>Bada</td>
<td>2,478.5</td>
<td>2.2</td>
<td>920.6</td>
<td>1.1</td>
</tr>
<tr>
<td>Microsoft</td>
<td>1,701.9</td>
<td>1.5</td>
<td>2,203.9</td>
<td>2.7</td>
</tr>
<tr>
<td>Others</td>
<td>1,018.1</td>
<td>0.9</td>
<td>1,991.3</td>
<td>2.5</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>115,185.4</strong></td>
<td><strong>100</strong></td>
<td><strong>81,132.6</strong></td>
<td><strong>100</strong></td>
</tr>
</tbody>
</table>

Source: Gartner (November 2011)

### 4.1.2.3.1 Security Mechanisms in Smart Phone Operating Systems Environment

This chapter will introduce the various security mechanisms employed by operating systems used in the current popular smart phones. In each of the chapters describing the smart phone operating systems, we will list the used security mechanisms.

#### 4.1.2.3.1.1 Capabilities

Capabilities are a concept used to restrict access to API functionality, sensitive data and hardware functions. During the development of an application the developer will need to decide which API function the application will use, which data to access, and which hardware components, such as GPS or camera to use. The specification of these used capabilities will typically result in some sort of document which is distributed alongside the application. Since this document is security relevant, the platform in question needs to make sure that the capability document is not tampered with. Also the platform needs to be equipped with a mechanism to enforce the specified capabilities.

There are different ways to secure the capabilities.

- Capabilities are implicitly granted by the user at the time of installation. This will effectively grant the applications the rights to access all the functionality specified for the time it remains on the device. (Android)

- Granting capability through external (trusted) entities. Instead of involving the user, an external entity verifies the different capabilities used by the application. This typically involves some form of review. After the review process is passed, the capabilities document will be secured by the external party. This typically involves the external party digitally signing the document. These external parties will most likely be operating a distribution hub such as an App market. (Symbian, Apple)
Self-signing capabilities is also possible, if an external trusted third party is not available, or no CA is directly operated. This can also be combined with implicit involvement of the user granting the capabilities. (Android)

Capabilities can also be directly granted by the device manufacturer.

The particular capabilities can vary in their granularity. For example the Android OS allows specifying read/write control to the SD card, but not a general restriction to certain directories on the SD card. Somewhat similar to the concepts of capabilities, application level access controls are decided during development of the application. Upon installation of the application, the necessary access controls to run the application may be presented to the users for confirmation. Deciding the level of granularity to be used is a trade-off between security and not overstraining the user with complex decisions.

4.1.2.3.1.2 Remote Device Activation

To control the device usage, some vendors require the activation of the device via a remote server. This is typically used when the vendor is also controlling the application distribution. It allows to link the device to the user. This can later be used to control the data of the user, and also allow the remote deactivation.

4.1.2.3.1.3 Remote Device Wiping & Deactivation

Remote wiping and deactivation techniques are useful in several cases.

- Stolen devices
- Infected devices
- Malicious device
- Illegal software on devices

The motivation for the vendor of the devices is obvious. He wants to control which data can reside on the device and which services can be used. Network operators are more interested in deactivating malicious devices that may attack the network infrastructure. Stolen or infected devices are more of concern to the owner/user. Remotely wiping the device can be useful if direct disinfection does not work. Also, in case of the device being stolen, the user can wipe the device without fearing his data to be disclosed.

4.1.2.3.1.4 Code Signing

Code signing is used to be able to confirm who has released a program or an app. The user still has to make his own decision whether to install the code or not. Additionally, code signing enables to check whether the code has been altered or corrupted since applying the signature. Signing is typically done by using digital signatures which are based on cryptographic hash mechanisms. The security of the signature scheme is the basis for the security of the code signing mechanisms. Every security issue related to digital signature is therefore also relevant when applying them to code.

Whether the trust imposed on the code is backed up by a trustworthy certification authority depends on the use case. There can also be solutions in which the code is signed based on self-signed certificates. These solutions may be able to protect against corruption and alteration of the code, but they do not necessarily provide means to trust the author of the code in question. This would have to be decided by the user that installs or runs the code on his device.
4.1.2.3.1.5 Trusted Boot (Chain of Trust)

Trusted or secure boot schemes allow developing trust in an incremental way across the components involved in the boot process of a device. Typically there are cryptographic checks to be performed after each stage of the boot process. If one of the checks fails, the boot process will be aborted.

An example for a “Chain of Trust” is the secure boot mechanism employed by the ARM TrustZone Software Architecture [ATZ]. The chain of trust has to start with an implicitly trusted component. This could for example be an OEM PUK entered at the first stage of the boot loader or the secure world software image. The root of such a chain of trust has obviously to be placed in a very secure location.

![Figure 12: A typical Boot Sequence of a TrustZone-Enabled Processor](image)

When the device is powered on, the secure world checks are done all prior to the normal world having an opportunity manipulating any security critical memory regions. Most of the SoC (System on a Chip) designs start to load a ROM based boot loader which will check relevant hardware and memory controllers. After the secure world has successfully passed all checks, a security baseline has been established. This allows to transition to the normal world which can execute the normal boot loader and commence to the operating system. The figure above shows a simplification of the boot sequence.

Trusted Boot mechanisms such as the above (which is using the ARM TrustZone) resemble a feature from the platform security world. Platform security is one of the key points of ASMONIA AP2 Deliverable 2.1. Details on the Trusted Boot feature can therefore be found in Deliverable D2.1.
4.1.2.3.1.6 Sandboxing

In the context of operating systems, sandboxing means to separate different components by putting them into logically separated areas in the system. The sandbox typically provides strict rules which resources can be accessed. The most famous example for a sandbox is a virtual machine. Virtual machines allow emulating a computer host which can be equipped with many different operating systems. Sandboxes are typically designed to emulate a specific computing architecture such as ARM or x86. The sandbox itself will run on a host system which may have a completely different computing architecture. Sandboxes are typically considered as a jailed environment from which the sandboxed component should not be able to break out. The notion of a controlled environment is also of importance, since it allows imposing strict limitations on the communication from and to the sandboxed component.

In context of smart phone operating systems, sandboxing can be done with different granularity. The whole operating system can run in a sandbox, as well as only different applications, or even operating system processes which house the application at runtime.

4.1.2.3.1.7 Data Caging

Data caging is a form of access control in order to separate data from code. In context of smart phone operating systems, it refers to applications and users only having access to certain predefined areas of the file system. If different applications run isolated from each other, it is typically the case that they own private folders to which only the owner application has access to. There may also be shared folders by intent. The Symbian OS e.g., has the following separation:

- */sys/ is restricted to be accessible only by TCB (Trusted Computing Base) processes
- */private/<SSID>/ is restricted to non-TCB processes with SSID
- */resource/ is read-only for non-TCB processes
- */all the rest/ does not impose additional restrictions

4.1.2.3.1.8 Read-only System Image

Restricting access to the operating system image is important to prevent multiple attacks e.g., replacing the original image with a custom tailored or malicious image. In order to achieve this, the responsible mechanism typically uses trusted boot or TPM components to secure the image at early stages.

4.1.2.3.1.9 Privilege Separation

Privilege separation aims at minimizing the risk of processes misusing components to inflict damage to a user, user data or the system itself. Privileges are typically separated into regular users and root user. In contrast to processes running with regular user privileges, root processes will not be restricted by security mechanisms such as access control or data caging.

Maliciously escalating privileges is typically considered an important step to gain control over a system. Vendors restrict the operating system and the users, namely the applications, to run in user mode for security purposes. The process of “rooting” a phone is regularly done and fairly automated. It is primarily aimed at users that want to “unlock” the devices’ features that are not accessible for the regular users, for example, low level operating system functions that enable backup of user data.
4.1.2.3.2 Symbian

The Symbian Platform has been around as an operating system for smart phones for some time. It was originally intended for smart phones and PDAs. The platform is currently at version “Symbian^3” which has been released in 3rd quarter of 2010. As opposed to previous Symbian releases, the current version was aimed to be fully Open Source. Vendors such as Nokia, Sony Ericsson, Motorola, and NTT DoCoMo have joined in the Symbian Foundation to advance the development of the platform. After Sony Ericsson and Samsung left the Symbian Foundation in 2010, Nokia will take over the development platform in 2011. The improvement of Symbian as a platform is currently supported by the EU project SYMBEOSE. The next release Symbian^4 in 2011 will mostly focus on introducing a new user interface based on Qt, which is a cross platform application and UI framework.

The security concepts of Symbian are focused on software security. It is to make sure that the software does what its (flawless) specification declares. The specifications should also be fulfilled by the software under attacks. The most prominent security mechanism, Symbian Signed, is discussed in detail in the ASMONIA Deliverable D2.1-

4.1.2.3.2.1 Security Features and Concepts

- **Capabilities** (basic, extended, full) which force the applications to declare which resources it will use.
- **Data Caging** based on file system access rights and process user ids
- Control on removable disks
- **Code signing** by the Symbian Signed process. Controlled applications distribution.
- **Trusted computing base** to enforce capabilities and data caging

4.1.2.3.3 Android

The development of the Android OS has primarily been driven by Google. Its goal is to provide an Open Source smart phone operating system allowing many different hardware vendors to use it on their devices. Its core component is the Linux Kernel based on version 2.6. It has first been released in 1Q2009 and its current version is 2.3 which is based on the 2.6.35.7 Linux kernel. Just as the iOS for the Apple iPhone, Android offers the possibility to install 3rd party applications via the “Android Market” which is similar to the “Apple App Store”. Applications do not run natively, but rather use the Dalvik virtual machine [Bornstein2008] as a runtime sandboxing layer. The core OS is protected from the applications, as well as applications from each other.

Applications can either directly be installed from the market, or manually by downloading them onto the device. Untrusted applications i.e., ones without a signature (by using the help of self-signed cert) cannot be installed on the device by default. However, this can easily be changed.

The figure below gives a general overview of the Android architecture. The lower level is represented by the Linux kernel. On top of that we have the necessary system libraries to interact with the kernel and the Android runtime, which consists of a set of core libraries (to interface with the native system) and the Dalvik virtual machine. On top of that we have the actual programming API accessible to application programmers.
Figure 13: Android Architecture [AndroidDev]

4.1.2.3.3.1 Security Features and Concepts

For a detailed description on how the features below are instantiated by the Android systems, the reader is referred to [AndroidDev].

- Security embedded into application model (Activities, Service, Broadcast receivers, content providers, intents, inter-process messages)
- Sandbox via the Dalvik VM running the applications
- Code signing using self-signed certificates in the absence of a responsible CA
- Capabilities as specified by the application manifest
- Inter-process communication only by OS mechanisms
- Data caging (user ID per proc)
- Read-only system image
- Non-overwriteable “.apk” files (Application install files)
- Remote wiping and installation of software initiated by Google
- Privilege separation (user vs. root)
- Underlying Linux security features such as file system access controls
4.1.2.3.3.2 Custom ROMs

Not only device manufacturers appreciate the versatility of Android, but also individual mobile software enthusiasts who build custom firmware versions based on Android. One benefit of these custom ROMs is a usually much higher update and patch frequency. Moreover, these ROMs typically have unnecessary software removed that has been installed by the manufacturer or carrier, which results in an improved performance. The most important advantage of custom ROMs, however, is that in many cases there is still an up-to-date version of the operating system available for phones with discontinued support by the manufacturer. Those phones are not going to receive any official updates anymore and will be forever vulnerable to any future discovered security gap and cannot profit from improvements in newer OS and SDK versions that do not depend on new hardware. The latest official Android version for the T-Mobile G1, for example, is 1.6, which was released in September 2009 and uses Linux Kernel 2.6.29.

One major downside of custom ROMs is that there is the risk of leaving the phone in a bricked state. In most cases, however, the device can be restored to factory settings. Nevertheless, there are some manufacturers trying to prevent users from installing another than the official firmware to their devices. Reports say that HTC recently started locking down the boot loader, which needs to be modified in order to install any custom firmware. Motorola followed this idea and protected the boot loader of its Droid X phone using a technique called eFuse, which will take the phone into recovery mode, if a modified firmware was detected. Only after installing an official firmware, the phone can be used again.

4.1.2.3.3 Android Application Structure

Android applications consist of several building blocks. There are four main blocks; activities, services, content providers and broadcast receivers. Other important components are the AndroidManifest.xml file and intents which enable communication between the components and processes (“inter component communication”, ICC).

4.1.2.3.3.1 Activities

Activities are all parts of the application that the user can interact with. They feature a user interface and handle user input. Activities are single screens and can be regarded as sub-programs. Applications typically consist of several activities. In many applications, these activities can be called directly. Moving to another screen also means switching to a new activity. Activities can return results to the previous activity.

4.1.2.3.3.2 Services

Services are processes without a user interface that run in the background. They often perform operations like playing music or downloading application data such as database updates.

4.1.2.3.3.3 Content providers

It is not possible for two applications to share the same address space due to the sandboxing enforced by the system. Content providers keep a data connection to a persistent or online storage and provide the data not only to other application components, but also to other applications if desired.
4.1.2.3.3.4 Intents

Intents are used for inter-process communication. Explicit intents address a particular receiver and are typically used inside the same application to communicate between the different components. The components are addressed explicitly in the code via their class name. An explicit intent can be used to start a new activity or to bring another to the front.

Implicit intents are sent without knowing if there is a receiver going to take care of the intent, so they are not addressed to a specific component. They are fired and typically used to notify another unknown application that listens to this intent type of a particular event.

Intents consist of an action (VIEW, EDIT, DELETE, ...) and some data to use. In the example of Figure 14 an Intent is created that tells the system to find an application that wants to handle the geo URI scheme. URIs, Uniform Resource Identifiers, identify names or resources in the system or the Internet. In the default case, this will be the Google Maps application that will show the given coordinate on a map.

```java
Uri uri = Uri.parse("geo:50.7789435,6.0773018");
startActivity(new Intent(Intent.ACTION_VIEW, uri));
```

*Figure 14: Starting the Google Maps app with predefined coordinates*

4.1.2.3.3.5 Broadcast Receivers

The third type of intents are broadcast intents. Initiated by the system, implicit broadcast intents are sent to notify applications of certain events like an incoming phone call or SMS, a time zone change or an almost-drained battery. Broadcast receivers are application components that listen to these intents and handle them. Common actions reach from the launch of a new activity to changing an entry in the persistent storage. Broadcast receivers do not come with a UI.

4.1.2.3.3.6 AndroidManifest.xml

Information about inter process communication is included in the `AndroidManifest.xml` file. In this file, every component of the application must be defined, intent filters describe which intents this application wants to handle and every permission the application needs can be found. As soon as any kind of intent is fired, the system searches all installed applications for a matching intent filter.

4.1.2.3.3.4 Dalvik Virtual Machine

The Dalvik Virtual Machine is the heart of the Android runtime. In the layer model, it resides on the second level next to the core libraries. It was developed by Google employee Dan Bornstein since 2005. It bases on the Java Virtual Machine Apache Harmony (which is the open source variant of the Java VM), yet there are major differences.

The main difference between the two virtual machines is the architecture, which results in different byte code for the two machines. Dalvik follows the register machine model to generate its `dexcode`, whereas the JVM uses a stack-based architecture, akin to a microprocessor.

For both virtual machine types, the programming language is Java and both use the `javac` compiler to build `.class` files from the source code. When compiling for Dalvik, the resulting class files are packaged into one single `classes.dex` file which eventually contains the Dalvik
byte code. Several optimization steps are performed in order to deal with the constrained resources on the mobile device. Battery power was one key concern when developing the Dalvik Virtual Machine.

In contrast to J2ME (Java Platform 2, Micro Edition), which was an attempt by Sun to make Java run on mobile handsets, Dalvik does not strip many libraries from the Java language to make it more lightweight, but it makes use of the device’s particular hardware. Most smartphones use a register-based ARM RISC microprocessor. Dalvik is able to access the processor’s registers directly, which the Java VM is not. This significantly improves the overall performance, despite Android does not use the hardware-acceleration for Java virtual machines that is built into the ARMv6 chips that are working in Android devices.

Every Android application runs in its own Dalvik instance. This has several advantages, e.g., there is no need for a shared address space of two applications. Every application can run in its very own address space and will not be interfered by another. This prevents memory corruption and unauthorized access to other applications’ data.

Moreover, it is very important that one misbehaving application does not affect others. If an application crashes, only its own virtual machine dies; so it cannot take down any other running application.

The Dalvik virtual machine must address the problem of constrained resources and must run on a slow CPU with little memory and no swapping space. Low-end Android devices feature only 64 megabytes of RAM, of which approximately 40 are left after the low-level system startup and 20 as soon as the whole system is up and running.

All Java .class files are being compiled into one single Dalvik executable file. At the beginning of the file, right after the header, there are constants pools. These pools store constant identifiers for, e.g., every single string in the code such as explicit strings or field and type names. Because several classes are compiled into one file, there must be also a class definition section. At the end of the file, there is the actual byte code. Between these sections, there are many references in order to only have to store everything once and use it later multiple times. In particular, strings and method signatures that are the same when a method is overridden in subclasses only need to be saved once, in contrast to the standard .class files where each maintains its own constant pool. This primarily keeps the file size small. According to Dan Bornstein, head developer of the Dalvik VM at Google, the size of a Dalvik executables is approximately half of the size of standard .jar file.

The main concept to keep memory usage low is the implementation of the zygote process. It is designed to minimize the problem that Android applications need a lot of memory for exclusive use (private memory). It is one of the first processes started on system boot and it loads many classes that are commonly used by many applications. zygote waits for applications that the user wants to start and on demand forks a new process that becomes the desired application. Applications can then use the shared (read-only) classes pre-initialized by the zygote and do not have to waste limited own memory to load the classes themselves.

4.1.2.3.3.5 Software Installation

The installation of applications to Android devices is done in software: the PackageManger does not use any user interface and performs every single installation and uninstallation action, keeps track of what applications are installed and their state. Moreover, it does not take care of any permissions, but instead assumes that they have been approved by the user. This makes it necessary that the PackageManager checks if the application that calls pm with an installation request was granted the system permission INSTALL_PACKAGES.
This permission, however, has protection level 2, i.e., the application requesting the permission must be signed with the same certificate as the application that defines the permission - which is in this case the Android system itself, signed by the firmware developer. Thus, only vendor-approved applications can install other applications. On a standard Android system, the only application holding the INSTALL_PACKAGES permission is the PackageInstaller which belongs to the core system. It asks the user for approval of the permissions during the installation request.

4.1.2.3.3.5.1 Android Market

The Market divides the available software into applications and games. The user can browse the categories and is displayed the name of the author of the application and its price. The Market application presents a description and screenshots. If the user decides to install the application, he taps the price tag. The button then turns into an OK button and the text label above then says Accept permissions, while the permissions requested by the application are unveiled.

**Quality Management:** Compared to the other two operating systems with a similar Market application, iOS and Windows Phone 7, the Android Market employs a very liberal policy. In order to be able to publish applications to the Android Market requires a one-time registration fee $25 fee. The amount must be paid by credit card using Google Checkout. By using prepaid credit cards, the registration can be completely anonymous.

The Android Market's security model relies entirely on the community to identify and report malicious applications. This implies that there is a certain number of users required to "sacrifice" themselves and try a new application. There is no further validation or review of the application, which makes it very easy to offer malware to a wide range of customers. Users have the chance to flag an application as inappropriate with one of the explanatory statements sexual content, graphic violence, hateful or abusive content or other objection. However, malware applications typically hide their malicious routines behind useful functionality; many users will not notice the secret features and never flag the application as inappropriate. One of the first malware applications found was phishing for bank accounts developed by Droid09. In February 2011, Google had to remove a group of more than 50 malware-infected applications.

The Android Market is an easy and convenient way for attackers to reach a multitude of devices with potentially valuable content at once. An approval process similar to the one Apple and Microsoft enforce for applications on their App Store or Marketplace, respectively, could have easily prevented the admission to the official Android Market. Static code analysis would have uncovered the hidden functions.

4.1.2.3.3.5.2 Alternative Software Shops

**Web-based Market:** In February, 2011, Google introduced a new additional web-based interface for its Market application to be usable from a desktop computer as well. Users can now access the market via a browser from a desktop computer and buy and install applications remotely.

Google installs software that has been bought via the mobile Market application using the INSTALL_ASSET intent to Android handsets, which means that the installation is triggered remotely by Google, not locally by the Market application. This is also true for applications purchased from the web. As soon as the user accepts the permissions requested, his handset starts downloading and installing immediately. No further interaction with and thus not even physical access to the device is required.
3rd-Party Markets: Besides the Android Market, there are several other application shops for Android that follow a similar approach like the Android Market by offering applications via a client application. The platform makes this possible by allowing third-party applications to install other packages with the user’s consent. The alternative shops must equalize their disadvantage that the number of applications they offer is significantly smaller than the official Market's. For example, many accept other payment methods like PayPal or Amazon Payments.

The most important alternative shops are GetJar, AndroidPIT and the Amazon Appstore. After all, the alternative software shops act inside the granted permissions by the user and they are prevented by the system to install applications on their own without asking the user. Installations are eventually handled by the Android system service PackageManager. Hence, they are not relevant from a security perspective, because they do not open a new attack vector, but use the legitimate operating system means.

4.1.2.3.5.3 Reverse-engineering Applications

Reverse engineering Android applications is very easy. All application packages are stored on the device in the /data/app directory; the important system applications are located in /system/app. Via the Android Debug Bridge, they can be copied to a desktop computer that the device is tethered to.

The .apk file is nothing more than a simple ZIP archive. It contains a binary version of the manifest file, the Dalvik byte code in one classes.dex file and all resources used by the application, such as images, databases and media files.

For performance purposes, the application package contains a binary form of the manifest file. An Android XML decompiler is available (http://code.google.com/p/android4me/). After decoding the binary file, it becomes the perfectly human-readable original version again.

In the root directory of the application archive, there is the classes.dex file that contains all compiled classes after some optimization done by the dx tool. The tool dex2jar (http://code.google.com/p/dex2jar) helps to undo the optimizations and convert the file to a Java archive. This archive can eventually be read and decompiled by a standard Java decompiler such as JD (http://java.decompiler.free.fr/).

It is possible to run a particular activity inside an application directly. This can be useful if the application uses some kind of lock mechanism, e.g. requiring the user to enter a password. This way, an activity that is normally only intended to be launched after successfully entering the password can be launched bypassing the password activity. Using the dumpsyspackage command in the ADB shell, all activities known to the system are displayed. The ActivityManager can be instructed to launch a new activity from the command line. This technique is useful to find hidden screens and settings in applications.

4.1.2.3.6 Protection Mechanisms

Android implements the three main security features code signing, sandboxing and a permission model. It does not employ a certificate authority and hence relinquishes proper non-repudiation and integrity protection in favor of openness. It uses fine-grained permission models for applications. 100+ different permissions can be requested by applications and more can be added by developers.
4.1.2.3.3.6.1 Safe Mode

An important security feature is the so-called safe mode the device can boot into. Most devices enter this mode by holding a particular key pressed while booting. In safe mode, no third-party applications are allowed to run. Only those packaged with the system image can be launched. This enables the user in most cases to remove malware and to clean an infected system without being interfered by protection mechanisms of the malicious software. However, malware applications that root the phone and copy their application package to the system partition will reinstall the malware at the next boot.

The safe mode can be left by rebooting.

4.1.2.3.3.6.2 Code Signing

Code signing in Android is in use and thus the system only allows the execution of code from signed application packages. Android does not employ a central certification authority that must sign all developer public keys, but allows self-signing instead.

Developers are able to generate many different key pairs and the corresponding certificates to sign their applications. So, the most important use of Android code signing is for the developer to be able update existing applications. All updates must be signed with the same key the original version is signed with. Developers can prove to be the same for two different applications, which enables him to share data between his applications in a secure way.

The lack of a CA makes it very easy to develop and distribute applications anonymously. Installations are not only possible via the Market, but also by just copying the application package to the SD card or by having the user download the application directly via a browser. To protect the user, installation from non-market sources is disabled by default, but this setting can be easily overridden in the system settings. Even if the application is supposed to be distributed via the Market, the barrier is not too high. By using prepaid credit cards, Market registration can be achieved anonymously.

Android does not signature check every code that is executed, but only the one bundled with the application package. Application can download a binary from the Internet and execute it within its standard permissions.

4.1.2.3.3.6.3 Sandboxing

Application-level security is in large parts achieved by a (pseudo-)sandboxing mechanism. Android does not enforce a real sandbox by virtualization, but instead uses UNIX user IDs to enforce restrictions.

Every application gets a unique Linux user assigned at installation time. The user names use app_ as prefix, followed by a consecutive number. The user ID does not change as long as the application is installed on the device. When an application is installed, the package contents are extracted into a directory below /data/data named the same as the application's bundle identifier. Only the newly created user that is associated with the application is allowed to read and modify data inside this application directory. Android sets the file permissions accordingly. The same technique is used for processes. The application process is launched with the associated user name and the according rights. Hereby is the application isolated from the others on the system and locked in a de-facto sandbox.

The only way for two applications to read each other's data is to be assigned the same user ID. This can be requested by an entry in the manifest file. The Android system, however, is only going to acknowledge the request if the requesting package is signed with the same certificate as the application whose user ID is requested is. The benefit of shared user IDs is
that two applications with the same user ID can read and write each other's data directly in terms of file system permissions.

4.1.2.3.6.4 Lock Screen Patterns and Pass Codes

Android provides several security mechanisms that can be influenced by the user. Besides the PIN lock of the SIM card that every mobile phone supports, the user can define a screen lock. If such a lock is used, the respective unlocking code must be used to access the device. The lock can become active every time the screen is turned off or after a predefined time of inactivity. This code can either be a four to 16 character long numerical PIN or alphanumerical password or a graphical pattern. The input screen for a graphical pattern consists of nine dots in a 3 by 3 matrix, of which at least four must be connected by drawing with the finger. The order in which the dots are connected is the code to unlock the device.

4.1.2.3.6.5 Permission Model

In contrast to iOS, Android employs a very fine-grained application permission model. As of API version 11, 116 different permissions are predefined. Developers are free to add custom permissions to protect their applications. An application that uses an SQLite database to store addresses may want to allow others to access the datasets, but only if the user agrees. An application has to explicitly request every single permission it is going to use in its manifest file. This model is supposed to make it impossible to hide undesired activities for applications.

If an application component attempts to perform an action that is protected by the need for a particular permission, Android raises a `SecurityException` and the action will fail. Developers can at any time check if the calling process has more fine-grained permissions that have been defined by calling the context's `checkCallingPermission()` method.

**URI Permission:** Content providers often protect themselves with permissions, but may want to pass a URI to another application to work with the data. For example, the mail application protects the emails from being read without permission; however, a third-party image viewer wants to display an image attachment. Of course, a normal image viewer does not hold the permission to read emails. In this case, the image viewer is handed a URI to the data with the `Intent.FLAG_GRANT_READ_URI_PERMISSION` flag set by the caller. This enables the receiver to read the data at the given URI.

**Protection Levels:** Permissions are categorized in four protection levels, 0 to 3: Category-zero permissions are normal permissions with a low-risk factor. Dangerous permissions belong in category 1. On this level, there are higher-risk permissions that for instance allow costly services like initiating a phone call or access to the device's sensors, the Internet and sensitive user data. Protection level 2 holds permissions which are granted only if the installation candidate is signed with the same certificate as the application that defined the permission is. These signature permissions are useful for developers to make sure that third-party applications cannot be granted this permission, even if the user would actually consent. Permissions on the highest level 3 can only be granted by the system to applications that are contained in the system image or are at least signed with the same certificate as the system image.

**Granularity:** The most important weakness of Android's permission model is that the user is only able to grant or deny all permissions at once. There is no chance to grant or deny particular permissions only. This forces the user to refrain from installing an application that might be useful, but requests too many permissions. Moreover, permissions that have been granted can only be revoked by uninstalling the affected application. This is a strong plea to
the user's discipline. He might want to test an application and not really care about the permissions dialog.

4.1.2.3.6.6 Google Apps Device Policy

Device policies allow the enforcement of security policies on devices. This includes setting a device PIN or password or requiring a minimal password strength, locating the device on a map, remotely wiping it and enforcing a timeout after which the device locks automatically. Once the optional app has been downloaded from the Android Market, it cannot be uninstalled and the policy cannot be nullified without the code that was defined on enforcing it, unless the phone is reset to factory defaults. It allows administrators to enforce the local security policy on devices. Administrators will have the ability to require users PINs on the device, lock screen and idle timeout, and wipe lost or stolen devices. Resetting PINs, ringing and locking the device, as well as locating the device are optionally possible.

4.1.2.3.7 Jailbreaking Android

There have been several exploits that can be used to root current Android devices. With some minor modifications, these exploits can be used in applications that are distributed via the Market. A simple way to acquire root privileges would be to set the setuid flag of the standard Android shell, such that the shell is always executed with root permissions. Applications with root access can read any data and post it to any service they want. Be it SMS messages, emails, address book entries and alike.

If the rooting functionality is hidden behind an appealing game or other useful application, the user may not even notice that his device is rooted. Android applications are easily reverse engineered. This makes the effort even less, because an existing application can be taken from the Market and the rooting functionality can be added.

The motivation for jailbreaking devices cannot be the execution of unsigned applications, because applications can easily be signed legitimately by oneself. The top reason to root a device is probably access to features and options that are not intended to be used or changed by end users. Many OEM carriers add a lot of software to the original Android version in order to customize it and enforce a branding. One can get rid of this additional, unnecessary software that often consumes a lot of battery and computation power. An example for available, but disabled features is multi-touch gestures. The Android system supports them, however, some manufacturers decided to not use it for devices such as the HTC Spica (also known as HTC Galaxy Lite). On a rooted device, multi-touch input can easily be enabled again.

Zygote Exploitation: The zygote jailbreak zimperich follows a similar structure as the exploit which targets the udev device manager, but exploits a different vulnerability. On Android, there is a maximum number of user processes that are allowed. In case of the HTC Desire Z running Android 2.2, this limit is 2983, which can be found by opening a shell on the device via the USB debug mode and issuing the ulimit -a command.

The exploit forks processes until the maximum number of allowed processes is reached. As the exploit is triggered by a Dalvik application, the zygote is responsible for setting the correct Linux user ID for the newly forked process. If, however, the maximum limit of user processes is reached, the setuid() system call performed by the zygote fails -- its intent was to limit the privileges of the new process by setting the user id to the (unprivileged) one associated to the application. However, the zygote process does not check the return value of the setuid() call. That is, if the call fails (which it will, because the user ID is not allowed to
spawn any more new processes), the newly forked process will remain with root privileges that originate from its parent process zygote.

Once the privileges have been escalated, this exploit creates a copy of the shell in /system/bin/rootshell with the setuid flag set, just like the udev exploit does.

The last Android version that is affected by this vulnerability is 2.2.

4.1.2.3.4 Apple iOS

Since the introduction of the iPhone in 2007 the smart phone industry went from business-only customers to mass market. The original iPhone was initially not allowed to be customized with 3rd party applications. But in mid 2008 popularity and general acceptance of the platform was increased by the introduction of an App Store. It enables easy distribution, installation, and management of applications. The general App Store concept allows controlling which applications can be installed on the end systems by the users. Initially the App Store only contained 500 applications, now it contains more than 500,000. Unless the user circumvents mechanism deployed by Apple on the iPhone, he will only be able to install applications directly from the App Store. Committing valid applications into Apple’s App Store requires obtaining a developer license which costs 99$ per year. The license includes a developer certificate which is also issued by Apple and used to create a signature for the application. In addition to signing the application, the application will undergo an undisclosed review mechanism by Apple itself.

![iOS Security APIs](image)

Figure 15: iOS Security APIs [iOSSec]

4.1.2.3.4.1 Security Features and Concepts

Some of the security features discussed below are described in [iOSSec]. Other security features, i.e. remote wipe and activation, are not documented for the general public.

- **Remote Activation** via iTunes to “unlock” the device for usage
- **Remote wiping** via Apple
Code Review of AppStore applications
- Code signing based on developer certificates that can be obtained by purchasing a development license from Apple
- FairPlay DRM as a digital rights management mechanism
- Encryption in baseband to make it harder to break into the firmware
- Chain of trust in boot process in order to detect low level integrity violations as early as possible
- Sandboxing on a per application basis
- Data Caging using permissions on memory regions
- Privilege separation (user vs. root) to minimize possible malicious impact

4.1.2.3.4.2 System Architecture

Apple's iOS is based on the desktop operating system Mac OS X. The kernel of the first iOS version reports to be Darwin version 9.0.0d1; following Apple's naming convention, the system is Mac OS X Leopard. Darwin is a free open-source operating system developed by Apple. It is the basis for Mac OS X and uses a monolithic XNU hybrid kernel, based on Mach 3.0. The Mach microkernel has been modified to include parts of the monolithic FreeBSD kernel for performance reasons. The I/O Kit has been added, which provides the driver infrastructure. Monolithic kernels do not only include functionality for process and memory management, but also device drivers and other features, in contrast to micro kernels. Its performance advantage over a microkernel results from the fact that there is no need for external driver software; however, monolithic kernels are more likely prone to malfunctions, because a faulty part cannot be restarted separately, but takes down the whole system. The Mach and BSD parts are written in C, while the language for I/O Kit is Embedded C++.

Before the release of the iPhone, Mac OS X only supported four architectures: Intel and PowerPC, each in 32- and 64-bit mode. With the introduction of the iPhone, support for its ARM architecture was added. Until the iPhone 3GS, all iPhones used ARM processors, the iPhone 4 and iPad use the Apple A4 system-on-a-chip which also includes an ARM processor.

Mac OS X Leopard passed the POSIX conformance test and is now allowed to use the UNIX trademark, although it does not use any AT&T UNIX code and its basis, XNU, stands for "X is not UNIX".

4.1.2.3.4.3 Boot Sequence

The very first code that runs on the iPhone is the bootROM, which is read-only. In the normal boot sequence, it is going to load a low-level boot loader (LLB). The LLB performs the setup of a basic file system and loads the next stage boot loader, iBoot. Its task is to load the device tree (a directory where the devices are located) from the flash, load the kernel and provide it with the device tree, because the kernel is not going to probe itself which hardware is available. The rest of the boot process is then done by the kernel and is similar to the Mac OS X boot sequence. The first process is launchd, which is the analog to Linux' init process. It starts daemons and registers IPC services. Other important processes that are started very early are the CommCenter, which manages the communication with the baseband processor and provides an interface to it, the USB communication daemon lockdownd and the user interface manager SpringBoard.
Apple uses a chain of trust that is described in a patent. Code signing is already involved in the boot sequence, as iBoot signature checks the device tree and the kernel including all resources; iBoot itself has previously been signature checked by the LLB and the LLB, in turn, by the bootROM, which uses the embedded Apple root certificate to verify the authenticity of Apple's public key.

The bootROM provides the option to boot into the so-called DFU (Device Firmware Upgrade) mode. In this mode, a firmware upgrade or downgrade is possible and the screen is just black. The device can communicate with a computer over a serial line. The DFU mode bypasses the boot loader, which makes it possible to downgrade the official firmware under certain conditions.

Moreover, a recovery mode exists, which enables flashing a new firmware. The recovery mode is a serial console via USB that is entered when one step of the boot process cannot load or fails otherwise.

### 4.1.2.3.4.4 Protection Mechanisms

The iPhone is known to be a very restrictive system that provides a lot of security features and protection mechanisms. This section describes the protection mechanisms of iOS in detail.

#### 4.1.2.3.4.4.1 Device Backups

Apple provides users the opportunity to synchronize their devices with a desktop computer and create full backups of all user data. These backups include everything that has not been present in the factory state of the phone, such as contacts, calendar entries and application data. They can be restored at any time, e.g., after a successful firmware update or manually on request by the user. A normal backup is stored in a new directory with the name of a unique device identifier (UDID) and a date. The UDID is calculated by using the SHA-1 hash over the device's serial number, IMEI, Wi-Fi address and Bluetooth address. Inside the backup directory, there are no more subdirectories. All files have different names, which can be restored with information contained in the `Manifest.mbdb` file. This file allows to retrieve the original file name, the source path and other attributes. The file name in the backup corresponds to the SHA-1 hash of its complete path.

By default, the files are only named differently, but the contents are not encrypted in any form. However, on explicit user request, iTunes can create an encrypted backup. Unfortunately, security researcher Jonathan Zdziarski was able to show that it is possible to circumvent the backup encryption completely [CF_ARS2009].

