//

Mobile Authentication Subspace Travel

Markus Vervier

May 28th, 2015

whoami

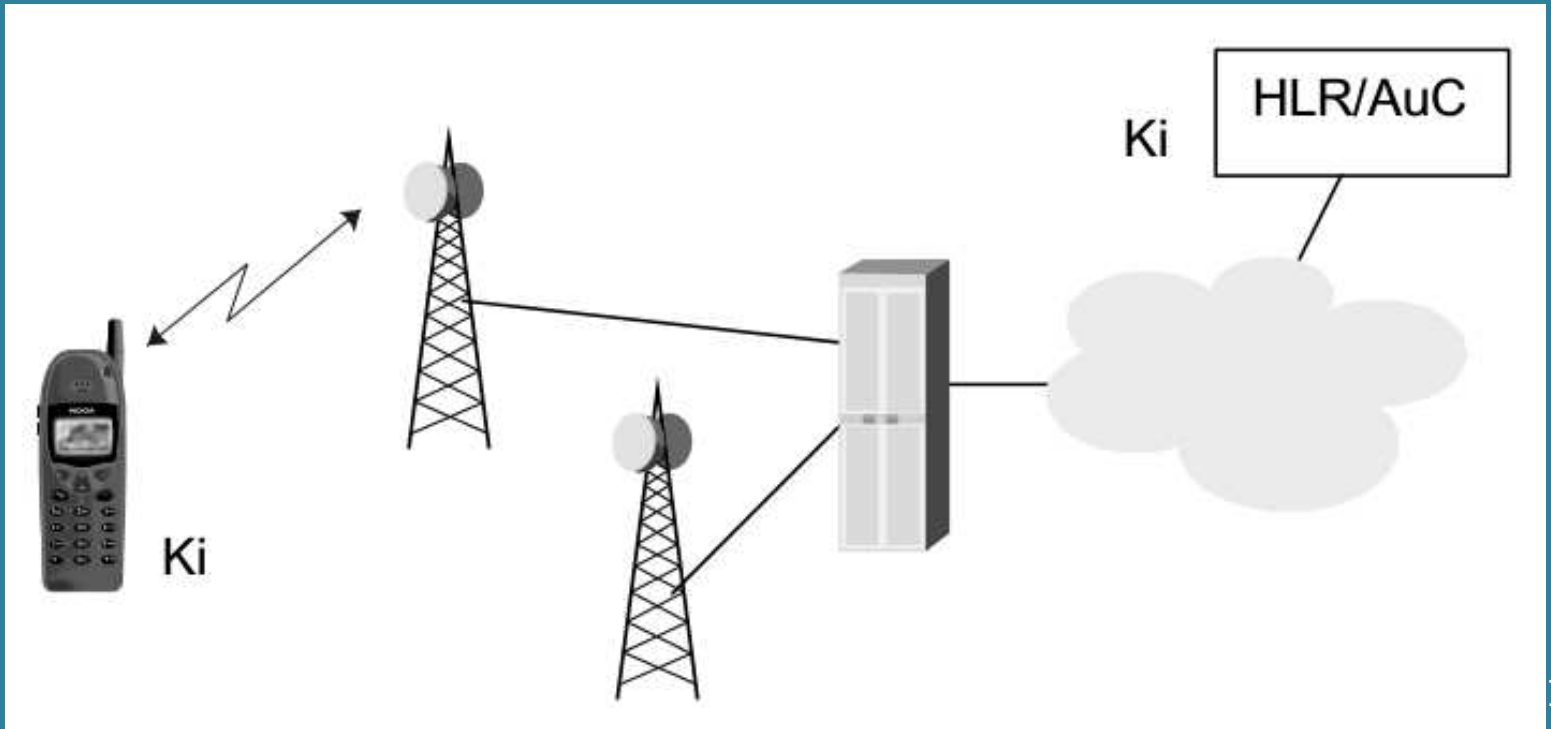
- Markus Vervier / @marver
- Background in security for over 10 years
- Main interests:
  - Firmware
  - Network Security
  - Mobile Networks
  - Finding Bugs
  - Security Design
- Working as Security Researcher and Penetration Tester for LSE Leading Security Experts GmbH

Intro / What it's all about

Topics of this Talk

- Authentication in mobile networks
- How millions of devices are exposing SIM-Cards
- How to have fun with baseband firmware
- Using this to foward mobile network authentication

Authentication (Birds Eye)



[1]

- SIM-Card authenticates a user / his contract
- Provider AuC and SIM-Card share a secret key *Ki*
- Challenge-Response Network-Authentication between Mobile-Equipment (ME) and Network
- `Users have no access to` K_i

[1] Source: UMTS Security, Valtteri, Niemi and Kaisa Nyberg

May we Borrow your Identity for a While?

