



All Your Baseband Are

over-the-air exploitation of memory corruptions in GSM software

Ralf-Philipp Weinmann

Laboratory for Algorithmics, Cryptology & Computer
Security

University of Luxembourg

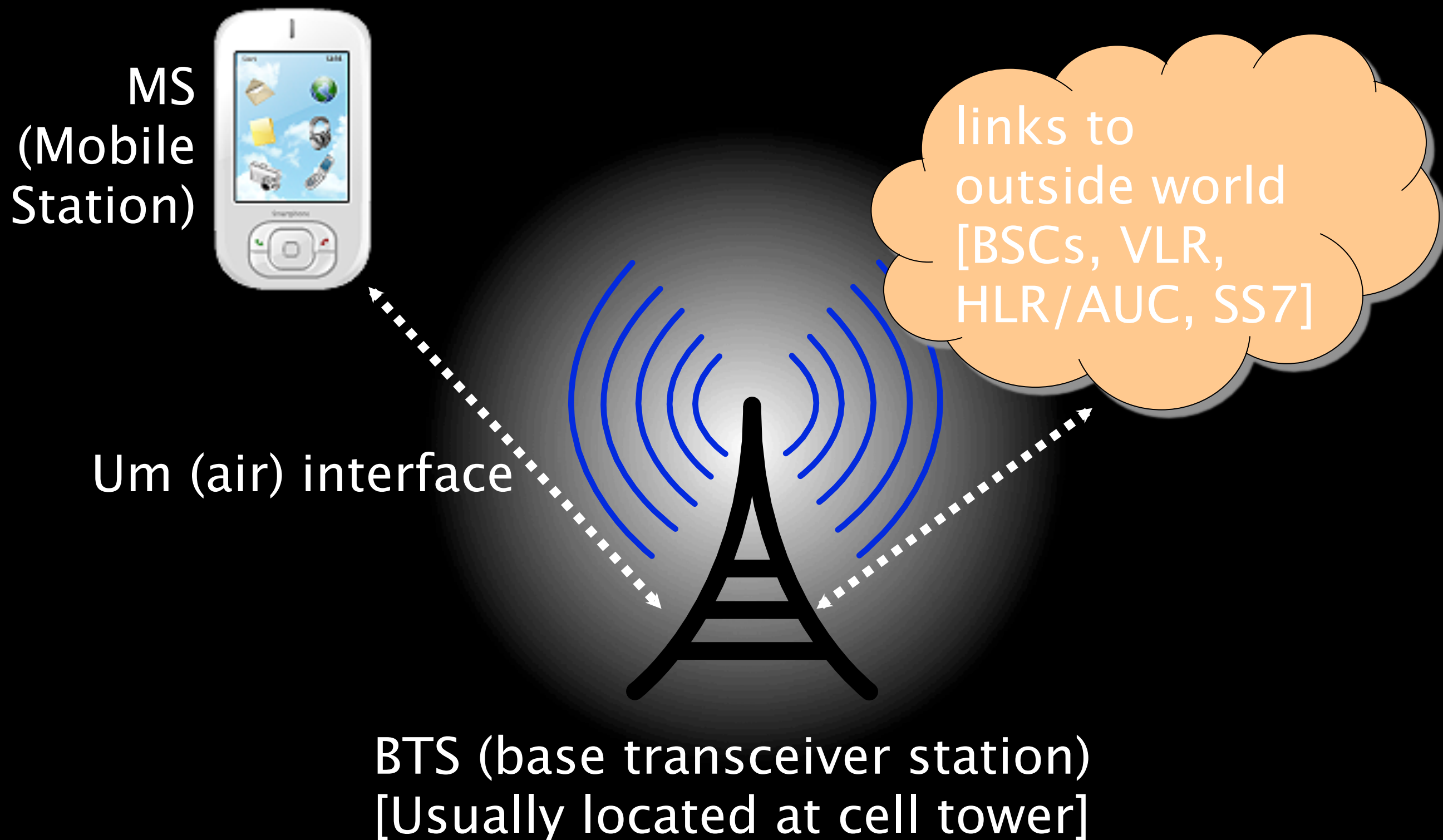