#### 4.1.2.3.4.4.2 Code Signing

Apple requires developers for their mobile platform to register and pay a fee of $99 per year. In return, Apple's certificate authority issues a certificate for the public key of the developer. He can now create an iOS application and sign it using the corresponding private key. The Apple-signed certificate is embedded into the application bundle such that the device is able to verify the code signature and check if the public key has been signed by Apple. In fact, signing is not only an option, but iOS requires every application to be signed by the developer. Before the execution of code, it checks if the signature is valid, if the certificate has not expired yet and if the public key was signed by Apple. Only if all these criteria are met, the application is allowed to run on the device.
Code signing provides authentication, because the platform is able to check if an application really originates from the claimed developer, and integrity, because it can be verified that the code has not changed after creating the signature. Additionally, it prevents repudiation by the developer, because a valid signature can only be generated with access to his private key.

The iOS kernel enforces the code signature check on every call of the system function `execve()`. It inspects the Mach-O binary (which is the binary format on iOS) and looks for `LC_CODE_SIGNATURE` segments. The segments carry the relevant code signature for the binary. If no signature is found for the address range where the code is located, the signature from the binary will be loaded and the pages verified.

One more important use case of code signing is that it is impossible to map libraries directly from the memory, because the signing is linked to the memory page permissions. They can still be loaded from the disk, but must be signed in order to have executable memory pages [CM_BH2009].

### 4.1.2.3.4.3 Sandboxing

Every application on iOS is executed in an own sandbox. The sandbox shields the application from the system and other applications in order to make it unable to do any damage to other system components. For this purpose, Apple uses a hosted hypervisor; a virtualization technique that does not need own device drivers, but uses those from the system.

The sandbox is realized as an extension for the TrustedBSD framework called Seatbelt. It is very flexible and allows profiles to define precisely which permissions an application is supposed to have or not to have. The same mechanism was introduced in desktop Mac OS X in version 10.5. Applications are installed in a dedicated directory that is named after a globally unique identifier (GUID). Below this directory, there are the directories `Documents`, `Library` and `tmp` and the `.app` package including resources, binaries and the code signature. The application is allowed to write its own directory, however the package is read-only.

### 4.1.2.3.4.4 Memory Protection

One security feature of ARM processors is the so-called XN (execute never) bit. This bit controls if a memory page is executable or not. iOS makes heavy use of it and protects every page on the stack as well as on the heap by making them non-executable. Moreover, no page can have the permissions read, write and execute (RWX) at the same time; only RW_ and R_X are possible [CM_BH2009]. This makes sure that no application can write code to memory and then execute it later, like, e.g., Flash does.

A frequently used attack type to circumvent non-executable memory is not executing code that the attacker wrote to memory himself, but re-use common fragments that are already loaded by the system. These return-to-libc attacks use buffer overflows to jump to, e.g., the C library that provides many useful routines for attackers. Using these routines, malicious and privileged programs can be composed. iOS 4.3 introduced address space layout randomization (ASLR) which takes care that the libraries are loaded to random addresses in memory every time they are loaded. On 64-bit systems, this makes return-to-libc attacks very unlikely, but on 32-bit systems, this improves the situation only insignificantly [SH2004]. Indeed, the ASLR implementation on iOS 4.3 has been defeated only a few days after its release.
4.1.2.3.4.4.5 DRM

Apple uses its *FairPlay* digital rights management to protect applications. Every application is encrypted using a *master key*, which is stored with the package in an encrypted fashion. There exist *user keys* that can be used to decipher the master key and eventually the application bundle. When a customer purchases an application, a new user key is generated to encrypt the master key. Hence, user keys are individual per user and application. This user key is stored with the Apple ID on an Apple server. Every device that wants to use an application must hold the user key in order to decrypt the respective application's master key. The keys are transferred to devices on synchronization with iTunes. Likewise, the user keys are stored in an encrypted manner, such that they can only be read by Apple software. In practice, users are allowed to install a purchased application to any number of iOS devices he owns.

However, FairPlay has proven to be defective and can be circumvented. The application binary is the only part that is encrypted. This especially means that the program code in memory is unprotected, hence, an attacker can easily use a debugger to suspend the program and create a dump of the memory to get the unencrypted binary code. Finally, the encrypted part of the binary is replaced with the plain one and the application's manifest file *Info.plist*, which contains declarative information on the application, is modified such that the system does not try to decrypt the binary (because it is not encrypted anymore) and signature checks are turned off, as the signature is invalid after the modification. There have been applications similar to Apple's official App Store that allowed downloading pirated applications for free. Of course, this only works if the device allows the execution of unsigned code, because the installer application would never be approved for the App Store and the unencrypted applications additionally carry an invalid signature. That is, the device must be *jailbroken* in order to turn off signature checks and install the illegitimate software repositories.

4.1.2.3.4.4.6 App Store

The very first iPhone generation did not allow any third-party applications to run natively on the device. Instead, developers were encouraged at the Worldwide Developers Conference 2007 by Apple CEO Steve Jobs to write *web apps*; applications that ran on the web, but mimicked the iOS UI elements and behaved like they were running natively on the device. An advantage of this policy is security, because no third-party code is actually run on the device. However, due to slow network speed (the classic iPhone did not support 3G networks), user experience suffered. Yet, iPhone web apps were from a user perspective oftentimes still superior to normal websites that were retrieved over a fast mobile data connection on other phones.

With the release of the iPhone 3G and iOS version 2.0, Apple decided to provide an SDK and allow native third-party applications on its devices. Developers were then able to register with Apple, pay an annual $99 registration fee and distribute their applications via the new online software shop App Store. The new client application allows the purchase, download and installation directly from the device. The App Store can also be accessed from the iTunes software from a desktop computer. All software that is bought for an iOS device is connected to the user's Apple ID. If the user logs in to the App Store with the same Apple ID he uses for his device, the software will be installed on the device at the next synchronization. In January 2011, Apple announced that 350'000 applications are available for download. Now, a year later in January 2012 it is over 500'000.

The App Store is the only Apple-legitimate source to install third-party software on the device.
Review Policy: The App Store Review Guidelines [APPDEV11] clearly state that Apple will "reject Apps for any content or behavior that we believe is over the line". Applications that Apple does not want on its App Store will be rejected. Common reasons for a rejection are that an application is malfunctioning, unstable, uses private API calls or behaves otherwise maliciously. For the end user, this is a huge quality improvement and security advantage, because Apple performs a set of analysis techniques to unveil unwanted or malicious behavior and prevent large parts of malware to be released to the devices, albeit the review process is not bullet-proof31.

During the review process, Apple performs a detailed analysis of the binaries submitted by developers. This analysis involves both automatic and manual tests for malicious behavior of any kind (for instance, loading and executing additional code or trying to write outside the allowed space), private API calls and poorly programmed software that does not function or does not comply with Apple's human-interface guidelines. Moreover, trial versions and software that Apple simply does not want on the App Store will be rejected [APPDEV11]. Apple's review process is known to be very strict and hence created large customer confidence in the store. Not only are the users getting applications of at least acceptable quality and user experience, but also functional software. This makes people trust the applications downloaded from the App Store. However, it does not protect from applications that serve a legitimate purpose, but use hidden malicious features, e.g., posting phone details to the Internet.

The review process and the App Store concept also have some downsides, just like the Marketplace for Windows Phone. The process sometimes seems to be heavily subjective and applications like Google Voice or Google Latitude had serious trouble getting approved for the App Store. Moreover, applications that enable, e.g., recording video with the iPhone 3G camera (which is officially not supported) are rejected, although the recording function is technically possible. Another example is the famous open-source media player VLC, which is available for several desktop and mobile platforms. Its source code is released under the GNU General Public License (GPL) and has been taken off the App Store again, because the GPL is incompatible to Apple's digital rights management used to protect applications from piracy. Brett Smith, member of the Free Software Foundation, says, "The GPL gives Apple permission to distribute this software through the App Store. All they would have to do is follow the license's conditions to help keep the software free. Instead, Apple has decided that they prefer to impose Digital Restrictions Management (DRM) and proprietary legal terms on all programs in the App Store"32. As the App Store is the only official way to install software on iOS, open-source software licensed under the GPL will not be able to be ported to iOS. Furthermore, legitimate applications that require full access to the device, such as firewall and anti-virus software, are excluded, because this kind of software that needs to take direct influence on other applications is excluded.

4.1.2.3.4.4.7 Permission Model

The permission model of iOS is implemented in an implicit fashion. iOS restricts access to sensitive frameworks. For instance, each application is automatically allowed to access the Internet or use the camera. Developers do not need to care about permissions that have to be requested, because the system displays the security prompt at the first use of a protected framework automatically. If the user allows access to the framework, iOS remembers this

32 http://www.fsf.org/blogs/licensing/vlc-enforcement
decision and does not ask again. If the user denies access two consecutive times, access is permanently denied for this application. Access to the address book cannot be revoked once it is granted.

4.1.2.3.4.5 Jailbreaking

To be able to run unsigned code, we must circumvent the checks in the kernel, which are enforced as soon as the execve() system call is execute. In addition to that, all applications run as the unprivileged user mobile, which means that even if there is a code execution exploit that allows running unsigned binaries, there is still the need for a privilege escalation exploit in order to be able to change important system parts.

The kernel is loaded from the system partition, which is mounted read-only. However, the kernel cache, which contains the kernel and all its linked extensions from the last boot, can be changed as root user. The kernel cache is used on Mac OS X to speed up booting if the hardware has not changed and the same kernel extensions (which provide driver functionality, comparable to Linux kernel modules) as before are to be loaded. And still, even if the manipulation succeeds, the kernel cache is signed as well and a modification would cause the signature check by iBoot to fail and prevent the device from booting.

Everything except the bootROM is signature checked. So, the most promising way to jailbreak is to attack the bootROM itself and sequentially patch out all signature checks in the chain. The great advantage in bootROM exploits is that they cannot be fixed by Apple, because a hardware revision would be needed in order to eliminate the vulnerability. This means that a jailbreak is always possible in future iOS versions, even if the vulnerabilities have been removed from the LLB, iBoot and the kernel. The signature checks just have to be patched out again.

Besides this permanent jailbreak technique, there are others which are reversible through a firmware update. Depending on which component of the boot sequence exhibits vulnerabilities, every stage is a potential entry point for jailbreakers.

Star Jailbreak: The most famous user land jailbreak is Star that is used by the website jailbreakme.com. It is able to jailbreak iOS versions 3.1.2 through 4.0.1 (without the iPad-specific 3.2.2).

It exploits a vulnerability (CVE 2010-1797) in the Compact Font Format (CFF) font parser that allowed unsigned code execution and additionally used a privilege escalation exploit for the IOSurface kernel extension. The user land process MobileSafari was used as injection vector, because it automatically opens PDF files. In the Star case, the user was redirected to a PDF file that included a malformed font in order to exploit the vulnerability in the CFF library libCGFreeType.A.dylib using a simple stack buffer overflow attack, where a long payload for a buffer in cif_decoder_parse_charstrings() allowed the attacker to control the program counter. Having now code execution, the payloads for gaining root access and the post-install instructions could be deployed.

The Star source code has been published (https://github.com/comex/star) by the author. The vulnerabilities have been fixed in iOS 4.0.2, however, there are still many devices in use that are not upgraded to a patched iOS version, such as first-generation iPhones. A patch for those devices is available for jailbroken phones, but no official one.

This kind of attack is very serious, because it demonstrates how to gain root access and arbitrary code execution by exploiting a user land application. If there is a similar vulnerability that has not been patched yet, it can be used by an attacker to drive-by infect devices and do anything he likes with full access.
4.1.2.3.5 Windows Mobile & Windows Phone 7

Microsoft Windows Mobile has been around as an operating system since 2002. It has originally been designed for PDAs. It is based on Windows CE 3.0 which is an operating system for PDAs and embedded systems. X86, MIPS, ARM and SuperH are supported to serve as the computing architecture. It is a closed source system which is programmable by an SDK released by Microsoft. With the release of Windows Mobile 5.0 in 2005, the .NET Compact Framework can be used for developing applications. Besides “native” programming, also JAVA, Python, and other programming languages are supported by the platform.

The newest release Windows Phone 7 supports applications running in a sandboxed fashion. They can be developed in C# or VB.net. Their respective runtimes offer different security features which will not be discussed in detail at this point. As opposed to previous versions, Windows 7 Phone claims to have moved critical operating systems components into the kernel space. The figure below shows the OS architecture layout of Windows 7 Phone.

![OS Layout Diagram](image)

The key concept of Windows Phone 7 is to “provide the capability to validate whether an identity may have requested access to a particular resource”. The concept of “chambers” is introduced between which there can be complex access controls deployed.

4.1.2.3.5.1 Security Features and Concepts

As many of the features mentioned below are based on the Windows Embedded CE 6.0, the reader is referred to [WMCE6] and [ValsWindows7] for a detailed description. Also [WP7DEV] offers details information of new features of Windows Phone 7.

- "cab"-signing (< Windows Phone 7)
- Sandboxing (C#, VB.net)
- Caging (Chambers, Code)
• Privilege separation (chambers + least privilege required)
• Capabilities (what APIs to use, ACL based)
• Security policies (Restrict what can run on the device)

4.1.2.3.5.2 System Architecture

The kernel is based on Windows Embedded Compact (formerly known as Windows CE) in version 7. It is an operating system for embedded devices and should be considered an own system rather than comparing it to a reduced desktop Windows version.

On the kernel layer, device drivers are placed. As a novelty compared to former Windows Mobile systems, most of them are written by Microsoft and only the very silicon specific drivers must be written by terminal manufacturers. Also, security and networking implementations can be found on the first layer on top of the hardware foundation.

The next layer provides the basis for all applications running on Windows Phone 7 and consists of three parts. Cloud integration is an important factor in Microsoft's new platform and there is a close connection of a phone to a Microsoft Live ID account, which handles tasks like holding licenses for applications and performing backups of, for instance, contacts and calendar entries. Bing is used for searching the device, the Internet and locations nearby. The UI model bases on a pages principle that can be compared to browsing the Internet and on the design language Metro.

On the applications layer, the basis is a reduced version of Microsoft's established Common Language Runtime. It compiles the source code of applications into a Common Intermediate Language which is then translated to native code using a just-in-time compiler. Windows Phone applications can be written in any .NET language. They cannot access devices or sensors directly, but must use the provided API and frameworks instead. The XNA framework allows developing games, Silverlight can be used to write applications.

4.1.2.3.5.3 Protection Mechanisms

The Windows Phone 7 security architecture uses isolation and least privilege and introduces chambers as a concept. A chamber provides a security and an isolation boundary inside which processes can run. Different security policies can be defined for chambers and be used to create a security level hierarchy. Four different types of chambers exist. The most permissive one is the Trusted Computing Base (TCB) chamber, allowing unrestricted low-level access. The kernel and device drivers run in TCB chambers. The exclusive capability of the TCB chamber is the possibility to modify the security policy. This separates it from the Elevated Rights Chamber (ERC), which is intended for user-level device drivers and services that provide phone-wide functionality or shared resources that are going to be used by other applications. Consequently, applications providing non-global functionality, run in the third chamber, Standard Rights Chamber (SRC), which is the default for preinstalled system applications. Finally, the Least Privileged Chamber is the chamber in which every third-party application will run in.

4.1.2.3.5.3.1 Capabilities

As applications request permission to accessing protected APIs during the deployment process, resulting capabilities are declared in a so-called application manifest file. The creation of the capability list is done by a detection utility shipped with the developer tools.
The requested capabilities are displayed to the user in the Marketplace application on the phone and he is asked to explicitly approve them prior to the installation process. Minimal capabilities are automatically granted, which can only affect the application's own scope. For example, writing an isolated storage file is allowed without special request. The permission checks are enforced at runtime.

The use of the capabilities detection utility ensures that all and not too many capabilities are declared by the application. However, granting the permissions is still the user's responsibility. The automatic generation of the capabilities list does not add additional security benefits, because if an application tried to perform an action that uses a protected resource without declaring the appropriate capability, the execution would fail at runtime.

4.1.2.3.5.3.2 Code Signing

Only packages carrying a valid Microsoft signature will be installed, according to the capabilities defined in the manifest file. This file is also inspected during the review process of Microsoft. Only after the review applications will be admitted to the Windows Phone 7 Marketplace. Also, the application cannot take control over the update or deinstallation process; the decision if and when any of these actions are performed, is entirely left to the user. However, the Marketplace still can intervene. The licensing mechanism also supports trial versions of applications. The Marketplace also checks for license revocations. If a license turns invalid or an application's license is revoked, the Marketplace can initiate the deinstallation.

Windows Phone 7 does not require the developer to sign his code, but leaves this task to Microsoft. After reviewing an application on submission and deciding to approve it, the application package is signed by the company.

4.1.2.3.5.3.3 Sandboxing & Isolation

Applications cannot influence each other and cannot interfere. It is always guaranteed that there are enough resources available for the application to run, because the foreground application receives full priority. Isolation also ensures privacy to the application data. It cannot be modified or even read by other installed applications. Using a sandbox, the system prevents applications from accessing native APIs.

The system provides a host process for the application to be launched in. Before each execution, the code is checked for integrity by the runtime. Only if the check confirms the validity of the code signature, the software will be allowed to run. At runtime, applications are supervised by the Execution Manager. It monitors the application's usage of system resources and may terminate the process if it considers the application misbehaving or unresponsive.

4.1.2.3.5.3.4 Windows Phone 7 Marketplace

The Marketplace acts as a gatekeeper for third-party software, being the one central source of third-party applications for their devices. It is the only official software shop. Developers must submit their application packages to Microsoft for review, before the application is published on the Marketplace. The main purpose is to exclude malware and find poorly implemented applications that disrupt the user experience. Hence, the user is supposed to only see well-written and high-performance applications and gain the confidence that Marketplace applications are of good quality and will not do any harm to his device.
If the application complies with the rules, Microsoft decides to approve the application, it is going to digitally sign the application package and issue an appropriate license. This license also provides a copy protection functionality to ensure that a legally purchased application cannot be transferred to another phone and run there without paying again. In order to be able to post applications to the Marketplace, developers must register with Microsoft and pay an annual $99 registration fee. In return, Microsoft will sign their applications and issue licenses.

The Marketplace also offers non-repudiation. Developers must prove their identity before posting applications to the Marketplace. Depending on the key that is used for the code signature, applications can always be tracked back to the original developer.

Marketplace also provides a protection of intellectual property. As software piracy becomes increasingly significant on mobile devices, there is the need for such a mechanism. Every application is not only signed by Microsoft, but is also has an according license that is needed in order to run. This license is saved with the Windows Live ID and queried at every application launch. So, even if attackers manage to sideload applications, there is still the need for the respective license in order to run.

4.1.2.3.5.4 Jailbreaking

So far, there has only been one known vulnerability used for jailbreaking Windows Phone 7. However, this vulnerability enabled hackers to install applications that have not passed the review process. Those have not been signed by Microsoft and never appeared on the Marketplace. ChevronWP7 for desktop PCs allows users to sideload applications that are not available in the Marketplace. This is normally a feature that is exclusively available to registered and paying developers. The unlock software employs a fake Microsoft server tricking the device into thinking that it is legitimately registered as a developer phone.

ChevronWP7 bypasses the code signing and licensing mechanism of Windows Phone. Microsoft has acknowledged the issue and it has been fixed with the NoDo version of Windows Phone 7. After all, the vulnerability was not too serious, because no security mechanism of the system was compromised. This functionality can also be legitimately achieved by developers for the yearly fee of $99. Moreover, Windows Phone 7 calls home to Microsoft every two weeks in order to find unlocked developer devices that must be locked again, because the developer is not a legal subscriber in the developer program (anymore). For this purpose, a device identifier is sent to Microsoft and compared to the ones stored on Microsoft's servers. The ones belonging to ChevronWP7-unlocked phones will not appear on this white list and the devices will be re-locked.

4.1.2.3.6 MeeGo (Linux)

Instead of considering all of the many Linux smart phone operating systems we will turn to MeeGo as an example operating system. Instead of targeting only smart phones, MeeGo also has netbooks in mind. It is a combination of Nokia's Maemo and Intel's Moblin project. MeeGo supports X86, Intel Atom and ARM. Its current version is at 1.1.1 which was released in November 2010. The figure below shows an overview of the architecture of MeeGo.
Applications for MeeGo are envisioned to be distributed as rpm, the well known Red Hat Package Manager format, via Intel AppUp, Nokia’s Ovi Store and other community repositories. Development of the applications can be done using Qt or QtQuick (JavaScript-like). The key security concept of MeeGo is based on the “Mobile Simplified Security Framework” (MSSF) which was developed by Nokia.

4.1.2.3.6.1 Security Features and Concepts

The security features of MeeGo are based on the Mobile Simplified Security Framework (MSSF). [MSSF] provides a detailed description of all the features that are mentioned below.

- **Chipset security** at OS level provides tamper resistant secure services similar to TPM
- **Trusted computing base** to facilitate e.g., secure boot
- **Capabilities** restricting access to critical resources
- Inter-application privacy protections
- **Data caging** for process based on file system access rights (possibly encrypted)

4.1.2.4 Comparison of the Security Features of the Most Popular Operating Systems for Current Smart Phones: Android, iOS and Windows Phone 7

This section covers a comparison of the to date most popular smart phone operating systems, Android, iOS, and Windows Phone 7. It discusses the different security features of each smart phone operating system.

4.1.2.4.1 Hardware Features

All three platforms are currently based on the ARM architecture. The processor features protection mechanisms that are hardware-supported. One example is the **TrustZone** of ARMv6KZ and newer architectures (Cortex-A series and ARM1167 processors). It provides
two virtual processors. Security-relevant operations can then happen in the more trusted world and be prevented from leaking to the less-trusted one.

This technique can be used to run the main operating system of a Smartphone in the less-trusted world and important security code in the other. iOS uses a similar approach by hiving off the radio communication into an own baseband system by adding an extra chip, despite the integrated Cortex-A8 processor provides a similar feature with the TrustZone. The TrustZone is not in use by any of the systems.

In addition to the TrustZone, ARM processors offer memory page protection using the execute-never (XN) bit. Apple makes intensive use of it, Android does not protect its memory using this technique. Its stack and heap are executable and memory is not randomized at all.

### 4.1.2.4.2 Jailbreaking

The motivation for jailbreaking device is essentially the same for all platforms, which means that some users consider it desirable to circumvent certain security mechanisms on the phones. The most important reasons are that certain features that the devices' hardware supports are disabled or that some software features are completely missing. These features can often be (re-)enabled or installed with third-party software that is granted the appropriate rights to do that. Granting some of these rights, however, is impossible on regular systems and requires the device to be jailbroken.

The approaches to get the desired privileges for applications differ. The challenge for jailbreakers on iOS and Windows Phone 7 in the first place is that they must run code that has not been approved by Apple or Microsoft, respectively. On Android, this is not a problem, because the system accepts self-signed certificates. Arbitrary code execution on Android does, however, not mean that the running code is automatically privileged. It will still run within the application's sandbox. Consequently, the problem reduces to finding a privilege escalation exploit that allows to run a jailbreaker-originating process as root user.

So, jailbreaking iOS adds the extra burden to have to find a way to bypass the signature checks the system employs. The chapter on iOS showed that the checks are compiled into the operating system kernel. The two options are now to create a valid signature for every binary that is supposed to run on the system on the fly or to apply a patch to the kernel to disable the signature checks completely. The first is computationally infeasible, which leaves only the patching option. Patching can be done in the kernel memory in the running system (so-called user land jailbreaks) after using a privilege escalation exploit, this patch has to be re-applied after every reboot though; or permanently in the kernel representation on the flash memory. The second, however, invalidates the kernel signature which is checked in the chain of trust in the boot sequence which requires patching the previous boot stages as well. The first code that is run is the bootROM, which is directly loaded from the flash memory and cannot be signature checked. Exploiting a bootROM vulnerability is the most desirable option, because the jailbreak can then be permanent and bootROM vulnerabilities cannot be fixed without a hardware revision.

For Windows Phone 7, similar conditions as for iOS hold. There is the need for unsigned code execution and a privilege escalation exploit. However, there has been one way to run code not approved by Microsoft. By tricking the phone into thinking that it is a developer phone, it runs every code. This "jailbreak" enables normal users to gain the same privileges legitimate developers have after paying the registration fee. There is no protection mechanism that has been circumvented and the registration as a developer phone is non-permanent, because the device contacts the Microsoft servers in regular intervals in order to
confirm its state as a developer phone. If this confirmation fails, the device is returning to the locked state.

4.1.2.4.3 Code Signing

All systems are checking the signature of each application before executing it, yet only iOS and Windows Phone 7 use a reasonable code signing policy that satisfies the needs of a solid protection mechanism for mobile devices. The two platforms employ a certification authority that must sign a certificate for the developer's public key. These certificates are issued only after paying a yearly registration fee. Third-party developers are thus known to Apple and Microsoft, respectively, and applications can always be tracked back to the developer, because digital signatures prevent repudiation.

Android developers are free to generate as many key pairs and certificates as they want, because Android does allow self-signed certificates. Even worse from a security perspective, there is not even a need to register as a developer. A registration enables developers to post their applications on the Android Market and sell them, but there exist many other ways to distribute an application, especially those involving social engineering. Pointing a user to a URL to the application will prompt him to install it. By default, the installation of applications from non-market sources is disabled. The user, however, can override this setting in the system settings. A notification box suggests how to turn off this security mechanism if he tries to install a non-market application with active protection.

4.1.2.4.4 Sandboxing

Sandboxing is the system creating a dedicated environment for all applications that are running. Applications are entirely shielded from each other and are thus typically not aware of other applications that are running. Android, however, makes an exception to that by allowing applications that are signed with the same key to run in the same sandbox. This is done by assigning the same Linux user ID to the two applications, because Android implements the sandboxing principle by normally assigning a new user ID for each application and setting the file system and process owners and permissions according to the application's user ID.

Old iOS versions used a kernel module in order to enforce their Seatbelt sandbox, but newer versions integrate the functionality directly into the kernel. The sandbox is enforced by using virtualization and an appropriate hypervisor. Android does also use virtual machines to run applications, but they must not be confused with a sandbox, because it does not prevent applications from breaking out. On iOS, there is no way for two applications to share the same sandbox, even if they originate from the same developer.

Windows Phone 7 adds a so-called ExecutionManager service to the sandbox. It is supposed to supervise applications and to intervene if it detects suspicious behavior in one of the applications. While Apple and Google have similar features in terms of a watchdog that kills unresponsive applications, the Windows Phone ExecutionManager is explicitly designed to enforce the system security policy.

4.1.2.4.5 Remote Service Connections

Apple, Google, and Microsoft keep a persistent connection to the devices in order to be able to execute remote commands. While this seems like a deep intervention into user privacy, these connections also realize some important security features that are connected to the account with the respective vendor. It is possible to locate the phone on a map, lock it or even completely erase all user data in case it is stolen. Moreover, applications can be
removed remotely and push notifications are delivered over the connection that is held over cellular or Wi-Fi networks. However, the GSM-capable devices must have the SIM card installed in order to identify the device for highly security-relevant actions affecting only one single device.

There is one specialty with Android's GTalkService connection. It is also used to remotely install applications and even more, this is the normal case. As soon as the user decides to install an application from the Android Market, an INSTALL_ASSET intent is sent to the phone. Whether the other two platforms have the capability to remotely install applications is unknown.

4.1.2.4.6 Permission Models

The permission model is the main security mechanism that is in use at an application's runtime. It protects security-relevant resources and services and grants access to them only if the request has been approved by the user before. Android and Windows Phone 7 use an explicit implementation; both ask the user to approve the requested permissions before installing an application and cancel the installation if the user denies. This leaves the user with the uncomfortable choice to either grant all permissions or none. The two systems use a very fine-grained permission model (Android itself defines 116 different permissions), while iOS only protects very few frameworks and follows the implicit way. It does not ask for any permission at install-time, but instead at runtime of an application as soon as it wants to use a protected framework for the first time. Apple additionally relies on its review process to prevent malicious applications from ever entering the App Store. This complies with its policy that Apple devices must be easy to use and ideally run without any configuration necessary by the user.

The different approaches yield different security implications. One common problem of all three platforms is the missing granularity. Even though Android defines many permissions in comparison to iOS, its model is still not fine-grained enough for some cases. For example, it is only possible to disallow Internet access for an application, however it can make sense to restrict the Internet access to only incoming traffic or to a certain host or network. None of the systems provides the chance to restrict the access in such a way. At least, the Android platform allows the development and integration of anti-virus software and personal firewalls. On the other two platforms, this kind of software is excluded in advance due to the lack of real multitasking. The Android firewalls still face the problem that they cannot escape from their sandbox and thus cannot influence other running applications. So, root access is required.

4.1.2.4.7 Software Distribution

The platforms disagree on the question if there is protection prior to the installation necessary. For Android the only requirement to applications is that they must be signed by the developer. This can be done using a self-signed key, which means that developing an application can be completely anonymous. It suffices to copy the application package to the device and open it in order to install the software. If developers additionally want to distribute their application via the Android Market, they pay a one-time $25 fee upon registration for the Market and are immediately able to upload applications, which will also immediately appear on the Market ready for download. The Android Market does not employ reviews or tests on applications posted to the Market. It entirely relies on the user community to flag inappropriate or misbehaving applications. This means that some users must install a malicious application, notice its malicious behavior and report it, before it is being pulled from the Market.
iOS and Windows Phone 7 follow a very restrictive strategy in their software shops. The App Store and the Marketplace, respectively, are the only legitimate sources for applications. This concentrates the malware sources and forces applications to pass Apple’s or Microsoft’s approval process which include automatic as well as manual tests of the submitted binaries. No review or analysis can be done on the code, though. Developers only submit the application package which contains the binary, but no source code. The approval processes can discover most kinds of malware, e.g., by analyzing which features and frameworks are used. Applications that attempt to read system data or to elevate their rights can then be identified and rejected. This filters a large part of the malware that would otherwise appear on the shops.

In addition to the review process, the required use of the software shop on both platforms and the corresponding registration duty for developers, applications can be linked to the respective creator. The annual $99 fee that both vendors charge for being able to distribute applications on the shop is only a minor obstacle on the way to publishing malicious applications.

There are several ways to pass the review process despite carrying malicious code. The simplest idea is to plant a time-bomb, to disable the malicious features until a certain condition holds true in order to not be discovered in the review process. For example, the features can turn active only after a certain date or after verification with a developer-hosted server. In this case, all code has been present (but inactive) the whole time, reviewed and approved by Apple and carries a valid signature. Users are not going to expect malicious behavior from App Store or Marketplace applications.

4.1.2.4.8 Attacks

This section covers a selection of historic attacks, as well as an exemplary attack on the permission model of Android composed of a collection of individual attacks. For jailbreak related attacks, the reader is referred to the subsections of the relevant operating system discussion of the previous chapters.

4.1.2.4.8.1 Historic Attacks

4.1.2.4.8.1.1 Russian SMS Trojan

Trojan-SMS.AndroidOS.FakePlayer.a was one of the first Android malware. The application pretends to be a music player. A background service sends SMS messages to expensive Russian premium numbers. As being one of the earlier - not yet too sophisticated malware - it did not try to compromise any other security mechanisms of the phone. On installation, it requested permission to send SMS messages and it relied on the fact that users oftentimes do not read the potentially long list of permissions an application requests and just approve it. The application showed no indication of attempts to silently spread to other phones or harm the device in any other way.

4.1.2.4.8.1.2 Jailbreak-Dependent Malware

The 2009-published ikee worm was the first malware discovered for the iOS platform [CL_2009]. The worm did not damage the phone, but changed its background image to a photograph of Rick Astley. Additionally, it tried to spread to other devices on the network. Ikee only attacks jailbroken iOS devices with SSH server installed. It exploits the design flaw that the iOS root account is enabled by default and uses the weak password alpine, which is the same on all devices and still active on the newest iOS version. The SSH server now
opens the channel to the login prompt. This makes it very easy for the worm to find other vulnerable iOS devices and copy itself. This technique is very simple, has been exploited multiple times [MO_2009] and the consequences have not always been as harmless as ikee's. There have been worms that steal the user's SMS and other data and one particularly more dangerous version. It uses the same SSH entry point, but changes the root password and thus locks out the user from his device. It has the option to load more code from the Internet and hence to add more attack capabilities. In addition to the local network, it also searches known IP address ranges of Internet Service Providers for vulnerable devices.

This is a serious attack vector, because the attack is very simple, full access to the device is immediately available and central security features have been disabled. However, this device state must have been procured by the user by actively jailbreaking the device and additionally installing the SSH server. So, despite the attack is serious, only a small amount of devices are actually vulnerable to the attack. A default root password is generally a bad idea, however, if the attacker never gets the chance to log in (e.g. by disallowing the user to log in or not providing a shell login at all), this seems to make a default password acceptable.

4.1.2.4.8.1.3 Stealing Market Authentication Tokens

Jon Oberheide reported in a talk [OL_SC2011] at ShmooCon 2011 that it was possible to install applications without user consent using Google's INSTALL_ASSET intent. Instead of directly attacking the SSL-encrypted GTalkService connection and trying to inject a payload, Oberheide suggested imitating the request that the device sends to the Market servers on legitimately purchasing an application. He reverse-engineered the installation part of the Market API and found out that the needs are similar to the ones for the application package download. It could be shown that the illegitimate installation uses the Market authentication token and the device ID, both of which can be read directly from the Android device. In addition, the ID of the application that is to be installed is needed. Using this, he was able to construct the Protocol Buffer request, Base64-encode it and send it to the Market servers via HTTP POST. A proof of concept application was uploaded to the Market. It installed three more applications that were granted arbitrary automatically without the user ever approving it.

The vulnerability has already been fixed with Android 1.6 Donut, which now requires the permissions READ_ACCOUNTS and USE_CREDENTIALS to read the Market authentication token. In addition to that, the Market servers now only respond with HTTP code 400 (bad request), even with the valid tokens.

Neither Windows Phone 7, nor iOS allow third-party applications to authenticate with the respective software shop via a token that applications can read in a legitimate way and thus, the systems are not susceptible to the attack.

4.1.2.4.8.1.4 DroidDream

In February 2011, Google had to remove several applications from the Market, because they were infected with malware [MA_2009]. The developers took legitimate applications from the Android Market and added rooting functionality to the applications. The modified applications were re-bundled.

Applications infected with the so-called DroidDream malware install a rootkit exploiting one of two different vulnerabilities, depending on the running operating system version. One is the well known udev vulnerability of Android. The other exploits basically the same vulnerability as used by the zimperlich exploit. The Android Debug Bridge (ADB) exhibits the same vulnerability as the zygote process does. ADB uses a device-side daemon used to open a
debug shell from a USB-connected computer to the device. Just like zygote, it is allowed to spawn processes. It also suffers from the return value check of the setuid() call vulnerability and is thus prone to exceeding the maximum allowed number of processes. This exploit is also known as RageAgainstTheCage.

All DroidDream applications define two services, com.android.root.Setting and com.android.root.AlarmReceiver. The first performs the decryption of a byte buffer, which has simply been XORed with a key defined in the class adbRoot. The plaintext contained an IP address and the server URL that the data collected by DroidDream should be sent to. The DroidDream-infected application itself posts only details on phone and SIM card to the server, such as IMEI, IMSI, the SDK version and the device model. Additionally, it installs an application that it brings which is then the actual rootkit and able to post more phone details and download instructions and content.

This is probably the most serious kind of attack; an application silently roots the phone, disguising its malicious intention by looking exactly like a legitimate application and imitating its functionality in a way that a user is unable to tell the difference between the original and the infected version. Even though reverse-engineering iOS and Windows Phone applications is more difficult, the two platforms are in general susceptible to these attacks as well. However, for Apple’s iOS there is the need for unsigned code execution in addition to the privilege escalation exploit, because it is under normal circumstances impossible for applications to run unsigned code. For Windows Phone 7, there have not been any indications of a privilege escalation exploit so far.

4.1.2.4.8.2 Attacks on the Android Permission Model

This section cover a composed attack Android’s permission model aiming at silently rooting the phone. The basis for this attack is a non-jailbroken Android device and the attacker’s application is going to be installed with limited permission. In the past there has been research on the permission model of Android ranging from survey studies by Felt et al., [Felt_1_2011] to explicit attacks, e.g., [Felt_2_2011], [Shin_2010], and [Chin_2010]. Here we compose an attack path based on previously known vulnerabilities, as well as new vulnerabilities in a comprehensive way.

