Playing with the GSM RF Interface

26C3

Dieter Spaar

Agenda

- Motivation for playing with GSM

- Introducing GSM related projects

- What about an „open“ phone ?

- Sniffing the air traffic using the hardware of a phone (work in progress)

- DoS attack to the GSM network using a phone with modified firmware

- Demonstration of the DoS attack

Motivation

- GSM system and protocols not that well researched compared to TCP/IP (although GSM in use since 1992 in Germany)

- Nearly impossible to "play" with GSM without major effort

- Situation has changed:

  * used equipment can be found
  * OpenBTS
  * OpenBSC
  * Airprobe

- Specification is available from 3GPP or ETSI (huge, about 2000 documents)

Simplified GSM Network Structure diagram to better understand the abbreviations and acronyms

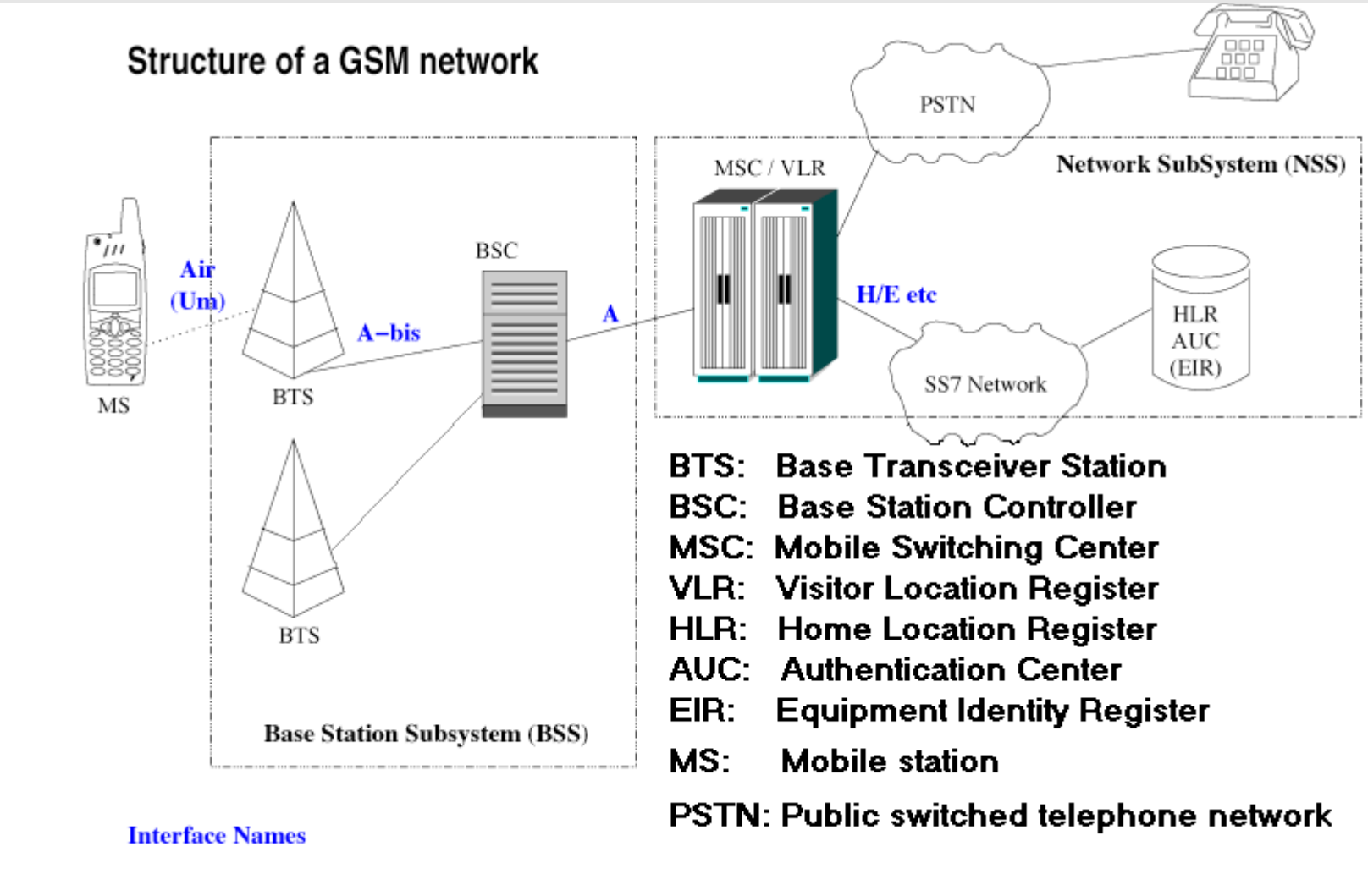


Diagram derived from Wikipedia (http://en.wikipedia.org/wiki/File:Gsm_network.png), GNU Free Documentation License applies to this diagram