<https://cryptolux.org>

Outline

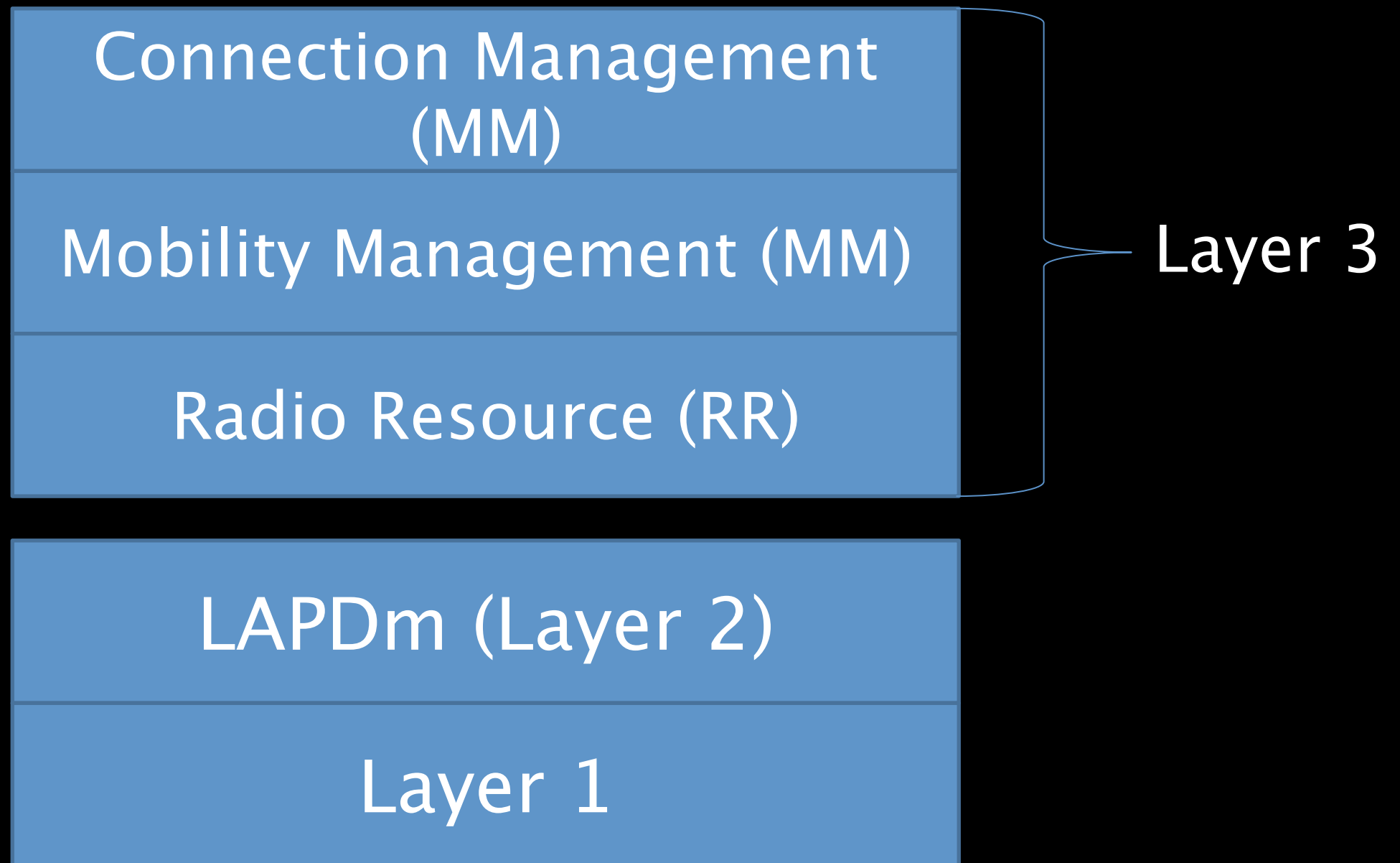
- GSM / Smartphone basics
- Baseband software (in)security
- How to find bugs
- Practicality of exploitation
- Scenarios for the “baseband apocalypse”
- Disclosure, outlook & conclusions

