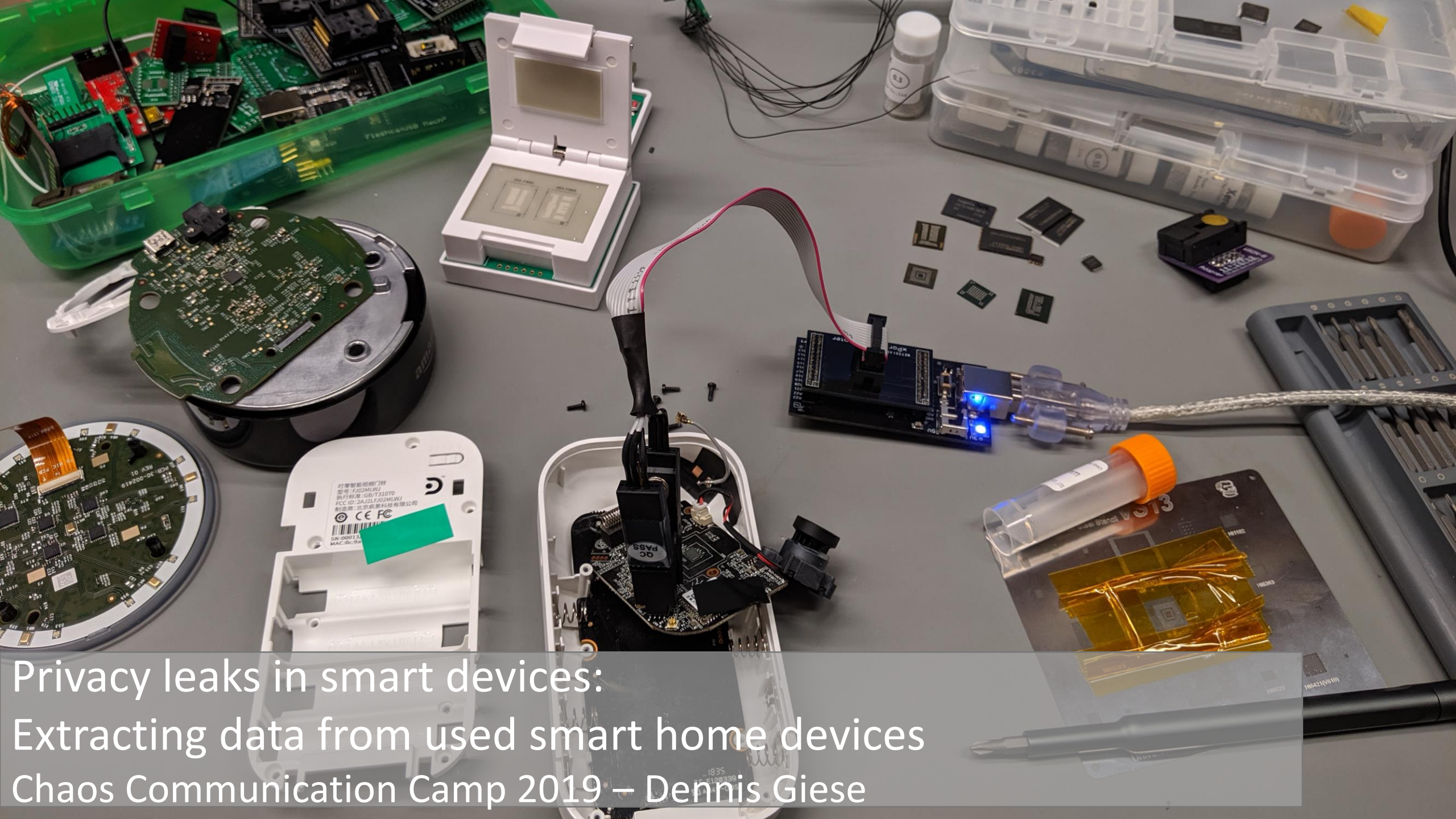


Privacy leaks in smart devices:
Extracting data from used smart home devices

Chaos Communication Camp 2019 – Dennis Giese

Outline

- Motivation
- Data on IoT devices
- Storage on IoT devices
- Reset states of used devices
- Data extraction methods
- Device analysis

About me

- PhD student at Northeastern University, USA
  - Working with Prof. Guevara Noubir@Khoury
- Grad student at TU Darmstadt, Germany
  - Working with Prof. Matthias Hollick@SEEMOO
- Interests: Reverse engineering of interesting devices
  - IoT, Smart Locks
  - Physical Locks ;)

Northeastern University
Khoury College of
Computer Sciences

TECHNISCHE
UNIVERSITÄT
DARMSTADT

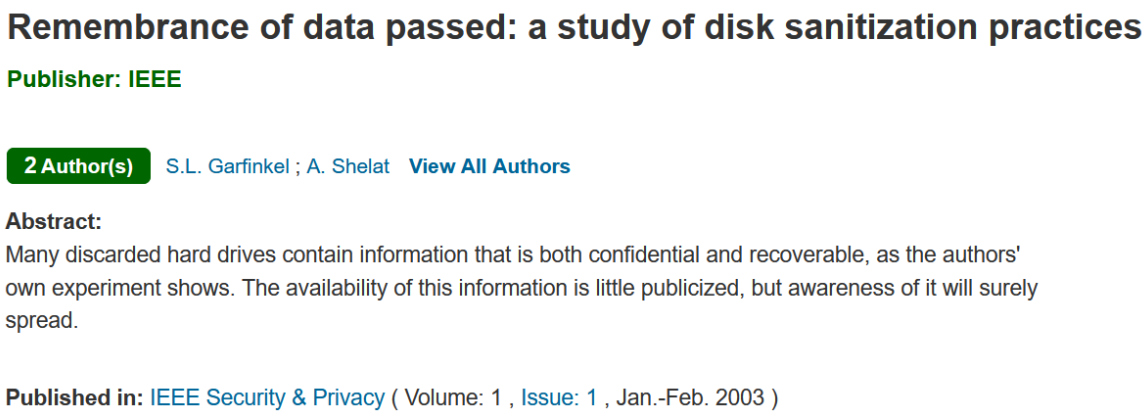
Side notes

- This is not a Xiaomi bashing talk, issues applies to all vendors
- Most methods already well known
- For ethical/legal reasons, I had to censor most of the data
- Use methods on your own risk
- Some technical aspects are simplified
  - NAND flash is more complex, but I simplify there a lot
  - Actual NAND data interpretation/reassembly out of scope
  - Device-specific rooting out of scope

MOTIVATION

Old problem: data on used hard drives

- Traditionally: second hand hard drives contain still data
  - Issue existed forever, increased with platforms like eBay
  - Devices still contain data like: personal information, emails, pictures and other media, sensitive documents
  - Awareness was raised in early 2000's



Remembrance of data passed: a study of disk sanitization practices

Publisher: IEEE

2 Author(s) S.L. Garfinkel ; A. Shelat **View All Authors**

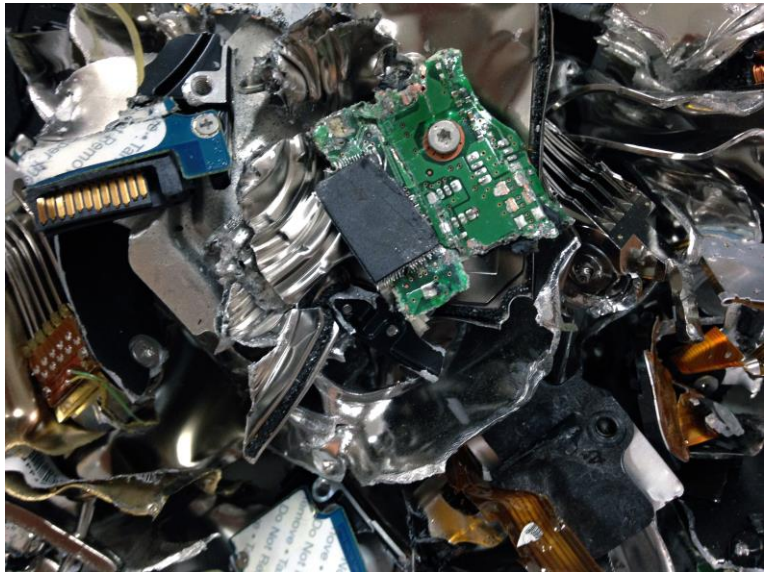
Abstract:
Many discarded hard drives contain information that is both confidential and recoverable, as the authors' own experiment shows. The availability of this information is little publicized, but awareness of it will surely spread.

Published in: IEEE Security & Privacy (Volume: 1 , Issue: 1 , Jan.-Feb. 2003)

https://doi.org/10.1109/MSECP.2003.1176992

Old problem: data on used hard drives

- Also affected: Multifunction printers, Lab instruments
- Published standard: NIST SP 800-88 (2006)
- Solutions:
  - Wipe hard drives
  - Sell used devices without hard drive
- Remaining problems:
  - Lack of knowledge or awareness
  - Carelessness
  - Broken devices



IT Liquidators [CC BY-SA 3.0]
https://commons.wikimedia.org/wiki/File:Destroyed_Hard_Drive.jpg

https://csrc.nist.gov/publications/detail/sp/800-88/archive/2006-09-01

Old problem: data on used hard drives

- Problem still exists today
  - Of 159 second-hand HDD's/SSD's 66 (42%) still contained sensitive data

PC REVIEWS BEST PICKS HOW-TO NEWS SMART HOME BUSINESS SHOP

#SiriRecordings #IceLakeBenchmarked #GalaxyNote10 #BacktoSchool #5G

News & Analysis

Many Used Hard Drives Sold on eBay Still Contain Leftover Data

Data removal company Blannco sponsored a study that analyzed 159 SSD and HDD storage drives purchased on eBay and found that many still contained leftover data from the previous owners.

By Michael Kan April 26, 2019 4:36PM EST

https://www.pcmag.com/news/368029/many-used-hard-drives-sold-on-ebay-still-contain-leftover-da

Smartphones: sensitive data to go

- Phones store much sensitive information:
  - Pictures, Messages, Account credentials, call lists
- Device storages were not encrypted by default
  - Introduced with iOS 8 (2014), Android 6.0 (2015)
- Factory reset was not wiping all the data
  - Paper “Security analysis of android factory resets” (2015): Android < 4.0 does not wipe data correctly
- Addressed with new NIST SP 800-88 Rev. 1 (2014)

http://www.cl.cam.ac.uk/~rja14/Papers/fr_most15.pdf
https://csrc.nist.gov/publications/detail/sp/800-88/rev-1/final

IoT is everywhere

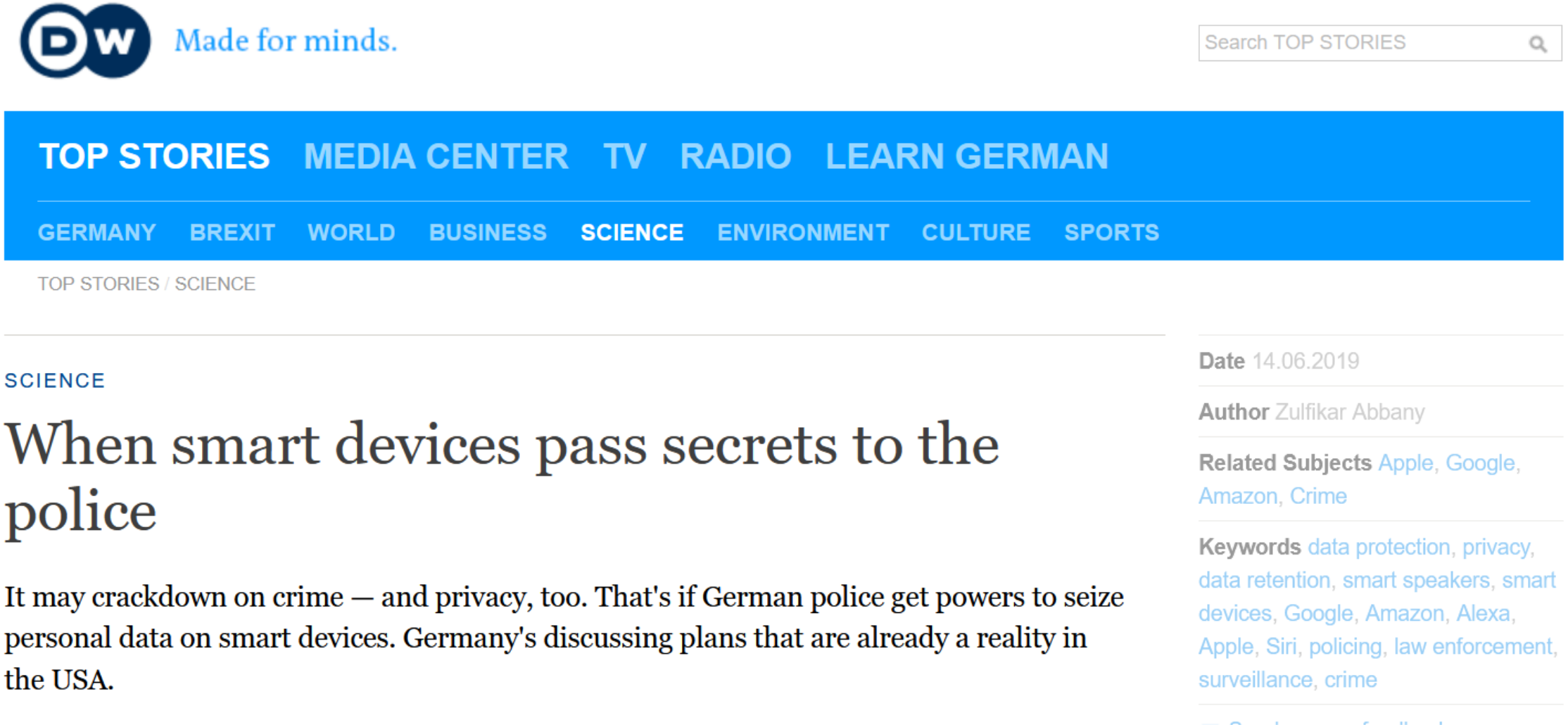
- In contrast to smartphones/PC:
  - Smaller or no user interface
  - Data on device not directly accessible
  - Unclear which data is collected in the first place
- Factory resets
  - not fully verifiable
  - Implementation unclear, depends on version and vendor