First the attacker’s application will take over the UI of the device to be able to trick the user into installing a malicious application. Next, the attacker’s goal is to run the application directly after it has been installed, but without further interaction by the user. Also, the attacker makes sure that his application is automatically started once the phone is rebooted. Next, the attacker will craft a bidirectional internet connection without requiring the actual INTERNET permission. This allows to download and send arbitrary data, e.g., to down a root exploit. After having acquired the root exploit, the attacker can silently root the phone effectively compromising the whole system.

4.1.2.4.8.2.1 UI Takeover

The idea of the UI takeover is to intercept all keystrokes and keep the current application in the foreground. Thus, the device is effectively blocked as there can only be one activity running at a time. We implemented this in the so called KeyInterceptor as a proof of concept.

The Android operating system allows applications to intercept all key presses for further processing. As soon as a key press occurs, the current foreground activity is queried whether it wants to handle the key press, or if the event should be forwarded to the next receiver in the queue. For this purpose, the activity’s onKeyDown() method is called automatically. In this method, the application will handle the key press and finally indicates if the application
did indeed handle the key press, or if it is supposed to be forwarded. If the method returns false, the press will be forwarded, otherwise it will be discarded. The proof of concept KeyIntercepter application uses this mechanism to intercept all key presses. However, it will just discard them and thus disables all keys by indicating to handle them, but doing nothing.

The onKeyDown() method is called for every key except the Home button. If the Home button is pressed, the system will immediately pause the activity in the foreground and return the user to the home screen of Android. Holding down the Home button for a few seconds will show a list of recently used applications. In either case, the current activity will be paused, because the Home screen or another application comes to the foreground. As a result, the KeyIntercepterActivity of the KeyIntercepter will be notified by automatically calling the onPause() method. In this method, the application uses an intent (the very same it was originally started with) to restart the KeyIntercepter and finishes. Thus, the KeyIntercepter will be able to intercept all keystrokes again. Depending the implemented logic, such an application can effectively control which buttons the user is allowed to press (as following in the next paragraph).

When installing an application from the Android Market the user can browse and gather the author and the price of the applications. If the user decides to install the application, he taps the price tag which is also the install button. The button turns into the OK button and the necessary permissions of the application will be shown. Thus, the same button is used to present the details of an application as well as to approve permissions. This behavior can be exploited by the UI takeover attack. As we have detailed, some components of applications may run in the background. Thus, using the KeyIntercepter, an attacker can either force the user into only being able to press the button which the attacker chose, or block the UI for an amount of time. This can be experienced as the device reacting sluggishly. With a high probability the user will tap more than once thinking his first tap has not been recognized or he missed the button. When the system finally becomes responsive again, it interprets the second tap and installs the application.

4.1.2.4.8.2.2 Post-Install App Invocation

The Android operating system sends out broadcast intents in order to notify installed applications of events that have taken place. Examples for these events are the date change at midnight, the battery status getting below a threshold, or the fact that the system has finished booting. These broadcasts are a type of inter-process communication. Typically, users expect to have to explicitly interact with the device in order to run applications for the very first time. Applications can be started automatically by using the Google Analytics SDK. This framework allows tracking of referrers in the Android Market since versions Android 1.6 of the operating system. For this purpose, the Android system sends the intent INSTALL_REFERERRER right after the installation of an application to this particular tracking application. If the attacker’s application implements a corresponding receiver method, it will also receive the intent and handle the referral information sent to it. The way it was originally intended to be used is in combination with the Google Analytics SDK, which itself is supposed to deal with the information for the user in the final application.
The Android operating system documentation suggests the use of the Google Analytics broadcast receiver by integrating it in the way shown in Figure 18.

It is also possible to implement a broadcast receiver to listen to the INSTALL_REFERRER intent and handle it, e.g., in an attackers' application. It is only necessary to replace the name of the Analytics broadcast (in shown in the figure above). Since Android version 2.2 the broadcast receiver is directly notified after the installation of an application. Therefore, as soon as the custom broadcast receiver is instantiated, in order to be notified of the referral, it can execute any code it desires to. Thus, applications can immediately be started after their installation.

4.1.2.4.8.2.3 Post-Boot App Invocation

The Android operating system allows to send out broadcast intents to other applications or their components. Concerning the boot sequence of devices, Android uses the BOOT_COMPLETED broadcast intent. This intent is sent after the devices successfully booted. Applications can register a broadcast receiver enabling them to react on this intent.

Applications can be prevented from illegitimately starting after boot by requiring the RECEIVE_BOOT_COMPLETED permission. However, enforcing a check for this particular permission is a tricky thing to remember during implementation.

Applications can successfully register to listen for the BOOT_COMPLETED intent without asking for the necessary permission. The user will not notice that this intent has been sent at all.

4.1.2.4.8.2.4 A Bidirectional Communication Channel

Android restricts access to the Internet by forcing applications to request a permission called INTERNET. The requesting application may freely access the Internet without being restricted by the system if it holds this permission. A particular issue is caused by the transitivity of Android's permission model.

Applications are able to define custom permissions and to protect their components. For instance, when the browser component of Android accesses the Internet, it requests the INTERNET permission. The browser is also intended to be launched by other applications that desire to display a website. However, using the browser is not protected by a separate permission, although it in fact enables Internet access. This is an example of the transitivity of permission.
After firing the VIEW intent for a website URI, the Android operating system will launch a new browser activity and load the requested page. Despite not holding the INTERNET permission the application can use the browser to load a potentially malicious website.

However, in order to be able to receive incoming data as well, a custom URI scheme needs to be registered for the attacker's application. URI schemes that an application desires to handle must be defined in the application's manifest file and are not shown to the user.

Once the application is installed and has been launched, it can launch a browser and have it start loading an attacker-controlled website. This website could return data by redirecting the browser to the custom URI scheme which the application listens to. The browser automatically hands over the return data to the application that registered for attacker's scheme.

4.1.2.4.8.2.5 Covert Jailbreak

A very popular method for rooting Android is zimperlich jailbreak which is able to root devices running Android versions prior to 2.3. While the original version creates a copy of the default shell /system/bin/sh, we set the setuid bit on the shell executable which is owned by root. That is, every instance of the shell runs with root privileges.

The exploit binary has to be compiled from zygote.c in order to run on Android devices.

Once the source code of the native (as in non-Java) jailbreak code is compiled, it must be named accordingly with a lib prefix and carry the file extension .so. The file must be placed in the libs/armeabi directory of the Android application project. Our proof of concept implementation uses the so called JailbreakActivity.

The application starts an infinite loop which executes the jailbreak binary over and over again until it succeeds, i.e., Android throws an exception, as the current user to which the process belongs is exceeding the maximum number of allowed processes. The new process will keep its root permissions that it inherited from the process it was forked by, the zygote. Now that the maximum allowed number of processes is reached, it only remains to have the zygote fork a new process. This means that there must be a new application component that is instantiated as soon as the exception is caught.

A dummy application called Permissions, which only requests the permissions to access the Internet and to initiate phone calls is used. The Installer demonstrates that it is possible to root the device and install a package requesting any permission completely without user consent or even notification. With a root shell, it is already possible to read all desired data like contacts, SMS messages and device internal information. Installing additional applications makes it more comfortable to use Android's frameworks.

4.1.2.4.9 Summary Comparison of Android, iOS & Windows Phone 7

We have compared many of the protection mechanisms that are employed by the systems. This section is going to contrast the implementations of the platforms. Table 2 lists the protection mechanisms and evaluates the solutions of the single systems with the ratings ++, +, 0, - and -- from best to worst.

The sandboxing mechanisms implemented have proven to be only partially secure. All platforms allow the interaction of applications with daemons, libraries or frameworks that are running natively and are privileged. This makes it possible to exploit vulnerabilities that can allow privilege escalations.
In terms of memory protection, Android has a lot of room for improvements. Neither does it protect its stack or heap (both are executable), nor does it use a technique like ASLR in a reasonable way. Apple, in turn, makes heavy use of the XN bit of the processor and never allows any application memory page to be writable and executable at the same time. Since iOS 4.3, it even employs ASLR. Windows Phone 7 only runs managed code. However, if unsigned code execution is possible, it also runs native code that leaves the Common Intermediate Language in which the apps are programmed.

The code signing mechanism is implemented very differently. Apple and Microsoft score with their involvement of a certification authority, while Android's solution only enables developers to prove his authenticity in order to be able to share data between his applications and to submit application updates. iOS enforces the checks at every system call that starts a new process and hence outscores Windows Phone in this discipline.

On all three platforms, service connections are encrypted and the public-key certificates are checked for validity, which results in proper authentication of the remote station. None of the systems allows the use of user-installed certificates for the service connection. This mitigates possible man-in-the-middle attacks using self-signed CA certificates.

<table>
<thead>
<tr>
<th>Protection Mechanism</th>
<th>Android</th>
<th>iOS</th>
<th>Windows Phone 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sandboxing</td>
<td>○</td>
<td>○</td>
<td>+</td>
</tr>
<tr>
<td>Memory Protection</td>
<td>--</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Code Signing</td>
<td>-</td>
<td>++</td>
<td>+</td>
</tr>
<tr>
<td>Service Connection</td>
<td>++</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>(Application) Copy Protection</td>
<td>+</td>
<td>-</td>
<td>++</td>
</tr>
<tr>
<td>Application Shop</td>
<td>--</td>
<td>++</td>
<td>++</td>
</tr>
<tr>
<td>Security</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Permission Model</td>
<td>-</td>
<td>--</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: Comparison of Protection Mechanisms

iOS’s second issue is the copy protection. Despite it encrypts all binaries, they reside in memory in the clear as a whole, which enables attackers to attach a debugger and dump the binary to disk. Android and Windows Phone score with the option for developers to contact the vendor's servers from an application in order to verify if the account associated with the phone indeed owns a license for the application.

The most important discipline from a security perspective is preventing applications from gaining full access to the device, i.e. preventing a jailbreak. Android and iOS have deficits due to the fact that applications are allowed to run natively (as in not using application specific language), while Microsoft will not allow applications containing native code on its Marketplace. Reported malware incidents on the App Store only involved minor privacy issues such as unique device identifiers being transmitted. As such Apple seems to be less affected by malware incidents to date.

Google permits every application on the Market as long as the developer is registered with Google. There is no review whatsoever and the entire security concept relies on the user community to identify and flag malicious applications. This has caused the Market to be
flooded with applications that are malicious in any kind. Google has proven to maintain very short reaction times if a really dangerous application is spotted on the Market.

None of the platforms found a satisfying permission solution that is robust in its application (i.e. applications without a particular permission cannot gain it otherwise), fine-grained enough to restrict applications to the least privileges they need and flexible enough to allow the user to revoke particular permissions at the same time. Apple fails horribly regarding the granularity by only protecting address book, location and push notifications. The permission models suggest the presence of application-level security to the user that is not actually there in an acceptable manner.

4.1.2.4.10 Risk Evaluation of Threats to Android, iOS & Windows Phone 7

Jailbreak was described as the worst attack that an attacker can mount on a mobile device. For Android devices, we have seen that it is very likely that applications are and in the future will be able to jailbreak the device from an application in various ways. The consequences of an application-based jailbreak for the other two systems are equally fatal.

Privacy issues are one of the key concerns when designing a security concept for a smartphone platform, as they carry large amounts of sensitive or valuable data and many applications are granted access to sensors, files and other information. Unfortunately, none of the compared operating systems allows control over how the information is processed and there is no chance to prevent an application with these permissions from sending the information to a server on the Internet. Consequently, this kind of attack is very common on all platforms.

Malware oftentimes tries to spread to other devices. This implies danger for contacts that are stored in the address book of an infected phone. If their private details can be read, malware can, for instance, try to spread via SMS or email. On Android, reading the contacts and sending email is possible by rooting the phone, which can be done silently. While simple spreading of malware by sending emails does not suffice for an installation on the remote victim device, it is perfectly enough to send spam emails. The other two systems employ a reasonable protection for their address book. Because unwanted jailbreaks from the user land are unlikely, stealing contact information on iOS devices and attacking third parties can be considered low-risk.

On all platforms, most applications request access to the Internet and this permission is easily granted by users. On iOS, access to the Internet is not even protected and the user cannot prevent applications from communicating over the Internet unless he turns off all Internet traffic for the device. As no platform supports the limitation of Internet traffic to a particular host, network or type, attacks to devices thereon could be simple. However, in order to harm a device on the network, there must be a vulnerability that the victim exhibits. This lowers the success rates of these attacks dramatically.

Malicious applications like to target the permission model of the devices. We have seen that Android’s model is compromised by rooting the device and installing applications with any permission they like. For some permissions, this is not even necessary, because they can be gained otherwise. The iOS permission model is by far too permissive and poses a risk by leaving many parts of the system unprotected. Microsoft has found a model that makes a good approach in protecting the system, even though there are permissions that would deserve finer granularity.
Browsers encounter malicious or manipulated websites on a daily basis. Despite the browsers are all running in their own sandbox, attacks attempting a drive-by infection with malware are very likely and may cause severe damage, depending on the type of malware. Other traffic originating from networks seem to be less dangerous for iOS and Windows Phone, because neither of the two allows real multitasking and hence, there cannot be a service listening for incoming connections, unless the application is currently running.

The risk of eavesdroppers and attackers intercepting radio communication is always high in mobile communication devices. While Android and iOS support numerous encryption standards and protocols, Windows Phone 7 does not allow a secure connection to a network via VPN, There is severe danger to the confidentiality of the communication by adversaries on the same network.

While a denial-of-service attack is of course easily possible on Android with root access, the other two platforms will not allow applications that manage to restart the device on their shops. Even if such an application finds its way to the live system, it is only able to restart the phone at most once, because no application is allowed to run at system startup and without explicit user interaction. Android, in turn, does not even require root access or a special permission to start an application at boot time. Android applications that exploit a vulnerability to restart devices can create a reboot loop and hence a denial-of-service attack. Finally, Android has proven to be a high-risk system, because it is possible to root the device once an appropriate exploit binary is available. In the previous sections we have indicated how it is possible to transfer binary files to an Android device without requesting a single permission. The almost-constant rooting danger implies a high security risk for most system components. The other two platforms have reached risk evaluations that coincide in large parts. All platforms are prone to attacks targeting the user privacy and must be aware of the risk of drive-by infections from malicious websites. Both iOS and Windows Phone employ reasonable protection from denial-of-service attacks and appropriately shield the address book entries from unauthorized access.

**4.1.2.5 PCs with 3G/4G Module**

Considering PCs with 3G/4G modules is manifold. The operating landscape is very diverse and different security concepts for the many platforms have been extensively studied to date. Applications can be installed from many vendors and there is no regulation involved how to trust the application or the application programmer. Discussing the implications of PC operating system flaws and PC application flaws is not within the scope of this document. From a mobile operator’s point of view, PCs with mobile network connections are in principle devices that have more computing resources than the regular phones and will most likely only be creating data traffic on the network.

From a control point of view, operators will most likely be able to influence only the security of the baseband part used in relevant hardware module.

**4.1.3 Threat Analysis**

This following section gives a concise threat analysis based on the previous section on terminal security. Different from the previous version of this document, we decided not to apply the threat categories introduced in section 3.2. While these threats are very useful in assessing the risk of network elements, they turned out not to be fully suitable to capture the vast complexity of the group of themes related to terminals. Therefore, we rather offer an informed and concise view on the topic with regards to the manifold challenges, e.g., the
balkanization of Android versions deployed on phones of different vendors, or the inconsistent security of the software distribution mechanisms of Google, Apple, and Microsoft.

All present baseband stacks in mobile terminals are backward compatible to GSM. As a result, the conceptual flaws in GSM are still the major security related problem for 3G/4G basebands. Furthermore, only a few different software implementations exists, which are very complex and lack a thorough and independent analysis. Our practical tests regarding the fuzzing of baseband implementations have shown that many of today's devices are highly vulnerable in terms of buffer overflow attacks. Common hardening or integrity protection mechanisms are not present.

The security of the application part of mobile terminals largely depends on the type of the devices. Machine-to-Machine applications will become more and more important in the future. Depending on the machine type and functionality, very different values for likelihood of attacks, vulnerability of the device and impact of a successful attack are possible. While feature phones are still most widely used and sold, a more attractive target for an attacker seems to be a smart phone.

The security of the smart phone operating system largely depends on the composition of the individual security features as a whole. In the previous section it has become obvious that of Google's Android, Apple's iOS and Microsoft's Windows Phone 7, each offer a different level of security.

Jailbreaking for instance, is possible on Android, iOS and Windows Phone 7, however, especially on Android, this threat is more serious. The can be constituted to the openness of Android itself, whereas the other two platforms employ more strict safeguards to prevent jailbreaking. Non-native 3rd party applications, i.e., those that can be downloaded from markets or obtained from 3rd party markets, have also shown to be able to jailbreak devices just by being run. These rather convenient applications - at least for developers and users willing jailbreaking their devices - are proof that also malicious applications are capable to jailbreak devices silently and fully compromise the device. Recent Android malware has proven to employ such techniques. As such, this kind of malware presents a serious threat.

With respect to threatening the privacy of the end users and their associates, all of the discussed platforms have rather serious issues. Obtaining and extracting user data from the device and transferring them to a remote location is easy, very likely, and has been proven to be done by malware writers to date. Also, as the threat of malware jailbreaking the devices is rather significant, eavesdropping becomes a trivial task as the devices are fully compromised afterwards.

The threat of malware infected devices threatening other device by spreading and infection is rather ambiguous. Even though there exist cases in which devices infect other devices, the most reliable attack vector still to remains to be official software markets, 3rd party markets, and some social engineering. More tightly controlled software distribution path, and the introduction of additional hurdles to installed non-official software may somewhat limited this effect.

Attacking single components of the device security models, e.g., the permission model or the (pseudo) sandboxing mechanisms, is a serious threat that is commonly exploited by malware. The most convenient way to attack an Android phone is to distribute applications with excessive permissions which the users is still very likely to accept. However, this user behavior might change over time as user become more educated, also be personal experience, e.g., data loss. But, as the pervious sections have shown, fully compromising an Android device can also be achieved without excessive permissions.
Not only malicious applications residing on the terminal are a threat – mobile terminals are also endangered by all kinds of threats arising from being connected to the Internet. Even though the scope of this document is not to address the abundance of threats that exists around the Internet, it should be mentioned that the discusses smart phone operating systems are threatened as well. For instance, popular browser libraries, e.g., WebKit, are commonly reused by e.g., Android. Existent exploits that for these libraries have proven to be effective against their smart phone counterparts as well. As such, the manifold threat of browser exploits can largely affect smart phones as well.

4.2 Access Network

Access networks comprise the radio base stations deployed all over the range covered by a mobile network and support the main characteristic of mobile networks, i.e. mobile communication without a wire line. Access networks also comprise so called controllers, i.e. nodes located between the core network and the base stations. Typically, each controller connects many base stations to the core network.

For packet switched services, 2G/3G radio access networks can be connected to an evolved packet core via an SGSN (see 4.3.3.1 for a description). At the same time, such radio access networks may be connected to the circuit switched domain of a 2G/3G network (see 4.3.4 for a description) to provide circuit switched services, in particular voice. It is assumed that such access networks will stay in use for a long time and thus will be part of future 4G networks. Therefore, 2G/3G access networks will be briefly discussed here. Following this, the components of 4G access networks will be assessed in detail.

4.2.1 2G Access Networks

The 2G radio access network is called the base station system and consists of base transceiver stations (BTS) which are connected to base station controllers (BSCs). The Figure below illustrates the base station system and the interfaces between the components in the base station system.

![Diagram of the BSS](image)

*Figure 19: Reference Architecture of the BSS*

2G access networks used to rely on TDM, ATM or Frame Relay for the traffic transport between the network elements. However, there is a trend to use IP networks also. Mostly, such IP networks would be private IP networks and not be part of the Internet.
For 2G circuit switched services, in particular voice, user and control plane traffic are only encrypted between the mobile station and the base transceiver station. Different encryption mechanisms are specified, but all of these are rather weak according to today’s standards and can be cracked with inexpensive equipment in real time.

For packet switched 2G services, encryption using one of a set of algorithms is standardized between mobile stations and SGSN. Two of these algorithms have been kept secret up to now, but some reverse engineering activities of the hacking community are ongoing. Even if these algorithms are not yet practically broken as of today, one shouldn’t count on their strength against future attacks. Another GSM encryption algorithm is public and did not show relevant weaknesses until today. As encryption is performed between mobile stations and SGSN, RAN elements play no role in protecting the 2G packet switched traffic.

Integrity protection on the radio interface is not supported in GSM. In addition, as authentication is only one-sided (MS to network, but not network to MS) and encryption is not mandatory, an attacker can easily impersonate a BTS to an MS. In this case, if the MS accepts usage of weak or even null encryption as the fake BTS would propose, the attacker will be able to intercept and modify all communication.

These GSM weaknesses and how they can be exploited are described in detail in chapter 4.1.1.4. It should be noted that as of today, the GSM weaknesses are not exploited to an extend that it impacts the usability or the business models of GSM networks – in fact, such exploits seem to be very rare.

4.2.1.1 Base Station Controller (BSC)

In the base station controller, no security-relevant functions are implemented. As a consequence no sensitive data like keying material is stored in a BSC. All circuit switched traffic is accessible in the BSC in the clear. Therefore a BSC and its (unprotected) interfaces are in principle attractive targets for eavesdropping and traffic manipulation attacks, assuming the attacker has somehow access to the BSC interfaces. This is rather not the case, as long as primarily TDM and ATM are used here, but may change with the adoption of IP networks as transport networks. Note that the encryption keys used between BTS and MS are also provided to the BTS over the interfaces of the BSC.

4.2.1.2 Base Transceiver Station (BTS)

For circuit switched services, the BTS in GSM is responsible for the encryption of user plane and control plane traffic. On the network side, no encryption is used, so the same considerations hold as for the BSC. Moreover, as discussed before, a fake BTS may be established by an attacker, using a strong radio signal, so the mobile stations will rather attach to the fake BTS than to one of the genuine ones.

4.2.2 3G Access Networks

The UMTS or 3G radio access network comprises the base stations, which are called NodeB (NB) here, and the radio network controllers (RNCs) over which the base stations are connected to the backbone network.
3G access networks may use TDM or ATM for the traffic transport, but there is also the option to use IP, and IP is gaining more and more importance here. However, IP based 3G networks will be private IP networks mostly and not be part of the Internet.

4.2.2.1 Radio Network Controller (RNC)

An RNC has mainly three types of interfaces: an interface to the NodeB, an interface to the MSC for voice traffic and an interface to the SGSN for IP traffic.

In 3G networks, user plane and control plane traffic is encrypted and control plane traffic is also integrity protected. This security holds between mobile station and RNC, i.e. the RNC is the termination point for encryption and integrity protection on the network side. Note that in contrast to NBs, RNCs are “big computers” and are mostly deployed at physically protected sites, like core network elements.

If based on IP rather than on ATM, the connection between an RNC and the MSC / SGSN can be protected with IPsec, according to the 3GPP concept for Network Domain Security on the IP layer (NDS/IP, see [3GPP_TS33210]). Correct implementation and usage of the IPsec framework results in full protection of the communication. Clearly, faults in the IPsec implementation, compromise of the devices implementing the tunnel endpoints or disclosure of secret credentials used for peer authentication may still defeat IPsec protection.

If IPsec is not applied on this interface, the interface is vulnerable to key disclosure as the encryption and integrity protection keys are provided to the RNC via these interfaces. The consequence is that user traffic could easily be eavesdropped on and manipulated, if the attacker could gain access to this interface. This is rarely the case, and there have been no reports of security breaches of this kind in practice. Gaining access to an RNCs memory directly and thereby to the keys currently stored and used by the RNC would have similar consequences. RNCs are however considered to be less easy to access than NodeBs as they can be more easily physically protected.

4.2.2.2 NodeB

No security-relevant functionality is implemented in the NodeB. As a consequence, a NodeB is generally not a very attractive attack goal for any other attacks than denial of service attacks that try to render connectivity in a particular geographical area impossible.

4.2.2.3 Home NodeB (HNB)

The HNB, often called femto cell, is a very small NodeB used to provide 3G services in a restricted area, e.g. in a home or inside enterprise premises. HNBs are expected to be
connected to the mobile core network via the Internet, e.g. using DSL on the first mile and infrastructures of Internet service providers up to an interconnection point of the mobile network. The HNB comprises RNC functions and thus critical security functions, for example it terminates the radio interface security. Moreover, it is heavily exposed to physical tampering.

The 3G HNB is endangered in a very similar way as the 4G HeNB. 3GPP has specified a specific security architecture for HNB and HeNB ([3GPP_Ts33320]). We describe this security architecture and assess the risks of the HeNB in section 4.2.3.3. These assessments hold in the same way for the HNB, unless otherwise stated.

4.2.2.4 Other Components for the Support of HNBs

The 3GPP architecture for HNBs specifies that they connect to the network via a Security Gateway (SeGW) and a HNB-Gateway. Moreover, there may be a dedicated HNB Management System HMS, and an AAA-Server performing certain authentication and access control functions. These components are very similar, if not identical, to the respective components for a 4G access network, which are assessed in sections 4.2.3.4 and 4.2.3.5.

Note however that there is one difference between the HNB-GW and the HeNB-GW: In contrast to the HeNB-GW, the HNB-GW, in certain scenarios, will perform access control, i.e. check whether a specific UE is allowed to use a specific HNB. (In other scenarios, this check will be done in the core network.) Therefore one could argue that a slightly higher risk might be associated with an HNB-GW than with a HeNB-GW.

4.2.3 4G Access Networks

A 4G access network, called Evolved UTRAN (E-UTRAN), mainly consist of base stations called eNBs (E-UTRAN NodeBs). There are no dedicated radio network controllers. Important, security critical functions like termination of the radio interface encryption are therefore located in the eNBs. As these eNBs are moreover assumed to be increasingly deployed at places with poor physical protection, it is obvious that they have the potential to pose a considerable risk to mobile networks. Consequently, 3GPP has specified a number of additional security features in the EPS security architecture, and has explicitly specified a set of security requirements to be fulfilled by the eNB.

As in 3G networks, there is also the option to deploy so called Home eNBs (HeNBs), i.e. very small eNBs that are deployed in homes or enterprise premises and connected to the operator network via the Internet. Usage of HeNBs requires deployment of a Security Gateway (SeGW), an optional HeNB-GW, a HeNB Management System (HeMS), and optionally a AAA server

4.2.3.1 E-UTRAN Node B (eNB)

4.2.3.1.1 Interfaces and Protocols

The base stations in a 3GPP 4G network are called eNBs. As illustrated in the figure below, eNBs have four interfaces: an interface to the mobile (called Uu), an interface each to the core network elements SAE-GW and MME (called S1-U and S1-MME, respectively), and an interface that allows for direct connections between different eNBs (called X2). Note that the mobile is often called User Equipment (UE) in 3GPP specifications.
Each eNB supports a protocol stack on the interface Uu. This protocol stack is illustrated for user plane as well as control plane traffic in the following figure.

Both user plane (i.e. IP traffic) as well as control plane traffic are carried on top of the Radio Link Control protocol (RLC) and the Packet Data Control Protocol (PDCP). The control plane uses the Radio Resource Control protocol (RRC). In addition, there is the so called Non Access Stratum (NAS) signaling between the UE and core network elements. Note that user and control plane traffic between UE and eNB are encrypted and control plane traffic is also integrity protected on the PDCP layer.

The interface S1-MME is used for control plane traffic, while the interface S1-U is used to transport user traffic via the SAE-GW towards the Internet. In both cases, the S1 interface is IP-based as illustrated in the following figures.
The logical point-to-point interface X2 between eNBs is mainly used to support fast handovers between eNBs. This is achieved by allowing two eNBs connected via an X2 interface to exchange signaling messages to prepare a handover of a UE between them. Another handover type (the S1-handover) makes use of the S1 interface and involves the MME in the handover preparation. This type of handover procedure is more time intensive due to the higher signaling overhead via the MME. The protocol stack supported on the X2 interface between two eNBs is IP-based and is identical to the protocol stack on the S1 interface illustrated above, except that the S1-AP is replaced by the X2 application protocol X2-AP. We therefore refrain from providing a separate figure illustrating the X2 protocol stack.

4.2.3.1.1 S1 Application Part (S1-AP)

The S1 interface describes the interface of an eNodeB (as of [3GPP_TS36410]), towards the core network, via which it is connected to the MME or SAE-GW. To achieve the security requirement separation, the S1 interface is (virtually) divided into two parts, providing the user plane (S1-U) and the control plane (S1-MME). The S1-MME interface supports the S1 application part (S1-AP) and supports functions like evolved radio access bearer (E-RAB) management, context and configuration transfers, NAS signaling transport and more, which can be found at [3GPP_TS36413]. Furthermore, S1-AP supports trace and location reporting functions.

Before the actual communication between an eNodeB and an MME takes place, an S1AP connection has to be set up. This is done by a S1 Setup Request, initiated by the eNodeB and followed by a S1 Setup Response message from the MME. After setting up S1-AP connection further information can be sent, for example a configuration request, which is followed by NAS transport messages for setting up a UE session. An example, including the full connection establishment and detach procedures is shown below.
In the following, the basic packet structure of S1-AP (and equivalent X2-AP) is explained. Both, S1-AP and X2-AP packet header, consist of some general fields, like message type, packet length and count of items, but also of QoS parameters like criticality. Furthermore, the following table describes the setup of the item fields, which are used to transport the contained information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message Type</td>
<td>The message type defines the content and buildup of the packet. The value is created out of two numbers:</td>
</tr>
<tr>
<td></td>
<td>• Type of Message, which may contain value 0x00 for initiating message or 0x20, which means a successful outcome.</td>
</tr>
<tr>
<td></td>
<td>• Procedure Code, defines the further content of the message, specified in</td>
</tr>
</tbody>
</table>
The criticality field is a QoS parameter, which may be “ignore” (value 0x40) or “reject” (value 0x00). No more values are defined yet in [3GPP_TS36413].

Length
Defines the length of the packet.

Count of Items
Defines the number of container items that the message contains. The container items depend on the message type.

Item 1
An item is a container of information, delivered to the target. Each item again has an own header, containing a Protocol Information Element ID, a Criticality field and the length of the container, followed by some information values.

Item 2
...

4.2.3.1.1.1 Non-access Stratum (NAS)

Non-access stratum (NAS) denotes the protocols supporting signaling and traffic between core network and UE. In 4G networks, NAS is responsible for signaling between MME and UE, including mobility management and session management procedures and is transported by S1-AP via the S1-MME interface (between an MME and an eNodeB). 3GPP defines two new NAS protocols which have many similarities to NAS used in 2G/3G networks [3GPP_TS24301]:

- EPS Mobility Management (EMM), supporting UE mobility, security and signaling connection management.
- EPS Session Management (ESM), supporting activation, deactivation, or modification of EPS bearers.

EPS Mobility Management (EMM) procedures are used for control of mobility, if the UE is using the E-UTRAN, and provides mechanisms for tracking and authenticating the UE towards the core network (especially the MME). That means requesting and delivery of identification and authentication data from UE, but also the delivery of UE capability information to the network. Another feature is the information of UE about specific services active in the network. EPS Session Management (ESM) procedures are used to handle EPS bearer contexts and to control the user plane bearers connecting the UE with a PDN. Therefore, it manages the bearers by establishment, modification and deactivation functions. In regular context, the procedures are initiated by MME, but the UE may also request the network to modify the bearer resource.

EPS Mobility Management (EMM)

EMM messages consists of a security header, a protocol discriminator (value 0111 for EPS mobility management messages), a message type and the related information elements. If
the EMM message is security protected, the protocol discriminator is followed by the Message authentication code and a sequence number. This feature is optional and can be used for ciphering and integrity protection [3GPP_TS24301].

The security header IE includes control information, related to the security protection of the NAS message [3GPP_TS24301] and may have values for integrity protection and ciphering.

**EPS Session Management (ESM)**

The format of ESM messages is similar to EMM, but instead of the security header, the ESM NAS message contains the EPS bearer identity, followed by the protocol discriminator (value 0010 for EPS session management messages), a procedure transaction identity and the message type. The EPS bearer identity is used to identify a message flow, defined in [3GPP-TS24007], and the procedure transaction identity which defines a unique number to distinguish up to 256 different message flows for a given protocol discriminator (as defined in [3GPP-TS24007]). The protocol discriminator and the message type are used the same way as in EMM messages.

Like the EMM messages, the ESM messages may be security protected in ciphering and integrity, using the optional message authentication code and sequence number.

### 4.2.3.1.1.2 X2 Application Part (X2-AP)

The X2 Application Protocol (X2AP) is defined in [3GPP-TS36423] which is interconnecting two eNodeBs within the Evolved Universal Terrestrial Radio Access Network (E-UTRAN) architecture. The protocol was specifically developed for providing signaling information across the X2 interface [3GPP-TS36420].

The X2 interface is the interface between eNodeBs components that are connected to each other. It can be used for data exchange, but also for exchanging management information. The functions of the X2 interface are defined in [3GPP-TS36420] and some of them are listed below:

- It's supposed to provide mobility management functions, e.g. for a handover of a certain UE from one eNodeB to another eNodeB, but also the handover cancellation or UE context release of the source eNodeB. Another function is the control of user plane transport bearers between source eNodeB and target eNodeB.
- It's meant to provide the exchange of overload and traffic load information of two eNodeBs.
- To provide the exchange of neighbor information to handle inter-cell interferences.
- It provides the ability to manage signaling associations between two eNodeBs. An example could be sending reset messages or the error indication in case of an error condition.

### 4.2.3.1.1.3 GPRS Tunneling Protocol on User Plane (GTP-U)

GTPv1-U, specified in [3GPP-TS29281], is the part of the GTP protocol responsible for carrying encapsulated user data and signaling messages between two tunnel endpoints.
GTPv2-U is similar to GTPv1-U, but contains some enhancements like support of IPv6 and therefore will not be explained in more detail here.

The endpoints of GTP-U are defined by the Tunnel Endpoint Identifier (TEID), which in normal communication of 4G networks will be the eNodeB and the PDN-GW. For tunnel establishment, Packet Data Protocol (PDP) context information is necessary. The PDP context contains information about the subscriber's IP address, subscriber's IMSI and the TEID of the used gateway (3G:SGSN and GGSN; 4G:SAE-GW) and is delivered from eNodeB via the S1-MME interface using the S1AP protocol from eNodeB to MME. In summary, the PDP context contains the subscriber's session information and may also contain information about QoS or other additional features. If the connection setup between UE and MME finishes successful, the MME starts a tunnel activation process by sending the GTP-C (GPRS Tunneling Protocol for Control Plane) message "PDP Context Activation" to the Serving GW (which is one part of the SAE-GW). By receiving this message, the Serving GW builds up the tunnel with tunnel endpoints eNodeB and the PDN-GW (which is the other part of the SAE-GW).

To inform the PDN-GW about the PDP context, another PDP context message is sent to the PDN-GW via the S5 interface. The protocol stack is shown in figure 23. It is based on the transmission protocol UDP and uses port number 2152.

Two different kinds of messages are defined in [3GPP-TS29281]: the transmission of PDU data and the transmission of signaling messages. Signaling messages are path management or tunnel management functions.

The GTP-U supported message types are shown below. GTP-U also provides the path management/tunnel management functions like Echo Request/Response and Error Indication functions. The End Marker message shall be sent to the specific tunnel endpoint for each GTP-U tunnel and indicates the end of the payload stream for a given tunnel [3GPP-TS29281].