Part I: GSM and smartphone basics

Lay of the GSM/UMTS land



Layers of the GSM Um

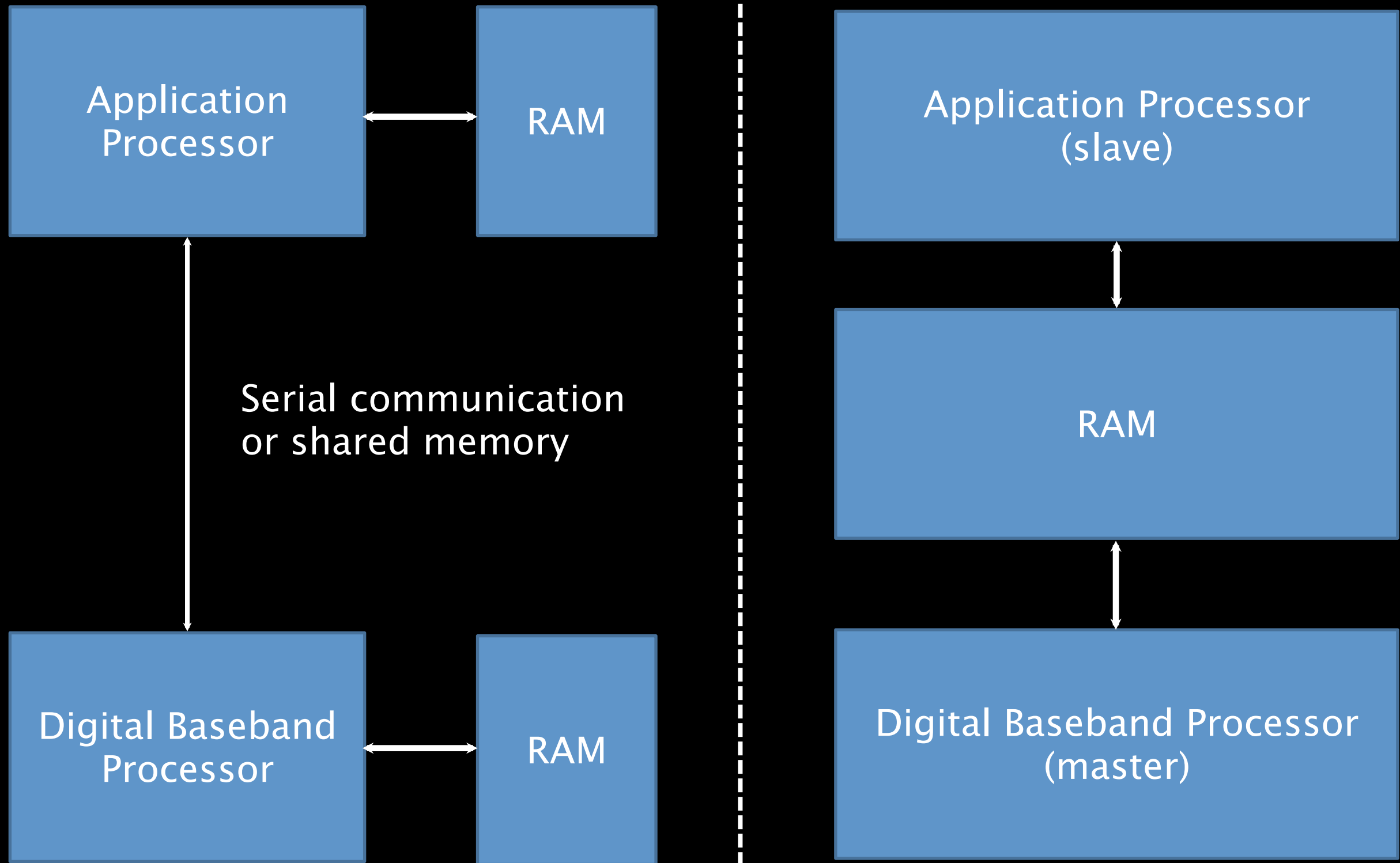


Smartphones

- Somewhen in the late 20th century, PDAs and cellular phones merged
- Result: smartphones
- Have driven PDAs into extinction
- Usually a multi-CPU architecture: application processor (APP) and baseband (BB) processor
- In 99% of all cases, ARM CPUs used for both
- Trend: single-chip APP/BB (for cost