Motivation for this talk

- During my master thesis @SEEMOO
  - Analysis of security of many IoT devices
  - Goal: root access to devices
  - When factory resetting devices:
    - traces of data were left on the device
    - sometimes all data was still available

Where you find used devices

- eBay, Amazon Warehouse deals
- flea markets
- Trash
- Family and friends
- (In your home)


Made for minds.
Search TOP STORIES
TOP STORIES MEDIA CENTER TV RADIO LEARN GERMAN
GERMANY BREXIT WORLD BUSINESS SCIENCE ENVIRONMENT CULTURE SPORTS
TOP STORIES / SCIENCE
SCIENCE
When smart devices pass secrets to the police
It may crackdown on crime — and privacy, too. That's if German police get powers to seize personal data on smart devices. Germany's discussing plans that are already a reality in the USA.
Date 14.06.2019
Author Zulfikar Abbany
Related Subjects Apple, Google, Amazon, Crime
Keywords data protection, privacy, data retention, smart speakers, smart devices, Google, Amazon, Alexa, Apple, Siri, policing, law enforcement, surveillance, crime

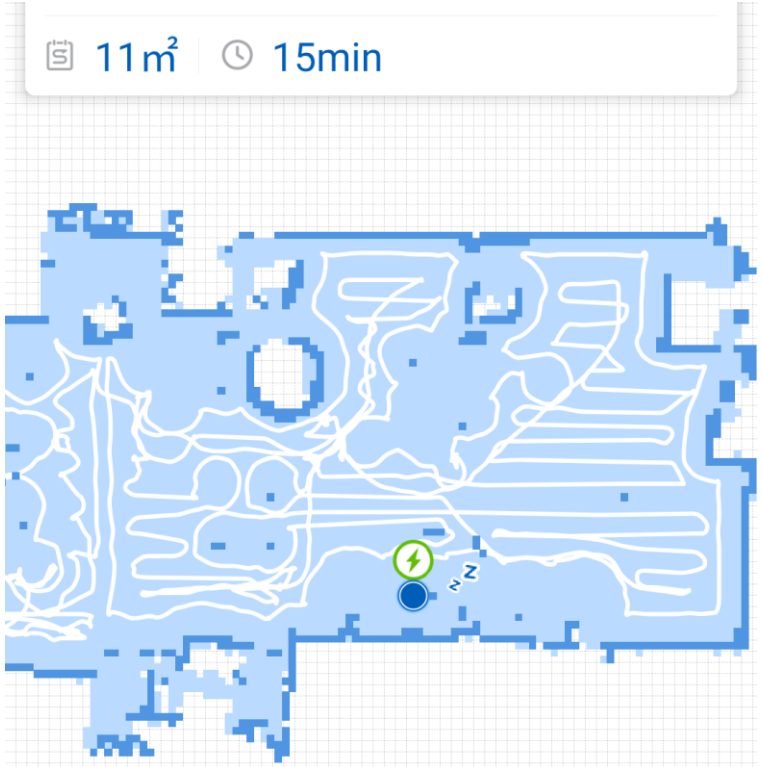
DATA ON IOT DEVICES

Data on IoT devices

- Data on individual devices depending on device type
- All IoT devices require: Wi-Fi credentials, Cloud credentials, Cloud bindings
- Rule of thumb: The more performance/functions/storage a device has, the more data is available on it

Vacuum cleaners

- Connection log files
- Maps
- Cleaning logs
- User ID



Smart Home Gateways

- Connection log files
- Sensor/actuators bindings
- Sensor/actuators log files
- Key material
- User ID

Cameras

- Cached snapshots/video clips
- Recorded video
- Event logs
- User ID
- Cloud storage credentials

Routers

- DHCP leases (MAC, IP, timestamp)
- Firewall configurations
- Media files
- Logfiles (connection, DNS, filters, etc.)
- Other credentials

Media players

- Connection log files
- Media libraries
- Playlists
- Cache
- Browsing history
- Other credentials/tokens
  - Google Play Store
  - Network shares

For ethical reasons I have to skip this device ☹

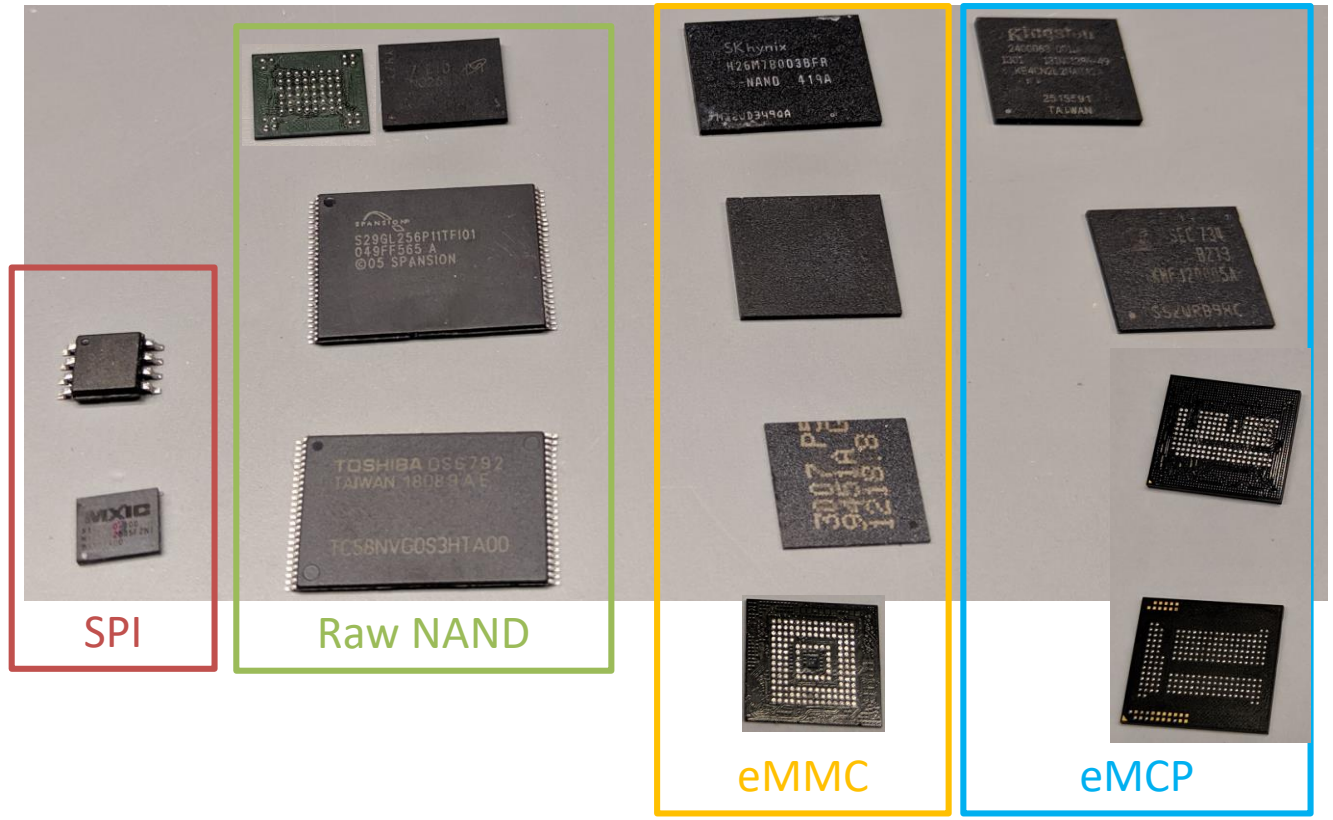
Toys

- WIFI credentials
- Configuration settings
- Video/audio data
- Usage logs

STORAGE ON IOT DEVICES

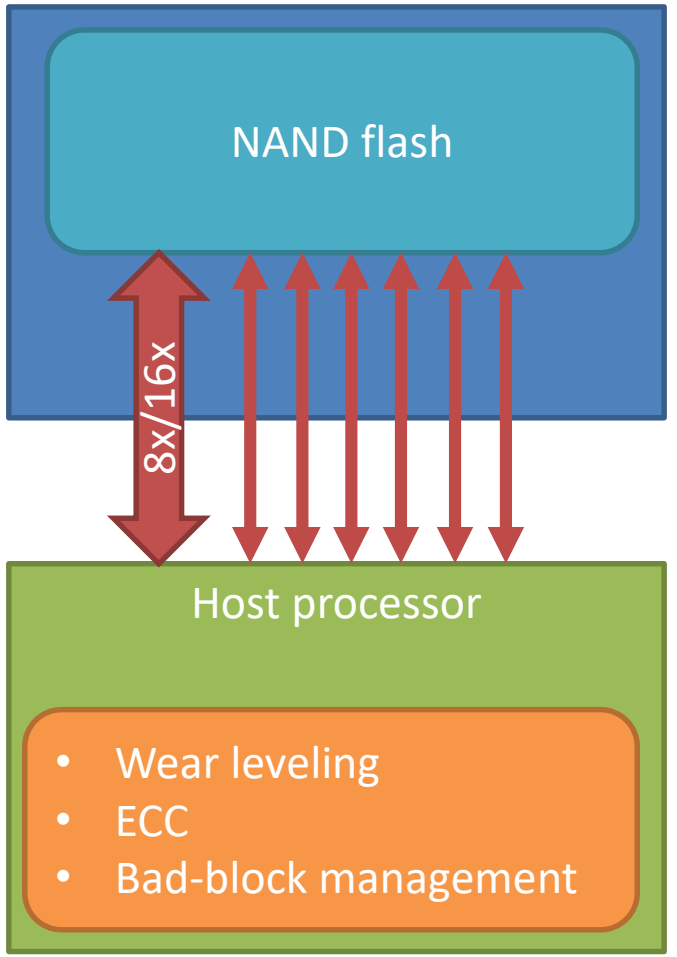
Storage on IoT devices

- 2 groups of storage types:
  - Raw flash
    - serial flash (SPI)
      - NAND
      - (NOR)
    - Raw parallel NAND flash
  - Block devices
    - eMMC
    - eMCP
    - (SD cards)
- Choice of storage type affects useable filesystems



Raw NAND flash

- SPI flash: typically sizes < 64MByte
  - Packages: SOP8, WSON8,…
- Raw NAND: typically 128MByte – 4GByte
  - Packages: TSOP-48, TSOP-56, BGA-63
- Cheap and fast storage, but Bit-errors
- Host processor/OS tasks:
  - Wear leveling
  - ECC (sometimes CPU accelerated)
  - Bad-Block management
- Abstraction under Linux
  - MTD subsystem (Memory Technology Devices)
  - Character device -> Block device



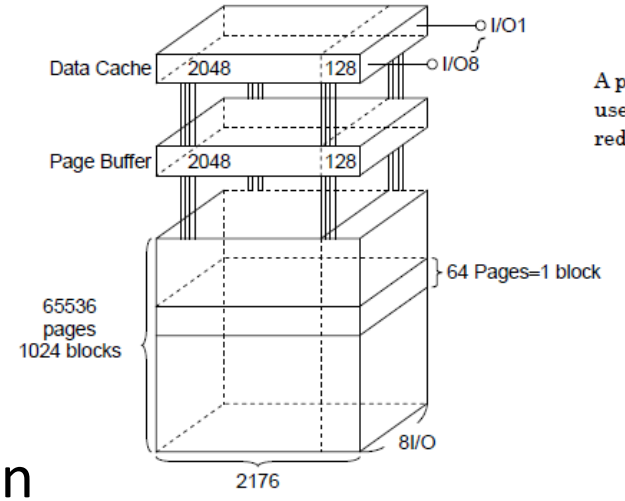
Raw NAND flash properties

- organized in blocks and pages
  - To erase data, a whole block needs to be erased
  - Erasing sets all bits to 1
  - Typical block sizes: 16-512 Kbytes
  - Typical page size: 0.5-2 Kbyte
  - Programming works on page level
  - OOB: management + ECC
- Flash contains additional spare blocks
- ECC is computed by Host CPU
  - Sometimes vendor specific computation

TOSHIBA TC58NVG0S3HTA00

Schematic Cell Layout and Address Assignment

The Program operation works on page units while the Erase operation works on block units.