A Misconception

- Naive Idea: Authentication is secured by having a ”secure“ SIM device that does it

A Misconception

- Naive Idea: Authentication is secured by having a "secure" SIM device that does it

A Misconception

- Naive Idea: Authentication is secured by having a "secure" SIM device that does it

- Temporary Authentication tokens are derived from the secret key K_i on the SIM
- `Then they leave the SIM!`
- They are valid on their own for a time!

SIM Access

SIM-Usage

- Baseband manages the SIM-Card
- Sends command APDUs to the SIM-Card and processes responses
- Passes stuff like SMS, SIM-Tookit, etc. to the AP

Retrieving Authentication

- No direct access to SIM by AP
- But there are indirect methods:
  - AT-Command-Interfaces accessible via Bluetooth / USB
  - Vendor specific: Internal Android RIL calls

SIM-Card-Access via AT+CSIM


```
Command Syntax:
AT+CSIM=<length>,<command>
Response Syntax:
+CSIM:<length>,<response>
```

- Nobody listened to security advice from 3GPP 27.007: "Care must be exercised in AT commands that allow the TE to take unintentionally control over the SIM-MT interface (e.g. +CSIM);"

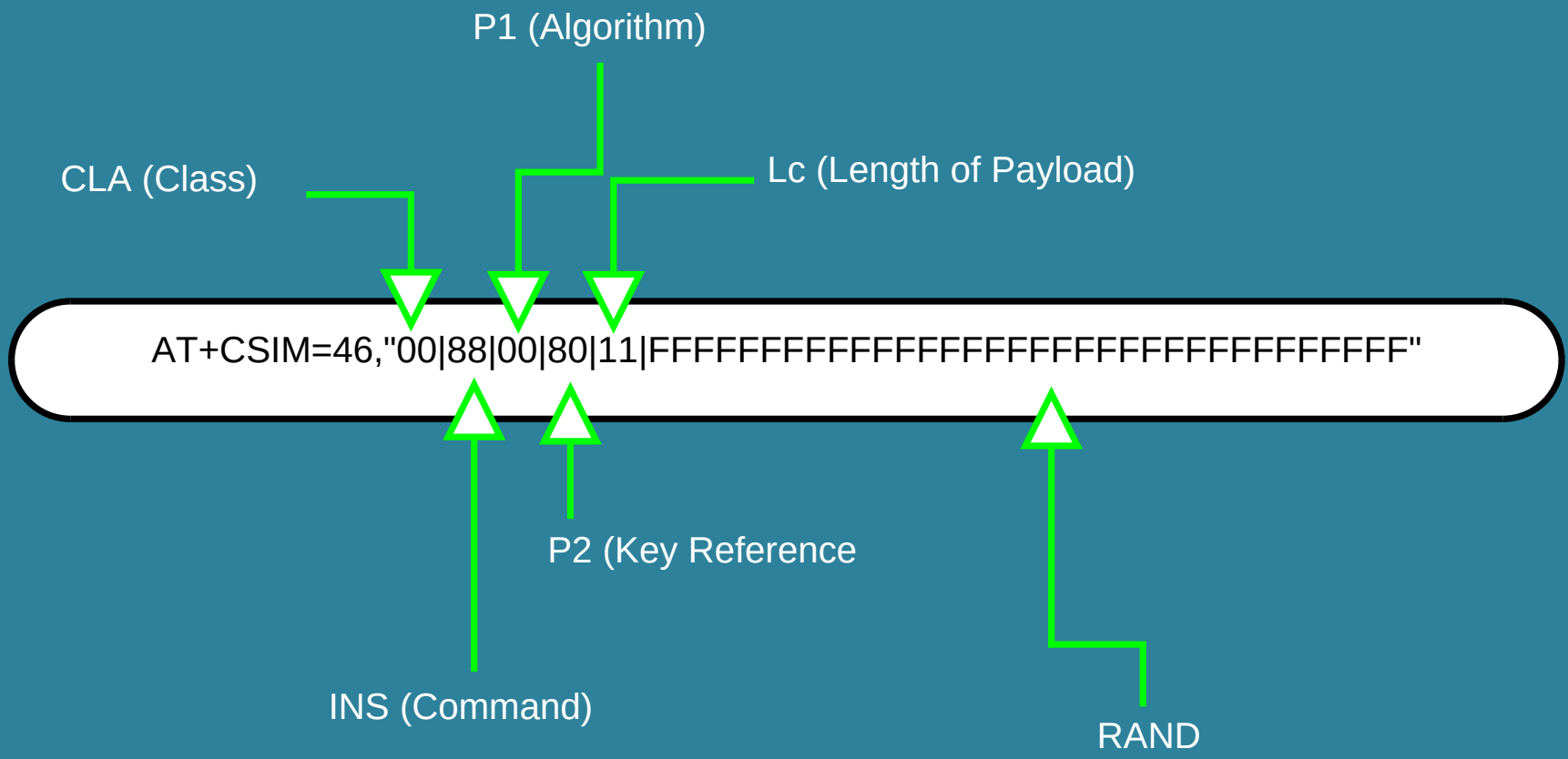
Unprivileged Apps can Talk to the SIM

- Should have no SIM access withtout privileges
- AT-Command-Prompt found on /dev/pts/XX on MTK-Devices
- Bug: Permissions 0777 on older Alcatel Android devices!
- *Unprivileged* apps can query the SIM-Card via AT+CSIM
- Also other methods for SIM-Access at other vendors (Samsung Galaxy S2 / S3)