<table>
<thead>
<tr>
<th>Message Type Value (Decimal)</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Echo Request</td>
</tr>
<tr>
<td>2</td>
<td>Echo Response</td>
</tr>
<tr>
<td>26</td>
<td>Error Indication</td>
</tr>
<tr>
<td>31</td>
<td>Supported Extension Headers Notification</td>
</tr>
<tr>
<td>254</td>
<td>End Marker</td>
</tr>
<tr>
<td>255</td>
<td>G-PDU</td>
</tr>
</tbody>
</table>

An important entry in the GTP-U (and also GTP-C) header is the TEID, which defines the target endpoint within a tunnel. As example, in direction from eNodeB to PDN-GW, the TEID includes the identifier for the PDN-GW, and in direction from PDN-GW to eNodeB, the TEID-U includes the identifier for the eNodeB. The TEID always will be unique. Including random numbers rather than predictable numbers into the TEID may provide some protection against attackers that have access to an interface where GTP is used and try to guess a TEID as part of their attack.
4.2.3.1.2 3GPP-specified eNB Security Requirements/Features

[3GPP_TS 33401] specifies the following security requirements/features for the eNB:

- There must be a secure environment inside the eNB to store sensitive data like keys, execute sensitive functions like encryption and decryption, and secure the boot process. For the secure environment integrity and confidentiality must be ensured. Any unauthorized access to the secure environment must be excluded.
- On the radio interface, the eNB performs ciphering for user and control plane traffic and integrity protection for control plane traffic. The eNB receives the required key material from the core network.
- Mutually authenticated security associations are established by the eNB for the S1 interface towards the core network. For this, IKEv2 and IPsec-ESP are used to provide confidentiality and integrity for user and control plane traffic. Optional certificate enrolment according [3GPP_TS33310] may be done: Based on a private key and certificate provisioned by the equipment manufacturer, this procedure allows to provide the eNB with an operator signed certificate that can be used for the IKEv2 peer authentication towards core network components. It should be noted that using IKEv2/IPsecESP is recommended, but not mandatory; it may be replaced by other mechanisms providing equivalent security.
- Deciphering/ciphering of user and control plane traffic must be done inside the secure environment, so transit traffic does not exist in the clear outside the secure environment on the eNB.
- The X2 interface towards other eNBs must be protected as specified for the S1 interface.
- O&M traffic must be encrypted and integrity protected between the eNB and the O&M systems. Like for user and control plane traffic, IKEv2/IPsec-ESP security associations are specified to be used between the eNB and some entity within the trusted operator network (core or O&M network).
- Only authorized data and software may exist on the eNB; all data and SW changes must be authorized, all software transfer towards the eNB must be integrity and confidentiality protected.
- The eNB boot process must be secured using the secure environment. For example, hashes of code segments to be loaded during boot may be compared with values stored in the secure environment to prevent loading of compromised code.

4.2.3.1.3 Practical Security Testing

While the implementation of the above security requirements and features is supposed to lead to a very low vulnerability of eNBs, it must be assumed that in real life environments equipment may fail to implement every function correctly and thus be vulnerable against attacks. To corroborate this assumption, some practical testing was performed, as described in the following.

4.2.3.1.3.1 Protocol Fuzzing for S1-AP and X2-AP

To test the implementation and robustness of the network components, fuzzing was used. The method of fuzzing is already described in section 4.1.1.7, and basically aims at finding implementation bugs using malformed/semi-malformed protocol message injection. In this
case, a custom fuzzing tool, called *dizzy*\(^{33}\), was used to modify the communication data in an active S1-AP/X2-AP session.

*Dizzy* uses a common S1-AP/X2-AP packet (like described above) and inserts all possible variants/entries into a specific message. This means for example, that all possible values will be inserted into the packet header and sent to a specific endpoint. This will test the implementation of the protocol stack and the handling of the delivered information. This includes QoS parameters and content of the packet, but also main parameters like message type or packet length.

The tests showed indications of implementation errors in different implementations. Such errors can lead to relevant risks for the availability of nodes, like eNBs (e.g. crashing may be possible).

### 4.2.3.1.3.2 Man-in-the-Middle attacks

It should be kept in mind, that there is no native authentication available in GTP, S1-AP and X2-AP. Rather, security relies on usage of IPsec wherever the access to these interfaces is not protected otherwise, e.g. by physical security measures. Therefore, if IPsec is not used and an attacker gets access to backhaul network, she might be able to simulate/spoof any component in network and do some kind of man-in-the-middle attacks.

Based on the knowledge of the session setup procedure (illustrated in figure 27) some man-in-the-middle attacks were successfully simulated, which confirms that authentication and integrity checks are really necessary.

Because there is no authentication on S1-AP/X2-AP, it is quite easy to get into an active S1-AP session and insert malicious S1-AP/X2-AP messages into the communication flow. Because both protocols depend on SCTP, it was necessary to get into the SCTP stream (and possibly IPsec) first. If an attacker gets into the stream after authentication of the UE (means after NAS uplink complete message), it is possible to send almost all S1-AP messages to the related devices.

Some examples of messages, which may cause damage to the network in case an intruder manages to get into an active S1-AP session are shown below.

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uplink NAS Transport</td>
<td>With Uplink NAS Transport messages, an attacker may intrude into the NAS signaling procedure (described above) and may send malicious NAS messages to the MME. However, per standard, NAS signaling must be end-to-end integrity protected between UE and MME using a dedicated key, so the MME will detect and discard faked messages inserted this way.</td>
</tr>
<tr>
<td>Reset</td>
<td>The reset message can be sent from the MME to an eNodeB, but can also be sent the other way round and simply does what the</td>
</tr>
</tbody>
</table>

\(^{33}\) [www.asmonia.de](http://www.asmonia.de)
name implies: resetting the S1 interface. This procedure does not affect the application level, but still may be a valid attack vector to cause a DoS (Denial of Service) condition, because all allocated resources on S1 and LTE-Uu related to the UE association are released.

**Error Indication**

An Error Indication message can be sent either from eNodeB or from MME and is sent due to report of errors. In management, it is used to create logs and detect errors to prevent upcoming failures. An attacker might use this message to flood the MME with failures of the network.

**S1 Setup Request**

The S1 Setup procedure is needed to set up a S1AP connection for exchange of application level data between eNodeB and MME (S1 interface) and already is described above. For an attacker, the S1 Setup procedure is necessary to initiate the other messages discussed here or simply to set up an own UE context.

**Handover Required**

Indicates that there is a handover required in current cell. May allocate resources on MME and initiate a Handover Request.

---

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>E-RAB Release</td>
<td>The E-RAB Release Command contains a List item, where E-RABs to be released are listed. If an attacker gets this information, he might be able to initiate an E-RAB Release procedure (or by sending all possible combinations, i.e. fuzzing).</td>
</tr>
<tr>
<td>UE Context Release</td>
<td>The S1AP UE Context Release Function is responsible to manage the release of the UE specific context in eNodeB and MME. If an attacker is able to send this message to one of these components, it should be possible to release the context information and therefore force and abort the connection of an UE.</td>
</tr>
<tr>
<td>Handover Request</td>
<td>The Handover Request message prepares a handover to a target eNodeB. This is not actually initiating a handover, which must be</td>
</tr>
</tbody>
</table>
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

initiated by the eNodeB. This message allocates resources and lets an eNodeB wait for a coming handover which may be injected/simulated by an attacker.

Downlink NAS Transport

The Downlink NAS Transport messages are the equivalent to Uplink NAS Transport messages, but in direction from MME to eNodeB.

Overload Start

The Overload Start message informs the eNodeB that there is an overload situation and admits eNodeB to initiate possible countermeasures. An attacker may use this message to grant a specific connection a higher/lower priority/bit rate towards the S1 interface or simply reject specific connections.

MME Configuration Update

The MME Configuration Update is the message, which must be send before MME Configuration Transfer. It is sent to the eNodeB to update configuration data (e.g. PLMN, ...). An attacker may use this message in addition to the MME Configuration Transfer to change the PLMN identity or similar configuration data.

Figure 26: S1-AP message examples in direction MME to eNodeB

The X2-AP protocol supports similar functions to S1-AP and therefore will not be described in more detail here.

4.2.3.1.3.3 Attacks on GTP-U

Because there is also no authentication and encryption supported in GTP-U messages themselves, several attacks to GTP-U might be possible. As described above, an attacker might be able to craft GTP-U messages and send them to the network to trigger answer messages and thus get information (e.g. about network topology), or just send malicious PDUs to the network. This may involve guessing a valid TEID (hijacking a TEID), unless the endpoints use non predictable TEIDs, which is not always the case.

To perform practical testing of these issues, a fuzzing script for GTP-U was written, using a hijacked TEID, and malicious messages were send to the network. This method tests the implementation of GTP-U speakers and showed, that in contrast to S1-AP and X2-AP (which are newly developed protocols and less field proven), most of the GTP-U implementations seemed to be robust against malicious GTP-U messages.
4.2.3.1.4 Assessment

T1 Flooding an interface

T1a Flooding the radio interface

Flooding the radio interface on any of the protocol layers can lead to a DoS attack on legitimate UEs which cannot be served any more as all resources on the eNB are consumed by the flooding device. As the radio interface is accessible for any user equipment and as the radio resource establishment procedures are naturally unprotected, malicious radio resource requests will be hard to detect on an eNB. Manipulating UEs into issuing fake requests in large amounts could be achieved with the help of mobile malware. It may even be possible to construct a malicious mobile terminal that allows to produce a flood sufficiently large to flood an eNB. (Note that there are open source software projects ongoing aiming at providing the baseband protocol stacks of mobile terminals—such software could be abused to construct malicious terminals.)

T1b Flooding the backhauling interfaces

The backhauling interfaces (S1, X2, O&M) are IP-based interfaces between an eNB and other network elements such as MMEs, SAE-GWs, and other eNBs. While each eNB may have interfaces to more than one MME, SAE-GW and more than one other eNB, the number of network components to which an eNB has an interface will be rather small and static. An eNB will therefore typically know from which IP addresses traffic should be accepted and from which not. If IPsec is applied on all interfaces, than fake traffic can easily be detected and be discarded, as this traffic will fail the integrity protection check. We assume here that the eNB can process the IPsec integrity check in wire speed—if this is not the case, the eNB would be vulnerable by a flood of rogue IPsec packets. If, however, IPsec is not applied, then flooding the backhauling interfaces with traffic from a spoofed valid IP address could be possible.

T2 Crashing a network element via a protocol or application implementation flaw

An eNB supports many different protocols many of which have been newly designed for E-UTRAN. The probability that an implementation of one of these protocols will exhibit a serious implementation flaw is therefore rather high. Also, as mentioned above, for attacks via the radio interface, open source software may help to construct malicious terminals that can be used to look for such flaws and exploit them. An attacker with some form of access to the backhauling link may also attack backhauling interfaces, but the attack surface will be smaller here, e.g. only IKE/IPsec may be supported here but no application layer protocols.

As the coverage area of a single eNB is not that large, a crash of a single eNB has a limited DoS effect on some legitimate users only. However, as it can be expected that the same implementation flaw will be included in neighboring eNBs of the same mobile network operator, an attacker may succeed in exploiting the implementation flaw across a larger coverage area.

34 See http://bb.osmocom.org/trac/
T3  Eavesdropping

T3a Eavesdropping on the radio interface
According to [3GPP_TS33401] control plane traffic as well as user plane traffic on the Uu (radio) interface can be encrypted on the PDCP layer with SNOW 3G or AES. While the use of encryption is recommended in the technical specification, it is not mandatory. In particular this opens up to the threat of encryption being turned off maliciously e.g. by an inside attacker on the network operator side or with the help of malicious code on the eNB.

T3b Eavesdropping on the backhauling interfaces
If the backhauling link is not encrypted, then in particular user security context information such as part of the currently used keying material will be revealed to an eavesdropper. Also, the user plane traffic would be available to eavesdroppers in the clear. Note that NAS control traffic will not necessarily be available in the clear, as the technical specification supports optional independent encryption of NAS traffic. However, as a rule we assume that network operators will make use of IPsec to protect the backhauling interfaces, or will provide equivalent protection, such that eavesdropping on these interfaces will not easily be possible.

T3.1, T3.2 Eavesdropping of control plane traffic versus user plane traffic
The impact of eavesdropping depends on what traffic is affected. Eavesdropping control plane traffic is more critical for the operator, as it may reveal information to the attacker that allows him to mount further attacks. The impact of eavesdropping user traffic is estimated following the conclusions made in section 3.3.

T4  Unauthorized access to sensitive data on a network element via leakage
The most sensitive data stored on an eNB is the keying material shared between eNB and currently active UEs connected with it as well as the keying material with which the IPsec connections to the core network elements or other eNBs are protected. Access to this keying material enables a variety of quite serious attacks such as impersonating the eNB in other locations, manipulating user traffic, faking user traffic, and flooding SAE-GWs or MMEs. As eNBs are rather accessible, in particular via the radio interface but also via the other interfaces, finding vulnerabilities in the implementation of the protocols and applications running on them may be possible.. As described above, the technical specification [3GPP_TS33401] states a variety of requirements for providing a secure environment for key handling on an eNB. However, it is unclear how these are or will be achieved in actual products.

T5  Traffic modification

T5a Traffic modification on the radio interface
On the radio interface integrity protection of most of the control plane traffic is mandatory. Therefore control plane traffic modification should not be possible on this interface (unless an attacker gets into possession of the currently used keys as described in T4 or some weakness in the implementation of the integrity protection mechanism can be exploited).
T5a Traffic modification on the backhauling link
As for T3 we assume that network operators protect the backhauling link as required by 3GPP standards, either using IPsec (with integrity protection) or other, equivalent protection. As a consequence we assume that traffic modification on the S1 and X2 interfaces is hardly possible.

T5.1, T5.2 Modification of control plane traffic versus user plane traffic
The impact of traffic modification depends on what traffic is modified. Modifying control plane traffic is highly critical for the operator, as it may lead to malfunctioning or loss of availability of the network. Modifying user traffic is estimated following the conclusions made in section 3.3.

T6 Data modification on a network element
Data modification on an eNB could include the modification of configuration data such as security algorithms supported. Data modification could also include modification of UE context information such as keys, algorithms and identifiers. As mentioned above, the technical specification [3GPP_TS33401] includes requirements to provide a secure environment for security relevant data and procedures on the eNB. However, it is unclear how these requirements are or will be met in future. Obviously, modification of configuration data can lead to undesired behavior of the eNB.

T7 Compromise of a network element via a protocol or application implement. flaw
Compromising the eNB without crashing it (as in T2) should be more difficult than just crashing it. However, the impact of such a compromise can be much more serious than in the case of crashing as a compromise gives attackers a lot of possibilities for further attacks on the network and on all mobile terminals using the compromised eNB.

T8 Compromise of a network element via a management interface
Although this is not a standardized feature, all eNBs will likely support a management interface that allows for remote or local configuration. Trying to find vulnerabilities in a management interface is a regular procedure of an attacker and will rather often be possible. The impact of a compromise is clearly serious (see T7).

T9 Malicious insider
A malicious insider may gain access to an eNB during installation or maintenance or via a management interface. A malicious insider is very hard to exclude and protect against. However, we assume that a malicious insider is somewhat more likely to target core network elements and will rarely attack single eNBs.

T10 Theft of service
The eNB mainly serves as a relaying entity, but also as a control anchor for handover of UE to other eNBs. Charging functionalities are not implemented by the eNB, therefore it is very unlikely to be attacked in order to commit theft of service. An attacker that can compromise an eNB may be able to commit theft of service by impersonating other subscribers, or by somehow sending traffic that evades charging mechanisms. We consider a compromise of an eNB not very likely, though (see assessment of T7 to T8).
The following table summarizes our results:

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vul. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1a Flooding the radio interface</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>30</td>
</tr>
<tr>
<td>T1b Flooding the backhaul interface</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>2 - 3</td>
<td>3</td>
<td>2</td>
<td>12 - 18</td>
</tr>
<tr>
<td>T3.1a Eavesdropping radio interface (control-plane)</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>T3.1b Eavesdropping backhaul interface (control-plane)</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>T3.2a Eavesdropping radio interface (user-plane)</td>
<td>4</td>
<td>2</td>
<td>1 - 3</td>
<td>8 - 24</td>
</tr>
<tr>
<td>T3.2b Eavesdropping backhaul interface (user-plane)</td>
<td>3</td>
<td>2</td>
<td>1 - 3</td>
<td>6 - 18</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>T5.1a Traffic modification radio interface (control-plane)</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>T5.1b Traffic modification backhaul interface (control-plane)</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>T5.2a Traffic modification radio interface (user-plane)</td>
<td>3</td>
<td>2</td>
<td>1 - 3</td>
<td>6 - 18</td>
</tr>
<tr>
<td>T5.2b Traffic modification backhaul interface (user-plane)</td>
<td>2</td>
<td>2</td>
<td>1 - 3</td>
<td>4 - 12</td>
</tr>
<tr>
<td>T6 Data modification</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 3: Risk Assessment for the eNB

For the purpose of comparison with the assessments of other network elements, for T1, T3 and T5 an aggregated view on both interfaces, but with a differentiation of control plane and user plane is useful. Using this specific view, the following assessment holds:
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

Table 4: Risk Assessment for the eNB with aggregated view of interfaces

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>30</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>18</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control-plane)</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user-plane)</td>
<td>3 - 4</td>
<td>2</td>
<td>1 - 3</td>
<td>6 - 24</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c-plane)</td>
<td>2 - 3</td>
<td>2</td>
<td>4</td>
<td>16 - 24</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u-plane)</td>
<td>2 - 3</td>
<td>2</td>
<td>1 - 3</td>
<td>4 - 18</td>
</tr>
<tr>
<td>T6 Data modification</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
</tbody>
</table>

4.2.3.2 Relay Node (RN)

4.2.3.2.1 Architecture

The following picture shows the Relay Node (RN) embedded into a 4G RAN.

![Relay Node Architecture](image)

*Figure 27: Relay Node Architecture*

The main purpose of the RN is to improve radio coverage in areas that are not or cannot easily be covered by regular eNBs. The RN behaves like any eNB towards the UE, i.e. it terminates the radio interface (LTE-Uu). However, it is not connected to the rest of the mobile network by a fixed line, but uses a radio interface called Un to connect to another eNB, called Donor eNB (DeNB). Un is based on LTE-Uu; it uses the same protocol stack up to layer 2. At the Un interface towards the DeNB, the RN transports...
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

- user traffic from UEs using GTP-U/UDP/IP (i.e. the same protocol stack as on S1-U);
- control plane traffic using S1-AP/SCTP/IP (i.e. the protocol stack of S1-MME) in the user plane of the LTE-Uu interface.

X2 traffic is treated analogously, i.e. it is carried via the Un interface like S1 traffic.

So the DeNB needs to comprise SAE-GW functions to access the user plane traffic of the RN in order to relay the UE user plane traffic and extract the control plane traffic towards the MME.

4.2.3.2.2 Security Concept

3GPP has conducted a study on RN security which is documented in [3GPP_TR33816]. One part of it is a threat analysis. See Annex A of this document for a summary. Subsequently, 3GPP has specified the security architecture for the RN, documented in an appendix to [3GPP_TS33401].

In its function as an eNB, the RN must comply with the applicable eNB security requirements. In particular, it needs a secure environment to store credentials and other critical configuration data, perform security critical operations etc.

A RN must perform a device integrity check on startup and must not proceed if this check fails.

The RN uses an UICC with a USIM, the USIM-INI, which can be used for an initial attach to the network, in the role of a UE. During this initial attach, the RN may be able to connect to some O&M server to receive configuration data. The subscription corresponding to the USIM-INI must be restricted in a way that it only allows connectivity to the relevant O&M servers. The O&M connection must be integrity protected and encrypted with the authentication based on certificates securely provisioned in the RN. The initial attach is not mandatory – the RN may be preconfigured with all necessary data. A USIM-INI is not needed in this case.

The UICC of the RN must hold another USIM, the USIM-RN. This USIM is used when the RN attaches to the network in the role of a relaying node. A one-to-one binding between an individual RN and a USIM-RN and a secure channel between USIM-RN and RN secure environment is implemented. The secure channel is a TLS connection with mutual authentication based on certificates. (There is also an alternative PSK based secure channel variant.)

On the LTE-Uu interface, the user plane is only encrypted, not integrity protected. In contrast to this, S1 and X2 control messages between RN and DeNB are also integrity protected using a dedicated algorithm key (derived from the key $K_{eNB}$ that is part of the EPS key hierarchy, see [3GPP_TS33401]).

All O&M traffic for the RN must be secured as required for eNBs.

4.2.3.2.3 Assessment

In the following we discuss the threats for the asset RN according to the threat categories defined in Chapter 5. As a rule, we compare the threat to the respective threat for the eNB. To avoid complexity, we do not assess threats per interface. However, we distinguish between control and user planes for threats T3 and T5.
T1 Flooding an interface
All interfaces of the RN are radio interfaces. They can be flooded in the same way as at the eNB. The likelihood of an attack may depend on the number of deployed RNs. The vulnerability may be somewhat higher as for the eNB, where it is more difficult to get access to the S1 interface. The impact may be somewhat lower as for the eNB, as less UEs are expected to be served by a typical RN than by a typical eNB. We consider these differences however hardly measureable in our metric.

T2 Crashing a network element via a protocol or application implementation flaw
The considerations of T1 apply, leading to the same result, i.e. the values for the RN are very similar to the eNB.

T3 Eavesdropping
The LTE-Uu interface is identical to this interface at the eNB. On the Un interface, the confidentiality protection is not implemented by IPsec but by the LTE radio interface confidentiality mechanisms. There is no indication that this may be stronger or weaker, resulting in the same vulnerability. No difference to the eNB is seen for the likelihood. The impact may be slightly smaller, as less terminals may be affected.

T4 Unauthorized access to sensitive information on a network element via leakage
The RN holds sensitive information, like the eNB. We assume that RNs are deployed at places that are even harder to secure physically than the average eNB deployment places, so physical access is easier. RN may therefore be more likely to be physically controlled by attackers. Also, the network may be more tolerant to the failure of an RN (caused by a physical attack) than to the failure of an eNB. We also assume RNs will be smaller, cheaper and have a weaker housing than typical eNBs. So there is a higher vulnerability wrt. physical tampering and consequently, the vulnerability wrt. T4 is higher.

We see no significant difference to the eNB regarding the impact of a successful attack, even if less mobiles are using an RN on average.

T5 Traffic modification
The considerations of T3 hold – no significant difference to the eNB.

T6 Data modification on a network element
The considerations of T4 hold – higher likelihood and vulnerability as for the eNB.
Note that we do not consider changing of the UICC to fall into this category. Using a different UICC is rather an attack against this UICC, or an attack against the RN to which this UICC belongs.

T7 Compromise of a network element via a protocol or app. implementation flaw
We see no significant difference to the eNB wrt. the likelihood and impact of this threat. As with threats T4 and T6, an increased exposure and consequently an increased vulnerability can be concluded when assuming that an RN is more likely to be controlled physically by attackers.
T8  Compromise of a network element via a management interface
Remote RN management should be secured to the same degree as eNB management. We assume that there is no active local management interface on an operational RN, and thus conclude no significant differences in the assessment compared to the eNB.

T9  Malicious insider
We see no significant differences to the eNB.

T10  Theft of service
Similar to the eNB, the RN is not very involved in charging, but with a compromised RN, the attacker may impersonate other subscribers or otherwise sent traffic that evades charging. The RN is somewhat more vulnerable against compromise, and thus also against theft of service.

The following table summarizes our results:

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>3</td>
<td>5</td>
<td>2</td>
<td>30</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>18</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control-plane)</td>
<td>3 - 4</td>
<td>2</td>
<td>3 - 4</td>
<td>18 - 32</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user-plane)</td>
<td>3 - 4</td>
<td>2</td>
<td>1 - 3</td>
<td>6 - 24</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c-plane)</td>
<td>2 - 3</td>
<td>2</td>
<td>4</td>
<td>16 - 24</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u-plane)</td>
<td>2 - 3</td>
<td>2</td>
<td>1 - 3</td>
<td>4 - 18</td>
</tr>
<tr>
<td>T6 Data modification</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>T7 Compromise via</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>implementation flaw</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>45</td>
</tr>
<tr>
<td>T8 Compromise via</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>management interface</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 5: Risk Assessment for the RN

4.2.3.3 Home eNB (HeNB)

4.2.3.3.1 Architecture and Security Concept
The HeNB is the 4G equivalent to the HNB, i.e. it is a very small eNB used to provide 4G services in a restricted area, e.g. in a home or inside enterprise premises. HeNBs are expected to be connected to the mobile core network via the Internet, e.g. using DSL on the first mile and infrastructures of Internet service providers up to an interconnection point of the mobile network. The HeNB has functions similar to the (macro) eNB which comprise critical...
security functions, as described above. Even more than the eNB, the HeNB is exposed to physical tampering.

Because of the strong similarity between HNB and HeNB, the considerations below hold not only for the HeNB, but also for the HNB. In fact, the practical security testing in the framework of ASMONIA as well as the “femto cell hacking” performed by certain researchers (described below) was applied to HNBs only, as HeNBs were not yet commercially available.

3GPP has conducted a study on H(e)NB security which is documented in [3GPP_TR33.820]. One part of it is a threat analysis. See Annex A of this document for a summary. Subsequently, 3GPP has specified the security architecture for the H(e)NB in [3GPP_TS33320].

The following picture taken from [3GPP_TS33320] illustrates the H(e)NB architecture. (The abbreviation H(e)NB refers to a node that is either a HNB or a HeNB.) Please refer to that document for a detailed explanation of this architecture and the functions of the different entities.

![Figure 28: H(e)NB Architecture](image)

All user plane and control plane traffic originating from a H(e)NB is routed over the insecure link (e.g. a connection through the Internet) to a security gateway SeGW. The SeGW may use the AAA Server/HSS to perform the optional hosting party authentication (see below). The traffic of many HeNBs may be aggregated by an HeNB-GW, or otherwise HeNBs may communicate directly with core components like MME and SAE-GW. (For HNBs, the use of a HNB-GW is mandatory.) A H(e)NB can be remotely configured with the help of the H(e)NB Management System H(e)MS, which can either be inside the operator’s core network or can be connected to the Internet directly (which may be the case for example for a H(e)MS that is operated by the vendor of the H(e)NB).

As a H(e)NB is exposed to all kind of physical tampering, 3GPP requires that there is a Trusted Environment (TrE) inside the H(e)NB with the following properties:

- The TrE is built on a HW-based root of trust and a secure boot, where only verified components can be loaded (e.g. the operating system must be verified).
- The TrE stores sensitive data and performs sensitive functions and thus facilitates device integrity check, device validation and device authentication.
- Only authorized access to TrE functions and data inside the TrE is possible.
- For the HeNB, the secure environment (as required for eNBs) is established assured by the TrE.

Upon boot and before connecting to the core network and/or the H(e)MS, the H(e)NB must perform a device integrity check. If an attacker has tampered with the device and the check
fails, the TrE does not allow access to the sensitive information needed for device authentication, so the H(e)NB cannot connect successfully to the SeGW or the H(e)MS.

Two different mechanisms are specified for mutual authentication between the HeNB and the operator’s core network:

- A mandatory device authentication between the H(e)NB and the SeGW using IKEv2 based on public key certificates. This requires suitable certificates to be provisioned at the H(e)NB, or alternatively, enrolment of such certificates via an H(e)MS outside the operator network. It is not mandatory, but strongly recommended to support the OCSP protocol in the H(e)NB to allow checking the revocation status of the SeGW certificate. The SeGW on the other hand may support OCSP or CRL checking or both to check the revocation status of certificates.

- An optional authentication of the party hosting the H(e)NB based on a removable hosting party module similar to a USIM card may follow the mandatory device authentication. If authentication of the hosting party is used, then EAP-AKA is used and the authentication endpoint on the core network side is the AAA/HSS. The SeGW acts as the authenticator.

The device authentication between H(e)NB and SeGW based on IKEv2 is used to establish an IPsec tunnel (using ESP in tunnel mode) between the H(e)NB and the SeGW. Note that IPsec is not mandatory to use but may be replaced by mechanisms providing equivalent security.

The SeGW may perform access control for H(e)NBs, e.g. it may blacklist or whitelist H(e)NBs for the access to the core network.

Usage of a given H(e)NB may be restricted to a limited set of mobile subscribers, either specified by a Closed Subscriber Group (CSG) or by an access control list (for the HNB only).

If the H(e)MS is located in the operator’s core network, then it resides behind the SeGW and the connection between the H(e)NB and the H(e)MS is protected with the help of the IPsec tunnel between H(e)NB and the SeGW. If the H(e)MS is connected to the Internet directly, all communication between H(e)NB and H(e)MS must be protected by a mutually authenticated TLS connection based on certificates. The necessary certificates must be provisioned at the H(e)NB in this case.

Due to national regulations on spectrum usage, a H(e)NB must only be used at allowed locations. To verify this, 3GPP has specified a number of different methods which are all optional, although at least one of them must be applied. Note however, that all these methods can be overcome by motivated attackers, i.e. such attackers may be able to operate H(e)NBs at arbitrary locations (and possibly abuse them to attack mobile device at these locations).

4.2.3.3.2 Practical Security Testing performed in ASMONIA

In the framework of the ASMONIA work, a practical penetration test of a commercial HNB was executed, but with limited effort only.

The first step here was information gathering about the device. For example, a hint on the usage of Open Source Software was given in the product documentation. Following this hint it was possible to find the respective source code in the Internet. These sources suggested that a Linux kernel is included in the HNB firmware. This information is relevant, as typically
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(I)-1.0

there are lots of exploits available for Linux kernels, and these could give the chance to somehow compromise the HNB.

In a second step, the hardware was investigated with the goal to find ways to manipulate the firmware and thus gain control over the device. An obvious step here is to look for a serial connection on the system board (which may be present as it may be required for testing purposes). In this case, it was possible to identify solder joints of a serial interface and solder a console cable to it that allows to connect the board to a computer with a terminal emulation software running on it.

The following figure shows the board with the serial cable.

![Serial Connection soldered to a HNB](image)

However, in this case it was not possible to use the serial interface – it may have been disabled or blocked by the system.

In the third step, the network connection of the HNB was investigated and attacked. The HNB was connected to a laptop, and a protocol analyzer software was run on the laptop to capture and analyze the messages coming from the HNB. The HNB first performed a DHCP request to get an IP address, which was answered by a DHCP server running on the laptop. After that, the HNB performed a DNS request for a host obviously belonging to the network operator who sold the HNB. Having received an IP address for that host, the HNB proceeded to initiate an IKEv2 handshake to set up a security association with that host. No weakness was found in this procedure that could be exploited for further attacks.

In a last step, a security scanner software was used to discover characteristics of the system and to find any open ports on the system. The result was that no open ports could be detected and the system could not be identified by the scanner.

To summarize, the result of this practical test was that the target system seemed to be following the standardized, secure procedures and didn’t show any obvious, exploitable vulnerabilities at the network interface.
4.2.3.3.3 Femto Cell Hacking (external work)

A number of research teams has worked on security testing of commercially available HNBs. Such activities are sometimes dubbed “femto cell hacking”.

The Hacker’s Choice Project

As early as mid 2009, at “The Hacker’s Choice” (THC) (see www.thc.org), a project was conducted aiming at a specific HNB product. Here, it was found out that a serial port was located on the board which was not disabled. Via a serial cable soldered to the board, it was possible to get a login prompt on a connected terminal (emulation). It turned out that root login was possible using a trivial password. From that point, it was possible to reconfigure the system to allow access using ssh via the network interface. Further manipulation of the system was used to show the possibility

- to catch IMSIs of subscribers in the range of the HNB;
- to intercept cleartext traffic (voice) running over the HNB;
- to impersonate subscribers while they were attached to the network via the HNB and perform call fraud, i.e. perform calls and have them charged to the attacked subscriber.

More details can be found at http://wiki.thc.org/vodafone (online, last checked 17.10.2011).

This example shows that the possibility of physical access to the HNB hardware significantly increases the risk related to this network element, as it may allow to detect and exploit vulnerabilities that may remain hidden to remote attackers.

Google Code Project

A project targeting another HNB product is documented on http://code.google.com/p/samsung-femtocell (online, last checked 17.10.2011). The documentation gives detailed instruction how to remove the casing, to access the console, which passwords to use etc.

T-Labs TU Berlin Project

Detailed research on HNB security weaknesses was performed by Borgaonkar et.al. at the Telecom Innovation Labs at the Technical University of Berlin (http://www.laboratories.telekom.com).

The product under test was analyzed in several steps.

- A serial port could be located on the board, but there was no login prompt enabled.
- Sniffing and scanning showed the usage of (unauthenticated) NTP, but also a web interface at port 80.
- It was possible to force a factory reset for the device, which made it request a new image via plain HTTP. A signed and encrypted image can be retrieved via this request, but it turned out that also a signed but unencrypted image was accessible by modifying the URL used for the image request. This allowed to further analyze the recovery procedure.
- During the recovery procedure, the HNB used HTTPS to request a parameter and firmware list. However, no authentication of the server was implemented at the HNB, which made it possible to provide a faked parameter and firmware list.
The HNB did only load encrypted and signed software, but via the parameter list, it was possible to provide it with faked keys that were subsequently used to “verify” faked software.

It turned out that a faked parameter list could be used to enable a login prompt at the serial interface on the board. The respective root password was found as an MD5 hash in the recovery image.

Ultimately, it was possible to make the HNB “recover” using an image controlled by the attacker.

Moreover, a "hidden" web interface provided by the device manufacturer was found, which allowed very convenient, unauthenticated access to all configuration options of the HNB, allowing it e.g. to impersonate arbitrary networks towards mobiles.

With a HNB compromised like this, very serious attacks could be performed on subscribers in the range of the HNB, including

- IMSI catching,
- interception of voice calls,
- interception and modification of short messages (SMS),
- impersonation of subscribers (e.g. placing calls, sending short messages),
- detaching subscribers from the network.

However, the attacks turned out not to be restricted to the local range. Rather, weaknesses in the OAM servers for the HNB network could be abused to retrieve information on other HNBs, and even to provide faked images that might subsequently have been loaded by other HNBs.

Moreover, a buffer overflow exploit was found for the web server running on the HNB. This would allow an attacker to become root remotely on arbitrary HNBs, which were all reachable from the originally attacked HNB. So all attacks mentioned above could be applied to the complete HNB network.

A presentation about this work given by the researchers in August 2011 is http://femto.sec.t-labs.tu-berlin.de/bh2011.pdf (online, last retrieved 17.10.2011).

Clearly, the target system of this research work failed to implement the security measures specified by 3GPP for H(e)NBs. However, the system was a commercial product actually deployed in a productive network. This shows that it is reasonable to assume that despite of the sound security concept as specified by 3GPP, H(e)NBs will in practice be vulnerable to a variety of attacks, and that the impact of such attacks can be significant.

4.2.3.3.4 Assessment

In the following we discuss the threats for the asset HeNB according to the threat categories defined in Chapter 5, taking into account the results of practical HNB security testing and of femto cell hacking done in external projects, as described above.

T1 Flooding an interface

Flooding the radio interface of an HeNB should be as easy as flooding the radio interface of an eNB. However, the impact of flooding a HeNB is more limited, as a HeNB will typically serve only a very limited number of UEs anyway. If a HeNB is connected to the mobile core network via the Internet, e.g. using a DSL, it can be flooded easily. However, the HeNB may
be able to handle such a flood by discarding all traffic that is not valid IPsec traffic immediately. However, such an attack may not be very attractive for attackers.

**T2  Crashing a network element via a protocol or application implementation flaw**

Similar to an eNB a HeNB supports many different protocols on the radio interface, many of which have been newly designed for E-UTRAN. The probability that an implementation of one of these protocols will exhibit a serious implementation flaw is therefore rather high. HeNBs can be physically controlled by attackers, so it may be easy for attackers to test them and find such flaws. However, as the coverage area of a single HeNB is even smaller than the coverage area of a single eNB, a crash of a single HeNB has a limited DoS effect on some legitimate users only. Nevertheless, we do not consider T2 as a very serious threat, also because crashing an HeNB does not seem to be a very attractive attack target for an attacker.

**T3  Eavesdropping**

As described above, control plane traffic as well as user plane traffic on the radio interface is expected to be encrypted (on the PDCP layer with SNOW 3G or AES). All traffic on the backhauling interfaces should be encrypted using IPsec or TLS (but the practical tests, as described above, show that also plain HTTP may be used for certain O&M traffic). The HeNB itself has access to the cleartext traffic anyway. Although a TrE should provide protection here, it can be assumed that there will be HeNB implementations where the TrE is too weak to resist attacks making use of physical access to the HeNB. This may give attackers a chance to eavesdrop traffic directly at the HeNB. The impact of eavesdropping the control plane may be somewhat more limited compared to the eNB. For the user plane the same considerations hold as for the eNB, assuming an attacker will be able to trick the targeted mobile into using the HeNB rather than the surrounding macro cell (i.e. the local eNB).