A page consists of 2176 bytes in which 2048 bytes are used for main memory storage and 128 bytes are for redundancy or for other uses.

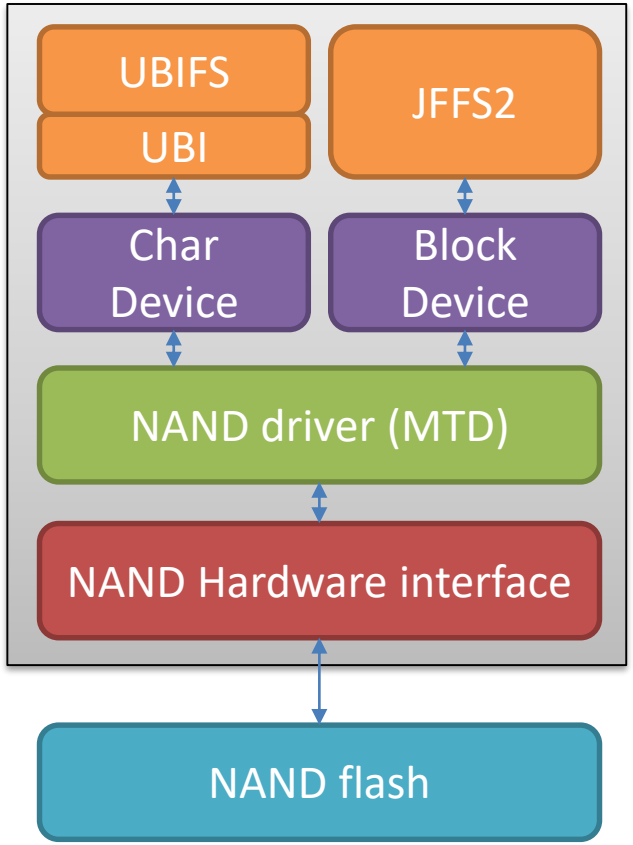
1 page = 2176 bytes

1 block = 2176 bytes × 64 pages = (128K + 8K) bytes

Capacity = 2176 bytes × 64pages × 1024 blocks

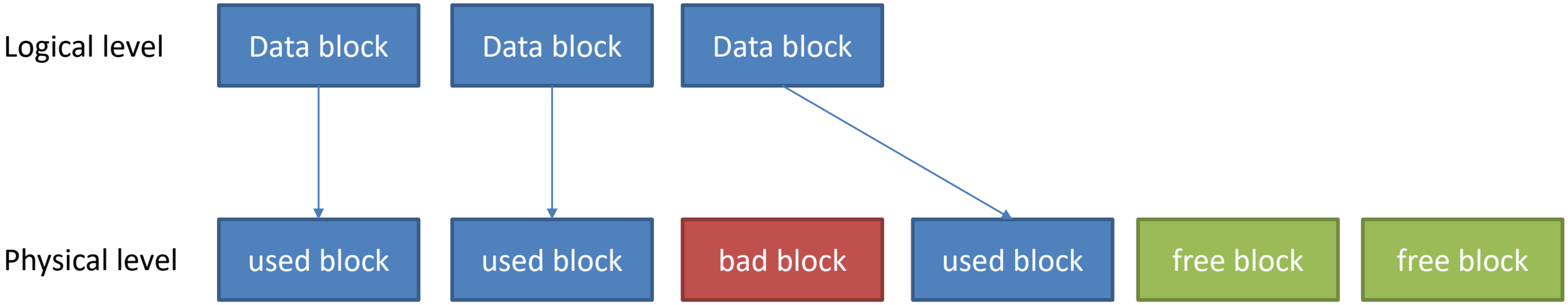
Wear-leveling for raw flash

- Problem: individual flash cell has limited writes
  - File-systems like Ext2/3/4 are not wear-leveling aware
  - Many writes can destroy the flash or corrupt the data
- Solution: Flash aware file-systems or additional layer
  - File-System (on partition level only): YAFFS, JFFS/JFFS2
  - Additional layer (on device level): UBI+UBIFS
  - Support of Bad-Block management and Wear leveling in OS
  - Idea:
    - Deleted blocks are not erased, but only marked as such
    - The changed information is copied into a new block
    - Garbage collector may clean up erased blocks if needed



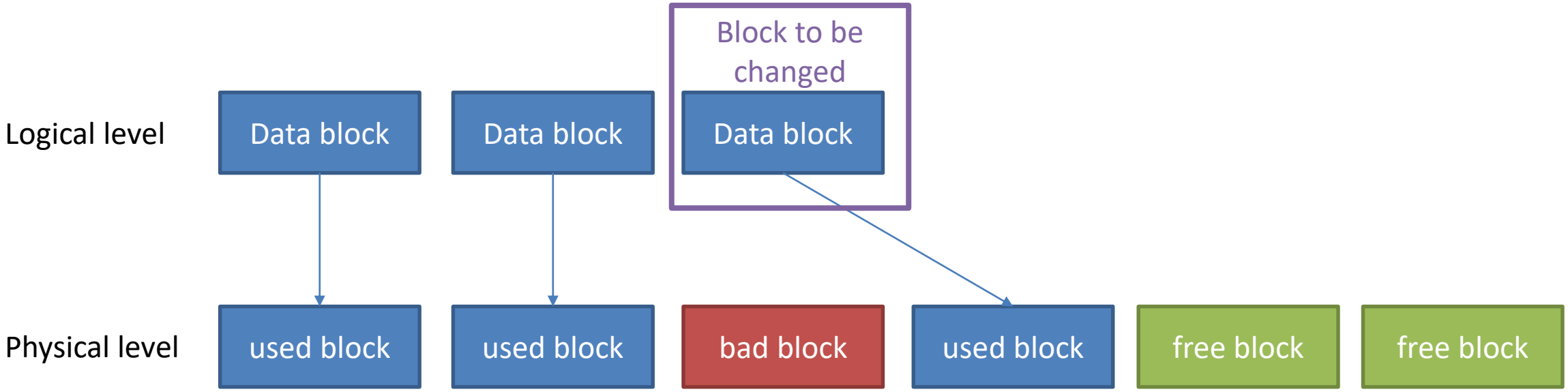
How Wear-leveling works

Simplified!

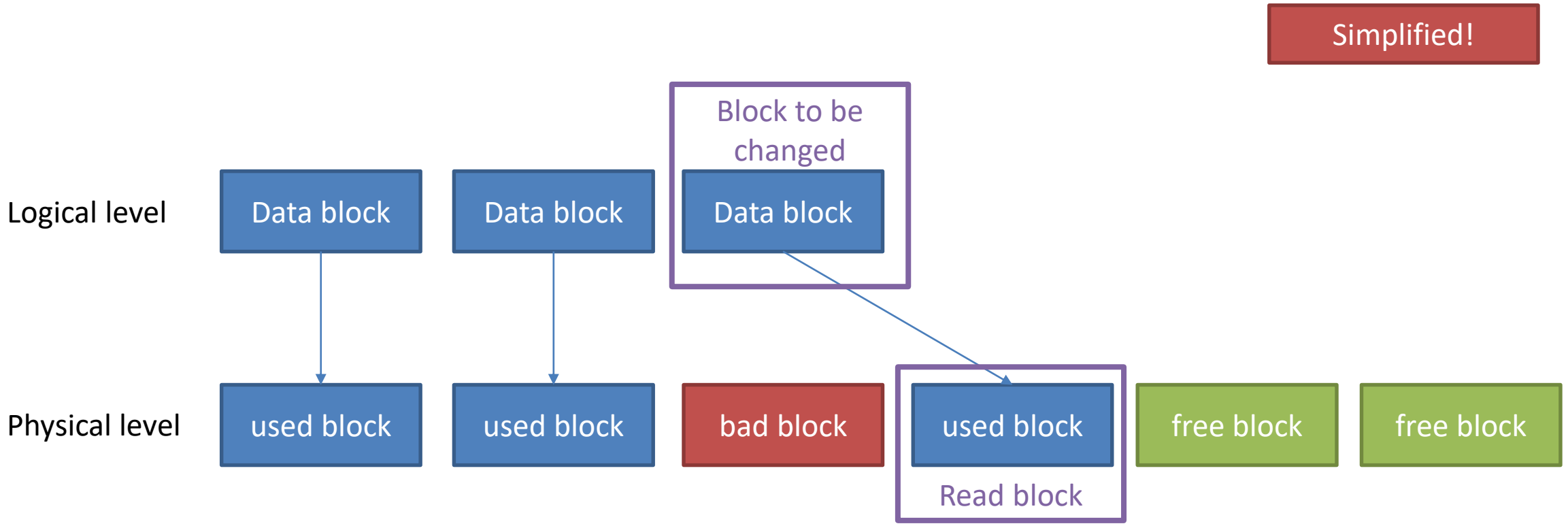


How Wear-leveling works

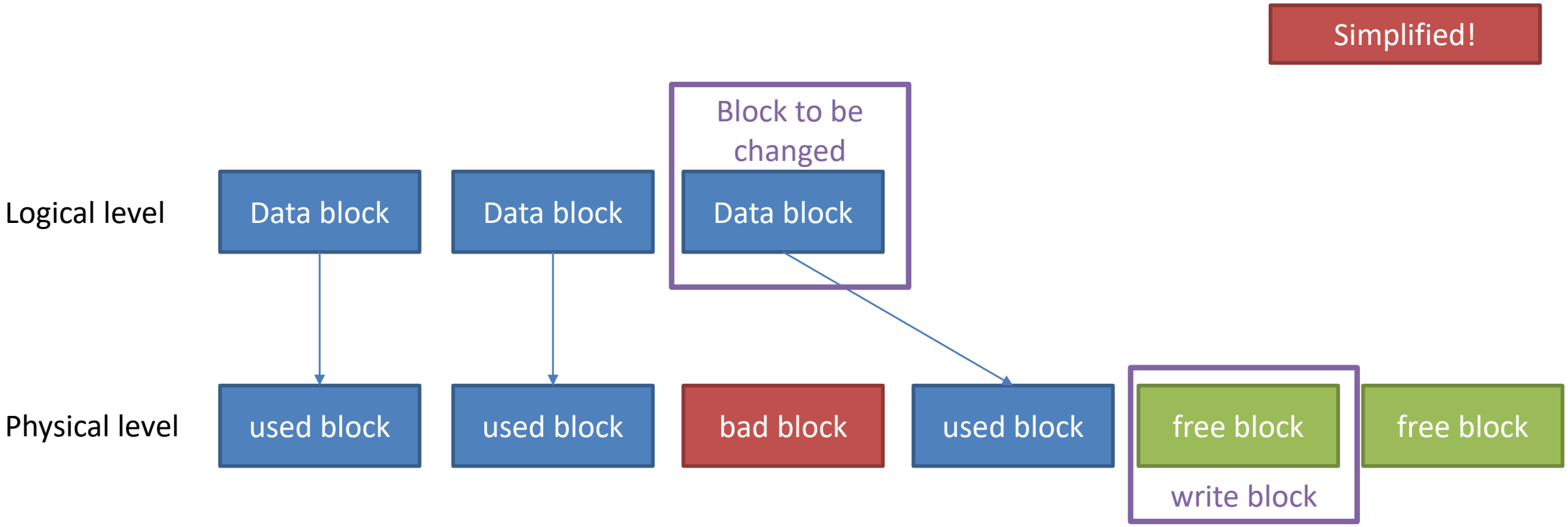
Simplified!