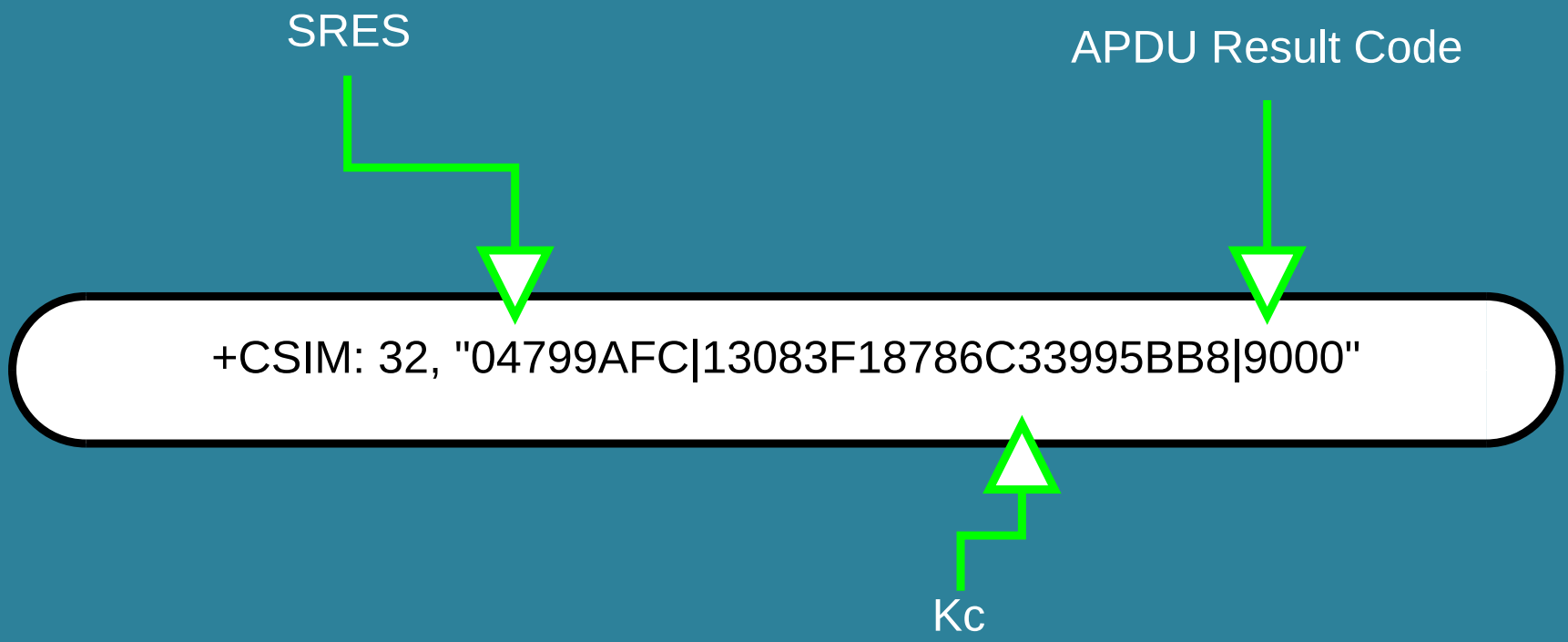
SIM-Card-Access Demo

DEMO

Command-APDU



Response-APDU



Enter AT+EAUTH

- Problem: AT+CSIM does not work on recent devices
- Solution: Vendors added new commands to help
- Dedicated commands for authentication: AT+EAUTH and AT+ESIMAUTH
- Used for EAP-SIM / EAP-AKA e.g. to authenticate to a WiFi using a SIM
- Also used to retrieve authentication to connect to a mobile network

SIM-Card-Access via BT-SAP

- Purpose: Interoperability Car↔Phone
- Solution: Sim Access Profile
- Allows remote SIM usage via Bluetooth
- Specified in Bluetooth DOC: SAP_SPEC
- Great! A Specified way to leak your network authentication!

Dial Up Networking

- USB or Bluetooth
- Works via AT-Commands
- Exposes a serial device
- Present on millions of older mobile phones
- Often exposed without user notification and interaction
- `What could possibly go wrong?`

Dial Up Networking

- USB or Bluetooth
- Works via AT-Commands
- Exposes a serial device
- Present on millions of older mobile phones
- Often exposed without user notification and interaction
- `What could possibly go wrong?`

Dial Up Networking

- USB or Bluetooth
- Works via AT-Commands
- Exposes a serial device