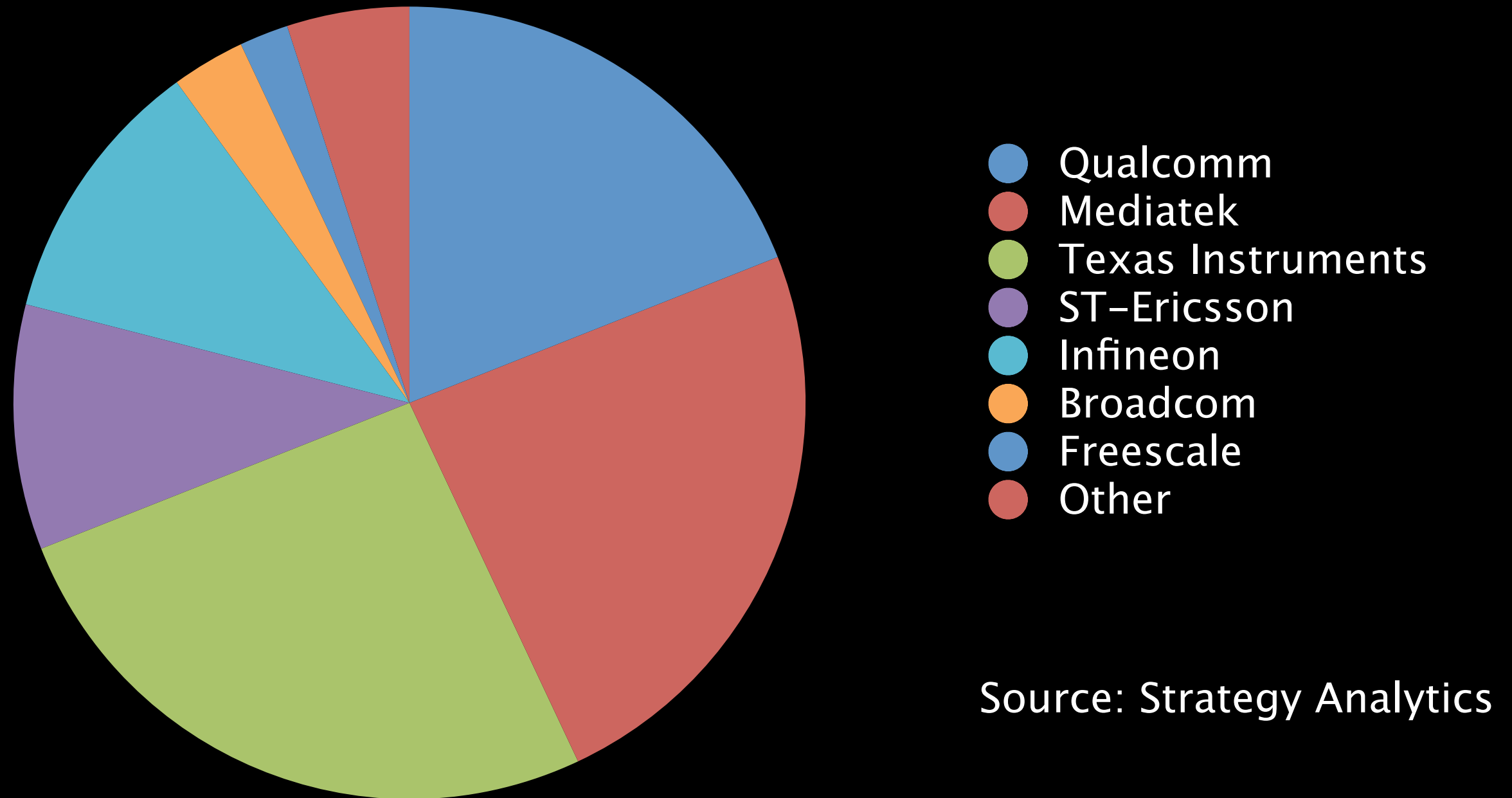
Dominant Smartphone archs

vs.



Let's do some quick market
research before we dive into the
technical details...