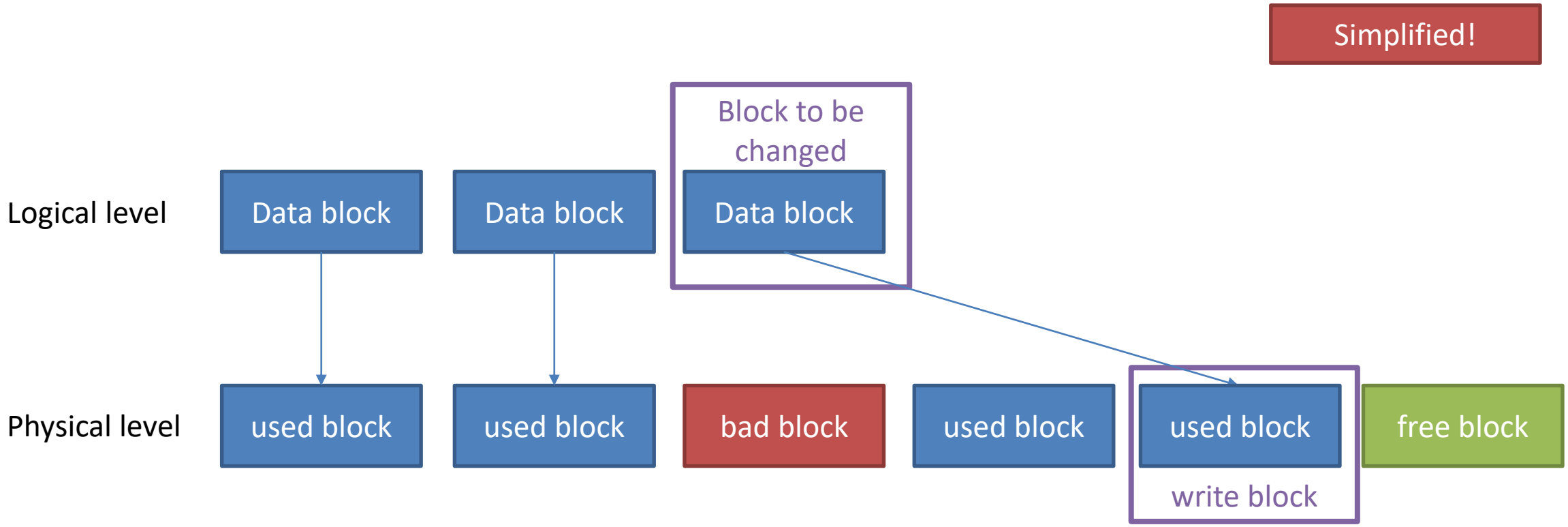
How Wear-leveling works



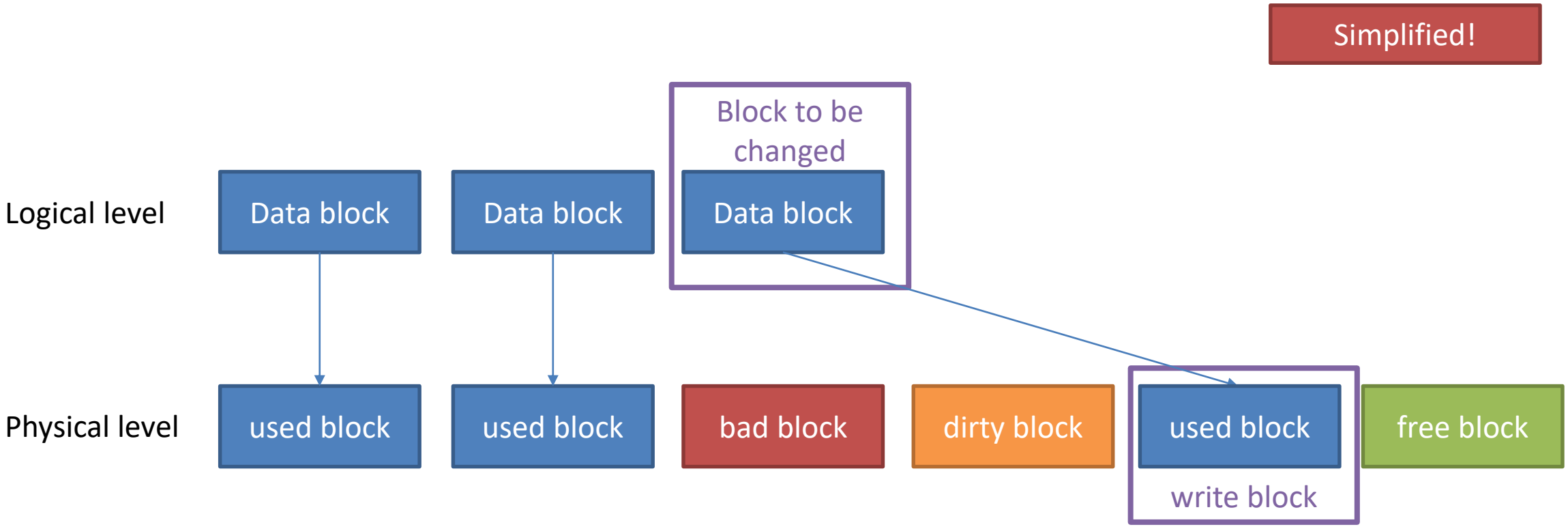
How Wear-leveling works



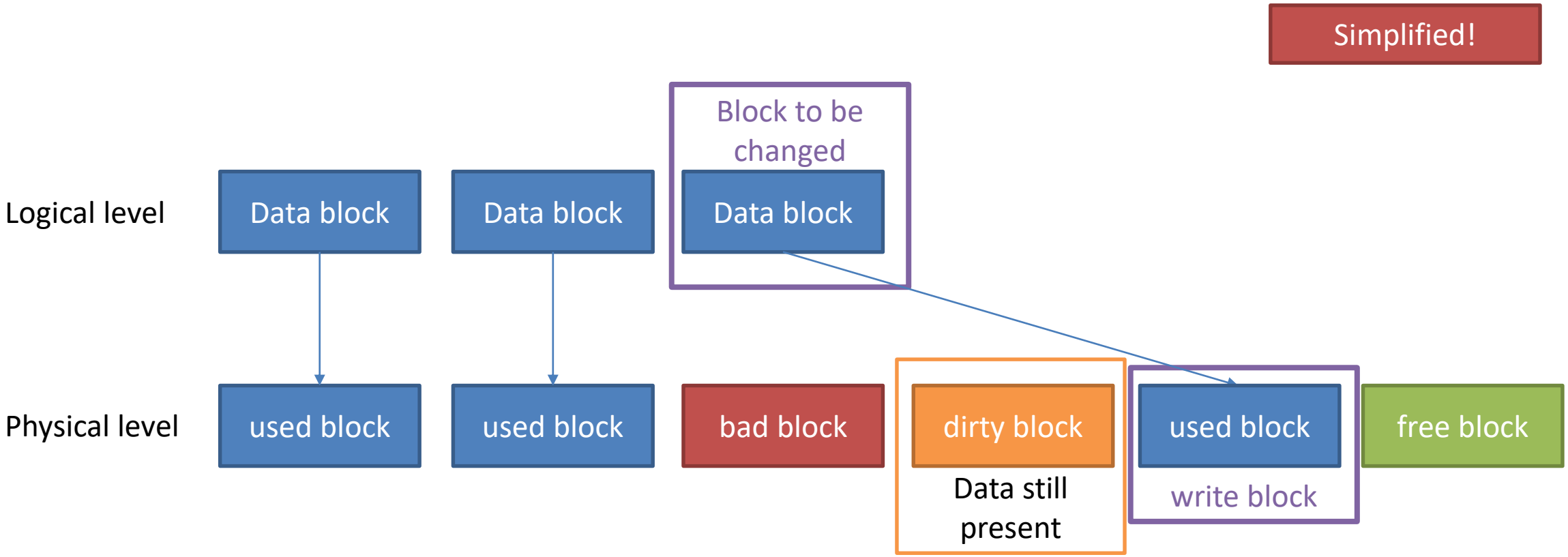
How Wear-leveling works



How Wear-leveling works



How Wear-leveling works



Interesting Wear-leveling properties

- Multiple copies of the data may exist
  - Data is not being erased as long as the block is not erased
  - Size of copies usually > 2KByte
  - Data changed regularly exists more often

"History" of changes remains

Recommended material about NAND

- Blackhat USA 2014: “Reverse Engineering Flash Memory for Fun and Benefit” by Jeong Wook (Matt) Oh
  - Intro in the communication protocol
  - Soldering/Unsoldering of NAND flash
  - How-to reverse engineer NAND formats

    https://www.blackhat.com/docs/us-14/materials/us-14-Oh-Reverse-Engineering-Flash-Memory-For-Fun-And-Benefit-WP.pdf
    https://www.blackhat.com/docs/us-14/materials/us-14-Oh-Reverse-Engineering-Flash-Memory-For-Fun-And-Benefit.pdf
- “From NAND chip to files” by Jean-Michel Picod

  https://www.j-michel.org/blog/2014/05/27/from-nand-chip-to-files

Sidenote

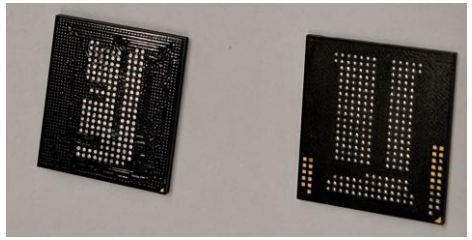
- Even device manufacturers are not aware of JFFS2 properties
- Example of leaked developer keys from my DC26 talk

```
0004cc10  e3 b5 3b e8 00 2c 23 20  63 61 74 20 2f 65 74 63  |..;..,# cat /etc|
0004cc20  2f 6d 69 69 6f 2f 64 65  76 69 63 65 2e 63 6f 6e  |/miio/device.con|
0004cc30  66 0a 23 20 64 69 64 20  6d 75 73 74 20 62 65 20  |f.# did must be |
0004cc40  61 20 75 6e 73 69 67 6e  65 64 20 69 6e 74 0a 23  |a unsigned int.#|
0004cc50  20 6b 65 79 29 70 00 00  4e 73 74 72 69 6e 67 0a  | key)p..Nstring.|
0004cc60  23 0a 64 69 64 3d 35 30  36 30 33 36 35 XX 0a 6b  |#.did=5060365.k|
0004cc70  65 79 3d 4e 41 37 4e 69  6d 4b 6f XX XX XX XX XX  |ey=NA7NimKo     |
0004cc80  69 58 6e 0a 6d 61 63 3d  32 38 3a 36 43 3a 30 37  |iXn.mac=28:6C:07|
0004cc90  3a 32 45 3a XX XX 3a XX  XX 0a 76 65 6e 64 6f 72  |:2E:      .vendor|
0004cca0  3d 6c 75 6d 69 0a 23 20  6d 6f 64 65 6c 20 6d 61  |=lumi.# model ma|
0004ccb0  78 20 6c 65 6e 20 32 33  0a 80 02 94 03 00 02 2e  |x len 23........|
0004ccc0  63 61 6d 65 72 61 2e 61  71 31 0a 70 32 70 5f 69  |camera.aq1.p2p_i|
0004ccd0  64 3d 41 2c 00 00 03 30  31 31 31 41 0a 11 00 00  |d=A,...0111A....
```

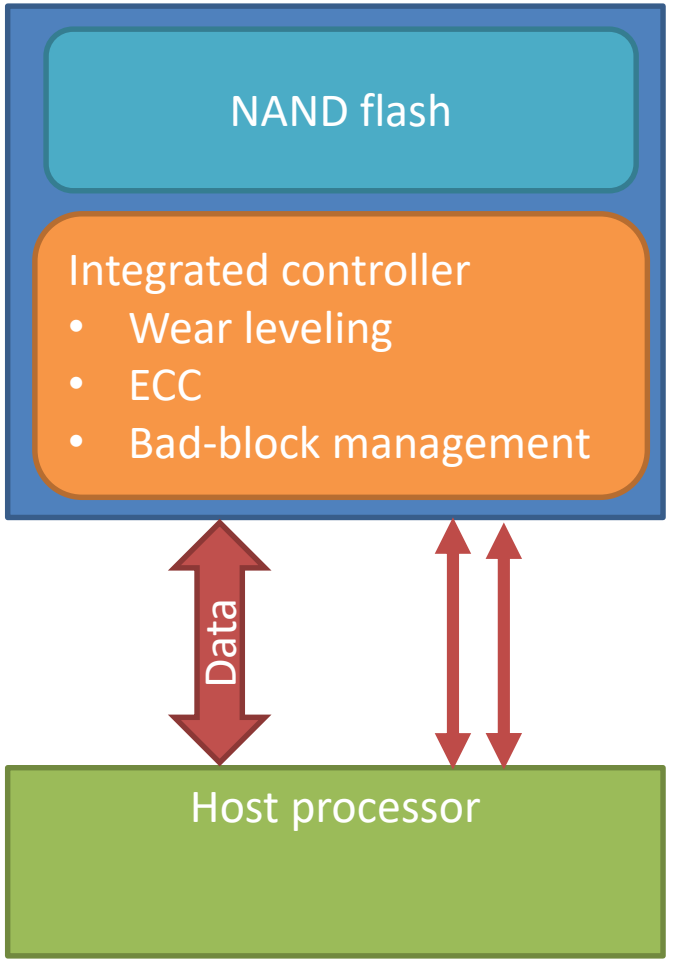
Block devices

- Also known as managed NAND
- Standards: eMMC 4.x, 5.x
- eMMC:
  - flash with integrated controller
  - Packages: FBGA-153