**T4  Unauthorized access to sensitive information on a network element via leakage**

As described for T3, access to data on the HeNB may be possible because of weak TrE implementations. Such data may be security critical and facilitate subsequent attacks on subscribers as well as on the network. We consider this a very likely attack with considerable impact.

**T5  Traffic modification**

The situation is similar as in T3: Traffic on the interfaces of the HeNB should be confidentiality protected, but the cleartext may be accessible at the HeNB itself despite of protection mechanisms. So the traffic may be modified there.

**T6  Data modification on a network element**

The considerations for T4 hold also for T6 — attackers may be able to exploit weak TrE implementations and flaws in the procedures trying to preserve the integrity of the HeNB and thus be able to modify data on the HeNB.

**T7  Compromise of a network element via a protocol or app. implementation flaw**

As discussed in the preceding threats, and demonstrated in practice, it is very likely that the security measures of a HeNB are not strong enough to be resistant against attacks based on the knowledge that can be gained by physically accessing and tampering with a HeNB. Such attacks can easily result in full compromise of a HeNB. In fact, this will be the primary goal of an attacker, and it will have significant impact such as attacks on the network via the control
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

or management plane, impersonation of UEs, modification of traffic, eavesdropping, etc. We estimate the impact of an HeNB compromise smaller than the impact of an eNB compromise because of the more restricted scope of the HeNB.

T8 Compromise of a network element via a management interface

As demonstrated in practice, there is a significant chance that local management interfaces like a serial port are available (e.g. because the manufacturer needs them for testing purposes) and can be enabled. Also, HTTP based management interfaces have turned out to be poorly secured and thus accessible in practice. We therefore estimate this threat like T7.

T9 Malicious insider

We estimate this threat similar as for the eNB, but, as discussed in T7, with a somewhat smaller impact.

T10 Theft of service

Similar to the eNB, the HeNB is not very involved in charging, but with a compromised HeNB, the attacker can impersonate other subscribers or may otherwise send traffic that evades charging. The HeNB is somewhat more vulnerable against compromise, and thus also against theft of service. The scope and therefore the impact of the attack may however be more limited.

The following table summarizes our results:

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerability Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>2</td>
<td>4</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control-plane)</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>27</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user-plane)</td>
<td>3</td>
<td>3</td>
<td>1 - 3</td>
<td>9 - 27</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>48</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c-plane)</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u-plane)</td>
<td>3</td>
<td>3</td>
<td>1 - 3</td>
<td>9 - 27</td>
</tr>
<tr>
<td>T6 Data modification</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>48</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>64</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>64</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>32</td>
</tr>
</tbody>
</table>

Table 6: Risk Assessment for the HeNB
4.2.3.4 Home eNB Gateway (HeNB-GW)

The HeNB-GW serves as a concentrator for control plane traffic from possibly many HeNBs and appears as a single eNB towards the MME. This allows for a scalable connection of the HeNBs to the evolved packet core. The protocol stack for S1-U and S1-MME is the same as illustrated in Section 6.2.2.1.

As described in [3GPP_TS36300], the HeNB-GW supports the S1-MME interface towards the HeNB as well as towards the MME. In addition, the HeNB-GW may also support the S1-U interface. However, this interface may also be a direct interface between the HeNB and the SAE-GW. The following figure taken from [3GPP_TS36300] illustrates the logical interfaces of a HeNB-GW for this second case.

![Figure 30: Logical Interfaces of an HeNB-GW](image)

Even if S1-U is relayed through the HeNB-GW, we don't assume there would be a significant user plane traffic processing allowing to attack the HeNB-GW. However, access to the user plane would be possible, e.g. for a malicious insider, like for all network elements in the user plane (like the SAE-GW, routers etc.). We therefore include threats to user plane traffic in our assessment.

Although formally part of the RAN, the HeNB-GW can be assumed to be deployed on core sites rather, behind a SeGW and not exposed to external networks. If the above architecture is implemented correctly, the HeNB-GW will receive traffic from HeNBs (via secure tunnels terminated at the SeGW), but not traffic from external hosts in the Internet. As discussed above, HeNBs have a significant probability of getting compromised, and compromised HeNBs may be used to attack the HeNB-GW. However, except for DoS there seems not to be a very obvious goal for an attacker to achieve by such attacks.

A more detailed assessment per threat is given in the following.

**T1 Flooding an interface**

Flooding an interface of the Home eNB Gateway should be rather difficult as it is not directly reachable from the Internet as it resides behind the SeGW. However, malicious HeNBs, e.g. HeNBs that have been compromised by their hosting parties, could try to flood the gateway.
T2 Crashing a network element via a protocol or application implementation flaw
Crashing an HeNB-GW via a protocol or application implementation flaw could be possible. However, as in T1, only (malicious) HeNBs are in the position to attack. We assume that an HeNB-GW is not a very attractive attack goal. The impact of crashing an HeNB-GW is a DoS against all connected HeNBs and consequently a DoS against all UEs connected to any of these HeNBs.

T3 Eavesdropping
Eavesdropping on the interfaces of the HeNB-GW should not be possible for attackers that do not have access to the operator’s core network. We assume a low likelihood for such attacks. The impact would however be considerable, in particular for the control plane. The impact of eavesdropping user traffic is estimated following the conclusions made in section 3.3.

T4 Unauthorized access to sensitive information on a network element via leakage
As it mainly acts as a concentrator for control plane traffic, there is not much sensitive information directly stored on the HeNB-GW.

T5 Traffic modification
The considerations of T3 hold. It seems even less likely that an attacker tries to modify traffic on the HeNB-GW.

T6 Data modification on a network element
The only data on the HeNB-GW that seems to be a possibly attractive target for this type of attack is the configuration data. Changing the configuration could have a considerable impact. We do not consider this to be a very likely or easy attack.

T7 Compromise of a network element via a protocol or app. implementation flaw
An implementation flaw on the HeNB-GW may enable an attacker to gain control over an HeNB-GW, e.g. via a previously compromised HeNB. A compromised HeNB-GW is a serious incident that can lead to DoS and to further attacks on network elements, in particular MMEs.

T8 Compromise of a network element via a management interface
A management interface to the HeNB-GW is assumed not to be easily accessible by attackers. On the other hand, weak configurations of management interfaces are not uncommon in operational networks. The impact of such a compromise is the same as in T7.

T9 Malicious insider
We do not consider it very likely that a malicious insider targets a HeNB-GW. However, abuse of a HeNB-GW by a malicious insider cannot easily prevented – i.e. the vulnerability is high. Clearly the impact of a successful attack would be high, too (e.g. DoS for many HeNBs, further attacks on MMEs etc.).
T10 Theft of Service

The HeNB-GW, as the eNB and the HeNB, does not support charging functionality as it serves as a concentrator of the control plane traffic from many connected HeNBs. We do not assume the HeNB-GW can be easily abused for theft of service.

The following table summarizes the results of our threat analysis:

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>2 - 3</td>
<td>2</td>
<td>4</td>
<td>16 - 24</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>2 - 3</td>
<td>2</td>
<td>4</td>
<td>16 - 24</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control-plane)</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user-plane)</td>
<td>2</td>
<td>1</td>
<td>1 - 3</td>
<td>2 - 6</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c-plane)</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u-plane)</td>
<td>1</td>
<td>1</td>
<td>1 - 3</td>
<td>1 - 3</td>
</tr>
<tr>
<td>T6 Data modification</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>2</td>
<td>2 - 3</td>
<td>5</td>
<td>20 - 30</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>4</td>
<td>5</td>
<td>20 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

*Table 7: Risk Assessment for HeNB-GW*

4.2.3.5 Other Components for the Support of HeNB

The 3GPP architecture for H(e)NBs specifies the usage of a Security Gateway (SeGW) terminating the IPsec tunnels from the H(e)NBs on the network side. The SeGW is assessed together with the slightly different security gateway specified in [3GPP_TS33210], see section 4.3.7.

Moreover, there may be a AAA-Server performing certain authentication and access control functions. AAA-Servers are assessed in 4.3.1.5, and despite some differences in the functions, this assessment can be considered valid also for the AAA-server of the H(e)NB architecture.

The H(e)NB Management System HMS is a specific OAM server. OAM servers are assessed in section 4.4.1.
4.3 Core Network

The core network of a 4G mobile network according to the 3GPP architecture can be divided into several domains, as visualized by the figure below.

In a pure 4G network, the core network is fully packet based, and is called Evolved Packet Core (EPC). As 4G networks are expected to be evolved from existing 3G networks, core network elements specific to a 3G network (i.e., up to 3GPP Rel.7) may also be part of the network. 3G core networks comprise the Circuit Switched (CS) domain and the Packet Switched (PS) domain. The CS domain supports circuit switched services, in particular voice. The PS domain—similar to the EPC—provides IP connectivity between mobile terminals and IP service networks. These include external networks like corporate IP networks or the Internet. A specific IP service network that is typically not external, but part of the PLMN itself, is the IP Multimedia System (IMS).

Common to packet and circuit switched services is the need for subscriber management, including the location of mobile subscribers. 3GPP specifies the Home Subscriber Server (HSS) as the central component here. Functions like the Equipment Identity Register (EIR) complement the subscriber management domain.

Charging is an important function in mobile networks. 3GPP specifies the Policy and Charging Control (PCC) architecture, with the Policy and Charging Rules Function (PCRF) as the central control component. Charging systems are used to perform offline and online charging functions (for postpaid and prepaid service, respectively).

In addition to the domains and functions mentioned above, 4G mobile networks comprise various additional functions, including Messaging Services and Location Services.

The following sections discuss the risks for the various parts of a 4G mobile as summarized above. Note that there are many aspects that are similar for the different core network elements, e.g., they are located in a rather well protected area of the mobile network, they have many core-internal interfaces where external attackers cannot easily attack, they all provide functions to support operation, administration and maintenance, they are all located in physically protected premises etc. To avoid redundancy, this aspects are mostly discussed in an exemplary style for the SAE-GW, but not repeated for each core network element.
4.3.1 Evolved Packet Core (EPC)

In a pure 4G network, all user traffic is packet traffic and is transported via the EPC. The central user plane component is the SAE-GW, the control plane component for 4G access is the MME. Access via 2G/3G access networks to the EPC is possible via an SGSN. To allow for access via non-3GPP access networks, an evolved Packet Data Gateway (ePDG) may be used between core and access network, and a the 3GPP AAA-Server is specified to support user authentication in non-3GPP access scenarios.

*Figure 1* on page 9 illustrates the EPC as the core network of a 4G mobile network. Note that that picture shows only the most relevant components and interfaces.

4.3.1.1 SAE-GW

The following picture shows the SAE-GW embedded in the EPS.

*Figure 32: The SAE-GW within the Mobile Network*

Note that the SAE-GW can be split in a Serving Gateway and a PDN Gateway. The reference point between these components is called S5 if it is inside one PLMN and S8 if it is between two PLMNs (to support roaming). S5 and S8 are functionally equivalent. Only S8 is discussed in the following; S5 is treated like an internal interface.

The SAE-GW supports the following functions/protocols:

- Termination of user plane tunnels towards UEs (includes tunnel control, e.g. protocols to set up or tear down tunnels):
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

- GTP at S4 interface to S4-SGSN, Gn/Gp interface to Gn/Gp-SGSN, S8 interface to (other) Serving-GW
- GTP-c (i.e. control only) at S1-MME interface to MME
- GTP-u (i.e. user plane only) at S1-U interface to eNBs and S12 interface to UTRAN RNC or UTRAN NodeB with RNC function
- MIPv4 (SAE-GW is Home Agent, S2a interface)
- PMIPv6 (SAE-GW is Local Mobility Anchor, S2a, S2b interface, S8 to other Serving-GW)
- DSMIP (SAE-GW is Home Agent, S2c interface)
- GRE (S103 interface to CDMA2000 access)
- Note that IPsec is recommended at many of these interfaces, in particular at the S1 interface towards the RAN, or between core network components, if different security domains are involved (e.g. two different PLMNs)

- Switching of user plane tunnels as a Serving-GW only (GTP-U or PMIP over the S8 interface to the PDN-GW function in another SAE-GW, Serving-GW is Mobile Access Gateway in PMIP)
- Interconnection and IP forwarding between user plane tunnels to UEs and IP service networks, including MNO internal networks (e.g. an IMS core network) and external networks like corporate networks or the Internet.
  - Interconnection to IP service networks may be IP user plane directly on layer 2, or may involve the tunneling protocols GRE, IPsec or L2TP
  - IP routing protocols like OSPF may be supported
- RADIUS client using RADIUS to communicate with RADIUS servers in IP service networks
- DHCP relay agent using DHCP to communicate with DHCP servers in IP service networks and DHCP clients in UEs
- Interaction with 3GPP AAA-Server using Diameter (S6b interface)
- Policy and Charging Enforcement Function (PCEF), controlled by the Policy and Charging Rules Function (PCRF) using Diameter over the Gx or Gxc interface.
- Client to an Online Charging System (Diameter Credit Control, RFC 4006).
- Client to an (Offline) Charging Gateway using GTP' (GTP prime)
- OAM functions (like SSH server, (S)FTP client/server, HTTP(S) server, SNMP instance, or proprietary OAM functions) using respective standardized or proprietary protocols to communicate with various components of an operation and maintenance center, like element managers, backup&restore servers, logging servers etc.)
- Lawful Interception (receiving interception requests and transporting communication content and interception related information towards the law enforcement agencies' monitoring centers).

The SAE-GW typically communicates with network elements in external networks, e.g. SGSNs, SAE-GWs or ePDGs in other PLMNs, (e)NBs or RNCs in LTE or UMTS radio access networks (that may belong to different organizations, e.g. in the case of RAN
sharing), components in non-3GPP access networks, AAA servers or DHCP servers in external IP service networks (e.g. corporate networks), or external monitoring centers for lawful interception.

The SAE-GW may communicate with various remote communication peers over backbones that are owned by 3.partys, e.g. backhauling networks (that connect access network components like eNBs to the core), a GRX (GPRS roaming exchange network) that interconnects different PLMNs, or backbone networks that interconnect core network sites. It should be noted that (e)NBs, even when being part of the same PLMN as the SAE-GW, may be deployed in locations where there is not much physical protection, so there is a non-neglectable probability that these communication peers of the SAE-GW may be compromised by attackers via physical access.

Note that user traffic from UEs is mostly only forwarded but not further processed by the SAE-GW. The same holds for traffic from IP service networks, except for communication with a AAA server or DHCP server in an external IP service network.

There is one notable case, where the SAE-GW communicates with terminals and must do more than pure IP forwarding: If the terminal uses DSMIP for non-3GPP access, the SAE-GW acts as home agent and has to process DSMIP control traffic of the terminal. Note that it is assumed that the vast majority of terminals will not be connected to the mobile network via DSMIP.

In the following, the threat categories specified in chapter 3 are discussed for the SAE-GW.

**T1 Flooding an interface**

The SAE-GW has many interfaces. It is connected to various different network elements. It has to process traffic from external IP networks, including the Internet, and from UEs (that may be hostile, e.g. may even form a Botnet). On the other hand, an SAE-GW should be prepared to process traffic on user interfaces in wire speed, and should comprise sound overload control mechanisms.

If one of the interfaces is flooded, this may result in a DoS condition for many users, which may be long lasting.

**T2 Crashing a network element via a protocol or application implementation flaw**

The SAE-GW supports a multitude of protocols. There is a considerable chance of implementation errors in some of these protocols. Moreover, the SAE-GW communicates with hostile peers, e.g. hosts in the Internet and UEs. Mostly, this is only IP forwarding, but it may also include processing DSMIP control traffic.

The SAE-GW does not support any applications for end users. One function that maybe abused in the sense of this threat could be the RADIUS client function, which may have to process RADIUS replies from external AAA servers.

Crashing of the SAE-GW will mean DoS for many subscribers until the SAE-GW is restarted. As long as the exploited vulnerability is not fixed, an attacker may be able to crash an SAE-GW repeatedly.

**T3 Eavesdropping**

Most of the SAE-GE interfaces are internal, i.e. between core components only, in the well protected network core, and so external attackers have few possibilities to attack. On the backhauling interfaces, 3GPP mandates the usage of IPsec encryption, unless the link is otherwise sufficiently protected. It is assumed that operators follow this recommendation.
T3.1 Control and management plane
Control protocols may be cryptographically protected on all interfaces – however, the focus of this protection may be on integrity rather than confidentiality.
Most control information may not be critical and not be valuable for attackers, i.e. cannot easily be abused.

T3.2 User plane
It seems much more obvious to mount eavesdropping attacks against the user plane not at the SAE-GW interfaces, but rather at other links in the communication path, e.g. in the IP service network (may be the Internet).
Sensitive user information may be protected by means independent of the mobile network, e.g. online banking is typically protected by the usage of HTTPS.
User information classification can range between "public" and "top secret". There is no valid estimation of the impact of loss of confidentiality (from the user's point of view). However, it is assumed that highly sensitive user plane data are secured independently of the mechanisms provided by the mobile network, so eavesdropping, even when breaking some mobile network specific protection, will only reveal encrypted data. This somewhat limits the impact of such an attack.
On the other hand, it should be noted that a high number of users could be affected if an attacker is able to eavesdrop user plane traffic at the SAE-GW.

T4 Unauthorized access to sensitive data on a network element via leakage
No user plane data are stored on the SAE-GW. The control data on the SAE-GW may be of no direct use for the attacker, so there may be no significant impact, if an attacker e.g. is able to read static configuration data or dynamic user profile data etc.

T5 Traffic modification
Similar considerations as for eavesdropping (T3) hold. Control traffic at external interfaces should be cryptographically integrity protected according to 3GPP standards.
A successful attack may lead to DoS or theft of service.

T6 Data modification on a network element
No user plane data are stored on the SAE-GW. As described in T4, there is not much chance to abuse user applications.
Modification of control data may lead to DoS or theft of service.

T7 Compromise of a network element via a protocol or application implement. flaw
Typically, the SAE-GW will implement good protection mechanisms, adequate to the high importance of this network element. However, as an SAE-GW is highly complex, it cannot be safely assumed that there are no exploits possible via abuse of one of the various functions and protocols.

T8 Compromise of a network element via a management interface
It is typical behavior of attackers to go for management interface weaknesses.
An SAE-GW is expected to provide means to secure the management properly, and these means may be used to an extent that depends on the operator. E.g., a well organized, solid
network operator may achieve a low vulnerability. However, poorer operational practices may lead to a significant vulnerability, taking into account that the SAE-GW is rather exposed to attackers via its user plane interconnections.

T9 Malicious insider

The likelihood of a malicious insider is relatively low, but may depend on (hard to influence factors) like the social and cultural context. An SAE-GW can be an attractive target for an insider, given its many functions, in particular for the user plane, and its central place in the network.

Abuse of an SAE-GW by a malicious insider, in particular an attacker with administrator access cannot be prevented by technical means. It may be logged however, so the attacker may fear to be detected afterwards. The vulnerability against the malicious insider threat also depends on organizational processes within the operator organization, and on operational practices with respect to the network operation.

T10 Theft of service

As the SAE-GW performs charging for user plane traffic, attacking it with the goal to steal services is assumed to be pretty likely. On the other hand, the charging function should not offer any obvious interfaces available to attackers, at least external attackers, so the vulnerability is estimated only at medium level. It may be lower, if only (simple) volume or time based charging is done, it may be higher, if more complex schemes like content based charging are applied. The impact of theft of service is obviously high for the operator, although it may not endanger the availability of the network and its services.

The following table shows the assessment resulting from the above considerations.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>48</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control plane)</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user plane)</td>
<td>2</td>
<td>2</td>
<td>1 - 3</td>
<td>4 - 12</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c-plane)</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u-plane)</td>
<td>1</td>
<td>2</td>
<td>1 - 3</td>
<td>2 - 6</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>45</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>3</td>
<td>2 - 4</td>
<td>5</td>
<td>30 - 60</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>3</td>
<td>2 - 3</td>
<td>5</td>
<td>30 - 45</td>
</tr>
</tbody>
</table>

Table 8: Risk Assessment for the SAE-GW
4.3.1.2 MME

The following picture shows the MME embedded in the EPS. Note that the possibility of inter-operator interfaces (e.g. between MME and HSS) is not visualized to reduce complexity in the picture.

![Image of Mobility Management Entity (MME)](image)

**Figure 33: Interfaces of the Mobility Management Entity (MME)**

The MME has no user traffic interfaces; all interfaces shown in the picture above are control plane interfaces. They are all based on IP.

The MME performs mobility management, which includes:

- Termination of NAS (non access stratum) signaling with mobile terminals (includes authentication and key agreement). NAS signaling is transported using S1-AP over SCTP. It is protected by keys that the MME derives from a key received from the HSS (see below).
- Communication with eNBs at the S1-MME interface using S1-AP over SCTP (includes transport of the key $K_{\text{eNB}}$ to the eNB; $K_{\text{eNB}}$ is derived by the MME and used by the eNB to derive keys for radio interface protection).
- Communication with other MMES, SGSNs and the SAE-Gateway using GTP-C.
- Communication with the HSS and EIR to retrieve subscriber and terminal information. In particular the HSS passes authentication information as well as a key that is used...
as master key to derive the keys for NAS signaling protection and radio interface protection. The communication with HSS and EIR is based on Diameter.

- Communication with GMLC/E-SMLC to support location services, using Diameter/LCS-AP over SCTP.
- Communication with the CBC to support cell broadcast (e.g. of location service information) using SBC-AP over SCTP/IP.
- For SRVCC (single radio voice call continuity), the MME interworks with components of the CS domain (GTP towards the MSC-Server and SGs-AP over SCTP towards the MSC/VLR).
- For interworking with CDMA2000 networks, the MME communicates with the HRPD network using S101-AP over UDP, and tunnels messages between the UE and the CS IWS.

Other MME functions include OAM functions like at the SAE-GW, and Lawful Interception for signaling traffic (receiving interception requests and transporting interception related information towards the law enforcement agencies' monitoring centers).

The MME communicates with network elements in external networks, in particular with HSSs in other PLMNs in roaming scenarios, but also with (e)NBs in LTE or UMTS radio access networks (that may belong to different organizations, e.g. in the case of RAN sharing), components in CDMA2000 networks, or external monitoring centers for lawful interception.

Like the SAE-GW, the MME may communicate with various remote communication peers over backbones that are owned by 3rd parties, and communicates with the eNBs, which have a somewhat higher likelihood of being compromised by physical attacks, as they are – in contrast to the MME itself - deployed in locations where there is not much physical protection.

**Threat and Risk Assessment for the MME**

The MME has many interfaces – but no user plane interfaces. It is connected to various different network elements, but mostly, these are internal network elements, so external attackers have few possibilities to attack. There is one notable exception: The MME communicates with mobile terminals (NAS signaling), so hostile terminals may try to attack the MME via this communication, i.e. by sending malformed signaling messages or floods of signaling messages. However, MME-products are expected to implement this sensitive protocol in a very sound way.

On the backhauling interfaces, 3GPP mandates the usage of IPsec integrity protection and encryption, unless the link is otherwise sufficiently protected. On external interfaces (e.g. towards a HSS within another network), IPsec integrity protection is mandated. It is assumed that most operators follow these recommendations.

An MME should comprise sound overload control mechanisms; moreover, MMEs may be pooled and share the signaling load for each area of the network. Still, the impact of a failure of an MME may be quite significant, as a considerable part of the signaling capacity of the network is lost during such a failure.

The impact of compromising an MME is high. The MME, as a central control component, can cause a lot of damage when acting maliciously. One specific abuse would be to get subscriber keying material from the HSS.
The following table shows the assessment resulting from the above considerations.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flooding an interface</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>Crashing a network element</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>Eavesdropping</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Unauthorized data access</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Traffic modification</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Data modification on a network element</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Compromise via implementation flaw</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>Compromise via management interface</td>
<td>2</td>
<td>2 - 3</td>
<td>5</td>
<td>20 - 30</td>
</tr>
<tr>
<td>Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>Theft of service</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

*Table 9: Risk Assessment for the MME*

**4.3.1.3 SGSN**

The following picture shows the SGSN as part of the EPC.

*Figure 34: SGSN within EPC*
The SGSN supports interconnection of 2G and 3G radio access networks to the EPC (as well as to the PS domain of the 2G/3G core network, see below). The SGSN handles control as well as user plane traffic, where the latter may also bypass the SGSN (in the "direct tunnel" architecture). The two traffic types are combined on the interfaces towards the RAN.

Similar to MME, the SGSN performs mobility management, including termination of NAS signaling with mobile terminals (includes authentication and key agreement, communication with 2G/3G RAN components, communication with other SGSNs, MMEs, SAE-GWs and the registers HSS and EIR.

Towards the SAE-GW and the MME, the SGSN may either support "native" EPS interfaces, (S4, S3) which means that user and control traffic is separated, or it uses the Gn/Gp interfaces specified already for the 2G/3G packet core which combines user and control traffic. User IP traffic is however always tunneled within GTP; the IP layer terminated by the mobile terminals is not handled by the SGSN.

The Gp interface between SGSN and SAE-GW can be between two different operators (in case of roaming); an SGSN in a visited network must also retrieve information from the HLR/HSS in the home network. Also the interfaces to the RAN may be external interfaces, as the RAN may belong to a different organization, e.g. in the case of RAN sharing.

The SGSN may be connected to a legacy SS7 network for access to the HLR, the EIR and optionally to the CS domain (i.e. to MSCs). SS7 signaling may be TDM/ATM based, but may also use IP transport (SIGTRAN). The SGSN can transmit charging information to a charging gateway (Offline Charging System, see 4.3.6.1). Moreover, it comprises LI functions as described for the SAE-GW (see 4.3.1.1).

Compared to the EPC, the SGSN functions correspond to functions of the MME (control plane) and Serving-GW (user plane).

**Threat and Risk Assessment for the SGSN**

In general, similar considerations hold as for the MME. However, a few deviations must be considered. First, the SGSN may also handle user plane traffic. This is restricted to GTP tunnel switching and user plane data itself is not processed. Still, this seems to make the SGSN in general somewhat more vulnerable, e.g. against DoS attacks, and attacks may affect not only control traffic, but also the user plane traffic of existing sessions.

While MMEs may be grouped to pools, where in case of failure of one MME another MME can handle the same area of the network, this may not be the case for SGSNs, so an SGSN failure may not be compensated easily by other SGSNs.

The direct peers of the SGSN in the RAN are not thousands of Base Stations or Node Bs. Instead, the peers are BSCs and the RNCs. In particular, the RNC is a device typically deployed at the core network site, without a backhauling link between the SGSN and the RNC, which reduces the risk of external attacks at this interface.

Different from the MME, the SGSN may communicate with SAE-GWs in other PLMNs in roaming scenarios. It can also collect charging data and send them to an (offline) charging system (see below), so a successful attack on the SGSN may affect charging.

The SGSN may be connected to the SS7 signaling network of the mobile network, possibly based on IP (i.e. SIGTRAN) and use it to communicate with MSC, HLR and EIR. It is not assumed however, that the SGSN is significantly more endangered via such connectivity.

The following table shows the assessment resulting from the above considerations.
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control plane)</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user plane)</td>
<td>2</td>
<td>1</td>
<td>1 - 3</td>
<td>5 - 6</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c-plane)</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u-plane)</td>
<td>1</td>
<td>1</td>
<td>1 - 3</td>
<td>1 - 3</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>2</td>
<td>2 - 3</td>
<td>5</td>
<td>20 - 30</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 10: Risk Assessment for the SGSN

4.3.1.4 ePDG

The following picture shows the embedding of the ePDG into the 4G mobile network.
The ePDG demarcates the border between an EPC and an untrusted non-3GPP access network. (Whether an access network is "trusted" or "untrusted" is defined by the operator; it is not a property of the access network itself.) The interconnection is done on the IP layer. User equipment interconnecting to the EPC via an untrusted access network must use IKEv2 to establish an IPsec tunnel towards the ePDG, and must send all traffic via this IPsec tunnel. When establishing this tunnel, the UE is authenticated using EAP-AKA', where the ePDG is the authenticator and the 3GPP AAA server is the authentication server. (EAP-AKA' makes use of the shared secret on USIM/AuC, i.e. the 3GPP AAA server interacts with the HSS to get authentication information.)

PMIPv6 is used to support mobility for UEs using non-3GPP access networks. The ePDG acts as MAG (Mobile Access Gateway), the PDN-GW acts as LMA (Local Mobility Anchor).

For support of certain roaming scenarios, the ePDG in the visited network may interface a PDN-GW as part of an SAE-GW in the home network, i.e. the interface may be an inter-operator interface.

The ePDG could contain a PCEF, but the interface between ePDG and a PCRF is not yet specified by 3GPP (see 4.3.6.2 for PCRF and PCEF).

Threat and Risk Assessment for the ePDG

At the border of the EPC towards untrusted access networks, the ePDG is exposed to attacks via such networks. However, only IKE/IPsec must be processed at the (external) interface to an untrusted access network – other traffic can be discarded. The traffic arriving inside IPsec tunnels is only forwarded towards the PDN-GW but not otherwise processed.

The other interfaces of the EPC (towards the 3GPP AAA-server or –proxy and towards the PDN-GW), are internal core network interfaces (always within a single PLMN) and cannot easily attacked by external attackers. Moreover, the operator may use IPsec protection in particular for the Diameter based AAA traffic.

In certain roaming scenarios, the ePDG may also interface a PDN-GW that is in a different PLMN.

The impact of successful attacks on an ePDG, in particular if an ePDG can be compromised, is considered high, as this would open a hole in the perimeter of the mobile network that may be used for theft of service and various kinds of attacks, e.g. on the PDN-GW.

As discussed in 4.3.1.1, for user plane traffic or data, there is no valid estimation of the impact of loss of confidentiality or integrity (from the user's point of view). User plane traffic is however only endangered during transit – user plane data is not stored on an ePDG.

The following table shows the assessment resulting from the above considerations.
### Threats Likelihood Vulnerab. Factor Impact Risk

<table>
<thead>
<tr>
<th>Threaths</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>48</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control plane)</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user plane)</td>
<td>2</td>
<td>2</td>
<td>1 - 3</td>
<td>4 - 12</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>3</td>
<td>1</td>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c-plane)</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u-plane)</td>
<td>1</td>
<td>2</td>
<td>1 - 3</td>
<td>2 - 6</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>50</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2</td>
<td>4</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

**Table 11: Risk Assessment for the ePDG**

### 4.3.1.5 3GPP AAA-Server or –Proxy

Figure 35 in 4.3.1.4 shows the 3GPP AAA-Server embedded into the mobile network. It acts as the authentication server for EAP based non-3GPP access authentication. It interacts with the HSS to retrieve the required authentication information and to support user profile and location management.

When performing EAP based authentication for trusted access networks, the AAA-Server interacts with some entity within the trusted access network serving as the authenticator. For untrusted access networks, the ePDG acts as authenticator (see 4.3.1.4). (Note that there is also the option to perform an authentication with an entity in the untrusted access network acting as the authenticator. However, this may only be useful to authenticate users that want to use the access network without using the 3GPP core and service networks, e.g. for direct Internet access.)

The AAA server also interacts with the PDN-GW to exchange control information in case of non-3GPP access. In case of non-3GPP access with terminal based mobility support (i.e. MIPv4 or DSMIP), the PDN-GW also acts as EAP-Authenticator, with the AAA-server as authentication server.

In roaming scenarios, EAP-authenticators in access networks or the visited core network use a so called “3GPP AAA-proxy” in the visited network. The 3GPP AAA-proxy interfaces the 3GPP AAA-server in the home network, which in turn retrieves authentication and user profile information from the HSS. An AAA-Server in one PLMN may act as 3GPP AAA-server for authenticating subscribers of this PLMN and as 3GPP AAA-proxy for authenticating subscribers of other PLMNs roaming in this PLMN.
All communication of the 3GPP AAA-server/proxy is based on Diameter, using 3GPP specified Diameter applications.

Threat and Risk Assessment for the 3GPP AAA-Server/Proxy

The AAA-server/proxy interfaces only trusted network components (assuming that the option to perform authentication with an entity in an untrusted access network is not used). It may interface to an AAA-server/proxy in another PLMN, however.

The AAA-server/proxy supports only Diameter, with some specific Diameter applications. In particular, it is not exposed to the user plane traffic. The Diameter traffic may also be protected by IPsec.

The AAA-server/proxy handles sensitive data, including user authentication and profile information. Compromise of an AAA-server/proxy would have a significant impact, as it may allow unauthorized access to the services provided by the operator, and may be further abused to attack the HSS (e.g. request authentication information for arbitrary users).

The following table shows the assessment resulting from the above considerations.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>T3 Eavesdropping</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>2</td>
<td>1</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T5 Traffic modification</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20 - 30</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 12: Risk Assessment for the 3GPP AAA Server/Proxy

4.3.2 Subscriber Management

This section covers the HSS as well as the EIR.

4.3.2.1 Home Subscriber Server (HSS)

The HSS is the central database for user information (identification, addressing, location, security information, other profile information).
The most sensitive HSS part with respect to security is the Authentication Centre (AuC). For each subscriber, it stores a shared secret, i.e. a key. This key is also stored on the USIM. It is the basis for user authentication (mutual authentication between user and network except in the 2G case).

Another part of the HSS is the Home Location Register (HLR). It is accessed by core network entities like MME, SGSN, 3GPP AAA Server or MSC(-Server) to enable subscriber access to the network. For this, the HLR accesses the security information provided by the AuC.

Further, the HSS supports the control functions within the IMS; it is needed to enable usage of the IMS services by subscribers. For this, the HSS is accessed by the CSCF and by SIP application servers that are part of the IMS.

IMS and EPC entities perform access to the HSS via diameter, 2G/3G SGSNs and MSC use the SS7 protocol stack either over TDM/ATM or over IP (SIGTRAN). If needed, IP based access to the HSS should be protected using IPsec. At least IPsec integrity protection is specified as mandatory by 3GPP, if different security domains are involved (e.g. MME and HSS are in different PLMNs).

Threat and Risk Assessment for the HSS

There is no doubt that the availability and integrity of the HSS and the confidentiality of the data it holds are of essential importance for the mobile network. It is assumed that typically, security controls for the HSS are on a relatively high level (as compared to other areas of the same network).

In contrast to core network elements like SGSN, SAE-GW or MME, the variety of functions and the number of supported protocols is substantially lower at the HSS, reducing the potential for exploitable weaknesses.

Attacking the HSS directly from external networks seems hardly possible. Rather, an external attacker would have to compromise another core network element, like an MME or SGSN, or a SIP server within the IMS to be able to abuse it for attacking the HSS. Otherwise, it may be possible to use a large set of mobiles (e.g. a mobile botnet) to generate an overload condition at the HSS. However, such attacks are not yet known.

Taking this into account, the assessment shown in the following table can be derived.
### Threats, Likelihood, Vulnerability Factor, Impact, and Risk

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T3 Eavesdropping</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T5 Traffic modification</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>2</td>
<td>2 - 3</td>
<td>5</td>
<td>20 - 30</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

*Table 13: Risk Assessment for the HSS*

#### 4.3.2.2 Equipment Identity Register (EIR)

The EIR can be used to store the status of IMEIs (International Mobile Equipment IDs). In particular, IMEIs may be blacklisted in the EIR, meaning that the respective terminal is excluded from the usage of the mobile network. Note that this is different from excluding a certain subscriber – subscribers are identified via the IMSI (International Mobile Subscriber Identification) stored on the USIM.

The EIR is an optional component. An operator may go without it, meaning that there is no check of the status of terminals. Such an operator cannot offer the service to have stolen terminals registered and blocked. Note that the incentive for an operator to implement such a service may be low, as it has the potential to reduce revenues.

The EIR may be integrated with the HLR on a single platform.

### Threat and Risk Assessment for the EIR

Compared to the HSS, the EIR seems to be rather less attractive to attackers. The vulnerability may be at about the same low level.

The most important consequence off successful attacks on the EIR could be DoS, e.g. for terminals that have been falsely blacklisted in the EIR by an attacker. A more severe situation would be unavailability of the EIR as a whole, in particular if the mobility management procedures at components like MME or SGSN would be stalled by this. However, such procedures may be implemented in a way that in case of no response from the EIR after a reasonable time, the check is skipped. Also, the operator may have means to switch an EIR into an "always-answer-ok" mode to avoid blocking of terminals by malfunction of the EIR.
In the following table, Likelihood and Vulnerability Factor values have been taken over from the HSS-assessment. Only the impact has been estimated specifically for the EIR.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T3 Eavesdropping</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>T5 Traffic modification</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

*Table 14: Risk Assessment for the EIR*

**4.3.3 2G/3G PS Domain**

Similar to an EPC, the PS domain provides packet transport between access networks and IP service networks. The core components of the 3G PS domain are the Serving GPRS Support Node (SGSN) and the Gateway GPRS Support Node (GGSN). Compared with the EPC, the GGSN corresponds to the PDN-GW, while the SGSN corresponds to the MME with respect to the control plane and to the S-GW with respect to the user plane functions.