OpenBTS

- Hardware based on the USRP (Universal Software Radio Peripheral)

- Air Interface (Um) is a Software Defined Radio

- Does not model classic GSM architecture (BSC, MSC, ...) but uses a direct Um-to-SIP approach

- Uses Asterisk via SIP (Session Initiation Protocol) and Voice-over-IP

OpenBSC

- Implements the Abis protocol plus BSC/MSC/HLR

- Supports the Siemens BS11 microBTS

  * GSM 900, 2 W (can be reduced to 30 mW)
  * about 10 years old
  * cheap
  * heavy (30 to 40 Kg, depending on the configuration)
  * E1 interface for Abis

- Supports the ip.access nanoBTS

  * various bands, 200 mW for DCS 1800
  * small
  * GPRS/EDGE
  * Abis over IP

- Runs the 26C3 GSM network with four nanoBTS units

nanoBTS versus BS11

Airprobe

- Passively sniff the GSM Air Interface
- Based on the USRP and GNU Radio
- Analyze the protocols with Wireshark
- Can be used to capture the traffic for A5/1 GSM encryption cracking

What about an „open“ phone ?

- Project Blacksphere for Nokia DCT3 phones, no longer active ?

- TSM30, based on the TI Calypso GSM chipset, source code can be found. Approach:

  * don't use the source code, control the hardware on our own (Example: sniffing the air traffic)

  * modify the exisiting source code and see what can be done (Example: DoS attack to the GSM network)

- Openmoko GTA01/GTA02: GSM modem based on the TI Calypso however different version and different RF-Transceiver chip. Note: Even for „Open Source“ phones, no source code for GSM.

- Future plans: Take a GSM RF-Transceiver and Baseband chip, connect it to a DSP/FPGA board, develop true open software

TSM30

- Spanish phone (also distributed in Mexico and Chile), about 6 years old

- GSM, GRPS, WAP

- TI Calypso chipset: RF-Transceiver, Digital baseband, MCU (ARM+DSP core), leaked documents can be found

- Firmware C Source Code:

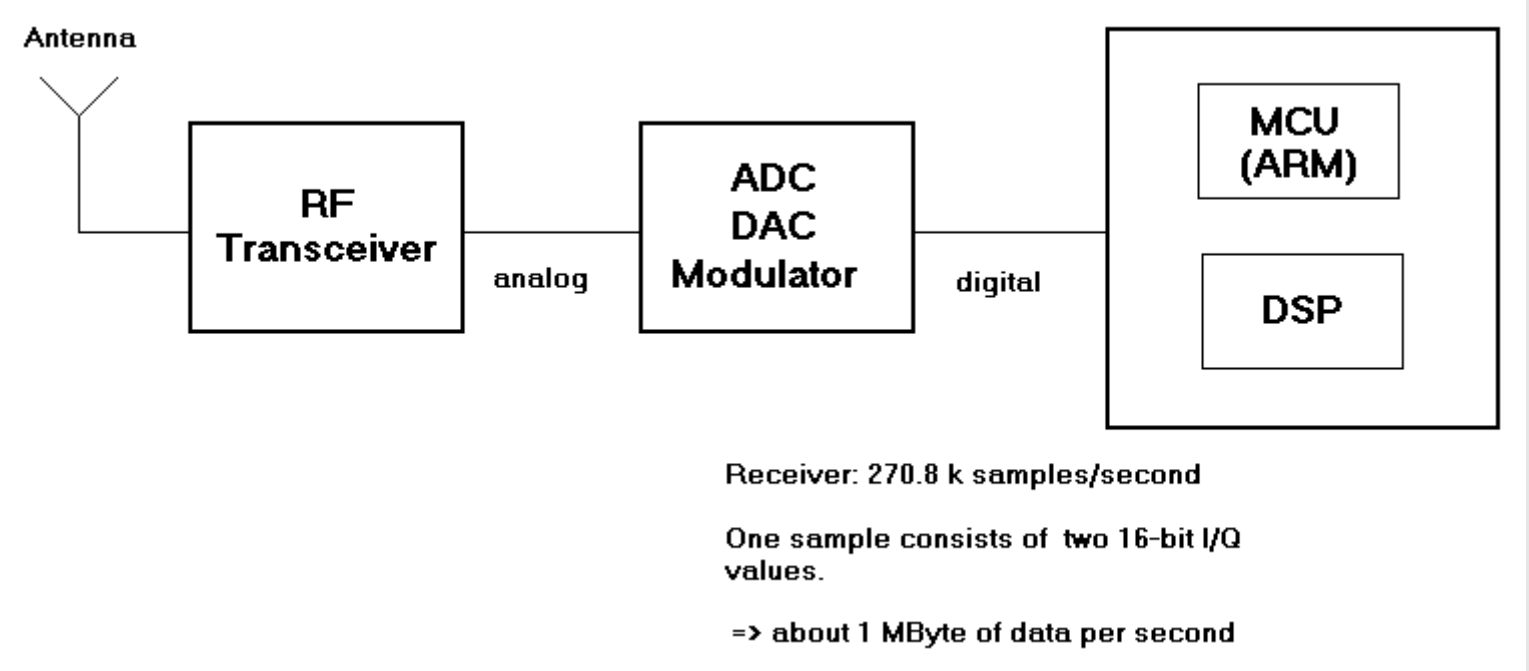
  * Calypso ARM: ca. 6500 files, 2.5 Mio lines of code