- eMCP
  - like eMMC, but with on-chip DRAM
  - Advantage: RAM + flash on one chip
  - Packages: FBGA-162, 221



- Under Linux: normal block storage device, supports Ext2/3/4
- Integrated wear-leveling, ECC, bad-block management by FTL



Access to deleted data

- eMMC controller does not allow raw access as for raw NAND
- eMMC/eMCP use raw NAND internally
  - Bypassing of the eMMC controller and direct attachment to NAND possible
  - Challenge: the data format of the eMMC controller
- Recommended talk: “eMMC CHIPS. DATA RECOVERY BEYOND CONTROLLER” by Rusolut
  - Summary: Even is eMMC is deleted, data is still present on internal NAND flash

https://rusolut.com/wp-content/uploads/2018/10/eMMCvsNAND.pdf

RESET STATES OF USED DEVICES

Reset states of used devices

- Reset states mostly depends on previous owner
  - No reset at all
    - Device still contains all data and configuration
    - Most probable cause: missing knowledge how to reset or device broken
  - Wi-Fi configuration reset
    - Device may contain data, but is in un-provisioned state
    - Many devices offer only a Wi-Fi reset, e.g. initiated by button
  - Device wipe
    - All data has been wiped, device is factory state (e.g. firmware)
    - There might be still traces of data
    - Not all devices support this

Wi-Fi reset vs. Device wipe

- Some devices support both
  - Wi-Fi reset is usually marked by special button
  - Device wipe is available via the App or via button combination
- Idea of Wi-Fi reset
  - Device can be reconnected quickly to a new Wi-Fi SSID
  - No long duration to reset
  - Most settings remain
- Device wipe: Should erase all user data and restore factory OS

DATA EXTRACTION METHODS

Data extraction methods

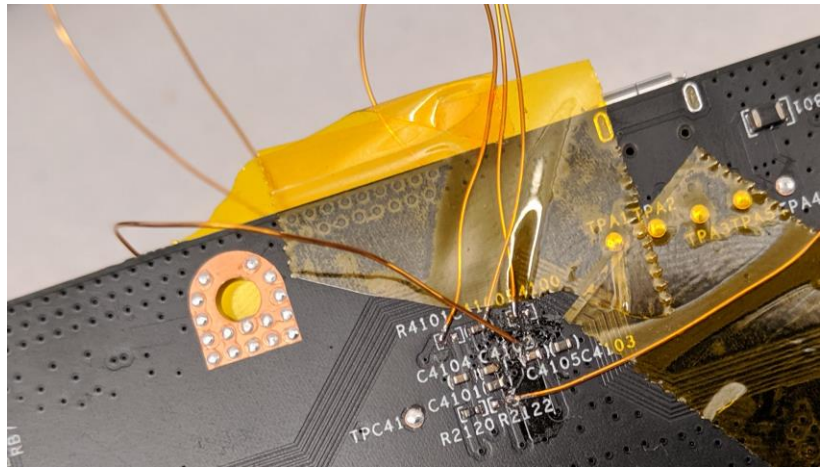
- Idea: extract all available data
- Methods:
  - Via software using root access
  - Dumping flash contents without de-soldering (ISP)
  - Dumping flash contents with de-soldering (Chip-Off)

Software methods

- For many IoT devices there are public rooting methods
  - Installation of custom firmware
  - Access via USB or UART
  - Boot via external media (e.g. SD card)
- Dumping flash contents using dd
  - dd is not flash aware, that is helpful in our use-case
  - Extraction of the data via external media, SSH or netcat
- Works good for JFFS2/UBIFS
- Disadvantage: low-level access on flash might be limited

Dumping flash without de-soldering

- Works mainly for SPI and eMMC flash
- Some devices allow In-system programming (ISP)
  - Flash can be accessed via test pins
  - Processor must not interfere
- Advantage: reduced risk of destroying hardware
- Disadvantage: requires knowledge of test pins or PCB traces

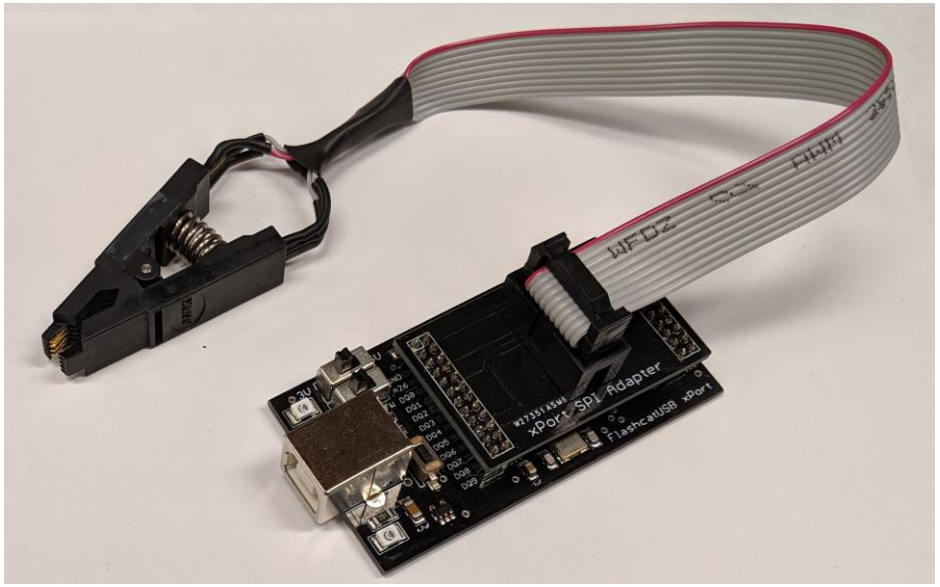


Dumping flash with de-soldering

- Works for all flash chips
- De-soldering:
  - Preheating of the whole PCB recommended
  - For accessible pins (e.g. TSOP): create low temperature alloy
  - For BGA chips: Hot air, IR or reflow soldering station required
- Disadvantage for BGA chips: re-soldering requires re-balling
- Raw NAND: requires the use of adapters due to pin count

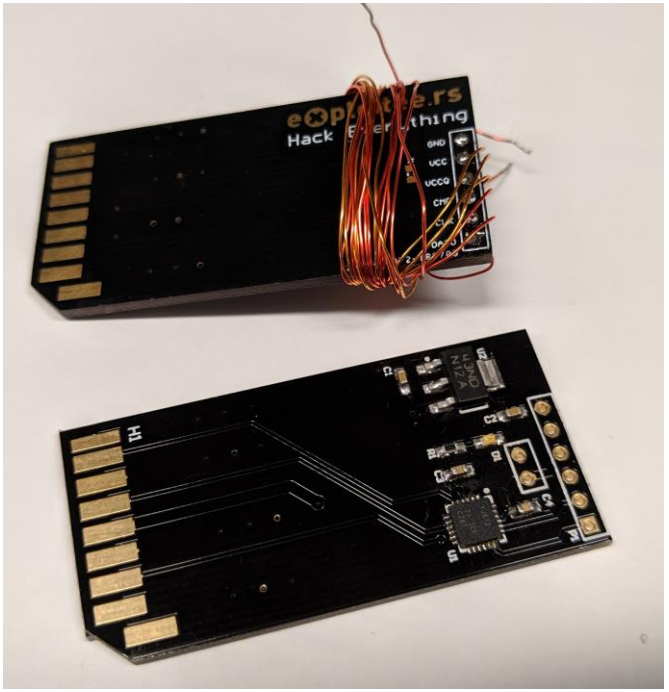
Tools for SPI flash

- Any device which supports bit-banging on GPIOs
  - Raspberry Pi, Arduino, Bus pirate, etc.
- My favorite: Flashcat USB



Tools for eMMC flash

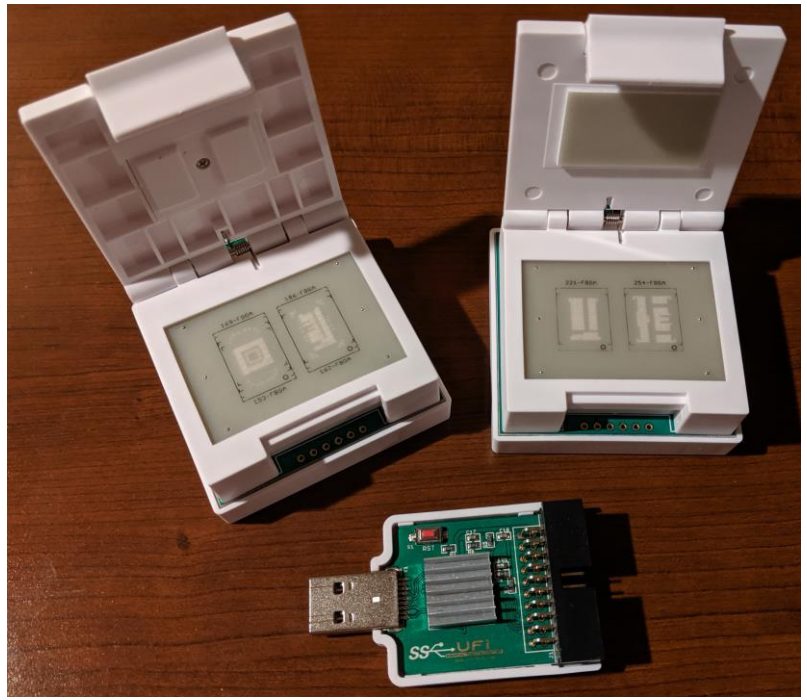
- (Some eMMC chips require a lower voltage!)
- Exploitee.rs eMMC adapters (~10USD)
  - Connection via SD card reader
  - Difficulty:
    - requires soldering with microscope or good eyes
    - May not be able to access all partitions

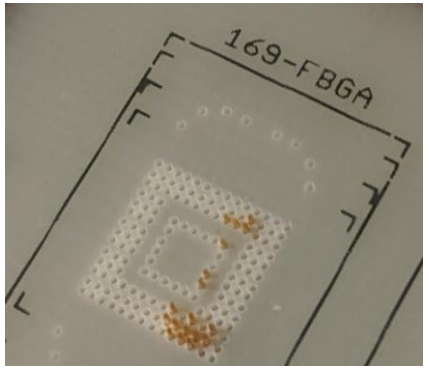


https://www.exploitee.rs/index.php/Exploitee.rs_Low_Voltage_e-MMC_Adapter

Tools for eMMC flash

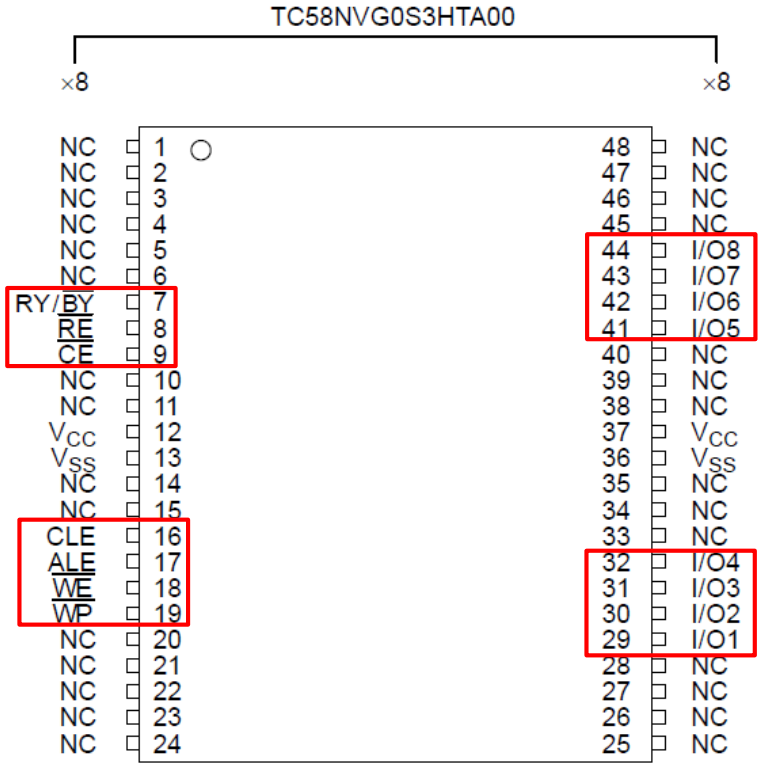
- UFI Box Lite with BGA sockets (~75USD)
  - Simple connection to BGA chips
  - Supports dual voltage chips
  - Disadvantages:
    - Needs many tries to find correct position for BGA
    - Original software for Windows questionable (detected as Malware)