- Present on millions of older mobile phones
- Often exposed without user notification and interaction
- `What could possibly go wrong?`

USB Modem Demo

DEMO

A Blackhat Telco Operator

- How many systems in the world are part of botnets?

A Blackhat Telco Operator

- How many systems in the world are part of botnets?
- *Over 9000* for sure!

A Blackhat Telco Operator

- How many mobile phones are connected regularly to these systems via USB?

A Blackhat Telco Operator

- How many mobile phones are connected regularly to these systems via USB?
- *A lot!*

A Blackhat Telco Operator

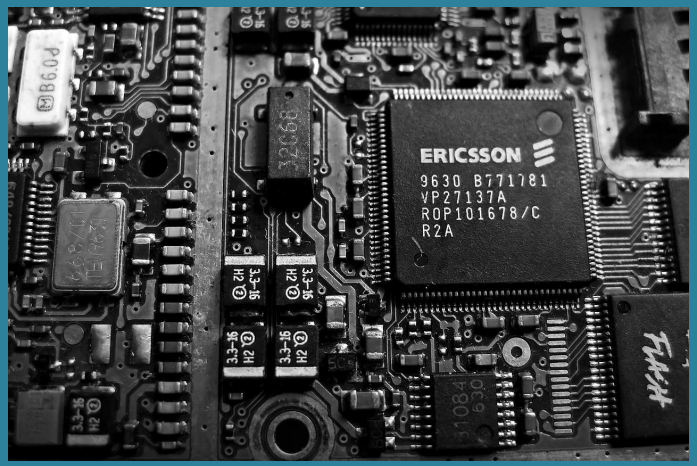
- How many mobile phones are connected regularly to these systems via USB?
- *A lot!*
- Attacker-Goal: Authenticate to a mobile network using stolen credentials
- As seen above: a lot of mobile phones expose their SIM cards
- A big pool of vulnerable devices available for malicious purposes!

Baseband

What about it?

- Acquire valid authentication vectors from a remote SIM
- What to do with it?
- We can forward authentication to a custom mobile device
- Boring - everyone wants off the shelf phones!
- So let's take a stock baseband firmware and modify it!