**4.3.3.1 2G/3G SGSN**

In a 2G/3G PS domain, there is no SAE-GW, but the GGSN. The SGSN connects to the GGSN, either within the same PLMN, or – in roaming scenarios – within another PLMN. Moreover, a pure 2G/3G SGSN will not interface MMEs, and it will not support EPC interfaces. The 2G/3G SGSN has fewer protocol options (at least on application layer – on the low layers it may support many protocols, e.g. IP, ATM, FR) e.g. it does not support diameter communication with the HSS, but only MAP based signaling with the HLR.

One can conclude, that on the level of the present document, no difference can be made between the risks of the 2G/3G SGSN and the SGSN in the 4G network, and the assessment in section 4.3.1.3 holds also for the 2G/3G SGSN.

**4.3.3.2 GGSN**

The GGSN terminates GTP based user traffic tunnels from the mobile terminals and routes user plane IP traffic between these tunnels and internal as well as external IP (service) networks. Simply put, it has the same role as the PDN-GW in the EPC, and therefore shares a lot of functions with it:
• RADIUS client
• DHCP relay agent (or DHCP client requesting IP addresses on behalf of mobile terminals)
• Policy and Charging Enforcement Function
• Client to an Online Charging System (Diameter Credit Control, RFC 4006).
• Client to an (Offline) Charging Gateway using GTP' (GTP prime)
• Lawful Interception

In contrast to the SAE-GW, the GGSN does not provide functions to support access and mobility via non-3GPP access networks.

**Threat and Risk Assessment for the GGSN**

The GGSN can be considered as a subset of the SAE-GW. Because it has fewer functions and supports fewer protocols, for some threats the vulnerability factor is somewhat lower as for the SAE-GW. However, in many cases, the vulnerability of the SAE-GW is already very low, so no difference can be made for the GGSN. Likelihood and impact are generally taken over from the SAE-GW assessment.

This leads to the assessment shown in the table below.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control plane)</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user plane)</td>
<td>2</td>
<td>2</td>
<td>1 - 3</td>
<td>4 - 12</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>3</td>
<td>1</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c-plane)</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u-plane)</td>
<td>1</td>
<td>2</td>
<td>1 - 3</td>
<td>2 - 6</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>3</td>
<td>2 - 4</td>
<td>5</td>
<td>30 - 60</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>3</td>
<td>2 - 3</td>
<td>5</td>
<td>30 - 45</td>
</tr>
</tbody>
</table>

*Table 15: Risk Assessment for the GGSN*
4.3.3.3 GPRS Tunneling Protocol for Control Plane (GTP-C)

GTP is used by many different elements inside mobile networks, like eNB, MME, SAE-GW, SGSN and GGSN, and is mentioned already in several sections above. This section describes a few general threats associated with the usage of GTP, or, more exactly, GTP-C that are not specific to single interfaces or network elements.

GTP-C is the control part of the GPRS Tunneling Protocol and uses the UDP port 2123 (assigned by IANA). Several control messages in the core network are delivered over GTP-C, for example session management, context information delivery and location information management. The protocol stack of GTP-C is the same as GTP-U (Figure 23).

Until now, 3GPP has specified two variants of GTP-C, based on GTPv1 and GTPv2. The specification of GTPv2-C can be found in [3GPP_TS29274] and updates the specification of GTPv1. Because of the downwards compatibility of the 3GPP architecture, both variants of GTP-C may be found in a provider’s network.

Because the SAE-GW physically may be one single component, it may be difficult to inject malicious packets at the S5/S8 interface (the interface between the two logical SAE-GW components Serving GW and PDN-GW, but there are other, accessible interfaces available. One example might be the Gn-Interface or S11 interface. The Gn-Interface is the connection between a 3G SGSN to the 4G PDN-GW, providing a mobile terminal access from a 3G access network to packet data networks via the 4G core. In context of evaluating security threats of the PDN-GW, this interface may listen for communication data, and therefore it might be possible for an attacker to inject malicious communication data. Another interface might be the S11 interface, which connects the MME to the Serving GW.

4.3.3.3.1 Attacks on GTP-C

The S11 interface transmits control information for session management and context information. It might be the main focus of an attacker who gets access to the core network. In this context we will focus on some messages (described in detail in [3GPP-TS29274]) that may be in the focus of an attacker and might be sent in a non regular context to get information or disturb the running system.

Path Management

Not only to find out if a tunnel endpoint is still alive, but additionally to build up the tunneled paths the first time, path management messages Echo Request and Echo Reply are implemented and used in GTP-C, acting with similar functions as traditional path management messages. An Echo Request message may be sent for each path in use and is responded by an Echo Response message. After receiving the response message, the node knows that the other node is still alive.

The tunnel management procedures, used to build up the GTP-U tunnel for the UE, might be even more interesting for attackers. A so called PDP Context is built up within the signaling process and sent from the SGSN to a GGSN as a part of the context activation procedure of an UE. In case of an already activated tunnel, the context information may change (for example during a location change of an UE). In this case, the SGSN sends an Update PDP Context Request message to the GGSN, but there may also be an Update PDP Context Request, sent from GGSN to SGSN, for instance, to re-negotiate QoS for a PDP context.

A Delete PDP Context Request shall be sent from a SGSN node to a GGSN node as part of the GPRS Detach procedure or the GPRS PDP Context Deactivation procedure, or from a
GGSN node to an SGSN node as part of the "PDP Context Deactivation Initiated by GGSN" (see [3GPP_TS29060]) procedure.

Perhaps, if an attacker could rebuild all valid entries, it might be possible to create an own PDP Context towards the PDN-GW/GGSN. Because the attacker is free to use Echo Request messages it might get information about network topology and send self-build PDP context's to a known GGSN. Furthermore it will be possible to send PDP Context Deactivation and terminate current session to disable/delete the context information.

Session Management

Concerning the session management procedures, tests with Create Session Request messages were made, which are used to set up a session (and therefore a GTP-U tunnel between eNodeB and PDN-GW). With the correct formed messages, an attacker can set up his own tunnel to possibly get access to a specific packet data network (over the PDN-GW).

Bearer Management

Another interesting procedure is the management of bearers (transport channels). This means the configuration, deletion and modification of bearer information at the Serving GW. Especially the Modification Bearer Request might be interesting for an attacker who could modify an active bearer information. He could disturb/modify the connection of customers by changing QoS parameters. With the right message, he can get higher priority, or, in a limited network, he might change the aggregated maximum bit rate or simply delete an active bearer located at the Serving GW.

Like attacks on S1AP, X2AP and GTP-U, also for GTP-C some implementation tests were made. All described attacks are possible, because there is no native authentication given by GTP. Like for GTP-U, the fuzzing tests of GTP-C showed that the implementations seemed to be robust against fuzzing attacks.

4.3.3.3.2 GTP in the Real World

Due to research of attacking 3G and 4G networks, some statistics about GTP-C speakers accessible over the internet could be made that were published in [Shmocon_MendeRey]. A tool called GTP-Scan\(^{35}\) was written, sending Echo Request messages to a target IP address. The tool sends the requests in all defined versions: GTPv1-C, GTPv1-U, GTPv2-C, GTPv2-U and GTP'.

According to standards, GTP-C communication should only be possible through security gateways as described as part of the Network Domain Security (NDS) IP mechanisms [3GPP-TS33210].

However, despite the fact that complying with the specification means not to be reachable over the Internet via GTP-C, the results of GTP-scan for GTP-C, as shown in the following table, show that many GTP-C speakers are reachable, which poses a relevant risks to the respective mobile networks.

---

\(^{35}\) GTP-SCAN is a tool testing GTP establishment; it can be found at https://www.asmonia.de
The reachability via GTP-U is not explicit forbidden in the specification, but will also lead to a higher risk of attacks. It should not be necessary to be reachable from the Internet, but again, GTP-scan revealed a huge number of GTP-U speakers, as shown in the following table:

<table>
<thead>
<tr>
<th>Regional Internet Registry</th>
<th>GTPv1-U</th>
<th>GTPv2-U</th>
</tr>
</thead>
<tbody>
<tr>
<td>AfriNIC</td>
<td>13809</td>
<td>13761</td>
</tr>
<tr>
<td>APNIC</td>
<td>585733</td>
<td>584156</td>
</tr>
<tr>
<td>ARIN</td>
<td>18348</td>
<td>18235</td>
</tr>
<tr>
<td>LACNIC</td>
<td>907736</td>
<td>907618</td>
</tr>
<tr>
<td>RIPE</td>
<td>1428574</td>
<td>1427899</td>
</tr>
<tr>
<td><strong>Summary</strong></td>
<td><strong>2954200</strong></td>
<td><strong>2951669</strong></td>
</tr>
</tbody>
</table>

*Table 17: Results of GTP-Scan for GTP-U*

Furthermore, a quick analysis of those addresses shows that a lot of GTP-Speakers were also listening to SNMP with the well-known community string *public*. That means that an attacker could possibly get access to internal addresses, internal routing tables, open ports, running processes, installed software (including the install date) and whatever is in the MIB.

### 4.3.3.3.3 Access Point Names

Once an attacker is in the core or backbone network of a telecommunication provider, the attacker may send several messages to find out certain information (like already described above). One piece of this information may be the Access Point Name (APN), which is a reference to a GGSN, registered in the network (this includes GGSNs that are not available to the public). The structure and role of the APN is defined in [3GPP_0303]. The APN defines to which IP network an UE is connected to when the context is established. Different APNs may be used e.g. for different corporate networks, and there may be an APN used to connect to the Internet.

The structure of an APN is composed of two parts as follows (see [3GPP_0303]):

<table>
<thead>
<tr>
<th>Regional Internet Registry</th>
<th>GTPv1-C</th>
<th>GTPv2-C</th>
</tr>
</thead>
<tbody>
<tr>
<td>AfriNIC</td>
<td>26</td>
<td>11</td>
</tr>
<tr>
<td>APNIC</td>
<td>81</td>
<td>97</td>
</tr>
<tr>
<td>ARIN</td>
<td>52</td>
<td>45</td>
</tr>
<tr>
<td>LACNIC</td>
<td>22</td>
<td>10</td>
</tr>
<tr>
<td>RIPE</td>
<td>129</td>
<td>94</td>
</tr>
<tr>
<td><strong>Summary</strong></td>
<td><strong>310</strong></td>
<td><strong>257</strong></td>
</tr>
</tbody>
</table>
Mandatory: The APN Network Identifier which defines to which external network the GGSN is connected to.

Optional: The APN Operator Identifier that defines in which PLMN GPRS backbone the GGSN is located.

An example may be “internet.asmonia.de”. Furthermore, to provide data roaming between GPRS-networks, the following name convention is used:

<operator-name>.<operator-group>.gprs

The use of an APN providing corporate network access is mostly restricted to specific subscribers identified by their IMSI’s. But, if an attacker may get an APN (e.g. due to bruteforcing) and the IMSI of a subscriber that is allowed to use the APN, he may get access to the normally prohibited network.

One tool to discover APNs in a GPRS backbone networks is apnbf, which guesses all available APNs in the network. For this, the tool sends PDP context request messages to the available SGSNs, including all possible variants for APN. If the APN is available, the SGSN will answer with a PDP context response message and if not, the session will be declined.

This makes it obvious that relying on the secrecy of APNs is not an effective protection concept and will pose a significant risk to networks reachable via these APNs.

4.3.4 CS Domain (2G/3G only)

The CS domain provides circuits interconnecting mobiles (via access networks) to each other or to terminals within ISDN/PSTN networks or within external VoIP networks. The user traffic is mostly voice. Voice may be transported using TDM technology, and, from 3GPP Rel 4, also IP. However, IP is not used by terminals for circuit switched voice service; it is only used between mobile network components.

Security issues within TDM based networks are mostly considered much less significant than in IP networks. For this reason, this document focuses on an IP based CS domain according to 3GPP Rel 4.

From this release, there is a separation of control and user plane. This means, the Mobile Services Switching Centre (MSC) is decomposed into MSC-Server (MSS), comprising the control plane functions, and Circuit Switched Media Gateway (CS-MGW), providing user plane functions.

The following picture visualizes the CS domain.

36 The APN bruteforce tool can be found at www.asmonia.de
The central control plane element in the CS domain is the MSC-Server (MSS). It uses SS7 based signaling protocols to communicate with various other entities, including:

- 2G/3G RANs
- the registers (HLR and EIR)
- other MSC-Servers
- the CS-MGW (media gateway control)
- optionally the SGSN

While SS7 may be transported using TDM or ATM, only transport over IP (SIGTRAN) is in the focus of this document. CS-signaling between mobile terminals and the MSC-S is not IP based and therefore out of the focus of this document.

The MSS may communicate with the MME of the EPC, as described in 4.3.1.2.

There can also be exchange of signaling traffic between the CS domain and an IMS. This signaling traffic is based on SIP. SIP may also be used for signaling between MSC-Servers, replacing the SIGTRAN/SS7 protocol stack.

Signaling traffic can be exchanged between control planes of different operator networks to create circuits interconnecting terminals in different operator networks.

The user plane of the CS domain is handled by the CS-MGW. In a typical deployment, access networks may be physically connected only to the MGW, but not to control plane components – in this case, the MGW "backhauls" the signaling traffic to the MSC-S. On the core network side, the MGW transports user plane traffic towards other MGWs, e.g. a MGW serving as a gateway to another network. MGWs are also used for interconnecting to the PSTN/ISDN, or to SIP based VoIP or multimedia networks, like an IMS. MGWs may perform certain forms of user traffic processing, like transcoding and provide additional services like announcements.
As mentioned above, the IP layer used to transport voice in the CS domain is always terminated by network elements – CS terminals do not use IP themselves. A MGW between the CS domain and the IMS may have to process user IP traffic on the IMS side – this is covered in section 4.3.5.

The CS domain network elements also provide IP based interfaces to support OAM, charging, and LI functions.

**Threat and Risk Assessment for the CS Domain**

CS domain entities used to communicate via TDM and have been much less in the focus of attackers than components in IP based networks. If based on IP, a CS domain may become a more attractive attack target, and also be more vulnerable than before. However, on the one hand, the knowledge necessary to mount successful attacks against network elements like an MSS or MGW is assumed relatively rare, as compared with knowledge about IP gateways or servers. On the other hand, the CS domain may still be sort of a walled garden, i.e. IP is used internally, but there is no IP interconnectivity to external networks – external peering uses TDM. In this case, only attacks from within the network are possible, e.g. via the IP based management interfaces.

If IP based external peering is used, the risk of the CS domain will be higher. This may however be somewhat compensated by usage of security devices like session border controllers that inspect for example all SIP messages traversing the network border, filter out malicious messages and – for outgoing messages – remove message elements that would reveal too much information about the internal network topology.

Assuming that external peering is either not based on IP or is suitably protected e.g. by usage of session border controllers, the risks to the CS domain can be evaluated as relatively low, as documented by the following table:
Threats | Likelihood | Vulnerab. Factor | Impact | Risk
--- | --- | --- | --- | ---
T1 Flooding an interface | 2 | 2 | 4 | 16
T2 Crashing a network element | 2 | 2 | 4 | 16
T3.1 Eavesdropping (control plane) | 2 | 2 | 3 | 12
T3.2 Eavesdropping (user plane) | 2 | 2 | 1 - 3 | 4 - 12
T4 Unauthorized data access | 1 | 1 | 2 | 2
T5.1 Traffic modification (c-plane) | 1 | 2 | 5 | 10
T5.2 Traffic modification (u-plane) | 1 | 2 | 1 - 3 | 2 - 6
T6 Data modification on a network element | 1 | 1 | 5 | 5
T7 Compromise via implementation flaw | 2 | 3 | 5 | 30
T8 Compromise via management interface | 2 | 2 - 4 | 5 | 20 - 40
T9 Malicious insider | 1 - 2 | 2 - 4 | 5 | 10 - 40
T10 Theft of service | 2 | 2 | 5 | 20

Table 18: Risk Assessment for the CS Domain

4.3.5 IP Multimedia System

The purpose of an IMS is to provide IP based multimedia services to the subscribers of a mobile network. The most basic service is a voice service. 2G/3G networks provide this service anyway via the CS domain, however, in pure 4G networks, no circuit switched service will be provided anymore, and voice will be handled by the IMS only.

With the IMS, a multitude of services may be offered, such as Multimedia Telephony, Presence Services, Instant Messaging, or Push-to-Talk services.

An IMS can comprise many different functional entities. It is beyond the scope of this document to discuss them in detail. Rather, a threat and risk analysis is done on a high level for the IMS as a whole.

The control plane of the IMS is based on SIP. It consists of a number of SIP servers or proxies, in particular the so called CSCFs (Call/Session Control Functions) and SIP-Application Servers. The media plane may comprise media servers that perform functions like transcoding or provide tones or announcements. Media gateways at the borders of the IMS provide interconnection to circuit switched networks like PSTN/ISDN or the CS domain of a mobile network.

IMS network elements interact with the HSS to retrieve subscriber information, with the PCRF for policy and charging control, and with charging systems to perform online or offline charging (see below). Obviously, the IMS network elements must support OAM functions and be interconnected to the operators operation and maintenance center. Also Lawful Interception functions are specified for the IMS.
In a mobile network, the IMS can be accessed by mobile terminals using the radio access network and the packet core (see Figure 1, page 9, and Figure 31, page 121). Alternatively, non-3GPP access networks may be used, including access via public hotspots and the Internet. IMS usage may also make sense for subscribers using a fixed access, without mobility support by the network operator.

As different levels of security may be provided by different access networks, 3GPP has specified access independent protection mechanisms for the IMS. IMS subscribers may be authenticated using an ISIM. The signaling between subscriber and IMS can be protected using SIP-digest, TLS or IPsec. RTP based media can be protected either end-to-end, or between terminal and the edge of the IMS, using SRTP. SIP and media interfaces within the core can be protected using IPsec or TLS.

At the borders of the IMS towards the subscribers and towards external VoIP networks, session border controllers may be used to provide perimeter security functions. However, such functions are not specified by 3GPP (as there is typically no requirement for interoperability of such functions with other network functions).

Such SBCs may act as so called back-to-back user agents, i.e. they do not just pass SIP messages but really terminate the incoming SIP protocol, evaluate each message, discard any message parts that are erroneous or must not be passed to the other side because of respective policies, and send a new, cleaned message towards the originally intended receiver. They may also perform functions like rate limiting or blocking of misbehaving SIP clients.

**Threat and Risk Assessment for the IMS**

The IMS handles end user traffic, even control traffic (SIP), and is thus pretty much exposed to attacks. SIP messages can be complex. The control plane may comprise different types of SIP application servers, meaning that it may be rather complex, with a significant potential for errors and security gaps.

The IMS media functions seem less vulnerable. Flooding is clearly an issue, but it is assumed that the available policy control and enforcement functions allow to mitigate this threat. While media may be cryptographically protected, this protection may not be seamless, i.e. full end-to-end cryptographic protection will mostly not be available. So there will be a certain vulnerability against eavesdropping. Purposeful modification of media on the other hand seems rather difficult.

It is assumed that a reasonable level of perimeter security control is available at the borders of the IMS.

In today's networks, the IMS may not be heavily used. In particular, the vast majority of voice calls is done via the CS domain, without need for the IMS. However, this will change in the long term. For the evaluation of the impact in this assessment it is assumed, that the IMS is an essential, heavily used part of the mobile network, but there is still also a circuit switched domain in the mobile network, and that the majority of users is able to make use of this domain for voice calls.

The following table shows the assessment resulting from these considerations.
Threats and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(I) - 1.0

Copyright © 2012 ASMONIA consortium. All rights reserved.
For example, as mentioned in 4.3.1.1, an SAE-GW uses GTP’ to interact with an OFCS and a Diameter Credit Control Application to interact with an OCS.

Offline charging ("postpaid") means collecting of charging information without real-time influence on the service. The collected charging information is transferred to a billing system where it is used for subscriber billing.

Online charging ("prepaid") means that authorization for the network resource usage must be obtained by the network prior to the actual resource usage. This authorization is granted by the Online Charging System upon request from the network.

Charging Systems may use various protocols to interact with the operator's billing systems, e.g. use FTP to transfer charging information from an OFCS to a billing system). Billing systems are considered to be part of the IT infrastructure of the operator organization and are not in the focus of this document. The same holds for inter-operator settlement of charges.

Charging systems are purely internal, they communicate only with other core network components. They use a restricted number of protocols for this communication, like Diameter or GTP’. No end user applications are running on charging systems. So the vulnerability factor is generally low. It is moreover assumed that the likelihood that an attacker has knowledge about a specific charging system and mounts an attack against it is rather low.
On the other hand, the impact of a successful attack can be very significant, as the operator may directly loose revenue, and may also break the law by wrong billing.

The following table shows the assessment resulting from these considerations.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flooding an interface</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Crashing a network element</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Eavesdropping</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>Unauthorized data access</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Traffic modification</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>Data modification on a network element</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>Compromise via implementation flaw</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>Compromise via management interface</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Malicious insider</td>
<td>1 - 2</td>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Theft of service</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
</tbody>
</table>

*Table 20: Risk Assessment for Charging Systems*

Notes:

T3, T5: Vulnerability Factor: Although cryptographic protection is possible for charging protocols, typical operators may not feel the need for it due to the internal nature of the traffic.

T4: Vulnerability Factor: Compared to threats like flooding or crashing, it seems particularly difficult to make a charging system leak sensitive data.

### 4.3.6.2 Policy and Charging Rules Function (PCRF)

The PCRF is the key component of the 3GPP Policy and Charging Control (PCC) architecture that allows applying policies (like QoS treatment) and charging rules to single IP flows. Figure 37 in 4.3.6.1 gives an overview how the PCRF is embedded into the mobile network.

The PCRF controls the Policy Enforcement Function (PCEF) within user plane entities like the PDN-GW, GGSN or (e)PDG. There may also be communication with a so called "Bearer Binding and Event Reporting" Function located in a serving gateway or an access gateway in a trusted non-3GPP access network.

The PCRF provides an interface that can be used by application servers to provide service information, like bandwidth and delay requirements.

In roaming scenarios, the PCRF in the visited network may communicate with the PCRF in the home network.
On all these interfaces, 3GPP specified Diameter applications are used that are transported over TCP or SCTP.

Concerning threats and risks, similar considerations hold for the PCRF as for charging systems, as attacks on the PCRF may influence charging, although maybe somewhat more indirectly. There is one difference however: The PCRF provides an interface to an "Application Function", i.e. application servers may communicate with the PCRF. This somewhat increases the exposure and thus the vulnerability factor with respect to various threats.

The following table shows the assessment for the PCRF, taking into account the assessment of charging systems and the increased exposure as described above.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>T3 Eavesdropping</td>
<td>1</td>
<td>3</td>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T5 Traffic modification</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>T6 Data modification on a</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>network element</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
</tbody>
</table>

*Table 21: Risk Assessment for the PCRF*

See Table "Charging Systems" for additional notes.

### 4.3.7 Security Gateway

The 3GPP security architecture specifies the function of a Security Gateway (SEG) in the context of the NDS/IP concept (see [3GPP_TS33210]). The function of the SEG is to terminate IPsec tunnels, in particular at the borders of so called security domains. The SEG function may be part of a network element like an SAE-GW, or it may be implemented as a dedicated network element. Simply spoken, such an SEG is an IPsec-VPN-Gateway.

A typical location where an SEG is deployed is the termination of the backhaul links from the eNBs, as IPsec support is mandated here by 3GPP standards ([3GPP_TS33401]). While in the context of NDS/IP, the SEG is focused on protecting control plane traffic, here, also user plane traffic is handled by the SEG.

Another likely location is at the border of an IP network interconnecting a PLMN with other PLMNs, e.g. a GRX – this is clearly also the border of the security domain of the specific PLMN, so according to [3GPP_TS33210], all control plane traffic via such an interconnecting IP network must be integrity protected using IPsec.
The SEG has some similarities with the ePDG discussed in section 4.3.1.4. However, an SEG need not support protocols like PMIP, and does not interact with network entities in other PLMNs like the ePDG does to support certain roaming scenarios. Also, a SEG typically does not interact with user devices. Therefore, the number of IPsec tunnels that are established at a SEG scales with the number of interconnected network elements, not with the number of supported subscribers.

As part of the H(e)NB architecture, 3GPP has specified a slightly different security gateway called SeGW (see [3GPP_TS33320]). The SeGW provides additional functions for the support of H(e)NBs. For example, if the H(e)NBs are equipped with hosting party modules (i.e. a module that – like a SIM card - holds credentials that uniquely identify the subscriber hosting the H(e)NB), the SeGW supports hosting party authentication via interaction with the AAA server and HSS (similar to what the ePDG carries out for subscriber authentication). In the following, no difference is made between the SeGW and the SEG, as the assessment does not differ on the level of detail chosen for this document.

In all cases, security gateways must at least provide integrity protection for control plane traffic. In some cases, more protection is mandated, e.g. confidentiality and integrity protection is mandated for all traffic towards the eNB or the H(e)NB.

A security gateway may be reachable from the Internet, or it may be reachable only from somewhat more protected networks, like a GRX.

The following table shows the assessment for the Security Gateway that results from these considerations. Obviously, the user plane traffic threats (T3.2 and T5.2) do only apply if the respective SEG handles such traffic.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control plane)</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user plane)</td>
<td>2</td>
<td>2</td>
<td>1 - 3</td>
<td>4 - 12</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c-plane)</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u-plane)</td>
<td>1</td>
<td>2</td>
<td>1 - 3</td>
<td>2 - 6</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>30 - 60</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

*Table 22: Risk Assessment for the Security Gateway*
4.3.8 Other Services and Functions

In addition to the functions discussed in the preceding sections, a number of additional services may be provided by the mobile network, like location services or (non IMS based) messaging services. To a significant part, such functions are deployed as parts of other network elements, like MME, SGSN or MSC. However, in some cases, also dedicated network elements are used. Such specific network elements or functions will be discussed in the following subsections.

4.3.8.1 Location Services (LCS)

A mobile network provides means to retrieve the geographic location of any mobile attached to the network. Different methods have been specified by 3GPP, and it’s up to each operator which of these to implement and use. Two classes of positioning methods can be distinguished:

- Control plane based methods, where control traffic between the mobile and access network elements is exchanged in order to perform the positioning.
- User plane based methods, where a user plane connection is established between the mobile and a server in the network which is used to exchange information facilitating the positioning.

Control plane based methods

Obviously, the network is always aware which cell a mobile is using. However, mostly better accuracy is required, which can be provided by access network specific methods. In 4G networks, the E-SMLC (Serving Mobile Location Center) manages the overall coordination and scheduling of resources required for the location of a mobile that is attached to E-UTRAN. It also calculates the final location and velocity and estimates the achieved accuracy. The E-SMLC interacts with the mobile in order to exchange location information applicable to positioning methods relying on functions inside the mobile (like comparing base station signal arriving times, or receiving GPS signals) and interacts with the eNBs in order to exchange location information applicable to positioning methods relying on functions inside the eNBs. An example of a positioning method is positioning based on the time difference of signals arriving at the mobile from the serving eNodeB and at least two neighboring eNodeBs.

Positioning is performed if it is requested by a so called LCS client, which may be part of a software application running on the mobile itself, or may be running on an external server that needs the positioning information to provide its service. Positioning is also relevant for network internal purposes like home location billing (i.e. specific tariffs apply when the mobile is at a specified home location). Positioning is also essential for emergency services.

The entry point for location requests from inside or outside the mobile network is the GMLC (Gateway Mobile Location Center). In theory, when an LCS client makes a location request, the network authenticates the client and verifies whether the client is authorized to do this request. It also verifies whether the affected subscriber allows the positioning to be done. In practice, such checks may not be executed explicitly for each location request. For example, it may be part of the subscription contract that subscribers agree to being located by the operator. In certain cases, this may be required by law, e.g. for the support of emergency services. Moreover, LCS authentication and authorization methods and policies may vary between operators.
If the performed checks do not fail, the request is passed to the E-SMLC which performs the positioning with the help of eNBs and/or the mobile and then sends back the required location information to the GMLC which passes it to the LCS client.

The following picture illustrates the LCS architecture for 4G networks.

![LCS Architecture Diagram](image_url)

*Figure 38: Location Services Architecture*

For more information on the LCS architecture of 4G networks see [3GPP_TS23271] and [3GPP_TS36305].

**User plane based methods**

The Open Mobile Alliance (OMA) has specified a method called SUPL (Secure User Plane Location). SUPL comprises a SUPL Location Platform (SLP), i.e. a server facilitating SUPL. A SUPL agent which may be located on the mobile or on some external server may send a location request to the SLP. The SLP authenticates the requesting SUPL agent. Between the mobile and the SLP a secure user plane connection using TLS is established, which is used to exchange the information required for the positioning. A typical piece of information that may be exchanged is the location data provided by a GPS receiver inside the mobile.

For more details on SUPL see [OMA_SUPLv3].

**Threats against location services**

Probably the most obvious threat is loss of location privacy, i.e. location information is leaked to an attacker. This may be effected either by eavesdropping of location traffic (T3), by retrieving location information out of network elements (T4), or by evading the authentication and authorization mechanisms at an interface where location requests can be done (T4). (Unauthorized access to location information on a mobile by a malicious application running on the mobile is also a valid threat, and may even be the most common attack on location privacy, but is not in the scope of the discussion in this section.)

Control plane traffic in the mobile networks is hard to intercept if it is protected as specified by 3GPP for the different interfaces. Encryption and integrity protection is standardized for the radio interface, the backhauling interface and also for core interfaces. SUPL traffic is
assumed to be protected by TLS. Extracting location information from eNBs, MMEs or the E-SMLC will probably require compromise of such network elements, which is also quite hard to achieve (see respective sections above). So the most promising target for attackers may be the GMPLS or the SLP. Here, flaws in the client authentication and authorization or general protocol vulnerabilities may be exploited to get unauthorized access to the location service. Also the traffic between such a server and an external LCS client may be somewhat more exposed to attacks as compared to the traffic on other LCS interfaces, as it may pass through untrusted networks.

Falsification of location data (T5, T6) is also a relevant threat, but we assume that it is less significant than attacks against the confidentiality of location information.

DoS attacks (T1, T2) or attacks aiming at compromise (T7-T9) of network elements performing the LCS functions may be possible – they are already covered in the respective sections above, except for the GMLC and/or the SLP. These two network elements are indeed significantly exposed to attacks, as they offer an interface to potentially malicious LCS clients on mobiles or external servers. At the same time, the impact of DoS or compromise attacks is probably the highest at GMLC and/or SLP, because the whole location service may be affected, i.e. either be blocked or owned by the attacker.

As location services may be relevant for billing (namely home zone billing), also theft of service by attacking location services may be possible. We assume however that this is not a very significant threat.

The following table shows the risk assessment for the location service that results from the above considerations. With respect to network elements, the table focuses on the GMLC and/or SLP as these are assumed to be the most exposed network elements and at the same time the most attractive targets for attacks.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>18</td>
</tr>
<tr>
<td>T3 Eavesdropping</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>18</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>18</td>
</tr>
<tr>
<td>T5 Traffic modification</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>T6 Data modification</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8-32</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

*Table 23: Risk Assessment for the Location Services*
4.3.8.2 Short Message Service (SMS)

The SMS\textsuperscript{37} is an important service provided by mobile networks. Its use ranges from casual social interactions to important business applications. Short messages are for example used to transmit passwords, like one-time-passwords for the authorization of online banking transactions ("mobile TAN"). Moreover, short messages can be used to transmit configuration data, multimedia content (like ring tones) or even firmware to mobiles. Short messages may be processed by mobiles without user interaction. (Such silent short messages are obviously a threat to mobiles if attackers succeed in sending them, but this is not in the scope of this section.)

The central function to implement the SMS is the SMS Service Center (SMSC). It stores and forwards messages between short message entities (SMEs), i.e. entities capable to send and/or receive short messages. Typical SMEs are the mobile terminals, but also servers outside the mobile networks, like web servers implementing SMS portals, i.e. web sites allowing users to send short messages to mobiles. Short messages from/to mobiles are transported in the control plane of the mobile network to/from the SMSC, which is connected to selected MSCs:

- SMS-G(ateway)MSC for delivery of short messages to mobiles and
- SMS-I(nter)W(orking)MSC for receiving short messages from mobiles.

3GPP has specified this interface, as well as the functions needed to transport the short messages through the mobile network. However, the SMSC and its various possible interfaces (to external SMEs or other SMSCs) are not covered by 3GPP specifications. A number of protocols exist e.g. for transmitting short messages from external SMEs to SMSCs, e.g. SMPP (Short Message Peer to Peer).

In 2G/3G networks, if the mobile is attached to the CS domain only, short messages are transported via the MSC to which the mobile is attached, using CS radio channels on the radio interface. If the mobile is also attached to the PS domain, the transport via SGSN and a PS radio channel is preferred because of higher efficiency. For 4G networks including a CS core, 3GPP has specified an interface between MSC and MME and a procedure to transmit short messages via the MSC and MME (i.e. using the control plane of the EPS).

Since 3GPP release 7, it is also possible to transmit short messages via the IMS. For this a IP-SM-GW (IP Short Message Gateway) interconnects on the one side to the SMS-GMSC/SMS-IWMSC, on the other side to the IMS control plane, i.e. one of the SIP servers inside the IMS. With this setup, UEs (mobile as well as fixed terminals) attached to the IMS can send and receive short messages via SIP messages, and need not be connected to a 3GPP access network at all (the UEs may connected to the IMS of the mobile network using a non 3GPP access network, e.g. the Internet).

The following figure illustrates the network architecture for the SMS.

\textsuperscript{37} Note that in this document, the abbreviation SMS stands for short message service and is NOT used to denote a short message transmitted via the SMS.
Note that the figure depicts components that may be outside the domain of the mobile network operator domain in blue. A mobile network is likely to comprise an SMSC and also SMEs that are not mobiles, but servers operated as part of the mobile network. However, mobile networks will mostly also connect to external SMSCs and SMEs to provide to the subscribers SMS to and from external networks also.

Note that 3GPP specifications do not cover the SMSC, SME, and the interface between them. This means that implementations will be proprietary, including the security mechanisms used to protect these components and their communication. Although many well known security measures exist that might be applied, we still assume that the absence of standards in general may lead to less secure systems in this area. In the following assessment, we assume an SMSC that is part of the mobile network, with the network operator aware of the security issues of the SMS.

**Threats against the SMS**

In the following, we use the generic threats to assess the various risks to the SMS.

Flooding an Interface (T1) is mainly applicable to the interfaces of the SMSC towards an external SME. (Flooding of other interfaces using the SMS is clearly possible, e.g. the radio interface, but this is covered in other sections already.) Typically, the SMS is charged, so we assume authentication of external SMEs and authorization of delivery requests, as well as overload control, which reduces the vulnerability against flooding.

Such measures will also somewhat reduce the vulnerability against attacks aiming to crash the SMSC (T2), although this threat is clearly relevant at this external interface, in particular if it is visible in the Internet, which may well be the case.

Eavesdropping short messages during transport (T3) is assumed to be difficult at the mobile network interfaces, if the protection mechanisms standardized for the various interfaces are used. The traffic could also easily be secured at the external interface of the SMSC, but this might be neglected, as no standards exist here. Access to short messages while they are
stored in the network (i.e. at the SMSC – other network components need not store messages for significant time intervals) seems to require either successful impersonation of the true receiver, which is difficult, or a serious flaw at the SMSC – which may be the more likely one of these two possibilities. In any case, it is primarily the user that is affected, not the network.

For the modification/faking of short messages during transit or on the SMSC (T5 and T6), similar considerations hold as for T3 and T4. However, it seems somewhat less attractive for an attacker to modify short messages, because faking short messages, like sending messages with faked sender id seems to be easy when using external interfaces to the SMSC, as in many cases, arbitrary strings are accepted at these interfaces as the sender id.