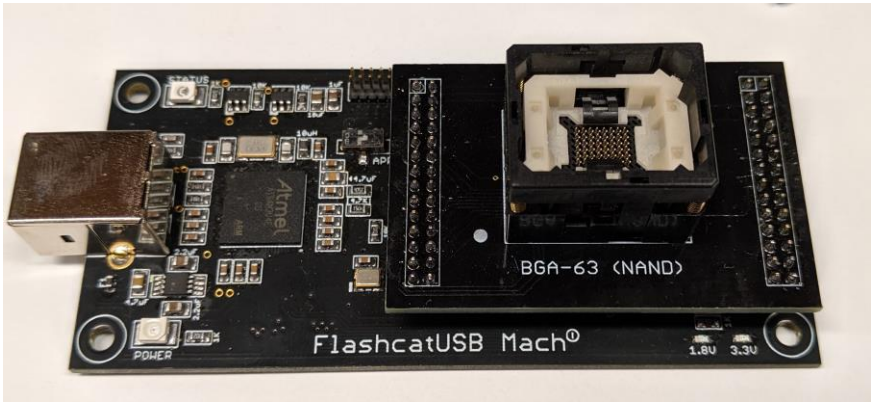
Tools for raw NAND

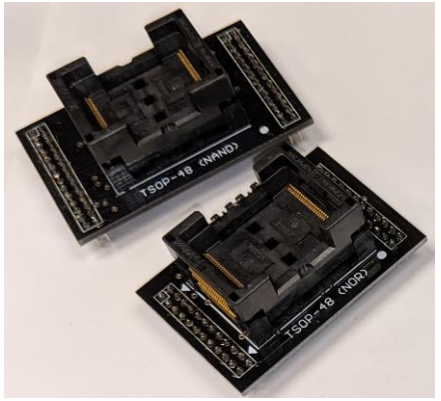
- High pin count makes soldering difficult
- Requires some sort of NAND controller
  - See also: “Reverse Engineering Flash Memory for Fun and Benefit“



Tools for raw NAND

- Flashcat USB with adapters
  - Supports all kind of raw flash chips
  - However: does not interpret proprietary ECC/OOB data





Tools for raw NAND

- Evaluation boards
  - Idea: soldering the flash on a board with similar SOC and OS
  - Enables read and writes for supported flash chips
  - Disadvantage: Boards are not always available

Analyzing the dumps

- Binwalk
- Hex editor
- raw NAND dumps:
  - Dumpflash by Jeong Wook (Matt) Oh https://github.com/ohjeongwook/DumpFlash
  - Nand-dump-tool by Jean-Michel Picod https://bitbucket.org/jmichel/tools
  - Problem: exotic OOB sizes and ECC data
- UBI images: UBIFS Dumper https://github.com/nlitsme/ubidump/blob/master/README.md
- JFFS2 images: Jefferson https://github.com/sviehb/jefferson

DEVICE ANALYSIS

Methods used

- Disassemble devices and dump flash
- Powering on devices and root devices (if possible)
- Connecting devices to the App
- Using devices and reset them
- Compare available data before and after reset

Ecovacs DEEBOT 900

- Brought 2019, factory reset by previous owner
- Platform:
  - OS: Linux
  - SOC: Rockchip RV1108
  - Flash: Toshiba NAND in TSOP48 (128MByte)
  - RAM: 128 Mbyte DDR3
- Approach:
  - Dumping NAND flash
  - Connecting over UART

Ecovacs verification of factory reset

- After unsoldering and dumping flash:
  - Confirmation that device has been factory reset

```
0  30 31 20 30 37 3A 33 38  3A 32 39 20 55 54 43 20  32 30 31 39 3A 20 66 61  Fri Feb 01 07:38:29 UTC 2019: fa
5  73 65 74 20 64 6F 6E 65  20 21 0A 00 00 00 00 00  00 00 00 00 00 00 00 00  ctory reset done !
```

  - Fragments of Logfiles, Keys, Maps and Wi-Fi credentials found

```
 32 30 39 38 2C 22 73  69 64 65 62 72 75 73 68  22 3A 32 32 39 31 2C 22  {"brush":2098,"sidebrush":2291,"
 38 7D 00 00 00 00 00  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  hepa":2098}
```

  - Problem: Unknown OOB method
- UART on the board is read-only, however found information:
  - Rockchip specific NAND driver with custom wear-leveling
  - Root partition uses SquashFS, Data partition uses EXT4

Ecovacs: extraction of credentials

- Factory credentials

```
0505EC40  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
0505EC60  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  FF FF FF FF FF FF FF FF  ÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿÿ
0505EC80  7B 0A 09 22 73 65 72 76  69 63 65 73 22 3A 5B 0A  09 09 7B 0A 09 09 09 09  22 74 79 70 65 22 3A 22  {  "services":[   {     "type":"
0505ECA0  57 49 46 49 22 2C 0A 09  09 09 09 22 73 73 69 64  22 3A 22 65 63 6F 76 61  63 73 5F 66 61 63 74 6F  WIFI",     "ssid":"ecovacs_facto
0505ECC0  72 79 5F 39 32 30 22 2C  0A 09 09 09 09 22 70 61  73 73 70 68 72 61 73 65  22 3A 22 65 63 6F 76 61  ry_920",     "passphrase":"ecova
0505ECE0  63 73 66 61 63 74 6F 72  79 66 61 63 74 6F 72 79  65 63 6F 76 61 63 73 22  0A 09 09 7D 0A 09 5D 0A  csfactoryfactoryecovacs"   }  ]
0505ED00  7D 0A 00 00 00 00 00 00  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  }
```

- Previous Owner

```
05E158A0  7B 0A 22 73 65 72 76 69  63 65 73 22 3A 5B 0A 7B  0A 09 22 74 79 70 65 22  3A 22 57 49 46 49 22 2C  { "services":[ {  "type":"WIFI",
05E158C0  0A 09 22 73 73 69 64 22  3A 22 57 4C 41 4E 2D 78  78 78 78 78 22 2C 0A 09  22 70 61 73 73 70 68 72  "ssid":"WLAN-xxxxx",  "passphr
05E158E0  61 73 65 22 3A 22 78 78  78 78 78 78 78 78 78 78  78 78 78 78 78 22 0A 7D  2C 0A 7B 0A 09 22 74 79  ase":"xxxxxxxxxxxxxxx" },  {  "ty
05E15900  70 65 22 3A 22 57 49 46  49 22 2C 0A 09 22 73 73  69 64 22 3A 22 57 4C 41  4E 2D 79 79 79 79 79 79  pe":"WIFI",  "ssid":"WLAN-yyyyyy
05E15920  22 2C 0A 09 22 70 61 73  73 70 68 72 61 73 65 22  3A 22 79 79 79 79 79 79  79 79 79 79 79 79 79 79  ",  "passphrase":"yyyyyyyyyyyyyy
05E15940  79 79 22 0A 7D 2C 0A 7B  0A 09 22 74 79 70 65 22  3A 22 57 49 46 49 22 2C  0A 09 22 73 73 69 64 22  yy" },  {  "type":"WIFI",  "ssid"
05E15960  3A 22 65 63 6F 76 61 63  73 5F 66 61 63 74 6F 72  79 5F 39 32 30 22 2C 0A  09 22 70 61 73 73 70 68  :"ecovacs_factory_920",  "passph
05E15980  72 61 73 65 22 3A 22 65  63 6F 76 61 63 73 66 61  63 74 6F 72 79 66 61 63  74 6F 72 79 65 63 6F 76  rase":"ecovacsfactoryfactoryecov
05E159A0  61 63 73 22 0A 7D 0A 5D  0A 7D 00 00 00 00 00 00  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  acs" } ] }
```

Ecovacs: network connection logs

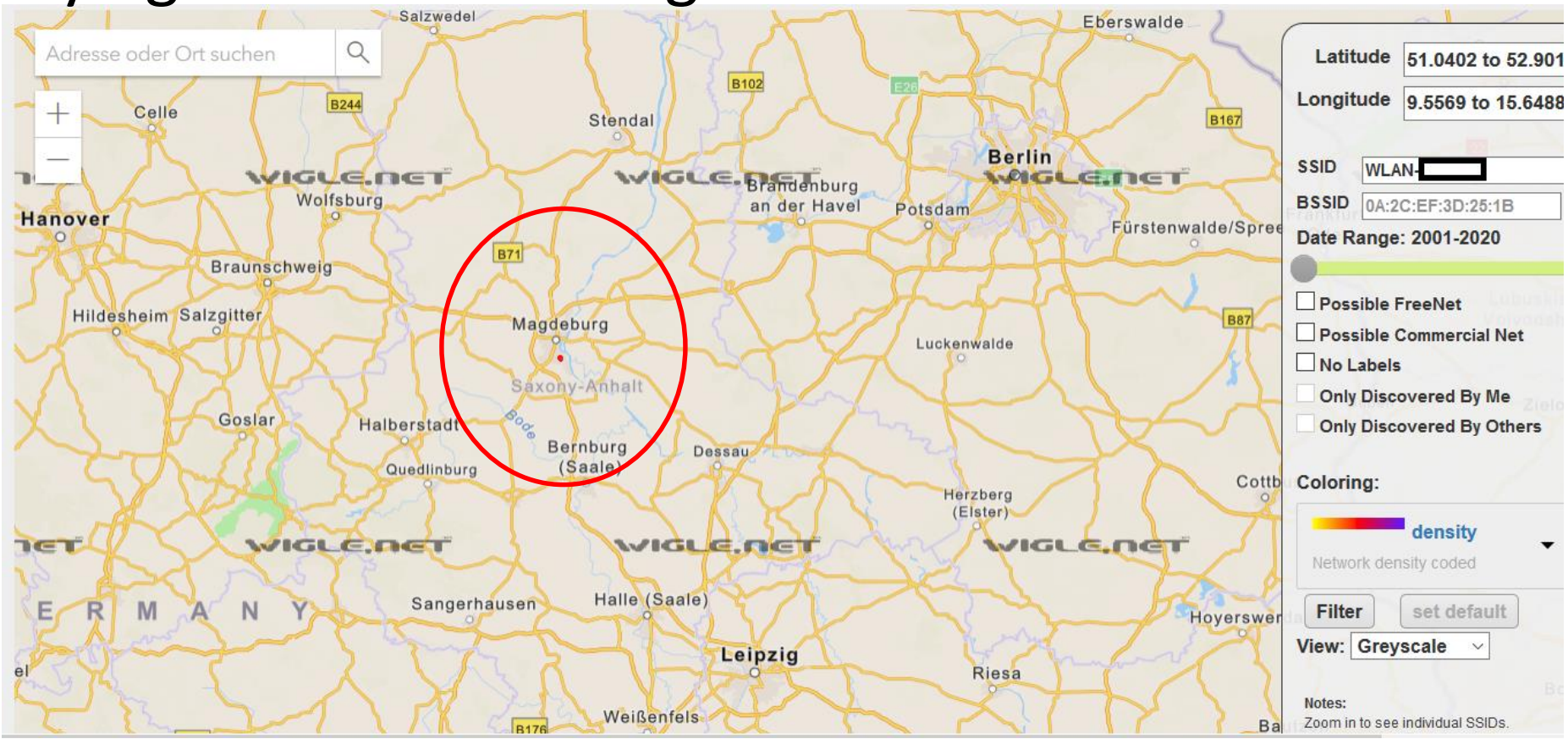
```
05E12A00  53 61 74 20 4E 6F 76 20  32 34 20 31 38 3A 35 31  3A 32 39 20 55 54 43 20  32 30 31 38 20 77 6C 61  Sat Nov 24 18:51:29 UTC 2018 wla
05E12A20  6E 30 20 61 70 5F 6D 6F  64 65 20 73 74 61 72 74  0A 53 61 74 20 4E 6F 76  20 32 34 20 31 38 3A 35  n0 ap_mode start Sat Nov 24 18:5
05E12A40  31 3A 33 34 20 55 54 43  20 32 30 31 38 20 77 6C  61 6E 30 20 73 73 69 64  3D 45 43 4F 56 41 43 53  1:34 UTC 2018 wlan0 ssid=ECOVACS
05E12A60  5F 78 78 78 78 20 61 70  5F 6D 6F 64 65 20 65 6E  64 0A 53 61 74 20 4E 6F  76 20 32 34 20 31 38 3A  _xxxx ap_mode end Sat Nov 24 18:
05E12A80  35 32 3A 31 32 20 55 54  43 20 32 30 31 38 20 20  3A 20 61 64 64 20 61 62  3A 63 64 3A 65 66 3A 67  52:12 UTC 2018  : add ab:cd:ef:g
05E12AA0  68 3A 69 6C 3A 6B 6D 20  31 39 32 2E 31 36 38 2E  30 2E 31 30 30 20 47 61  6C 61 78 79 2D 78 78 78  h:il:km 192.168.0.100 Galaxy-xxx
05E12AC0  78 78 78 78 0A 53 61 74  20 4E 6F 76 20 32 34 20  31 38 3A 35 32 3A 31 35  20 55 54 43 20 32 30 31  xxxx Sat Nov 24 18:52:15 UTC 201
05E12AE0  38 20 77 6C 61 6E 30 20  73 74 61 5F 6D 6F 64 65  0A 53 61 74 20 4E 6F 76  20 32 34 20 31 38 3A 35  8 wlan0 sta_mode Sat Nov 24 18:5
05E12B00  32 3A 32 34 20 55 54 43  20 32 30 31 38 20 77 6C  61 6E 30 20 6F 6E 6C 69  6E 65 20 57 4C 41 4E 2D  2:24 UTC 2018 wlan0 online WLAN-
05E12B20  78 78 78 78 78 78 0A 54  68 75 20 4E 6F 76 20 32  39 20 31 35 3A 33 39 3A  33 30 20 55 54 43 20 32  xxxxxx Thu Nov 29 15:39:30 UTC 2
05E12B40  30 31 38 20 77 6C 61 6E  30 20 6F 6E 6C 69 6E 65  20 57 4C 41 4E 2D 78 78  78 78 78 78 0A 54 68 75  018 wlan0 online WLAN-xxxxxx Thu
05E12B60  20 4E 6F 76 20 32 39 20  31 36 3A 34 35 3A 33 37  20 55 54 43 20 32 30 31  38 20 77 6C 61 6E 30 20  Nov 29 16:45:37 UTC 2018 wlan0
05E12B80  6F 6E 6C 69 6E 65 20 57  4C 41 4E 2D 78 78 78 78  78 78 0A 53 61 74 20 44  65 63 20 31 35 20 30 36  online WLAN-xxxxxx Sat Dec 15 06
05E12BA0  3A 32 36 3A 34 38 20 55  54 43 20 32 30 31 38 20  77 6C 61 6E 30 20 69 64  6C 65 0A 53 61 74 20 44  :26:48 UTC 2018 wlan0 idle Sat D
05E12BC0  65 63 20 31 35 20 30 36  3A 32 36 3A 35 31 20 55  54 43 20 32 30 31 38 20  77 6C 61 6E 30 20 6F 6E  ec 15 06:26:51 UTC 2018 wlan0 on
05E12BE0  6C 69 6E 65 20 57 4C 41  4E 2D 78 78 78 78 78 78  0A 53 61 74 20 44 65 63  20 32 32 20 32 30 3A 35  line WLAN-xxxxxx Sat Dec 22 20:5
05E12C00  31 3A 32 31 20 55 54 43  20 32 30 31 38 20 20 3A  20 61 64 64 20 61 62 3A  63 64 3A 65 66 3A 67 68  1:21 UTC 2018  : add ab:cd:ef:gh
05E12C20  3A 69 6A 3A 6B 6C 20 31  39 32 2E 31 36 38 2E 30  2E 31 30 30 20 47 61 6C  61 78 79 2D 78 78 78 78  :ij:kl 192.168.0.100 Galaxy-xxxx
05E12C40  78 78 78 0A 53 61 74 20  44 65 63 20 32 32 20 32  30 3A 35 31 3A 32 33 20  55 54 43 20 32 30 31 38  xxx Sat Dec 22 20:51:23 UTC 2018
05E12C60  20 77 6C 61 6E 30 20 73  74 61 5F 6D 6F 64 65 0A  53 61 74 20 44 65 63 20  32 32 20 32 30 3A 35 31  wlan0 sta_mode Sat Dec 22 20:51
05E12C80  3A 33 31 20 55 54 43 20  32 30 31 38 20 77 6C 61  6E 30 20 6F 6E 6C 69 6E  65 20 57 4C 41 4E 2D 78  :31 UTC 2018 wlan0 online WLAN-x
05E12CA0  78 78 78 78 78 0A 53 61  74 20 44 65 63 20 32 32  20 32 30 3A 35 31 3A 33  31 20 55 54 43 20 32 30  xxxxx Sat Dec 22 20:51:31 UTC 20
05E12CC0  31 38 20 77 6C 61 6E 30  20 69 64 6C 65 0A 53 61  74 20 44 65 63 20 32 32  20 32 30 3A 35 31 3A 33  18 wlan0 idle Sat Dec 22 20:51:3
05E12CE0  32 20 55 54 43 20 32 30  31 38 20 77 6C 61 6E 30  20 6F 6E 6C 69 6E 65 20  57 4C 41 4E 2D 79 79 79  2 UTC 2018 wlan0 online WLAN-yyy
05E12D00  79 79 79 00 00 00 00 00  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  yyy
```