Baseband market shares 3Q2009



Cellular Baseband Suppliers & their 3Q' 09 shipment share)

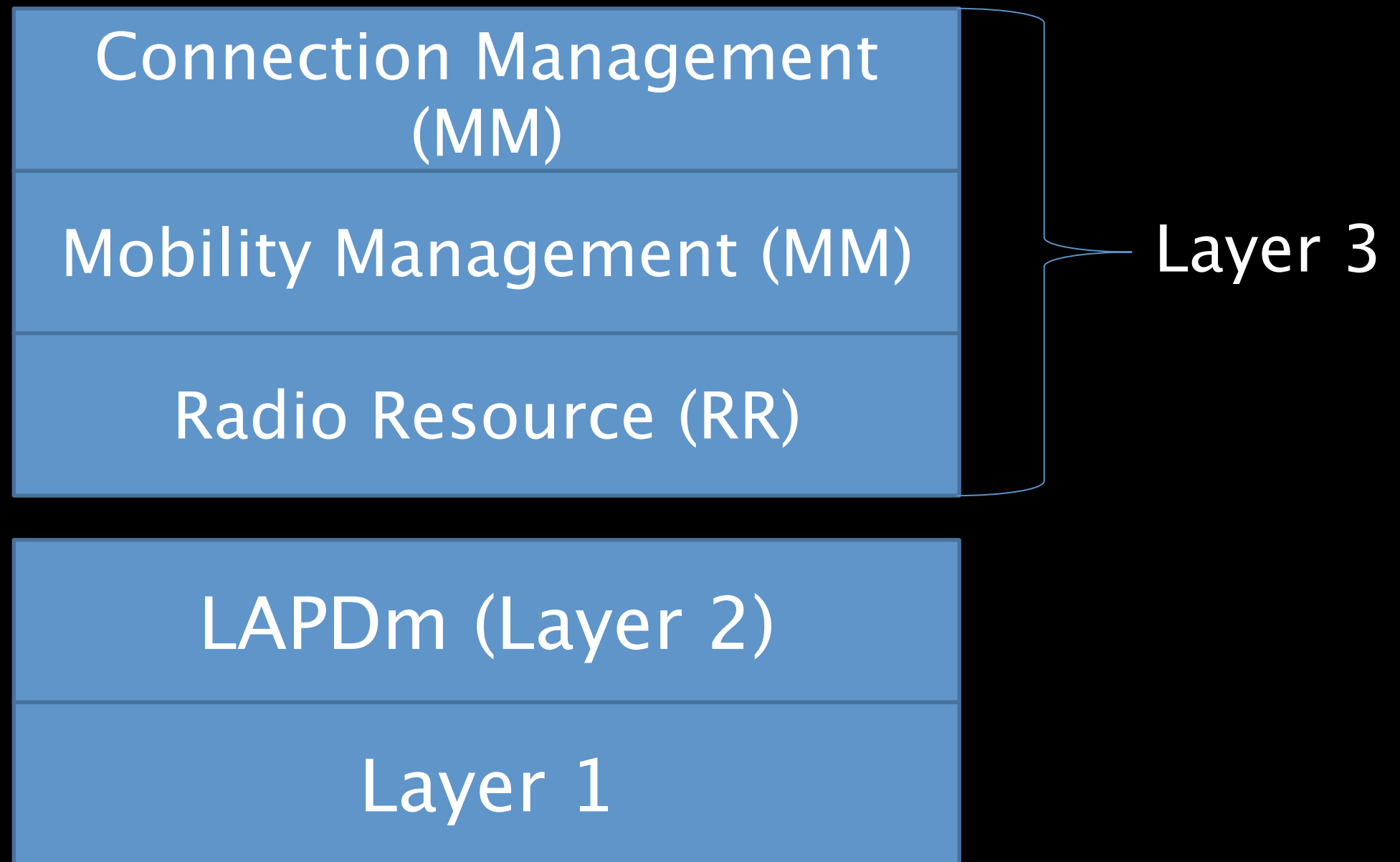
Part II: Baseband (in)security

Baseband (in)security

- Code base created in the 1990s...
- ... with a 1990s attitude towards security
- Network elements are considered trusted
- Both GSM and UMTS protocols have many, many length fields
- (Almost) no exploit mitigations
[one counter-example: XMM6180 on iPhone4 has hardware DEP enabled]

I know you forgot what the GSM protocol stack looks like, so let's see it once more before we proceed.