The threat of an SMSC being compromised (T7, T8, T9) seems quite real – we assume that vulnerabilities may well exist, and controlling an SMSC will give an attacker many options, like sending free short messages, distributing SPAM, or attacking mobiles with messages containing malicious configuration changes etc.

Theft of service in the context of SMS is assumed to mean uncharged usage of the SMS. It may most likely be achieved via vulnerabilities of the SMSC.

The following table summarizes these considerations.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>36</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>T3 Eavesdropping</td>
<td>3</td>
<td>2</td>
<td>1 - 3</td>
<td>6 - 18</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>3</td>
<td>2</td>
<td>1 - 3</td>
<td>6 - 18</td>
</tr>
<tr>
<td>T5 Traffic modification</td>
<td>1</td>
<td>2</td>
<td>1 - 3</td>
<td>2 - 6</td>
</tr>
<tr>
<td>T6 Data modification</td>
<td>1</td>
<td>2</td>
<td>1 - 3</td>
<td>2 - 6</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>45</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>45</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>45</td>
</tr>
</tbody>
</table>

*Table 24: Risk Assessment for the Short Message Service*

### 4.4 Application Servers

This section covers OAM servers and - as an example of user application servers – web proxies.

#### 4.4.1 OAM Servers

Operation, Administration and Maintenance of a mobile network is typically a highly complex task, as a mobile network is a highly complex system of a multitude of network elements of
various different types. Rather than a single network element management system, mostly a variety of different OAM servers is required, many of which are dedicated to perform OAM functions for a specific type of network elements only. Configuration management, fault management, performance management etc. may be combined for a set of network elements on the same management platform, but in addition, specific servers may be needed for tasks such as logging or backup and restore.

Typically, OAM servers are deployed on specific sites, called Operation and Maintenance Centers (OMCs), which are connected to all the different network sites and allow remote OAM for the complete network. This is mostly complemented by locally deployed OAM devices that may be needed for specific tasks like initial installation of systems.

Besides the interconnection to the managed network elements, OAM servers may also be interconnected to higher level management systems, like business support systems. In the context of the present risk assessment, such interconnectivity is not taken into account, however.

It is recommended security practice for mobile networks to provide dedicated network resources for the OAM network. In the core area, the OAM network may even be physically separated from the user and control plane networks. On the backhauling link towards the (e)NBs or 2G base stations, this is mostly not feasible, so there is only a logical separation of OAM traffic and other traffic types.

It is also highly recommended to use cryptographical protection of OAM traffic end-to-end between OAM servers and network elements. Many protocols are available for this, on different network layers, like IPsec, TLS, SSH, SFTP, HTTPS, SNMPv3 etc. However, there is a variety of different management protocols, and a variety of network elements, including “legacy” equipment that has been developed years ago, when security was not yet perceived as an essential property by many equipment providers and operators. Moreover, errors in the implementation of secure protocols may be exploited to break the protection. Therefore, it must NOT be assumed that all management protocols are truly secured.

Another recommended practice is to protect OMCs by additional firewalls against the network elements, to prevent that a network element that has been compromised by an attacker subsequently successfully attacks OAM servers. However, it cannot be assumed that such protection covers all possible connection and protocols and that it is, where available, flawless.

The impact of successful attacks on OAM network elements, in particular compromise, can be devastating. The network operator may lose control not only over single network elements, but over large parts of the network.

The following table shows the assessment resulting from these considerations.
<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T3 Eavesdropping</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>18</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>18</td>
</tr>
<tr>
<td>T5 Traffic modification</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>T6 Data modification on a network</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>T7 Compromise via implementation</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T8 Compromise via management</td>
<td>3</td>
<td>2 - 3</td>
<td>5</td>
<td>30 - 45</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1 - 2</td>
<td>2 - 4</td>
<td>5</td>
<td>10 - 40</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
</tbody>
</table>

**Table 25: Risk Assessment for OAM Servers**

### 4.4.2 Web Proxies

A number of operators have implemented HTTP ("web") proxies as part of their 3G/4G infrastructure. Those proxies mainly serve to cache HTTP-based content that is frequently accessed (by different users) and thereby preserve bandwidth needed to get this content from the Internet while at the same moment contribute to a better user experience by fulfilling requests (for the cached content) in a faster way. At times and depending on the exact service the operator provides (e.g. some offer "network based malware protection", usually for an extra fee) the proxies constitute centralized points of storage where scanning for malware can be performed.

**T1 Flooding an interface**

Attacking a web proxy serving user requests would mostly cause bad user experience (and - maybe - potentially subsequent loss of customers to the operator) but would not cause direct impact on the operator's core, revenue-generating services (that is still voice [services] – although this may change in the future – maybe even in the near future). Hence DoS attacks against this type of devices can rarely be observed which leads to a likelihood estimate of "2". Given the long existence of HTTP proxies in various networks both the underlying OSs (usually Linux or FreeBSD) and the application part of these components usually can cope well with high packet load (hence vulnerability factor estimated as "2"). In case of a successful attack no confidentiality or integrity violations would occur and most of the operator's core services would be not impacted, which gives an impact estimate of "2".

**T2 Crashing a network element via a protocol or application implementation flaw**

The same rationale as for T1 applies here. The likelihood is estimated as "2", the vulnerability as "2" (as the IP stacks and HTTP processing parts can be considered to be mature and stable) and the impact as "2".
Eavesdropping

Eavesdropping on cached (and subsequently delivered) HTTP traffic would possibly not be performed in the environment of the web proxies but in other parts of the networks that are in closer proximity to the user(s) to eavesdrop on. Therefore these attacks rarely occur in current operator networks (likelihood: "2"). Large parts of the traffic in question are unencrypted, but on the other hand an attacker would need access to the traffic path of the proxies which can be considered unlikely (overall estimation of the vulnerability factor "2"). As HTTP traffic served to users usually is public anyway the impact would be small ("2") as well.

Unauthorized access to sensitive data on a network element via leakage

The vast majority of data processed and delivered by the web proxies is public anyway so there's only a very small ("1") likelihood an attacker will try to get hold of the data stored on web proxies. Under certain circumstances (e.g. knowing an exact path or directory) it might be possible to retrieve some data as in general web proxies do not implement strict access controls. This leads to an estimate of the vulnerability factor of "3". Given the nature of the data the impact would be very small ("1").

Traffic modification

Modifying parts of the outbound network traffic of a web proxy (that is the traffic leaving the proxy, potentially destined to users) might serve as an attack vector for large scale malware spread (e.g. if an attacker can modify frequently accessed files like an update of a popular desktop component). This type of attack does actually happen in operator environments (likelihood "3"), however for the attack to be successful an attacker would need access to the traffic path of the proxies (see also discussion of T3). It should be noted that there's no integrity protection in place for large parts of HTTP based traffic (namely executables) which leads to an overall vulnerability factor of "3".

The impact could be severe, not so much for the operator itself but potentially for a large number of users (whose systems/terminals, being parts of a then-botnet, might in turn attack the operator). Therefore, the impact is rated as "4".

Data modification on a network element

This threat can be compared to T5 "Traffic modification". We estimate the difficulty of getting ("write") access to a web proxy to be roughly the same as to get into the traffic path so overall the same values can be applied.

Compromise of a network element via a protocol or application implement. flaw

In the last years only few vulnerabilities leading to a system compromise have been discovered in current TCP/IP stacks and HTTP processing entities (a notable exception might be the so-called Socksstress vulnerabilities disclosed in 2009). The likelihood of associated attacks is thus estimated medium ("3") taking into account this type of attack still happens, albeit not too often. While the web proxies’ IP stacks and HTTP processing parts can be considered to be mature and stable those systems are publicly reachable, at least to some degree (attacker might initiate a data connection from a terminal, fetching data from an attacker-controlled system which is then processed by an intermediate web proxy). This gives an overall vulnerability factor of "3". A successful compromise might not directly impact revenue-generating services of the operator but it would enable an attacker to modify the delivered data (see above) which leads to an impact estimate of "4".
**T8  Compromise of a network element via a management interface**

As web proxies deployed in operator environments usually run common-of-the-shelf operating systems (mostly Linux or FreeBSD) they are managed with the associated mechanisms (e.g. by SSH) and according to widely available Best Practices, including (IP-) restricted management access. Certainly attacker try to compromise these systems (likelihood "3"), however given the overall operational maturity they rarely succeed (hence vulnerability factor "2"). If successful the same considerations as above in the section on T7 apply (impact "4").

**T9  Malicious insider**

Probably a malicious insider would go for other goals and targets than web proxies (e.g. will perform billing fraud or redirection by LI interfaces [see so-called Athens affair] or sth.). Therefore we estimate a very small ("1") likelihood for most operator environments with regard to this asset. Still it should be noted that due to the underlying operating systems' architectures (with usually insufficient auditing and logging mechanisms) such systems are vulnerable to malicious insider attacks (therefore vulnerability factor rated as high, "4"). As for the impact the same considerations as above in the section on T7 apply (impact "4").

**T10  Theft of service**

Usually web proxies are not attacked by attackers going after theft of service. Those attackers go after weakly protected entry points (VoIP Gateways etc.). The likelihood for this asset is thus estimated to be very small ("1"). The vulnerability factor is considered to be very small as well as web proxies usually cannot be abused for this goal. Usually revenue-generating services will not be affected which gives an impact estimate of "2".

The following table shows the assessment resulting from the above considerations.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>12</td>
</tr>
<tr>
<td>T3 Eavesdropping</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>T5 Traffic modification</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

*Table 26: Risk Assessment for Web Proxies*
4.5 Network Infrastructure

This chapter focuses on network elements that are part of the network infrastructure, i.e. the IP networks interconnecting the platforms implementing the functions specified by 3GPP. These components are not standardized by 3GPP. The network infrastructure may comprise various different components. Two important and characteristic examples are discussed in this section: Backbone routers and DNS servers.

4.5.1 Backbone Routers

This type of devices provides basic (IP based) network connectivity and thereby connects EPC components with each other. It is assumed that such a device uses Multiprotocol Label Switching (MPLS) as operators may – at least, to some (growing) degree – use a common infrastructure for different services. The most important security objective (from a “4G mobile network point of view”) will be availability. Still, the integrity of such a device might be important as well (as, for example, modifying a device’s configuration may lead to [4G] traffic redirection). Furthermore the confidentiality of some types of processed data might be endangered in case of eavesdropping on traffic on such a device. This applies particularly to 4G network traffic not protected by cryptographic means, including GTP and potentially some protocols used for OAM (like Telnet or community-based SNMP).

In the following, the threat categories specified in chapter 3 are discussed for a sample backbone router.

T1 Flooding an interface

Such devices are frequently exposed to various attacks by incoming packet floods (hence a “4” for likelihood). On the other hand given their overall importance, and being long-existent in many networks, good operational (security) practice and mature packet handling capabilities can be expected (“2” for the vulnerability factor). Still, even a successful attack will probably only affect the (attack’s) ingress interface and subsequently potentially not the 4G-relevant parts (therefore impact: “3”).

In networks with a high IPv6 deployment rate probably the vulnerability factor would be rated higher due to yet-little-understood scaling effects of certain protocol aspects [e.g. NDP].

T2 Crashing a network element via a protocol or application implementation flaw

This happens less often than pure flooding (hence a “3” for likelihood), however if it happens successfully the impact is bigger than in case of flooding (means “4” as just availability will be impacted, not other security objectives).

Given stack maturity and operational practice with regard to this threat (e.g. infrastructure ACLs), a low vulnerability factor is considered.

In networks with a high degree of IPv6 deployment probably the vulnerability factor would be rated higher due to added protocol complexity.

T3 Eavesdropping

Obviously gaining access to (partially unencrypted [e.g. GTP within same the security domain]) 4G traffic seems an attractive approach for attackers (compared to e.g. compromising 4G network elements). So attack attempts can be expected in this space. However this would require that an attacker has already either compromised a device or has somehow access to network links that are usually quite well protected. Thus the likelihood
can be regarded quite small ("2"). On the other hand, the vulnerability is high as infrastructure devices usually do not implement integrity or confidentiality protecting mechanisms (due to performance penalties caused by those and the operational overhead associated with those mechanisms). That's why the vulnerability is rated "4". The potential impact is rated "1-3" for user plane traffic, as it can be expected that highly sensitive traffic should be (and usually is) protected by upper-layer mechanisms. The impact is somewhat higher ("4"), if sensitive control plane traffic is affected.

**T4 Unauthorized access to sensitive data on a network element via leakage**

Attacks in this space against infrastructure devices might be doable (namely by SNMP), but can rarely be seen in practice. Still those will happen more than once a year, thus the likelihood is estimated as "3". Overall we see a quite small vulnerability factor (hence "2") as there are not many vulnerable protocols and usually operational practice prevents them. A small ("2") impact is considered as well as the data extracted (for example routing information) might not be of much value for attacker, in particular in a 4G context.

**T5 Traffic modification**

See above at T3: this is potentially very hard to achieve in the discussed context. The likelihood is regarded even smaller than T3 (hence "1"). Overall the vulnerability factor is rated as medium as modifying information (routing protocol exchanges, MPLS labels) would require multiple layers or points to be modified to be successful. Still, if such an attack is successful for control plane traffic, this would have a very high impact. For highly sensitive user plane traffic, as in T3, additional protection is assumed, which restricts the impact to "1-3".

**T6 Data modification on a network element**

We rate a very small likelihood ("1") as this type of attacks usually does not work against infrastructure devices (given there is no user land applications and only very limited interaction capabilities at all) and there's very rare occurrences in the wild. The vulnerability factor is considered to be low ("2") as well due to the overall architecture. The impact might be severe as a successful attack might cause traffic redirection or DoS. It's probably very hard to cause large impact though (hence overall impact estimation "4").

**T7 Compromise of a network element via a protocol or application implement. flaw**

See above at T2: attackers regularly try to perform such attacks against infrastructure devices (therefore likelihood "3"), but due to various factors (protocol implementation robustness, general infrastructure protection approach) we see a low vulnerability factor ("2"). In case of a successful attack severe impact is to be expected to all security objectives.

**T8 Compromise of a network element via a management interface**

Probably most successful infrastructure compromises go back to this type of attack. So this is "the low hanging fruit every medium skilled attacker is after" with an associated likelihood of "4". The overall vulnerability depends on operational practice but can be considered low for most operators ("2" as weak configuration and public reachability must coincide). We expect a high impact if successful, for all types of traffic incl. unencrypted 4G traffic (GTP) or OAM access to the 4G infrastructure (e.g. by means of Telnet).

In case of a 4G infrastructure that is completely isolated from the Internet or other parts of the operator's infrastructure the likelihood might be smaller (e.g. "3").
T9 Malicious insider

“Malicious insider” is a rare, but serious threat. Probably a malicious insider would go for other goals and targets than infrastructure device access (e.g. will perform billing fraud or redirection by LI interfaces [see so-called Athens affair] or sth.). We estimate a very small likelihood for most operator environments with regard to this asset. On the other hand we consider a high vulnerability (given the simplicity of access methods and lack of trustworthy auditing mechanisms in most cases) and, obviously, potentially very high impact if successful.

T10 Theft of service

Usually pure infrastructure devices are not attacked by attackers going after theft of service. Those attackers go after weakly protected entry points (VoIP Gateways etc.). Likelihood for this asset is thus regarded to be very small (“1”). The vulnerability factor is expected to be small (“2”) as well as these devices usually cannot be abused for this type of attack. Still, the impact of this threat would be evidently very high for operators (leading to loss of revenue and potentially non-compliance with regulations).

The following table shows the assessment resulting from the above considerations.

<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1 Flooding an interface</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>3</td>
<td>2</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control plane)</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>32</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user plane)</td>
<td>2</td>
<td>4</td>
<td>1 - 3</td>
<td>8 - 24</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>T5.1 Traffic modification (c plane)</td>
<td>1</td>
<td>3</td>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>T5.2 Traffic modification (u plane)</td>
<td>1</td>
<td>3</td>
<td>1 - 3</td>
<td>3 - 9</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>30</td>
</tr>
<tr>
<td>T8 Compromise via management interface</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 27: Risk Assessment for Backbone Routers

4.5.2 DNS Servers

DNS servers can be regarded major infrastructure devices and in particular in 4G networks a number of services and functions operates by means of DNS names. Given their role obviously availability is the main security objective. Furthermore the integrity of their data is
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(I)-1.0

crucial as modifications might lead to large scale attacks like traffic redirection. Usually confidentiality is not an important security objective for DNS (servers and data).

In the following, the threat categories specified in chapter 3.2 are discussed for a sample DNS server.

T1 Flooding an interface
As DoS attacks against DNS servers have been quite popular for a long time, a certain occurrence rate of such attacks can be expected (at least once per year which gives a "3" for the likelihood). Some operators might use dedicated DNS servers for their core environments (not providing services for other parts of the network or for customers). Furthermore in general DNS servers can cope quite well with packet floods. The vulnerability is subsequently rated "2". In case of a successful attack the availability of the DNS service will be impacted (at least for the period of the attack), but due to factors like records being cached on resolving devices the overall impact is not rated higher than "3".

T2 Crashing a network element via a protocol or application implementation flaw
The most widely deployed DNS server software (ISC BIND) has been susceptible to various DoS attacks in the past, but in most networks attacks against DNS servers by means of malformed packets have only be observed quite rarely in the recent years. The likelihood is thus rated as "2". Given the overall maturity of the main DNS server variants (e.g. Microsoft DNS, ISC BIND, djbdns) the vulnerability is rated low (2) as well. As such an attack might require an affected system to be rebooted, the impact is rated slightly higher than in case of a flooding attack (see above).

T3 Eavesdropping
As DNS data mostly is public anyway, getting hold of it by means of eavesdropping is a very uncommon attack method (therefore likelihood = "1"). By default the data is transmitted in an unencrypted manner which would render it susceptible to eavesdropping attacks (if those were performed at all). As confidentiality of DNS data usually is not a main security objective, the overall impact of such an attack would be low ("2").

T4 Unauthorized access to sensitive data on a network element via leakage
There's a well known attack against DNS servers that abuses the (built-in and architecturally required) so-called “zone transfer” capability of DNS. If performed successfully an attacker gets hold of full DNS databases by just sending a certain request and might thereby identify systems to attack further or gain information about the overall infrastructure. This is a very common attack (likelihood = "4"). Most operators deny this capability to unauthorized systems though (therefore vulnerability rated "2"). If it still happens the impact is slightly higher than in the eavesdropping scenario as an attacker gets the full database (as opposed to some records).

T5 Traffic modification
If an attacker could modify DNS data in transit she might be able to perform traffic redirection or severely impact the availability of certain (core) services. This would require an attacker to be able to modify network traffic within core network segments which is hard to achieve which is the very reason why this attack cannot be observed frequently in typical operator networks (likelihood rated as "2"). Usually there's no integrity protection of DNS packets. On the other hand, again, the DNS traffic relevant for the correct function of 4G networks does
not cross network links that are easily accessible. The latter two factors combined lead to an overall vulnerability of "3". In case of a successful attack the impact would be high (reduced availability or redirection of some functions, which might still require other attacks to be performed in parallel, hence impact rated as "4").

**T6 Data modification on a network element**

While DNS in general includes some functionality for data to be updated over the network (see RFC 2136) this feature is (next to) never used in operator networks. Attacks in this space are very rare (likelihood "1"), the vulnerability is rated as "2" as the feature usually is not enabled (albeit if enabled usually does not require authentication). The actual impact would depend on the records to-be-modified. Overall, in case of a successful attack, it can be considered to be comparable to the "traffic modification" scenario.

**T7 Compromise of a network element via a protocol or application implement. flaw**

For DNS servers this threat is very much comparable to the threat "crashing a network element". Consequently the likelihood and vulnerability factor are rated in the same way. The impact of a successful compromise of a DNS server would potentially be very high for a 4G network.

**T8 Compromise of a network element via a management interface**

Given the overall high frequency of attacks against management interfaces such attacks will happen against DNS servers (as against all other types of devices as well), thereby likelihood is rated "4". Due to the overall importance of these devices and the fact they usually run on off-the-shelf operating systems (where security knowledge is widespread amongst operations personnel) the overall vulnerability to this type of attack can be considered low ("2"). Obviously the impact would be as high as in the "compromise via implementation flaw" scenario.

**T9 Malicious insider**

Probably a malicious insider would go for other goals and targets than DNS servers (e.g. will perform billing fraud or redirection by LI interfaces [see so-called Athens affair] or sth.). Very small likelihood is estimated for most operator environments with regard to this asset. We estimate a high vulnerability factor (given the simplicity of access methods and lack of trustworthy auditing mechanisms in most cases) and, obviously, potentially very high impact if successful.

**T10 Theft of service**

Usually DNS servers are not attacked by attackers going after theft of service. Those attackers go after weakly protected entry points (VoIP Gateways etc.). Likelihood for this asset is thus regarded to be very small ("1"). The vulnerability factor is expected to be small ("2") as well as DNS servers usually cannot be abused for this type of attack. Still, the impact of a successful attack in this space would be evidently very high for operators (leading to loss of revenue and potentially non-compliance with regulations).

The following table shows the assessment resulting from the above considerations.
<table>
<thead>
<tr>
<th>Threats</th>
<th>Likelihood</th>
<th>Vulnerab. Factor</th>
<th>Impact</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flooding an interface</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>18</td>
</tr>
<tr>
<td>Crashing a network element</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>Eavesdropping</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Unauthorized data access</td>
<td>4</td>
<td>2</td>
<td>3</td>
<td>24</td>
</tr>
<tr>
<td>Traffic modification</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>24</td>
</tr>
<tr>
<td>Data modification on a network element</td>
<td>1</td>
<td>2</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>Compromise via implementation flaw</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>Compromise via management interface</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>40</td>
</tr>
<tr>
<td>Malicious insider</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>Theft of service</td>
<td>1</td>
<td>2</td>
<td>5</td>
<td>10</td>
</tr>
</tbody>
</table>

*Table 28: Risk Assessment for DNS Servers*
5 Mobile Botnets – the Next Large Scale Threat to the Mobile Business

It is well known that many computers connected via fixed networks to the Internet are so called bots, i.e. are infected with a malware and via this malware controlled by malicious persons or organizations which abuse the set of infected computers, the botnet, to carry out a variety of attacks, in particular DoS attacks or mass distribution of spam. Now that an ever growing number of mobile terminals, the smart phones, are in fact mobile computers, botnets are no longer restricted to fixed networks. Attackers will try to turn smart phones into bots and thus establish mobile botnets (also called cellular botnets).

This chapter discusses mobile botnets, with the goal to understand the risks imposed by mobile botnets on the future mobile terminals and mobile networks.

5.1 Establishment and Operation of Mobile Botnets

In Chapter 4.1.2.4.8.2 we have seen how an end-user device can be infected by malware using known as well as new attacks. The proposed example attacks the Android smart phone operating system using the following steps targeting vulnerabilities of Android's permission model:

- Takeover the UI in order to trick the user into installing the malicious application of the attacker (cf. Chapter 4.1.2.4.8.2.1)
- Enable the automatic initiation of the malicious application after its installation. This makes sure that the attacker’s code will actually run. (cf. Chapter 4.1.2.4.8.2.2)
- Secure that the attacker’s application will also be running after the end-user’s device has been rebooted (cf. Chapter 4.1.2.4.8.2.3)
- Construct a bi-directional communication channel using the transitivity vulnerability of the permission model (as suggested, e.g., by Höberath et al.[Höberath_2011] by using the browser to access the internet and a custom URI scheme to direct incoming data to the attacker's application (cf. Chapter 4.1.2.4.8.2.4)
- Covertly jailbreak the end-user’s device by downloading jailbreak code (e.g., using publicly available jailbreak code for the respective device) via the previously established bi-directional communication channel (cf. Chapter 4.1.2.4.8.2.5)

Using the combination of these attacks allows to install arbitrary code on the end-user’s device, i.e., malicious code which could be botnet malware.

The botnet malware could be capable of accessing the SMS component of the system which can effectively be used to suppress incoming SMS. With the suppression deployed, SMS could be used as a covert communication channel between the command and control (C&C) server of the botnet the malware belongs to and the malware itself.

Of course there exist a variety of other way to facilitate a C&C mechanism for smart phone malware, e.g., using simple HTTP, IRC, or upcoming new forms of instant messaging available for smart phones.

Exemplary features of the malware might include:

- Sending premium SMS.
- Advertise the malicious app to other users from the contacts of the user.
- Obtain contact information of the user and the users in the contact database.
Monitor network traffic of the device for non-secured traffic in order to obtain credentials of the user.

Scam users into surrendering private (personal) information.

Nowadays a convenient way to spread malware is using the official markets of the different smart phone operating system or unofficial third party markets [Zhou2012]. As we have seen in Chapter 4.1.2.4 the spreading mechanism is less effective the more tightly the market is controlled. However, besides the official markets, there exist an abundance of alternative sources for software on the Internet. Typically, installing application from these "less trusted" sources entails disabling a security feature which prevents the users from doing so, i.e., on Android.

Figure 40: Distribution of Malware on Mobile Platforms

How many end users can be infected is hard to predict and would be of rather speculative nature. As we have seen in Chapter 4.1.2.4, the security features of the different operating systems, as well as the operating system as a whole, are unique in their potential vulnerabilities. Also, in the smart phone world there is no such thing as cross-platform, yet. Therefore, malware will typically affect only a single platform. According to Figure 11 on page 50, this could still be a very significant part of the set of all smart phones, in particular in the case of Android. Another aspect is the availability of infected devices for malicious activities. Here, it can be noted that smart phones tend to be always-on, and may therefore be more efficiently used as bots compared e.g. to privately owned PCs that may only be running a few hours per day. (On the other hand, smart phones can be assumed to have typically less processing capacity and less uplink bandwidth as compared to PCs.) However, although such considerations are relevant, they do not allow a reliable prediction on how many bots might be mounted in a future mobile botnet.

Figure 40 shows a very recent distribution of malware on current smart phone platforms. It becomes obvious that J2ME is the most popular platform to be attacked. More recent operating systems such as Android, iOS, and Windows Phone 7 are on a steadily rise. However, older phones running on Symbian, or phones using J2ME so sideload third party applications will still be around in the future for some time.

---

38 http://www.zdnet.com/blog/security/google-android-market-malware-problem-escalates/9001
5.2 Attacks Against End Users

As has been detailed in Chapter 4.1.2.4.8.2 infecting a device is the most important step for creating a botnet. Once installed, the malware could for example be used to steal MTANs (Mobile Transaction Numbers), an upcoming technology to provide two-factor authentication for doing online banking transactions by using SMS the user's phone to authenticate a current transaction. Recently, banking trojans have been discovered to gather MTAN related information.\(^\text{39}\)

Suppose the malware is part of a botnet whose purpose it is to gather MTANs and user credentials and phone numbers. This gathered information is then sent into the botnet for further processing. Once installed, the malware could use SMS to announce itself to the botnet it belongs to, obviously suppressing any notification to the user.

The botnet herder can now send transfer orders in the name of the infected user, assuming he has the necessary online banking credentials of the user. The malware can now suppress and redirect incoming SMS verification of the MTAN verification mechanism to the botnet and its herder - the user will not notice. The herder can than complete the transaction and the user has effectively lost money.

A similar attack is possible on Google accounts. Suppose a user with an Android smartphone. The user is very likely in possession of a personal Google account in order to use all the services to their full extent. In many cases, credit cards of the user will be directly connected to the account in order to purchase items, e.g., in the Google Marketplace. Recently Google introduced a two-factor authentication mechanism to their accounts which uses the user's smart phone for verification purposes.

Once the botnet herder is in possession of the account name of the user, he can initiate a password reset. Google will send a reset notification to the user's phone. This can be suppressed by the malware, as was the verification SMS in the MTAN case, and sent to the botnet herder. The herder can now use the reset to do the actual reset on the user's account. For some time the herder will be able to carry out transactions in the user's identity, possible inflicting large amounts of damage using the user's credit card which is associated to the Google Checkout service.

The above attacks do not necessarily require that there is a botnet in the sense of many infected devices. However, having many users that can be defrauded and the capability to do that in an automated way is a highly attractive goal for an attacker and justifies high efforts in developing and deploying the malware and the respective infrastructure.

5.3 Attacks Against Mobile Networks

5.3.1 Protection of Mobile Networks Against Malicious Mobiles

In general, a mobile network operator must not trust the mobile terminals but must protect the network against malicious behavior of terminals. However, typical subscribers do not have malicious intentions, or would be reluctant to execute attacks with their mobiles because they would fear to be identified and prosecuted. If however a terminal of a regular subscriber is controlled by an attacker, malicious behavior is much more likely, as the operator recognizes only the subscriber and typically has no way of identifying the true attacker. As a consequence of this increased likelihood of attacks, the risk for the network rises with the number of compromised terminals. The rise of this risk may be insignificant, if

the network is in general well protected against malicious behavior of mobiles, and the number of compromised mobiles stays small. If however the network has vulnerabilities that can be exploited by mobiles, then the risk can rise significantly when more and more mobiles get compromised by attackers.

Good protection against malicious behavior of mobiles may however not be sufficient, if the number of malicious mobiles becomes high, and if these mobiles can be controlled by a single attacker, as it is the case with botnets. The problem is, that even if each single bot performs legal actions only, these legal actions of all bots together may work as an attack on the network – that is to say, a DDoS attack.

5.3.2 Distributed Denial of Service (DDoS) Attacks

It is well known that DDoS attacks are hard to counter. In a not distributed DoS attack with only a few or even a single attack source, there is a chance to identify the attack sources and block them. If in a DDoS attack each single source produces only an amount of traffic or requests that also a regular device may produce, it is not easily possible to distinguish between attacking and well behaving devices. So there is no chance to lock out only the malicious devices, and DoS is inevitable as soon as the network is no longer able to handle all the requests. Clearly, suitable dimensioning of the network resources can mitigate this threat, but this approach is limited, because as a rule, a strongly over-dimensional network is not economically viable. (It may be possible however to boost the network performance temporarily during the time of the DDoS attack by taking advantage of elastic cloud resources – this approach is investigated in the ASMONIA project, see [ASMONIA_D32].)

DDoS attacks by mobile botnets may not be restricted to single mobile networks - several operators may be affected at the same time. While this increases the possible impact of the attack, it also opens up the chance to better fend off the attack by suitable cooperation of the different network operators. Such cooperation (with main focus on the detection of attacks) is a main topic of the research done in the ASMONIA project (see also section 2.1 of [ASMONA_D11]).

A DoS attack by a mobile botnet would most likely try to exploit potential bottlenecks inside the network. Such bottlenecks may include the radio interface of a cell, in particular the control channels on the radio interface, but also control plane components in the core network, in particular the HSS, or application services of the operator domain, e.g. the SIP servers within the IMS. The user plane resources of the core network are typically less limited than the control plane resources. So we assume it somewhat less likely that the user plane resources of the core network will be affected by a DoS attack of a mobile botnet, at least as long as the number of bots in the botnet is still low as compared to the totality of mobiles, e.g. less than 10 percent. We also assume that botnet malware would typically use the API to the baseband functions, e.g. to place calls or send short messages, but would not modify the baseband software itself, as the implementation of this software, in contrast to the API, is different on different devices.

Some potential bottlenecks that might be attacked by mobile botnets are discussed somewhat more detailed in the following.

5.3.2.1 Scenario 1: DoS Against the Radio Interface

Inherently, a radio interface is highly vulnerable to flooding (see also assessment in section 4.2.3.1). Prior research has shown that various attacks are possible against the GSM or the UMTS radio interface, not only by strong radio senders, but also by simple mobiles. In particular, control channels on the radio interface like the Random Access Channel that is
needed by any mobile in order to make any request, like placing a call or sending a short message (SMS), can easily be flooded by malicious mobiles (see e.g. [Sparar09]). Also dedicated control channels, needed e.g. to transmit the signaling for establishing of calls can be flooded, e.g. by sending or receiving many short messages. In [Traynor08], an attack is described where the short messages are originating from the Internet, but clearly, in a mobile botnet, the bots could be triggered to exchange short messages to a degree that heavily overloads the control channels and by this cause DoS for legitimate users in the affected cells.

The more widespread the botnet, the more cells can be affected by bots sending short messages. However, the messages can be sent to arbitrary mobiles, not only to bots, blocking also cells where no bots but mobiles receiving the short messages sent by bots are present. [Traynor08] shows how hitlists of valid telephone numbers of mobiles can be created, and how using such hitlists in an attack could take down the mobile network of a whole city or even a country. [Traynor08] shows how a HSS can be taken out by a DoS attack.

A mobile cannot communicate with the HSS directly. However, it can perform actions that cause messages to be sent to and processed by the HSS. One such action is changing the call forwarding settings. In [Traynor09], the authors estimated how many mobiles would be required to take down a HLR (essential part of the HSS). For this, they made measurements with real mobiles in real networks that revealed that a single mobile can submit new call forwarding settings about once per 5 seconds. Further, they modeled two types of HLRs, a low end one and a high end one. They exposed these two HLRs to a standard traffic mix as assumed to be generated by the activities of one million supported subscribers per HLR. They then simulated the attack by adding the malicious HLR transactions caused by the attacking mobiles changing their call forwarding settings once about every 5 seconds. They observed the decline in the throughput of the regular transactions in dependency of the number of attacking mobiles.

This resulted in the following figures:

- With the low end HLR, approximately 11,750 – 23,500 infected mobiles (depending on the initial load of regular traffic) would reduce the throughput of legal transactions by more than 90%.
- With the high end HLR, approximately 141,000 infected mobiles would reduce the throughput of legal transactions by more than 75%.

While it is out of the scope of this document to evaluate this work and these numbers in more detail, we still believe that they indicate a very significant truth: Mobile botnets acting against an HLR as described above may well be able to reduce the availability of the network...
significantly, depending on the size of the botnet and the dimensioning of the HLR. Current HLRs may not yet be prepared to thwart such attacks. So in the longer term, additional load control measures may need to be implemented to prevent such overload conditions.

5.3.3 Impact of Mobile Botnets on the MNO Business

Executing a successful DDoS attack and taking down parts or all of a mobile network has obviously very high negative impact on the operator business. However, it is less obvious how an attacker would profit from that. A competitor acting as an attacker would obviously profit, but we do not assume this to be a highly likely scenario, at least not in countries with reliable jurisdictions. Another criminal business model would be extortion of money – but it is largely unknown whether such criminal behavior could be successful, and it is out of the scope of this research work to investigate such matters.

But even if a botnet is not used for DoS attacks on the network, but rather to defraud and rob the owners of the infected mobiles, this will have significant impact on the operator business, if happening at a large scale. Obviously, users will be dissatisfied when being defrauded. Even if they have caused their problems themselves, like by jailbreaking their phone, installing apps from untrusted sources, or by accepting fraudulent terms and conditions without reading and understanding them. Users may tend to blame the operator, in particular if they have bought their smart phone via the operator. This can easily lead to increased user support efforts for the operator, to increased churn rates and to less usage of revenue generating operator services. Users may also refuse to pay their mobile services bills, e.g., if a malware has caused high costs by using premium services, and the operator may not be able to get back the money he may have already transferred to the premium service provider for providing the service.

In the long term, if fraud via mobile malware gets out of control, it may lead to a general decline in the trust in the reliability of mobile services and slow down the growth of the overall mobile business.
6 Conclusion

The assessments documented in chapter 4.1 show that mobile terminals will be increasingly endangered by a variety of threats. The vulnerability of mobile terminals against many of these threats is high, and successful attacks have often a high impact on the device. A successful attack on a single device, even when resulting in compromising the device and abusing it consequently to attack the network, has no significant impact on the network itself, assuming the network considers terminals as potentially malicious anyway and uses appropriate security controls against malicious terminals.

Although there are quite different types of terminals, there is a common trend: mobile terminals will become more and more powerful, but also more and more complex and therefore also more prone to flaws and security gaps. As we have seen in the previous sections, all platforms currently have a multitude of vulnerabilities. This is very likely to hold true for future revisions of these platforms. On smart phones, such vulnerabilities will be exploited by malware, which comes mostly in the form of malicious functions that are part of attractive applications which are installed by unsuspecting users. Successful compromise by such malware seems very easily possible, with severe impact on the user of the smart phone, whose data may be stolen and who may be impersonated in using expensive services or performing fraudulent and criminal actions.