Ecovacs: locating former owner

- Google's Geolocation API useful
  - Input: 2 MAC addresses and signal strength
  - Output: Location coordinates with accuracy rating
- Device contained only one BSSID in the log files ☹

Ecovacs: locating former owner

- After querying both SSIDs in wigle.net



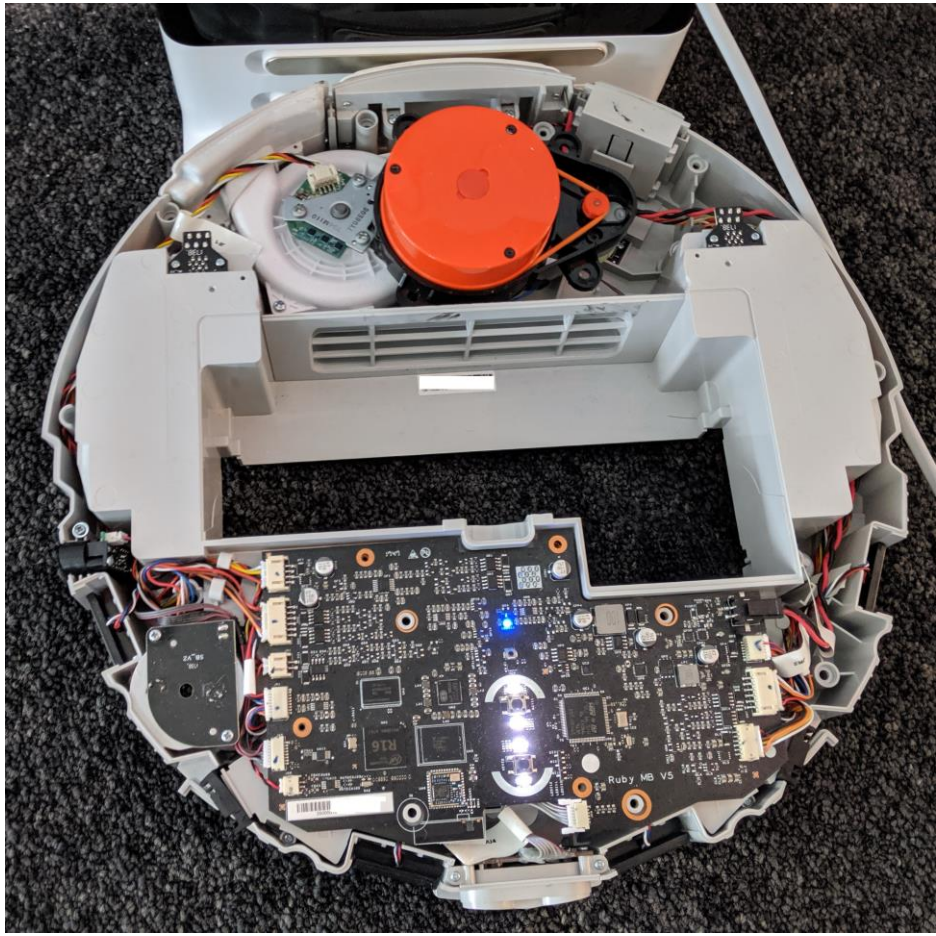
Screenshot from Website: Wigle.net

Ecovacs: summary

- Most of the user-data still exists on device
  - XMPP network logs, Maps, Credentials
- However: due to custom Map format reassembly difficult
- After resetting the device 3 times: Data fragments still readable
- Interesting aspect: factory logs were still stored
- Previous owner could be tracked
- Good news: App did not leak previous maps
- Very similar results with VIOMI Vacuum Robot V2

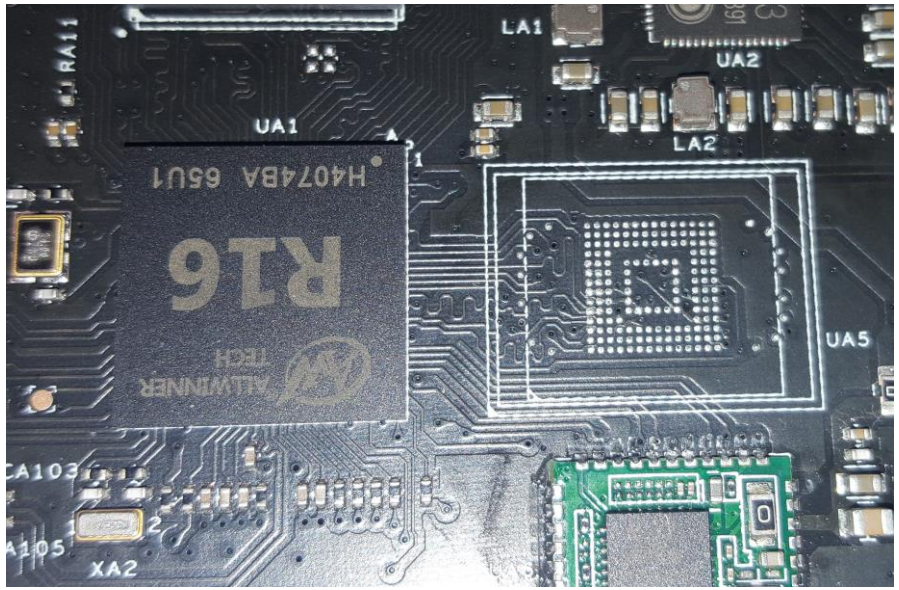
Xiaomi/Rockrobo Mi Vacuum Robot

- From 2018, unclear condition of device
- Platform:
  - OS: Ubuntu 14.04
  - SOC: Allwinner R16
  - Flash: eMMC (4GByte)
  - RAM: 512 Mbyte DDR3
- Approach:
  - Dumping partitions via UART
  - Connect device to cloud account



Mi Vacuum Robot data extraction

- Rooting methods exist
  - Root shell via UART or custom firmware
  - Extraction of data via SSH
- Alternative: removing and dumping of the eMMC flash



https://github.com/dgiese/dustcloud

eMMC layout

Label	Content	Mountpoint	Format
boot-res	bitmaps & some wav files		Ext4
env	uboot cmd line		Text
app	device.conf (DID, key, MAC), adb.conf, vinda	/mnt/default/	Ext4
recovery	fallback copy of OS		Ext4
system_a	copy of OS (active by default)	/	Ext4
system_b	copy of OS (passive by default)		Ext4
Download	temporary unpacked OS update	/mnt/Download	Ext4
reserve	config + calibration files, blackbox.db	/mnt/reserve/	Ext4
UDISK	logs, maps, Wi-Fi config, userID	/mnt/data	Ext4