    - GSM protocol stack: ca. 700 files, 400 000 lines of code
    - Layer 1 (without DSP): ca. 130 files, 130 000 lines of code

  * Second microcontroller: ca. 1500 files, 350 000 lines of code

Sniffing the air traffic

- (Very) simplified diagram of the Calypso chipset:



- Sample data can be processed with „gsm-receiver“ from Airprobe

- Problem: how to get the data in „real-time“ out of the Calypso

- Future plans: make use of the DSP ? (56 kByte RAM, ROM based GSM signal processing code, no source code)

DoS Attack

- Theory known for quite some time

- However no practical demonstration or implementation ?

- Attack possible because of the way an idle phone has to access the GSM network (details later)

- Affects one cell

- Very early stage, no authentication of the phone yet

- No information of the phone has been transferred yet (anonymous attack)

- Not much can be done against it, try to locate the attacking mobile

GSM TDMA (Time division multiple access)

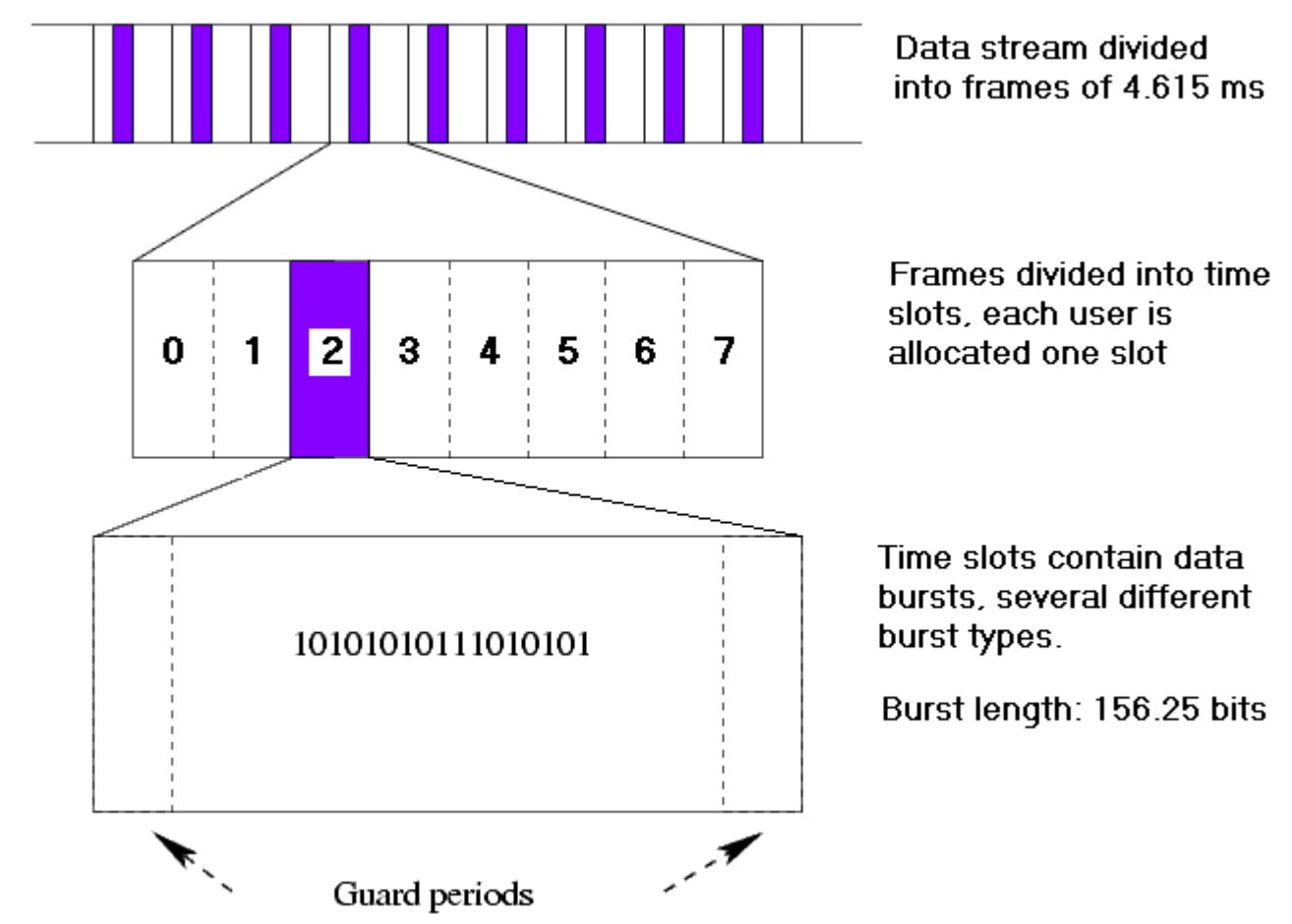
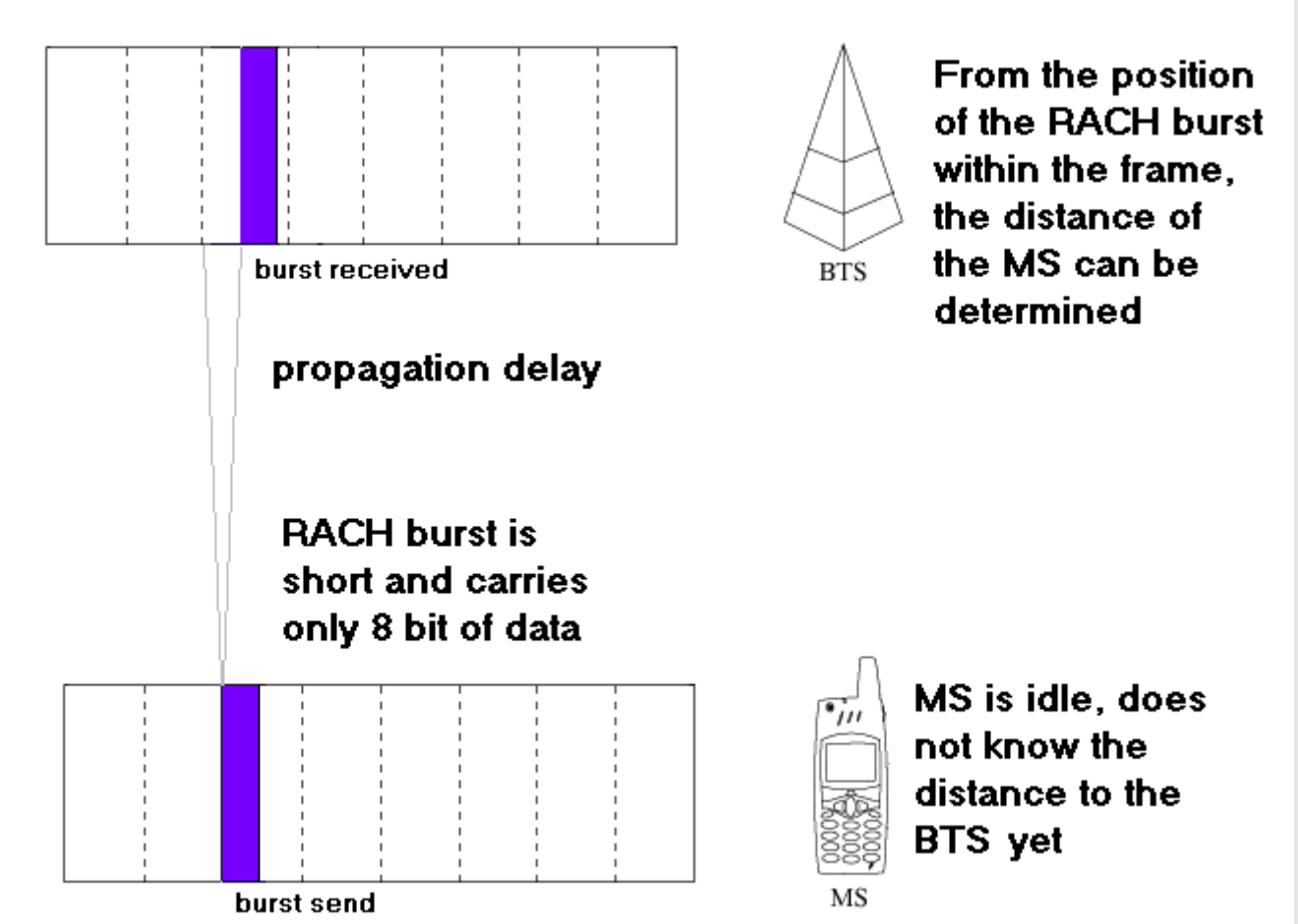