Layers of the GSM Um



Where to look for bugs

- Layer 1 not fruitful
- Layer 2: messages too short
- Layer 3: specified in GSM 04.08
 - allows for variable length messages (TLV and LV)
 - Maximum length: 255 octets (length field: one octet)
- However: ASN.1 used as well (e.g. RRLP)
- GPRS layer very fruitful as well
 - GPRS not supported by OpenBTS
 - layer 1 different

Where to look for bugs

- Layer 1 not fruitful
- Layer 2: messages too short
- Layer 3: specified in GSM 04.08
 - allows for variable length messages (TLV and LV)
 - Maximum length: 255 octets (length field: one octet)
- However: ASN.1 used as well (e.g. RRLP)
- GPRS layer very fruitful as well
 - GPRS not supported by OpenBTS
 - layer 1 different

Where to look for bugs

- Layer 1 not fruitful
- Layer 2: messages too short
- Layer 3: specified in GSM 04.08
 - allows for variable length messages (TLV and LV)
 - Maximum length: 255 octets (length field: one octet)
- However: ASN.1 used as well (e.g. RRLLP)
- GPRS layer very fruitful as well
 - GPRS not supported by OpenBTS
 - layer 1 different

Things get interesting



Initial Targets



Image credit: Yutaka Tsutano



Image credit: Jose A. Gelado

Apple iPhones
(Infineon
baseband)

HTC Dream [G1]
(Qualcomm
baseband)

How were the bugs found?

- Fuzzing was not successful
 - Lots of crashes, but no easy way to triage
- Static analysis
- No source code publicly available
 - exception: TSM30 src was available for some years
- Conclusion: reverse-engineer binaries

How do we start?

- Firmware updates often contain baseband firmware as well
- Packed multiple times, need to extract
- Tools for iPhone and HTC phones to do that
- Qualcomm firmwares: ELF files
- Infineon needs custom loaders/relocator
- Later: ability to dump memory/MMU

Reverse-engineering

- Bootstrap: use BinDiff to port symbols from known libraries (i.e. compiler runtimes)
- Identify functions that do memory transfers using REIL and BinNavi
- Lots of strings and assertions (!) in firmwares
- Often: clean-cut regions for RR/MM/AT command parser in binary

More reversing

- Identified functions handling GSM frames
 - Problem: apparently different tasks
 - Assertions/logging functions very helpful

Types of bugs found

- Many, many unchecked memory copies (can be found in binary once `memcpy()` et al. identified)
- Object/structure lifecycle issues (e.g. use after free, uninitialized variables, state engine confusion), can lead to infoleaks as well
- Protocol foo-bars: Code paths normally used for UMTS / CDMA can be triggered using GSM frames

An example (in ICE)

- TMSI reallocation:
 - TMSI always, always, always is 32 bits
 - nonetheless encoded as TLV
- Infineon stack uses length in L3 packet
- Results: heap overflow
- Somewhat tricky to exploit in stable way
- iPhone 2/3G/3GS vs. iPhone 4: different RTOS
 - old iPhones: Nucleus
 - iPhone 4: ThreadX

An example (in QCOM)

- GSM & UMTS use challenge–response auth
- Originally: fixed–length challenge in GSM
 - 16 bytes RAND
- 3GPP specification 24.008 added variable length challenge (AUTN)
- Functionality not needed in GSM!
- Allows to overwrite stack (limit 251 bytes)
- Result: remote code exec, pre–auth
- QCOM fixed after disclosure (pushed to OEMs)

Baseband Exploitation

- Baseband: what operating system?
- Unlock teams often have good info on this (iPhone dev team, XDA developers)
- Locate buffers used for GSM L3 messages
- Write custom code or use existing features (e.g. `AT+S0=x` handler in Infineon baseband)
- Debugging is hard, write own debugger

The AT+S0=n feature

- Hayes command to turn on auto-answer
- present in some software stacks (verified for Infineon & QCOM)
- Enable with `*5005*AANS#` on iPhones, disable with `#5005*AANS#`
- Excellent target to demonstrate memory corruptions
- Auto-answer can be made silent/

Part III: Practicality

Why should we care

- New base stations: expensive (cheapest: 25k USD)
- Old gear however often is sold on eBay
- Threat model has entirely changed: hardware has become cheap, open-source SW appeared
- Open-source projects for running GSM base stations: OpenBSC & OpenBTS
- OpenBTS provided service at Burning Man 2008–2010
- HAR2009 had OpenBSC test network



- Siemens BS11
- used by OpenBSC
- **HEAVY**
- E1/Abis interface
- cheap: EUR 250
- hard to come by now.