In particular, malware will turn mobile devices into bots. Large botnets of mobile devices have not occurred yet, but the trend has become obvious. The mobile devices offer an always-on connection which is very useful for the steady availability of mobile bots. Once a mobile botnet has a sufficiently large size it may endanger not only the users, but also the mobile network. Countering the spreading of malware is a difficult task because of the variety of vulnerabilities and the possibilities to exploit them.

Concepts that allow operators or vendors to exhibit a considerable amount of control on mobiles may be helpful in keeping the devices secured. However, it is assumed that such concepts will never be perfect, and evading this kind of control will be possible. Users are expected to try to turn off such control functions, in order to make full and unrestricted use of the capabilities of their devices. So security concepts for mobile terminals continue to be a very relevant field for future research.

The risk assessments for network elements (documented in chapters 4.2 to 4.5) can be summarized according to different criteria. Here, we will give evaluations on a per threat and on a per asset basis. In both cases, we will compare the risk, which is the main target of this document, but also have a look at the vulnerability factor, which reflects the "security level" of a component, and is the obvious target for any efforts to reduce a risk (compare section 2.2.3).

There may be different ways to compute average risk values. Here, taking into account that risks are computed as a product of three values in the range from 1 to 5, the cubed arithmetic mean of the 3rd roots of the risk values has been computed as an average risk value. (In case of ranges, the upper border was selected.) However, it turns out that other approaches do not yield significantly different results with respect to the ranking of risks. It should be noted that the figures given in the tables below do not have an absolute meaning, but only have the purpose to derive a ranking. Moreover, due to the inherent inexactness in the risk assessment, the tables should be interpreted as showing the trend rather than a strict absolute order.
Ranking of the Threat Categories

The following table shows the ranking of the generic threat categories according to the risk each threat category is associated with, in descending order.

<table>
<thead>
<tr>
<th>Threat</th>
<th>Vuln</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>T8 Compromise via management interface</td>
<td>3,1</td>
<td>39</td>
</tr>
<tr>
<td>T9 Malicious insider</td>
<td>4,0</td>
<td>33</td>
</tr>
<tr>
<td>T7 Compromise via implementation flaw</td>
<td>2,4</td>
<td>26</td>
</tr>
<tr>
<td>T1 Flooding an interface</td>
<td>2,7</td>
<td>19</td>
</tr>
<tr>
<td>T2 Crashing a network element</td>
<td>2,3</td>
<td>16</td>
</tr>
<tr>
<td>T10 Theft of service</td>
<td>1,9</td>
<td>15</td>
</tr>
<tr>
<td>T3.2 Eavesdropping (user plane)</td>
<td>2,1</td>
<td>15</td>
</tr>
<tr>
<td>T3.1 Eavesdropping (control plane)</td>
<td>2,0</td>
<td>13</td>
</tr>
<tr>
<td>T4 Unauthorized data access</td>
<td>1,6</td>
<td>10</td>
</tr>
<tr>
<td>T5.1 Traffic modification (control plane)</td>
<td>1,8</td>
<td>10</td>
</tr>
<tr>
<td>T5.2 Traffic modification (user plane)</td>
<td>2,1</td>
<td>9</td>
</tr>
<tr>
<td>T6 Data modification on a network element</td>
<td>1,6</td>
<td>9</td>
</tr>
</tbody>
</table>

Table 29: Ranking of the Different Threat Categories According to the Risk

It shows that the threats causing loss of control of network elements (compromising by attackers or abuse be malicious insiders) pose by far the most significant risks to the mobile network. Note that the malicious insider threat is even exacerbated when including also malicious insiders at arbitrary positions within the supply chain (e.g. equipment developers implementing malicious functions like backdoors). This result is not too surprising, as the consequence of losing control can be loss of all three essential security properties: availability, integrity and confidentiality.

Next in the ranking are the DoS attacks (flooding and crashing), followed by theft of service, loss of confidentiality and loss of integrity. It can be noted, that a ranking according to the vulnerability factor would show some deviations from the ranking according to the risk. E.g., when looking at threat T5.2, one sees that the user plane traffic is of somewhat "medium" vulnerability against modification; however, the threat poses only a very low risk because of its low impact (on the network operator) and the low likelihood of attacks trying to modify user plane traffic.

The threat showing the highest vulnerability factor is the malicious insider threat (T9). Several measures may be taken to mitigate this threat. Many of these are of organizational nature rather. However, there are also technical measures which can help. E.g., to prevent malicious network operator staff from changing the functions of network elements by modifying the software on a network element, e.g. install additional programs, software integrity protection mechanisms can be used. This is one of the topics that are explored in the ASMONIA project (see e.g. [ASMONIA_D21]; further work on this area within the ASMONIA project is still in progress and will be published on http://www.asmonia.de).
Ranking of the Assets

The following table shows the ranking of the network elements (assets) of the mobile network according to the risk related to each network element, in descending order.

<table>
<thead>
<tr>
<th>Asset</th>
<th>Vuln.</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>HeNB (4G home base station, “femto-cell”)</td>
<td>3,8</td>
<td>30</td>
</tr>
<tr>
<td>Relay Node</td>
<td>2,8</td>
<td>28</td>
</tr>
<tr>
<td>Short Message Service</td>
<td>2,8</td>
<td>25</td>
</tr>
<tr>
<td>eNB (4G base station)</td>
<td>2,5</td>
<td>22</td>
</tr>
<tr>
<td>SAE-Gateway</td>
<td>2,3</td>
<td>21</td>
</tr>
<tr>
<td>IP Multimedia System</td>
<td>2,3</td>
<td>20</td>
</tr>
<tr>
<td>Operation and Maintenance Servers</td>
<td>2,3</td>
<td>20</td>
</tr>
<tr>
<td>GGSN (Gateway GPRS Support Node)</td>
<td>2,1</td>
<td>19</td>
</tr>
<tr>
<td>IP/MPLS Router (e.g. core site router)</td>
<td>2,7</td>
<td>19</td>
</tr>
<tr>
<td>DNS-Server</td>
<td>2,6</td>
<td>18</td>
</tr>
<tr>
<td>ePDG (evolved Packet Data Gateway)</td>
<td>2,0</td>
<td>17</td>
</tr>
<tr>
<td>PCRF (Policy and Charging Rules Function)</td>
<td>2,9</td>
<td>17</td>
</tr>
<tr>
<td>Location Services</td>
<td>2,4</td>
<td>15</td>
</tr>
<tr>
<td>Security Gateway</td>
<td>1,9</td>
<td>15</td>
</tr>
<tr>
<td>Web Proxy</td>
<td>2,5</td>
<td>14</td>
</tr>
<tr>
<td>Circuit-Switched Core Network Domain</td>
<td>2,3</td>
<td>14</td>
</tr>
<tr>
<td>Charging Systems</td>
<td>2,4</td>
<td>14</td>
</tr>
<tr>
<td>HSS (Home Subscriber Server)</td>
<td>2,0</td>
<td>13</td>
</tr>
<tr>
<td>MME (Mobility Management Entity)</td>
<td>1,8</td>
<td>12</td>
</tr>
<tr>
<td>SGSN (Serving GPRS Support Node)</td>
<td>1,8</td>
<td>12</td>
</tr>
<tr>
<td>HeNB-Gateway</td>
<td>1,7</td>
<td>12</td>
</tr>
<tr>
<td>EIR (Equipment Identity Register)</td>
<td>2,0</td>
<td>11</td>
</tr>
<tr>
<td>3GPP AAA-Server/Proxy</td>
<td>1,8</td>
<td>10</td>
</tr>
</tbody>
</table>

*Table 30: Ranking of the Different Types of Network Elements According to the Risk*

This ranking puts the HeNB on the first rank, as the component with the highest risk. A very significant risk is also associated to the other base stations, i.e. the relay node and the
regular ("macro") eNB. It may not be intuitive why exactly the base stations lead the ranking. But although for example a DoS attack on such a network element has only limited, local impact, compromise of a base station has impact on the overall network, as in 4G networks, base stations perform important security functions and are fully trusted by the core network. As on the other hand the vulnerability of base stations is assessed to be high, in particular for the HeNBs, where non-expensiveness may become an important criterion for operators in the future, and at the same time full physical access by attackers must be assumed, a high resulting risk is a fully logical consequence.

A very high risk is also related to the short message service (SMS). Although often used for social interactions of low importance, it is also applied to convey highly security and business relevant information. The SMS is typically accessible from the Internet and thus highly exposed to attacks. However, exactly the crucial component for the SMS, the SMSC that stores and forwards short messages and provides the interface to external networks like the Internet, is not covered by the 3GPP security standards, which may lead to implementations without a sound security concept.

A high risk is related to components that handle aggregated user plane traffic and are moreover exposed to external IP traffic, namely SAE-GWs, GGSNs or routers. Also the IMS, handling user traffic not only in the media plane, but also in the control plane (i.e. SIP signaling), has a significant risk.

OAM servers rank also very high – this is due to their central position and importance for the operation and control of the overall network.

Conclusion

The assessments given in chapter 4 of this document show that there are various threats that pose significant risks for current and future mobile communication networks and for mobile terminals. Future large scale threats like mobile botnets as discussed in chapter 5 will cause a further rise of the risk level. As the role of mobile communication networks as part of the telecommunication infrastructure is expected to grow considerably over the next years, security concepts, including improved attack analysis concepts, as explored in the ASMONIA project, are of vital importance for the security of future telecommunication infrastructures and all the systems and applications relying on them.
References

"Security Threats and Requirements"

[3GPP_TS23271] 3GPP Technical Specification 23.271 V10.0.0  
"Functional stage 2 description of Location Services (LCS)"

[3GPP-TS24007] 3GPP Technical Specification 24.007  
"Mobile radio interface signaling layer 3; General Aspects"

[3GPP_TS24301] 3GPP Technical Specification 24.301  
"Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS), Stage 3."

[3GPP_TS29060] 3GPP Technical Specification 29.060  
"General Packet Radio Service (GPRS); GPRS Tunneling Protocol (GTP) across the Gn and Gp interface"

[3GPP_TS29274] 3GPP Technical Specification 29.274  
"3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunneling Protocol for Control plane (GTPv2-C); Stage 3"

[3GPP_TS29281] 3GPP Technical Specification 29.281  
"General Packet Radio System (GPRS) Tunneling Protocol User Plane (GTPv1-U)"

[3GPP_TS3120] 3GPP Technical Specification 33.120 V4.0.0  
"Security principles and Objectives"

[3GPP_TS33210] 3GPP Technical Specification 33.210  
"3G Security; Network Domain Security (NDS; IP Network Layer Security"

[3GPP_TS33220] 3GPP Technical Specification 33.220  
"Generic Authentication Architecture (GAA); Generic Bootstrapping Architecture (GBA)"

[3GPP_TS33320] 3GPP Technical Specification 33.320  
"Security of Home Node B (HNB) / Home evolved Node B (HeNB)"

[3GPP_TS33401] 3GPP Technical Specification 33.401  
"3GPP System Architecture Evolution (SAE); Security architecture"

[3GPP_TR33816] 3GPP Technical Report 33.816  
"Feasibility study on LTE relay node security"

[3GPP_TR33820] 3GPP Technical Report 33.820  
"Security of the H(e)NB"

[3GPP_TR33821] 3GPP Technical Report 33.821 V9.0.0  
"Rationale and track of security decisions in Long Term Evolved (LTE) RAN / 3GPP System Architecture Evolution (SAE)"

[3GPP_TS36300] 3GPP Technical Specification 36.300  
"Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description; Stage 2"

[3GPP_TS36305] 3GPP Technical Specification 36.305  
"Stage 2 functional specification of User Equipment (UE) positioning in E-UTRAN"

[3GPP_TS36410] 3GPP Technical Specification 36.410  
"Evolved Universal Terrestrial Radio Access Network (E-UTRAN); S1 general
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

aspects and principles"

[3GPP_TS36413] 3GPP Technical Specification 36.413 "Evolved Universal Terrestrial Radio Access Network (E-UTRAN); S1 Application Protocol (S1AP)"


[3GPP_TS44006] 3GPP Technical Specification 44.006 V9.1.0 "Mobile Station - Base Station System (MS - BSS) interface; Data Link (DL) layer specification"


[ASMONIA_D11] ASMONIA Deliverable 1.1

[ASMONIA_D21] ASMONIA Deliverable 2.1
Evaluating Methods to assure System Integrity and Requirements for Future Protection Concepts

[ASMONIA_D32] ASMONIA Deliverable 3.2
Work in Progress

[ATZ] ARM TrustZone:


[CM_BH2009] Charlie Miller, "Fun and Games with Mac OS X and iPhone Payloads", BlackHat Europe 2009


[ETSI_TS1021651] ETSI TS 102 165-1 V4.2.1 " Method and proforma for Threat, Risk, Vulnerability Analysis "

[CF_ARC2009] Chris Foresman, "New iPhone hardware encryption not even close to hack proof", ArsTechnica, http://www.webcitation.org/5yn84RxwX


Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(I)-1.0


Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0


[ITU_E408] International Telecommunication Union (ITU) E.408 [Edition as of 2004-05], "Telecommunication networks security requirements"

[ITU_X805] International Telecommunication Union (ITU) X.805 [Edition as of 2003-10], "Security architecture for systems providing end-to-end communication"


[MO_2009] Dan Moren, Third iPhone worm targets jailbroken iPhones in Europe, Australia, Macworld, http://www.webcitation.org/5ys9Xu18f


[WMCE6] Windows Embedded CE 6.0 Documentation

[WP7DEV] Windows Phone Development


Abbreviations

2G 2.Generation (→ GSM)
3G 3.Generation (→ UMTS)
4G 4.Generation
3GPP 3.Generation Partnership Project
AAA Authentication, Authorization, Accounting
ACL Access Control List
AES Advanced Encryption Standard
AKA Authentication and Key Agreement
AP Application Protocol
ARFCN Absolute Radio-Frequency Channel Number
ASLR Address Space Layout Randomization
ASMONIA Attack analysis and Security concepts for Mobile Network Infrastructures, supported by collaborative Information exchange (a project name)
ATM Asynchronous Transfer Mode
AuC Authentication Center
BSC Base Station Controller
BSS Base Station System
BTS Base (Transceiver) Station
CBC Cell Broadcast Centre
CDMA Code Division Multiple Access
CPU Central Processing Unit
CS Circuit Switched
CSCF Call/Session Control Function
DoS Denial of Service
DDoS Distributed Denial of Service
DHCP Dynamic Host Configuration Protocol
DNS Domain Name System
DSMIP Dual Stack Mobile IP
DSP Digital Signaling Processor
EAP Extensible Authentication Protocol
EDGE Enhanced Data Rates for GSM Evolution
EEPROM Electrically Erasable Programmable Read-Only Memory
EIR Equipment Identity Register
eNB E-UTRAN NodeB
EPC Evolved Packet Core
EPS Evolved Packet System
E-SMLC Evolved Serving Mobile Location Centre
ESP Encapsulating Security Payload
E-UTRAN Evolved UTRAN
FR Frame Relay
FTP File Transfer Protocol
FPGA Field-Programmable Gate Array
GERAN GSM EDGE RAN
GGSN Gateway GPRS Support Node
GMLC Gateway Mobile Location Centre
GPRS General Packet Radio System
GPS Global Positioning System
GPU Graphics Processing Unit
GRE Generic Routing Encapsulation
GRX GPRS Roaming Exchange network
GSM  Global System for Mobile Communications
GTP  GPRS Tunneling Protocol
GTP-C  GPRS Tunneling Protocol – Control Plane
GTP-U  GPRS Tunneling Protocol – User Plane
GUI  Graphical User Interface
GW  Gateway
H(e)MS  HMS or HeMS
H(e)NB  HNB or HeNB
HDD  Hard Disk Drive
HeMS  HeNB Management System
HeNB  Home eNB
HLR  Home Location Register
HMS  HNB Management System
HNB  Home NodeB
HRPD  High Rate Packet Data
HSS  Home Subscriber Server
HTTP  Hypertext Transfer Protocol
HTTPS  Hypertext Transfer Protocol Secure
ICMP  Internet Control Message Protocol
IETF  Internet Engineering Taskforce
IKEv2  IKE version 2
IMS  IP multimedia subsystem
IMSI  International Mobile Subscriber Identity
IMT  International Mobile Telecommunication System
IP  Internet Protocol
IPsec  “IP secure” (an IETF protocol framework for securing IP transport)
ISC  Internet Software Consortium
ISDN  Integrated Services Digital Network
ISO  International Organization for Standardization
IT  Information Technology
ITU  International Telecommunication Union
ITU  ITU Radiocommunication Sector
IWS  Interworking Solution
JTAG  Joint Test Action Group
LAN  Local Area Network
LCS  Location Service
LEA  Law Enforcement Agency
LFSR  Linear Feedback Shift Register
LI  Lawful Interception
LTE  Long Term Evolution
MAN  Metropolitan Area Network
MAP  Mobile Application Part
MME  Mobility Management Entity
MNO  Mobile Network Operator
MS  Mobile Station
MSC  Mobile Switching Center
MTAN  Mobile Transaction Number
MPLS  Multiprotocol Label Switching
NAS  Non Access Stratum
NB  Node B
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NDS</td>
<td>Network Domain Security</td>
</tr>
<tr>
<td>NFC</td>
<td>Near Field Communication</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute for Standards and Technology</td>
</tr>
<tr>
<td>OAM</td>
<td>Operation, Administration, Maintenance</td>
</tr>
<tr>
<td>OCS</td>
<td>Online Charging System</td>
</tr>
<tr>
<td>OCSP</td>
<td>Online Certificate Status Protocol (RFC 2560)</td>
</tr>
<tr>
<td>OFCS</td>
<td>Offline Charging System</td>
</tr>
<tr>
<td>OMC</td>
<td>Operation and Maintenance Center</td>
</tr>
<tr>
<td>PC</td>
<td>Personal Computer</td>
</tr>
<tr>
<td>PCC</td>
<td>Policy and Charging Control</td>
</tr>
<tr>
<td>PCEF</td>
<td>Policy and Charging Enforcement Function</td>
</tr>
<tr>
<td>PCRF</td>
<td>Policy and Charging Resource Function</td>
</tr>
<tr>
<td>PDN</td>
<td>Packet Data Network</td>
</tr>
<tr>
<td>PIN</td>
<td>Personal Identification Number</td>
</tr>
<tr>
<td>PKI</td>
<td>Public Key Infrastructure</td>
</tr>
<tr>
<td>PLMN</td>
<td>Public Land Mobile Network</td>
</tr>
<tr>
<td>PMIP</td>
<td>Proxy Mobile IP</td>
</tr>
<tr>
<td>PoE</td>
<td>Power over Ethernet</td>
</tr>
<tr>
<td>PS</td>
<td>Packet Switched</td>
</tr>
<tr>
<td>PSTN</td>
<td>Public Switched Telephone Network</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>RADIUS</td>
<td>Remote Authentication Dial In User Service</td>
</tr>
<tr>
<td>RAM</td>
<td>Random-Access Memory</td>
</tr>
<tr>
<td>RAN</td>
<td>Radio Access Network</td>
</tr>
<tr>
<td>Rel</td>
<td>Release</td>
</tr>
<tr>
<td>RF</td>
<td>Radio Frequency</td>
</tr>
<tr>
<td>RFC</td>
<td>Request for Comments (name for IETF standard documents)</td>
</tr>
<tr>
<td>RFID</td>
<td>Radio-Frequency Identification</td>
</tr>
<tr>
<td>RN</td>
<td>Relay Node</td>
</tr>
<tr>
<td>RNC</td>
<td>Radio Network Controller</td>
</tr>
<tr>
<td>SAE</td>
<td>System Architecture Evolution</td>
</tr>
<tr>
<td>SATA</td>
<td>Serial Advanced Technology Attachment</td>
</tr>
<tr>
<td>SCTP</td>
<td>Stream Control Transmission Protocol</td>
</tr>
<tr>
<td>SEG</td>
<td>Security Gateway</td>
</tr>
<tr>
<td>SeGW</td>
<td>Security Gateway (a different flavor than the SEG)</td>
</tr>
<tr>
<td>SGSN</td>
<td>Serving GPRS Support Node</td>
</tr>
<tr>
<td>S-GW</td>
<td>Serving Gateway</td>
</tr>
<tr>
<td>SIGTRAN</td>
<td>Signaling Transport (over IP)</td>
</tr>
<tr>
<td>SIM</td>
<td>Subscriber Identity Module</td>
</tr>
<tr>
<td>SIP</td>
<td>Session Initiation Protocol</td>
</tr>
<tr>
<td>SMS</td>
<td>Short Message Service</td>
</tr>
<tr>
<td>SNMP</td>
<td>Simple Network Management Protocol</td>
</tr>
<tr>
<td>SoC</td>
<td>System on a Chip</td>
</tr>
<tr>
<td>SRVCC</td>
<td>Single Radio Voice Call Continuity</td>
</tr>
<tr>
<td>SS7</td>
<td>Signaling System No.7</td>
</tr>
<tr>
<td>SSD</td>
<td>Solid-State Drive</td>
</tr>
<tr>
<td>SSH</td>
<td>Secure Shell</td>
</tr>
<tr>
<td>TCP</td>
<td>Transport Control Protocol</td>
</tr>
<tr>
<td>TDM</td>
<td>Time Division Multiplex</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
<tr>
<td>TMSI</td>
<td>Temporary Mobile Subscriber Identity</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>TPM</td>
<td>Trusted Platform Module</td>
</tr>
<tr>
<td>TrE</td>
<td>Trusted Environment</td>
</tr>
<tr>
<td>TS</td>
<td>Technical Specification</td>
</tr>
<tr>
<td>UICC</td>
<td>Universal Integrated Circuit Card</td>
</tr>
<tr>
<td>UDP</td>
<td>User Datagram Protocol</td>
</tr>
<tr>
<td>UE</td>
<td>User Equipment</td>
</tr>
<tr>
<td>UMTS</td>
<td>Universal Mobile Telecommunication System</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Serial Bus</td>
</tr>
<tr>
<td>USIM</td>
<td>UMTS Subscriber Identity Module</td>
</tr>
<tr>
<td>USRP</td>
<td>Universal Software Radio Peripheral</td>
</tr>
<tr>
<td>UTRAN</td>
<td>UMTS RAN</td>
</tr>
<tr>
<td>VLR</td>
<td>Visitor Location Register</td>
</tr>
<tr>
<td>VoIP</td>
<td>Voice over IP</td>
</tr>
<tr>
<td>WAP</td>
<td>Wireless Application Protocol</td>
</tr>
<tr>
<td>WLAN</td>
<td>Wireless Local Area Network</td>
</tr>
<tr>
<td>WiMAX</td>
<td>Worldwide Interoperability for Microwave Access</td>
</tr>
</tbody>
</table>
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

Revision History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>2012-02-01</td>
<td>Version for review by the ASMONIA Consortium</td>
</tr>
<tr>
<td>0.2</td>
<td>2012-02-17</td>
<td>Update according to review comments from the ASMONIA Consortium</td>
</tr>
<tr>
<td>1.0</td>
<td>2012-02-29</td>
<td>Minor corrections, editorial rework Released Version</td>
</tr>
</tbody>
</table>
Annex A: Threats to Mobile Networks Documented by 3GPP

It seems that 3GPP has not followed the approach to perform and document extensive threat and risk analyses for all parts of the mobile network. A technical specification "Security Threats and Requirements" has been discontinued after Release 4.

However, when introducing major new technologies or architectural concepts, in many cases threats were considered and documented, mostly in (informative) technical reports rather than in (normative) technical specifications. While some of these documents address wide scopes (e.g. the complete Evolved Packet System), others address only single components (e.g. the "evolved Node B"). Threat analyses related to a specific function are sometimes also included as informative annex to the technical specification describing the security concept for the specific function.

In the following, summaries of relevant 3GPP documents are given. These documents are all publicly available; they can be retrieved at http://www.3gpp.org/Specification-Numbering.

TS 21.133: "Security Threats and Requirements"

This 3GPP Release 4 document has not been continued in later releases. In its clause 6, it gives the threat categorization reproduced below.

**Unauthorised access to sensitive data** (violation of confidentiality)
- **Eavesdropping**: An intruder intercepts messages without detection.
- **Masquerading**: An intruder hoaxes an authorised user into believing that they are the legitimate system to obtain confidential information from the user; or an intruder hoaxes a legitimate system into believing that they are an authorised user to obtain system service or confidential information.
- **Traffic analysis**: An intruder observes the time, rate, length, source, and destination of messages to determine a user's location or to learn whether an important business transaction is taking place.
- **Browsing**: An intruder searches data storage for sensitive information.
- **Leakage**: An intruder obtains sensitive information byexploiting processes with legitimate access to the data.
- **Inference**: An intruder observes a reaction from a system by sending a query or signal to the system. For example, an intruder may actively initiate communications sessions and then obtain access to information through observation of the time, rate, length, sources or destinations of associated messages on the radio interface.

**Unauthorised manipulation of sensitive data** (Violation of integrity)
- **Manipulation of messages**: Messages may be deliberately modified, inserted, replayed, or deleted by an intruder

**Disturbing or misusing network services** (leading to denial of service or reduced availability)
- **Intervention**: An intruder may prevent an authorised user from using a service by jamming the user's traffic, signaling, or control data.
- **Resource exhaustion**: An intruder may prevent an authorised user from using a service by overloading the service.
- **Misuse of privileges**: A user or a serving network may exploit their privileges to obtain unauthorised services or information.
- **Abuse of services**: An intruder may abuse some special service or facility to gain an advantage or to cause disruption to the network.

**Repudiation**: A user or a network denies actions that have taken place.

**Unauthorised access to services**
- Intruders can access services by masquerading as users or network entities.
- Users or network entities can get unauthorised access to services by misusing their access rights.
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

The document then discusses how such threats apply to the following 3 parts of the system:

- Radio interface;
- Other part of the system;
- Terminals and UICC/USIM.

This results in close to 50 distinguished threats, which are mostly still of a rather generic nature and are described only shortly (see list of major and medium threats below – more detail is not given). In an annex focusing on active attacks on the radio interface, some possible attacks are discussed on a more technical level.

In clause 7, the document lists the threats with "major or medium value" (according to a risk assessment following a method described in an ETSI technical report [ETR332]). The list is reproduced below.

T1a Eavesdropping user traffic: Intruders may eavesdrop user traffic on the radio interface. (MAJOR)

T1b Eavesdropping signaling or control data: Intruders may eavesdrop signaling data or control data on the radio interface. This may be used to access security management data or other information which may be useful in conducting active attacks on the system.

T1c Masquerading as a communications participant: Intruders may masquerade as a network element to intercept user traffic, signaling data or control data on the radio interface. (MAJOR)

T1d Passive traffic analysis: Intruders may observe the time, rate, length, sources or destinations of messages on the radio interface to obtain access to information. (MAJOR)

T4a Masquerading as another user: An intruder may masquerade as another user towards the network. The intruder first masquerades as a base station towards the user, then hijacks his connection after authentication has been performed.

T5b Eavesdropping signaling or control data: Intruders may eavesdrop signaling data or control data on any system interface, whether wired or wireless. This may be used to access security management data which may be useful in conducting other attacks on the system.

T6e Manipulation of the terminal or USIM behavior by masquerading as the originator of applications and/or data: Intruders may masquerade as the originator of malicious applications and/or data downloaded to the terminal or USIM.

T9a Masquerading as a user: Intruders may impersonate a user to utilise services authorised for that user. The intruder may have received assistance from other entities such as the serving network, the home environment or even the user himself. (MAJOR)

T9b Masquerading as a serving network: Intruders may impersonate a serving network, or part of an serving network’s infrastructure, perhaps with the intention of using an authorised user’s access attempts to gain access to services himself.

T9d Misuse of user privileges: Users may abuse their privileges to gain unauthorised access to services or to simply intensively use their subscriptions without any intent to pay. (MAJOR)

T10a Use of a stolen terminal and UICC: Intruders may use stolen terminals and UICCs to gain unauthorised access to services. (MAJOR)

T10c Use of a stolen terminal: Users may use a valid USIM with a stolen terminal to access services. (MAJOR)

T10d Manipulation of the identity of the terminal: Users may modify the IMEI of a terminal and use a valid USIM with it to access services. (MAJOR)

T10e Integrity of data on a terminal: Intruders may modify, insert or delete applications and/or data stored by the terminal. Access to the terminal may be obtained either locally or remotely, and may involve breaching physical or logical controls.
T10f Integrity of data on USIM: Intruders may modify, insert or delete applications and/or data stored by the USIM. Access to the USIM may be obtained either locally or remotely.

T10k Confidentiality of authentication data in the UICC/USIM: Intruders may wish to access authentication data stored by the service provider, e.g. authentication key. (MAJOR)

TS 33.120: "Security principles and objectives"
This 3GPP Release 4 document has not been continued in later releases. It lists the following weaknesses in the security of GSM (and other second generation systems) and states that they will be corrected in 3G security:

1) active attacks using a “false BTS” are possible;
2) cipher keys and authentication data are transmitted in clear between and within networks;
3) encryption does not extend far enough towards the core network resulting in the cleartext transmission of user and signaling data across microwave links (in GSM, from the BTS to the BSC);
4) user authentication using a previously generated cipher key (where user authentication using RAND, SRES and A3/8 is not provided) and the provision of protection against channel hijack rely on the use of encryption, which provides implicit user authentication. However, encryption is not used in some networks, leaving opportunities for fraud;
5) data integrity is not provided. Data integrity defeats certain false BTS attacks and, in the absence of encryption, provides protection against channel hijack;
6) the IMEI is an unsecured identity and should be treated as such;
7) fraud and LI were not considered in the design phase of second generation systems but as afterthoughts to the main design work;
8) there is no home environment (HE) knowledge or control of how a serving network (SN) uses authentication parameters for HE subscribers roaming in that SN;
9) second generation systems do not have the flexibility to upgrade and improve security functionality over time.

TR 33.821: "Rationale and track of security decisions in Long Term Evolved (LTE) RAN / 3GPP System Architecture Evolution (SAE)"

The document is a report only, not a specification. Such documents may be a collection of ideas rather than an elaborated study result.

Clause 5 "Threats" discusses threats and possible countermeasures informally. Different contributors seem to have contributed and maintained their individual style (e.g. scope of the threat, structure of the discussion, level of detail). Obviously, this is by no means complete – there are many more potential threats.

Threats to UE:
- IMSI catching
- UE tracking
  - Tracking User temporary ID
  - User tracking due to Linkability of IMSI/TMSI and RNTI
  - User tracking due to IP-address linkability towards TMSI/IMSI/RNTI
  - Tracking based on new and old RNTI mapping
  - Tracking based on handover signaling messages
  - Tracking based on cell level measurement reports
  - Tracking based on packet sequence numbers
  - Tracking based on UE’s static IEEE MAC (Medium Access Control) address
- Forced Handover (to a compromised eNB)
- Forced Handover (to a legacy radio access technology)
- Unprotected bootstrap and multicast signaling broadcast of system information
Threats to eNB and last-mile transport links:
- User Plane packet injection attacks
- User plane packet modification attacks
- User plane packet eavesdropping
- Physical attacks on eNB
- (D)DoS attacks against eNB from the network
- (D)DoS attacks against eNB from UEs
- Threat to Radio Link Failure Recovery

Threats to MME/SAE gateway:
- (D)DoS attacks against MME from RAN side

Threats related to mobility management
- Unauthorised access to control plane data
- Privacy violations
- Unauthorised manipulation of control plane data
- Disturbing or misusing network services
- Unauthorised access to network services

TR 33.820: "Security of H(e)NB"

The document is a report only, not a specification. Such documents may be a collection of ideas rather than an elaborated study result.

The document is an example for a threat analysis considering very closely the technical details of the target system. Clause 5: "Threats Analysis" discusses 29 distinguished threats, listed below in several groups.

Compromise of H(e)NB Credentials
- Compromise of H(e)NB authentication token by a brute force attack via a weak authentication algorithm.
- Compromise of H(e)NB authentication token by local physical intrusion.
- User cloning the H(e)NB authentication Token.

Physical attacks on a H(e)NB
- Inserting valid authentication token into a manipulated H(e)NB.
- Booting H(e)NB with fraudulent software ("re-flashing").
- Physical tampering with H(e)NB.
- Environmental/side channel attacks against H(e)NB

Configuration attacks on a H(e)NB
- Fraudulent software update / configuration changes.
- Mis-configuration of H(e)NB
- Mis-configuration of access control list (ACL) or compromise of the access control list

Protocol attacks on a H(e)NB
- Man-in-the-middle attacks on H(e)NB first network access.
- Denial of service attacks against H(e)NB.
- Compromise of an H(e)NB by exploiting weaknesses of active network services
- Manipulation of external time source
- Attack on OAM and its traffic

Threat of H(e)NB network access

Attacks on the core network, including H(e)NB location-based attacks
- Changing of the H(e)NB location without reporting.
- Software simulation of H(e)NB.
- Traffic tunnelling between H(e)NBs.
- Misconfiguration of the firewall in the modem/router.
- Denial of service attacks against core network.
- H(e)NB announcing incorrect location to the network

User Data and identity privacy attacks
- Eavesdropping of the other user's UTRAN or E-UTRAN user data.
Masquerade as other users.
User’s network ID revealed to Home (e)NodeB owner
Masquerade as a valid H(e)NB
Provide radio access service over a CSG

Attacks on Radio resources and management
Radio resource management tampering
Handover to CSG H(e)NBs

Each threat is discussed in a fixed form, describing prerequisites, the way how to attack, probability, the impact (considering 3 "assets": H(e)NB, user, operator) and mitigation methods.

A "threat matrix" shows probability, impact and "risk level" as numbers between 0 and 1 which are also mapped to classes like "high" – "medium" – "low".

The document states that the results must be considered as preliminary.

TR 33.816: "Feasibility study on LTE relay node security"

The document is a report only, not a specification. Such documents may be a collection of ideas rather than an elaborated study result.

Clause 5 ("Threats") of the document establishes certain assumptions (like “A removable UICC is inserted into the RN to provide authentication between itself and the network”) and lists and discusses the following threats:

- Impersonation of a RN to attack the user(s) attached to the RN
- Attacks on the Uu interface between RN and DeNB
- Inserting a MitM
- Attacking the traffic
- Impersonation of a RN to attack the network
- Attacks on the interface between the RN and UICC
- Attacks on the RN itself
- DoS Attacks
- RN stays as UE after initial attach
- Attacks on NAS signalling and AS traffic

The threats are discussed in an informal way. Partly they are very short, like 5 lines of text or less. In some cases Editor’s Notes state that the discussion is incomplete. Also, there is no claim of completeness of the list of threats.

This threat analysis was used as input for specifying the security architecture for the H(e)NB, which takes into account and – as far as possible and reasonable - mitigates these threats.

Other 3GPP security specifications

TS 33.234 "Wireless Local Area Network (WLAN) interworking security" documents a threat analysis for this particular function in an informative annex. It describes the assets of the 3GPP network operator, the user and the WLAN access network provider and discusses informally threats to these assets. While only short descriptions of rather abstract threats are given, an additional clause "attacks" aims at giving more concrete examples of possible attacks.

TS 33.246 "Security of Multimedia Broadcast/Multicast Service (MBMS)" describes within an informative annex some 20 "security threats" related to MBMS. The description follows 2.1.1
Threat and Risk Analysis for Mobile Communication Networks and Mobile Terminals

D5.1(II)-1.0

TS 21.133 (see 0) in style, structure and level of detail. A risk assessment is not documented.

TS 33.220 "Generic Authentication Architecture (GAA); Generic bootstrapping architecture" gives in an informative annex "Information on how security threats related to known GSM vulnerabilities are addressed by the 2G GBA solution".

TR 33.978-800 "Security aspects of early IP Multimedia Subsystem" shortly describes "Threat Scenarios". However, this work seems to be incomplete and is no longer continued in 3GPP – the "descendant" document TS 33.178 is "withdrawn".

The following are examples of 3GPP security specifications that do not include a threat analysis:

TS 33.201 Security Architecture
TS 33.210 Network Domain Security; IP network layer security
TS 33.310 Network Domain Security (NDS); Authentication Framework (AF)
TS 33.320 H(e)NB Security (threats discussed in TR 33.820)
TS 33.328 IP Multimedia Subsystem (IMS) media plane security (also no threats in "antecedent" document TR 33.828)
TS 33.401 SAE Security Architecture (threats discussed in TR 33.821)
TS 33.402 SAE, Security aspects of non-3GPP accesses (threats discussed in TR 33.821)