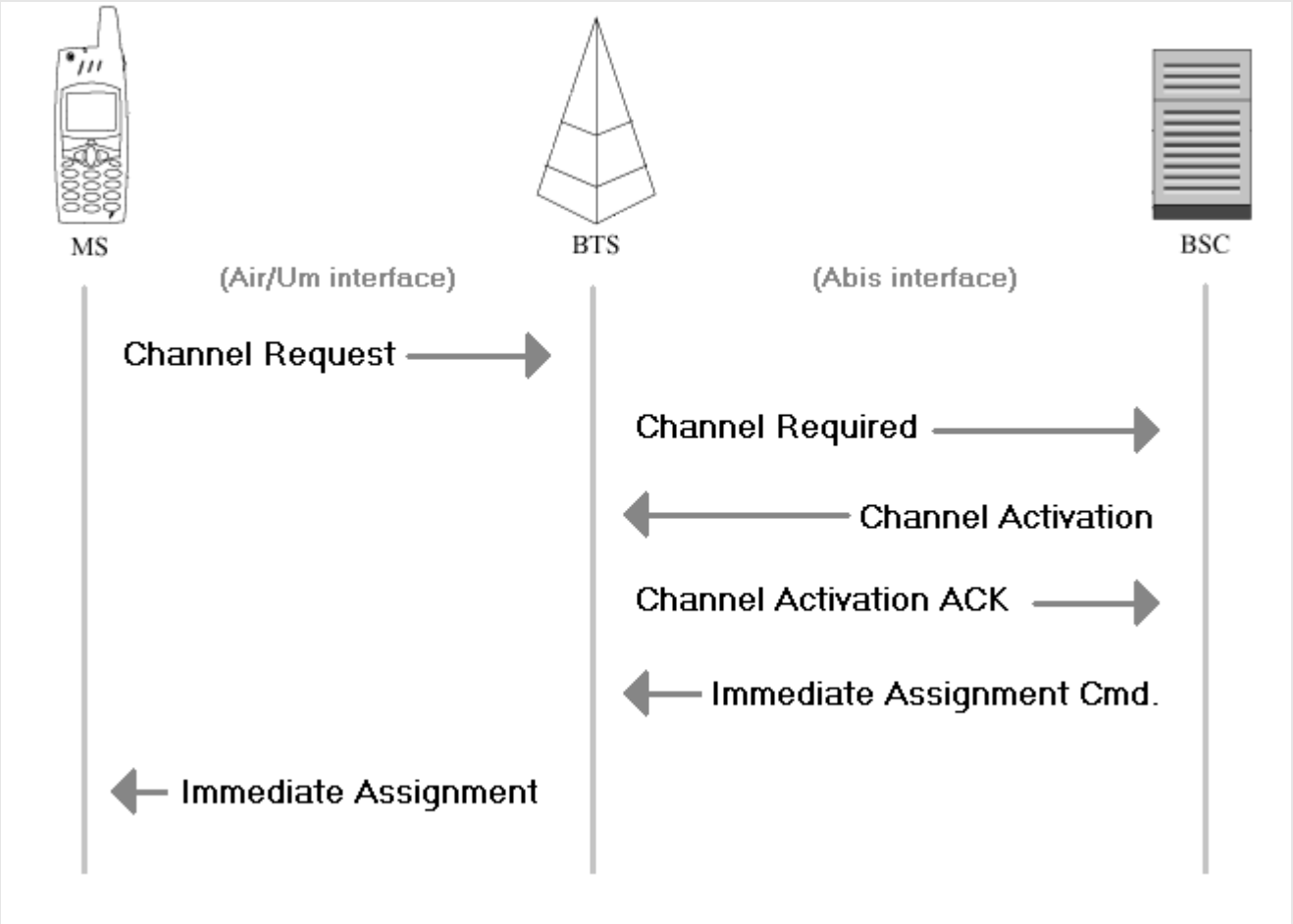
Diagram derived from Wikipedia (http://en.wikipedia.org/wiki/File:Tdma-frame-structure.png), GNU Free Documentation License applies to this diagram