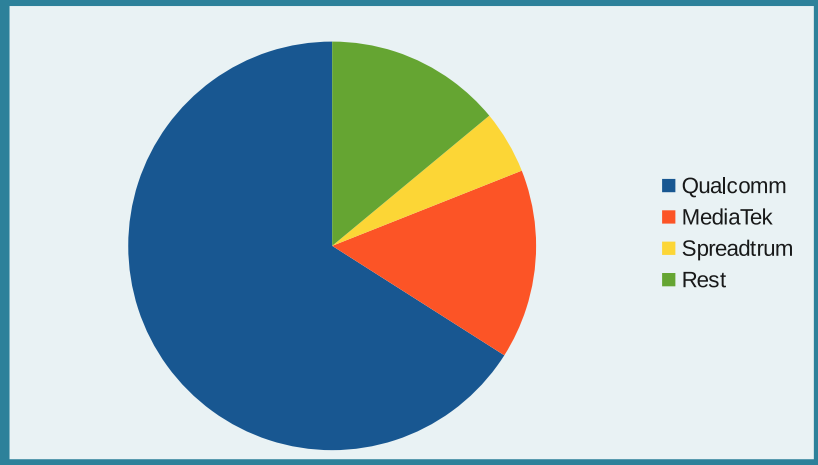
Baseband Overview



- Takes care of communication with the mobile network
- Has direct access to the SIM-Card
- Usually proprietary
- Runs on (somewhat) separate CPU

Baseband Hardware

- Only a few significant vendors: Qualcomm, MediaTek, Spreadtrum, Marvell and Intel
- Focus here: MediaTek Plaforms
- Other BaseBand vendors are more locked down today
- A lot of previous work regarding Qualcomm



Baseband Firmware

- MTK Baseband based on Nucleus RTOS
- Loaded at boot-time by the Android-System running on the AP from "/etc/firmware/modem*.img"
- MTK-Linux-Kernel-Module takes care of it
- Firmware on many MTK-Based-Phones not signed
- Logical separation between Baseband/Modem (BP) and Application-Processor (AP)
- Communication between AP and BP: Shared RAM, UART

Interfaces between AP and BP

- AP and BP are *logically* separated but they have a lot of intersections
- On the AP side exposed as char devices or via kernel (ioctls)
- Modem-RMMI: AT-Commands
- Debug-Output
- Firmware-Control via AP (Reset, Exception Handling, etc.)

CCCI / CCIF

- `CCCI (Cross Core Communication Interface)`: Handles data exchange between AP and BP
- Exposed as different kernel drivers on the AP side
- Character devices (/dev/ccci*)
- Low-Level (CPU to CPU Interface called CCIF according to MTK-Docs) for MT6582:
  - 16 Physical channels (8 AP→MD, 8 MD→AP)
  - One 256bytes dual port SRAM

Baseband Firmware - Structure

- Uncompressed raw binary
- Partial image of the memory space starting at address 0x00000000
- No virtual memory
- Contains a trailer at the end

Baseband Firmware - DEMO

DEMO

Adding Features

Remote SIM

- Goal: Transfer SIM commands to a remote mobile phone – but how?