Image credit:
Björn Heller



- ip.access nanoBTS
- supported by OpenBSC as well
- Abis over IPv4
- approx. USD 4500
- different versions for GSM900/1800, GSM850/1900
- supports GPRS



Image credit: Synthesis Studios

Our gear: Ettus USRPv1

- price: approx USD 1250 plus good clock

- software defined radio (SDR)
- versatile (different daughterboards)
- OpenBTS support, GSM850/900, GSM1800/1900
- no GPRS since layer 1 is different there
- clock: wrong freq (64Mhz) and imprecise

Part IV: Demo

Common failures (my experience)

- Lacking clock precision
- Misinterpreting stack traces
- Triggering the wrong bug ;)
- Overlooking code is placed in non-exec page

Some words about clocks

- Get a good one, seriously!
 - GSM spec requires 0.05ppm
 - equiv. to 50Hz in 900MHz band
- Time is too precious for fixing clock issues
- Using FA–SY on the road (EUR 40)
 - Si570 based design
 - not optimal: 20ppm uncalibrated
 - approx. 1ppm when calibrated

Part V: The Baseband Apocalypse

The “Baseband Apocalypse”

- Place fake BTS in crowded/sensitive areas: airport lounges, financial districts, near embassies
- Stealth room monitor: record audio, compress, store in RAM, piggy-back onto next data connection (mic/camera usually hang off BB CPU)
- Shared mem CPUs: compromise APP CPU as well, place backdoor/rootkit

The “Baseband Apocalypse”

- Ping-pong games: compromise cellphone, then BTS/BSC, infect more phones from there
- Brick phones permanently (e.g. erase SecZone on iPhone)
- No easy forensics possible in BB land (JTAG disabled to prevent easy unlocks). Need exploits to perform forensics

The scary bit

- How do we defend ourselves?
Turn off our cell phones? Hardly.
- Use a sound-proof enclosure for phone and encrypting Bluetooth Headset?
[approach allegedly used by a German company that produces “secure” end-to-end solutions for governments]

Is there still hope for the paranoid?

OsmocomBB

- Free Software GSM baseband stack
- implements layer 1–3
- target platform: Calypso chipsets
- present in OpenMoko phones and Motorola C11x/C12x (e.g. C123)
- current functionality: about GSM Phase 1
 - supports sending/receiving SMS
 - supports voice calls

Part VI: Disclosure, outlook, conclusions

Disclosure & Reactions

- QCOM was fantastic
- Apple fixed TMSI bug this week
- Vendor outreach by Microsoft
- ST-Ericsson:
“We have been using Coverity on our RTOS (incl. the entire L2/3 source code) for a few years – which may detect some of the vulnerabilities. And the canaries have always been there to enable the scheduler to detect stack overflows [...]”

Outlook

- Will see same problems for 3GPP/UMTS
- 3GPP uses mutual auth...
- Need Radio Resource Control (RRC) pre-auth
- RRC is about 1800 pages of specification!
- ASN.1 PER !!
- Only single vendor for the ASN.1 parser seen in stacks
- Femto cells as cheap attack platforms

Conclusions

- Memory corruptions over the Um interface: practical even with cheap hardware
- Vulnerabilities in GSM baseband codebases plentiful
- Small number of baseband vendors
- Malicious code execution on baseband CPU: compromises security
 - Shared memory between BB & APP: total compromise

Credits

- uni.lu for supporting me
- Anonymous Engineer and Jacob Appelbaum for the initial trigger
- Zynamics for providing great RCE tools!
- Harald Welte, Sylvain Munaut, André Stemper for technical help
- Karsten Nohl and Harald Welte for providing replacement gear
- Kestrel for writing OpenBTS
- Qualcomm and Apple for fixing bugs!