RACH (Random Access Channel) burst



MS sends a RACH burst to access the GSM network (Channel Request)

Message flow Channel Request

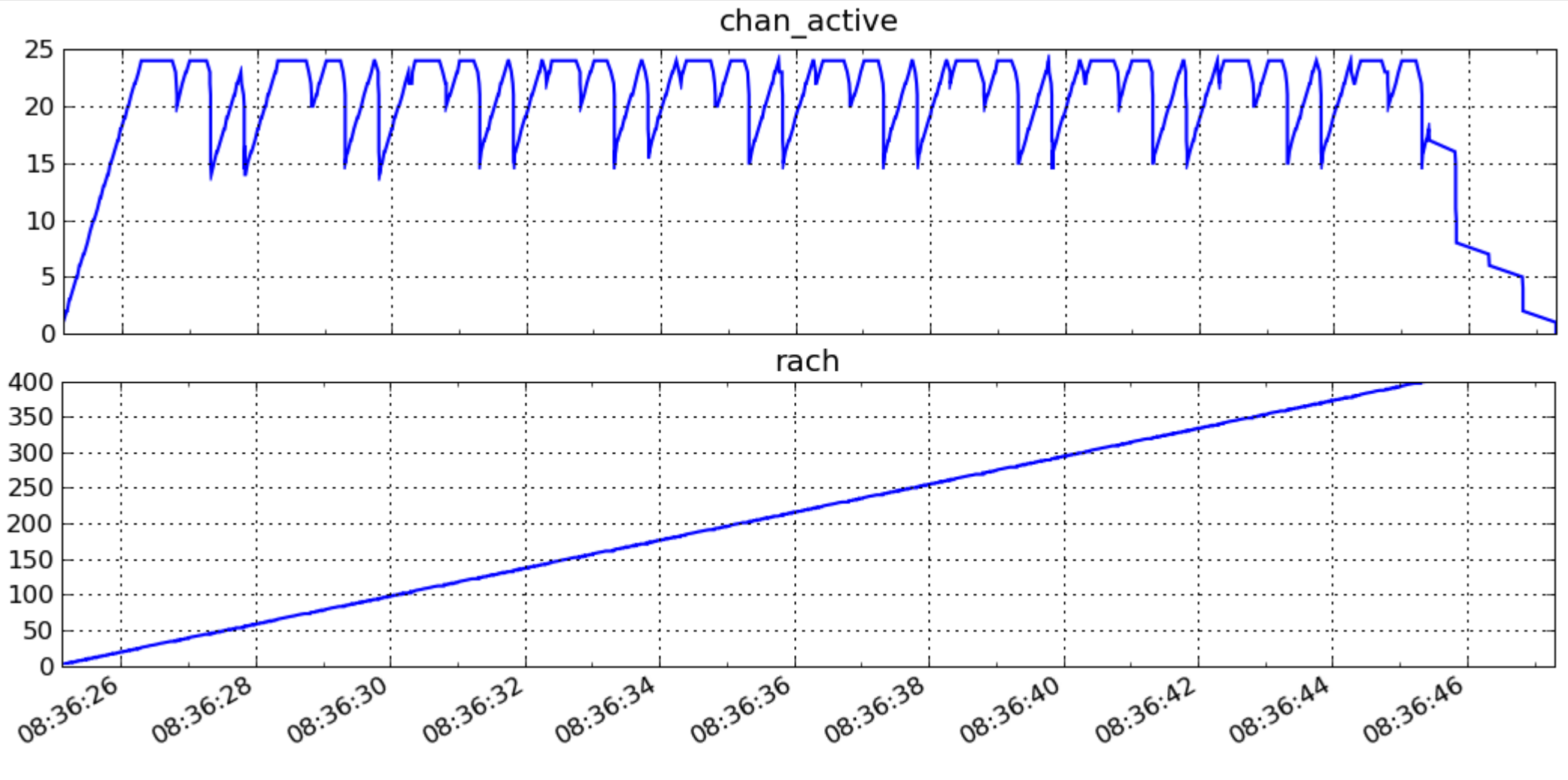


If the channel is not established within a certain time, it is released by the BSC