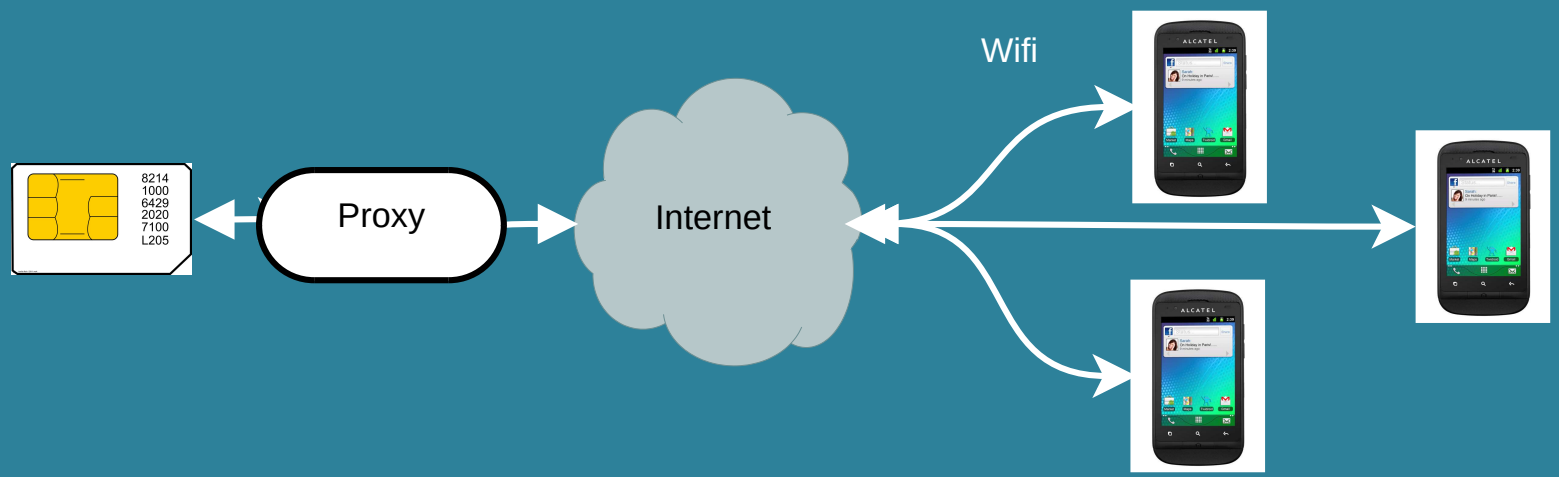
Remote SIM

- Goal: Transfer SIM commands to a remote mobile phone – but how?
- Modern phones have additional communication channels besides the mobile network
  - Bluetooth
  - Dual-SIM
  - Data Connection of a second SIM
  - WiFi

Remote SIM

- Goal: Transfer SIM commands to a remote mobile phone – but how?
- Modern phones have additional communication channels besides the mobile network
  - Bluetooth
  - Dual-SIM
  - Data Connection of a second SIM
  - WiFi
- BT-SAP (Sim-Application-Protocol) - works only for short distances
- SIM commands can travel through unintended channels i.e. over TCP/IP

Remote SIM Concept



ShadowSIM

- Allows usage of remote SIM-Cards
- Download from:
  `https://github.com/shadowsim/shadowsim`

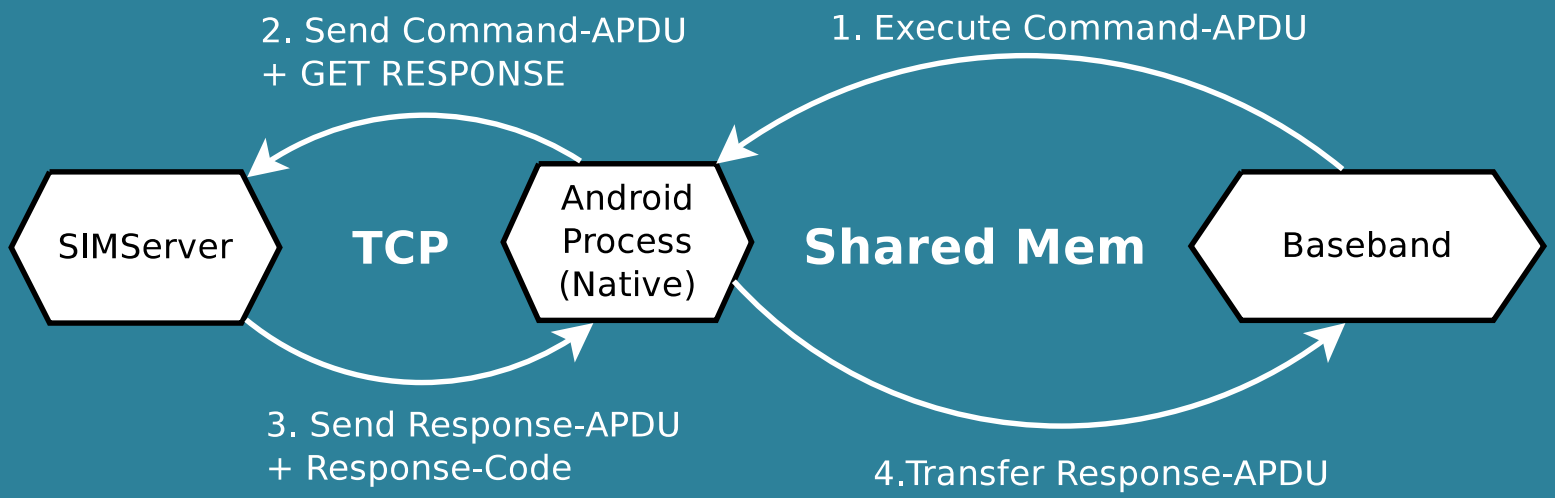
ShadowSIM

- Allows usage of remote SIM-Cards
- Download from:
  `https://github.com/shadowsim/shadowsim`
- Implements a virtual SIM-Card by patching the Baseband-Firmware of a Mediatek 6573 phone:
  1. Identify the code that enables SIM-Access
  2. Change it to send APDUs to the AP and read Response-APDUs