Effects of continuously sending RACH burst

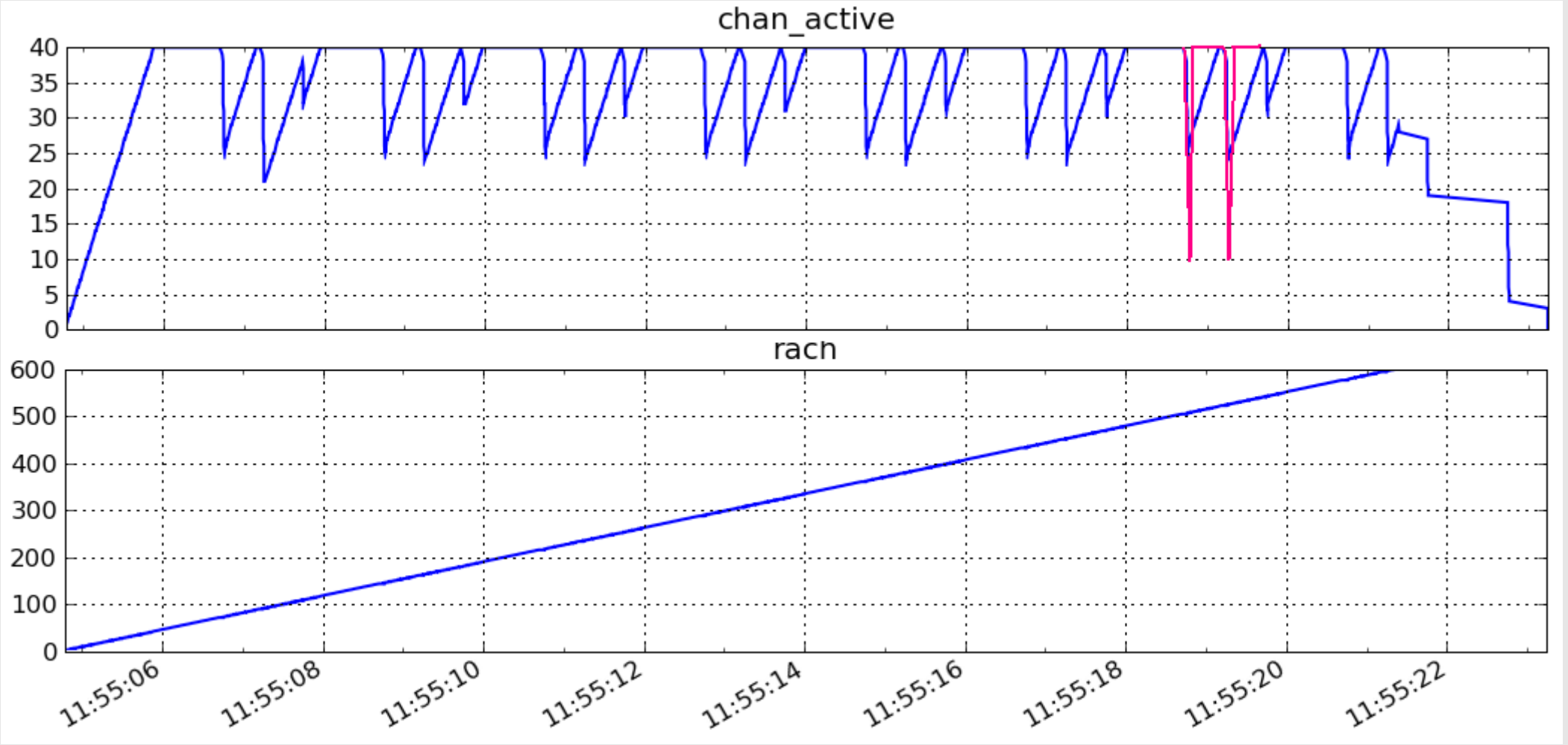
- Jamming (disturbing the signal of other phones)
- Resource allocation (BTS channels)
- No influence on already existing connections
- Difficult or impossible for other phones to access the network through the attacked cell
- Phones might switch to another cell
- No 100% guarantee that another phone won't access the attacked cell
- Useful application: attack IMSI catchers

Lab results, BTS with 2 TRX



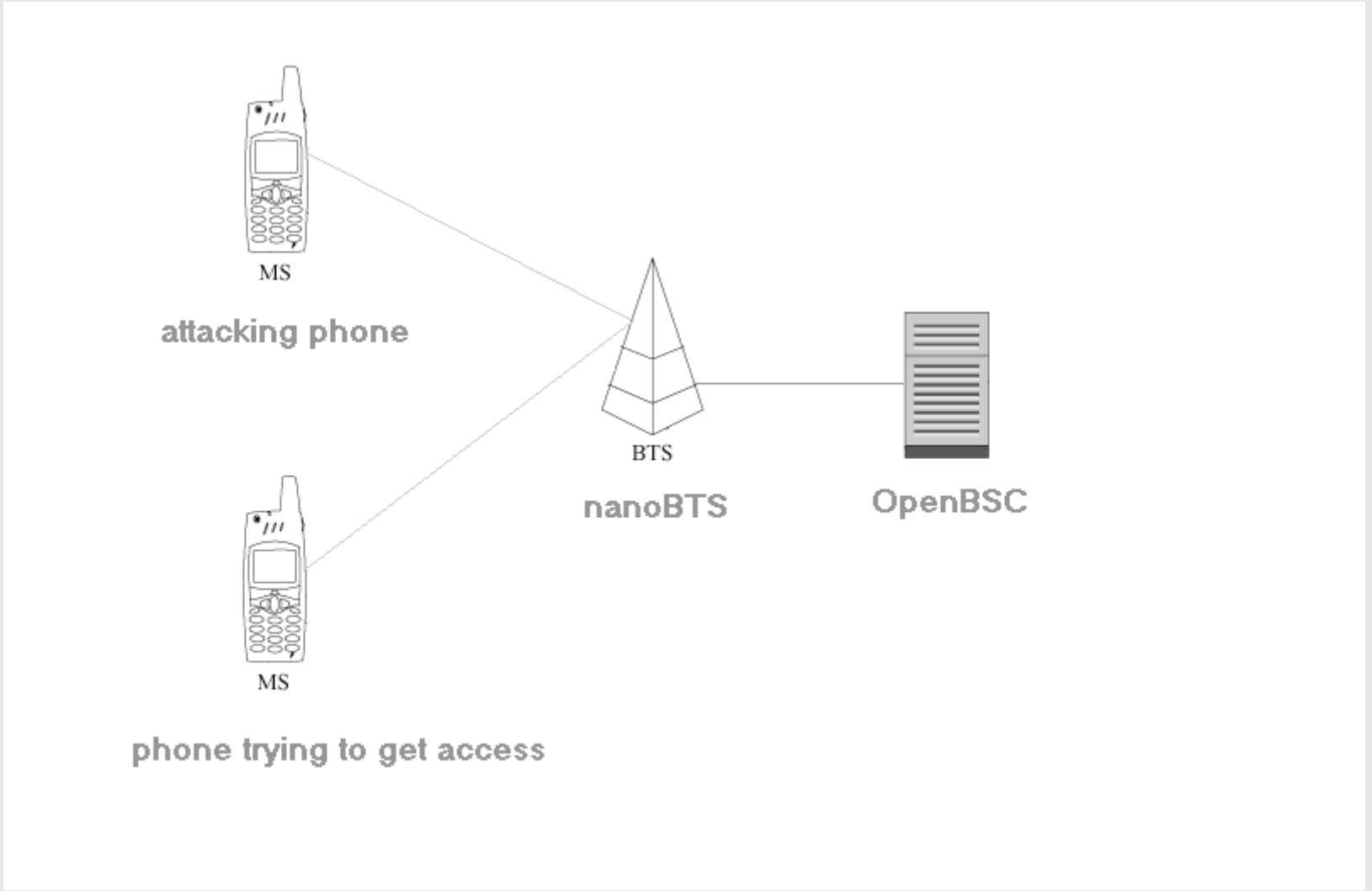
Note: At the time this measurement was made, the RACH density was not yet 100% as it is now

Lab results, BTS with 4 TRX



Note: At the time this measurement was made, the RACH density was not yet 100% as it is now
Red line: this is how it might look with 100% RACH density.

DoS Attack demonstration setup



Links

- OpenBTS: http://openbts.sourceforge.net/
- OpenBSC: http://bs11-abis.gnumonks.org/
- Airprobe: http://airprobe.org/
- 3GPP: http://www.3gpp.org/
- ETSI: http://www.etsi.org/

Thanks

Harald Welte, David Burgess

Feedback

<spaar@mirider.augusta.de>

Questions