ShadowSIM

- Allows usage of remote SIM-Cards
- Download from:
  `https://github.com/shadowsim/shadowsim`
- Implements a virtual SIM-Card by patching the Baseband-Firmware of a Mediatek 6573 phone:
  1. Identify the code that enables SIM-Access
  2. Change it to send APDUs to the AP and read Response-APDUs
- Implement a native Android-Application that processes APDU-Commands:
  1. Read a Command-APDU sent by the Baseband
  2. Send them over TCP to a remote system having SIM-Access
  3. Write the Response-APDU back to Baseband

ShadowSIM - Concept

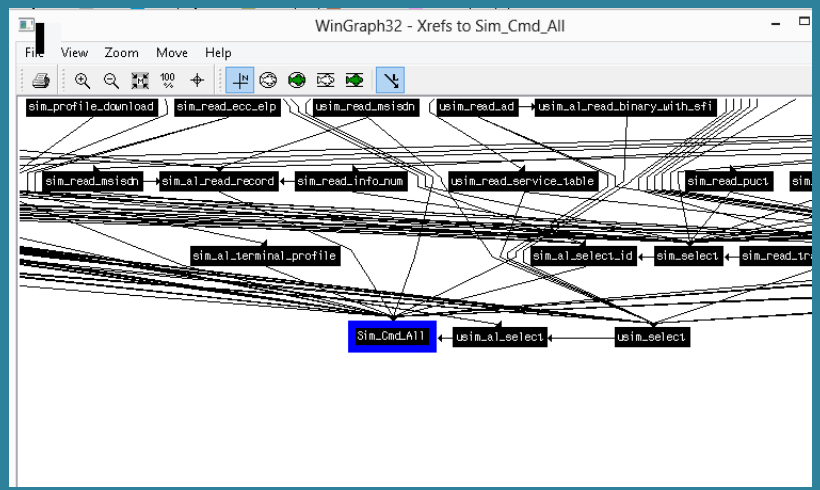


ShadowSIM - Baseband Communication

- First idea: Use one of the UARTs as a communication channel to AP
- Was a bad idea: UART communication is done asynchronous in Baseband, so lots of work writing and registering your own handler
- Easier: Using shared memory
- Vendor application for debugging: mdlogger already uses shared memory for log transfer
- Source code is published, so changing it for our purpose was easy

ShadowSIM - Firmware Modification

- Things that help:
  - Lots of assertions and debugging strings in the code
  - MediaTek firmware for various devices sometimes has Debug-Symbols
  - MediaTek reuses code a lot (made it easier to compare different firmwares)
  - No obfuscation
- In general the code is quite well structured and functionality is abstract - this makes patching easier




```
LDR             R2, =0x363
ADR             R1, aSim_driver__12 ; "sim_driver_interfaces.c"
MOV             R0, R7
ADDS            R2, #0x14
BLX             free_ctrl_buffer_ext
LDR             R2, [SP,#0x68+sim_context_struct]
ADD             R1, SP, #0x68+returned_apdu_code
STR             R1, [SP,#0x68+ptr_returned_apdu_code]
STR             R2, [SP,#0x68+cpy_arg_on_stack]
LDR             R2, [SP,#0x68+stack_r2_saved_ptr_apdu_resp]
LDR             R0, [SP,#0x68+stack_r0_saved_ptr_cmd_apdu]
MOV             R3, ptr_apdu_resp_size
MOV             R1, ptr_cmd_apdu_size
BL              sim_driver_cmd_api
CMP             R0, #0
BEQ             loc_26AA96
```

ShadowSIM - DEMO

DEMO

Wait a Minute

Goodie

Hardening

- What else can we patch?
- Objective: Have a more secure baseband firmware.
- Best way: Create a new one from scratch.
- In the meantime: Patch existing ones.
- Always an improvement for security: Reducing the attack surface
- So let's turn off stuff!

SIM Application Toolkit (STK / SAP)

- Can work outside of user control
- "value addded services"
- OTA commands sent to / via your SIM
- Used for attacks and surveillance
- Probably unwanted in "hostile" environments

Patching - DEMO

DEMO

Conclusion

Results-Recap

- Credentials can be acquired from a SIM card
- On many devices even over USB
- Dual-Use:
  - Bad: Bad guys may steal your network identity
  - Good: New applications that free users from SIM cards, allow them to share SIM cards
- Non-Repudiation is gone for good – a SIM-Card in a mobile phone proves nothing
- When your security model is from the 80s chances are high it doesn't work anymore
- If *YOU* have ideas on what features to add / remove in a baseband firmware, contact me!

THANK YOU