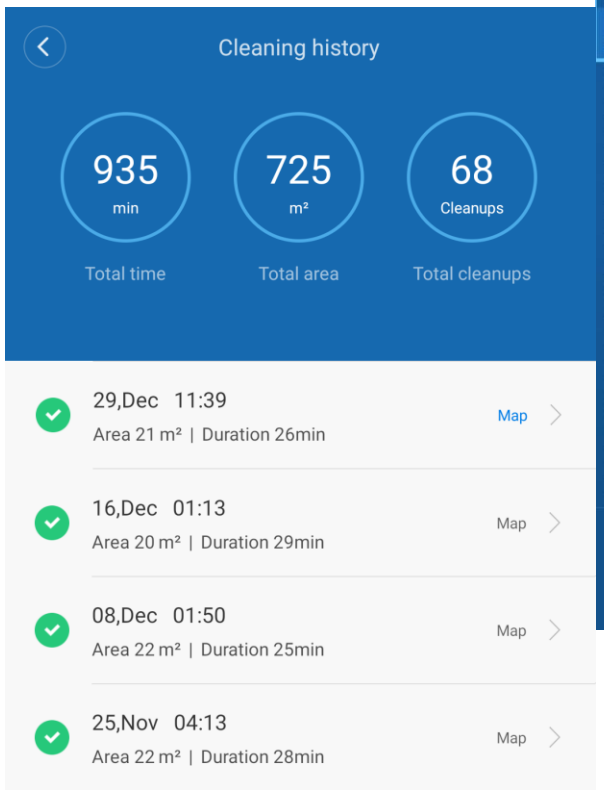
We are interested in this

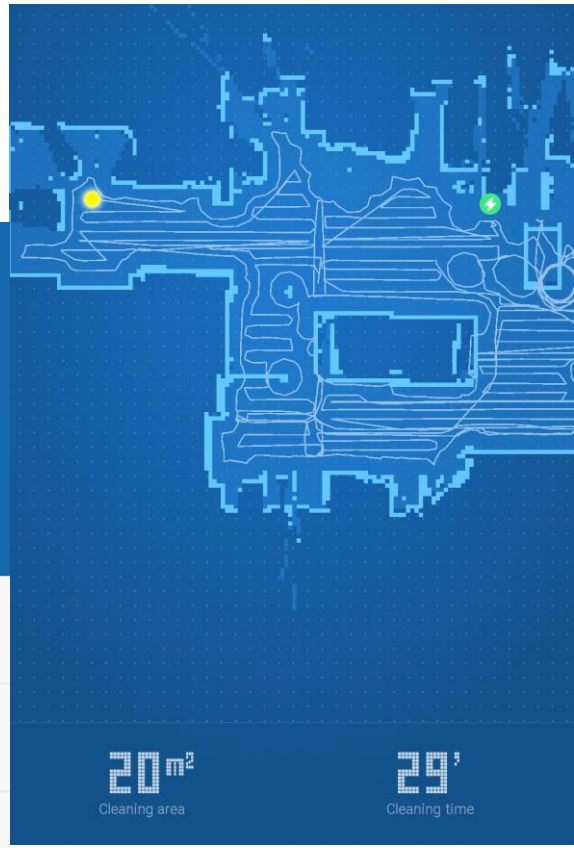
Mi Vacuum Robot reset methods

- Devices support Wi-Fi reset and Factory reset
- Wi-Fi reset: file with Wi-Fi credentials is deleted
- Factory reset:
  - Requires special procedure, mentioned in the manual
  - OS partitions are restored from Recovery
  - Data partition is formatted, but not wiped
  - Partition with usage data is not erased

Mi Vacuum Robot

- After provisioning of device with new account
  - previous data visible in App
  - Assumption: only Wi-Fi reset
  - Data reuploaded to the Cloud
  - Logfiles locally available
- After factory reset:
  - Maps were not visible anymore

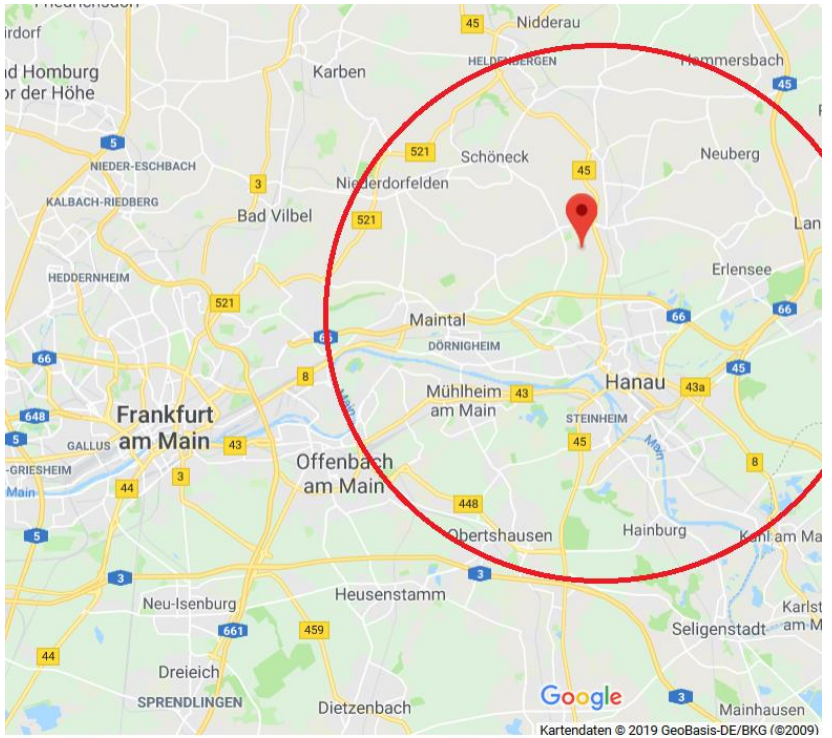




Mi Vacuum Robot: locating former owner

- Log files contained 2 BSSIDs
  - Google Geolocation API returned coordinates
- Wi-Fi credentials reveal part of address
  - Password contains personal data
- User-ID
  - Search via Mi Home App
  - Share device with user to reveal name


```
{
  "location": {
    "lat": 50.17[redacted],
    "lng": 8.90[redacted]
  },
  "accuracy": 20
}
```

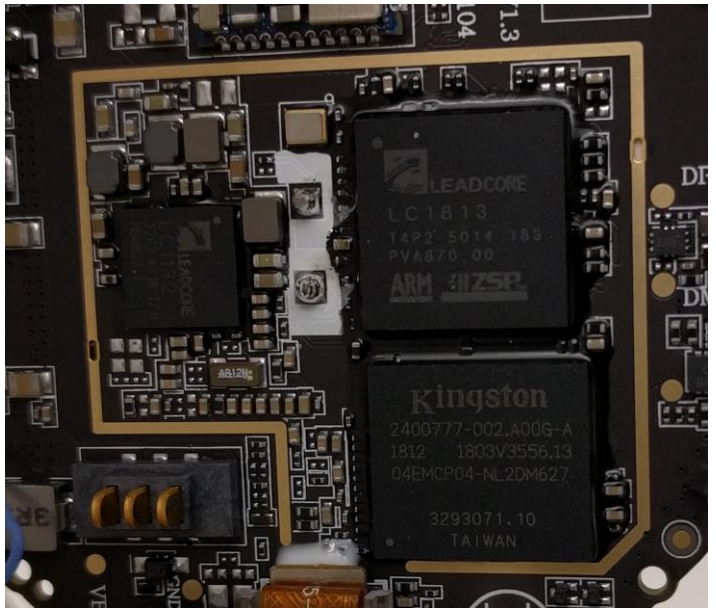
Source: Google Maps

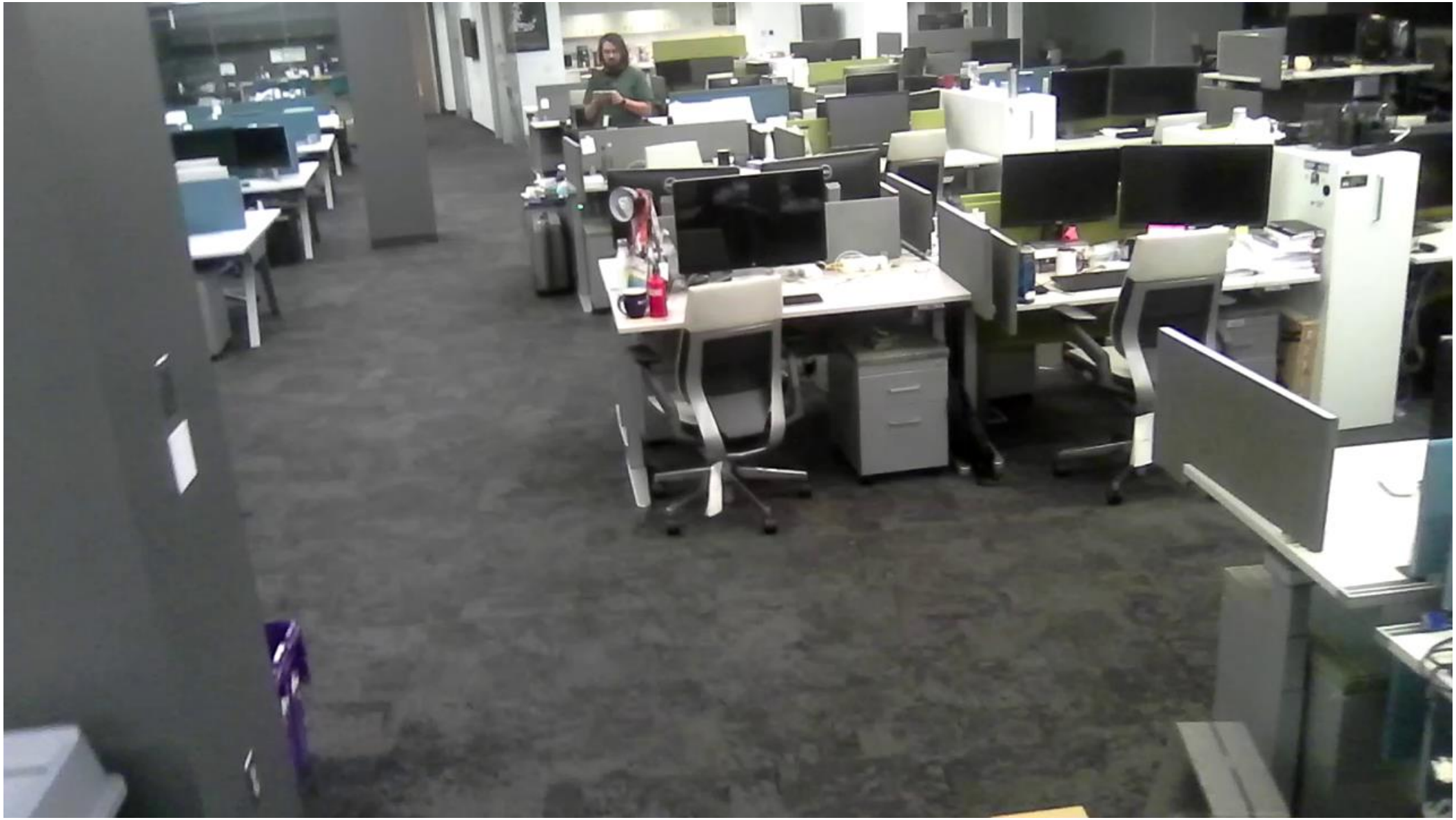
Mi Vacuum Robot: summary

- All data was still on the device
- Device was not wiped, instead only Wi-Fi reset
  - Reset button is misleading
  - Correct procedure is documented in the manual
- Previous owner could be tracked due to log files
  - Device creates very verbose log files and stores them locally

Other examples: MiTU Drone

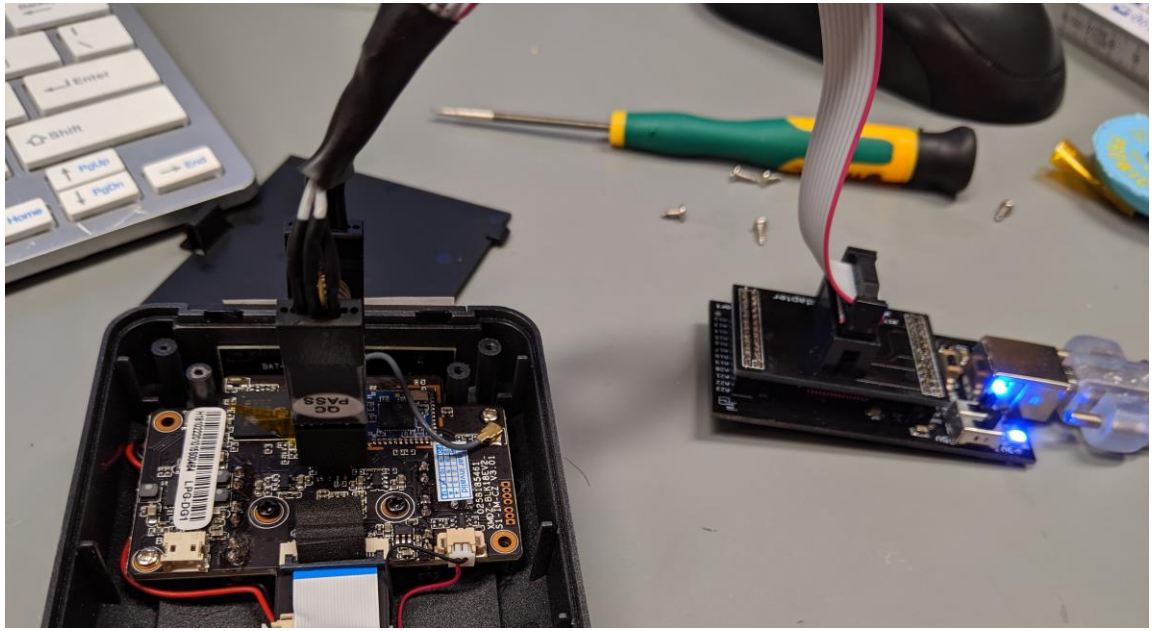
- Children toy, but powerful device
  - OS: Android
  - SOC: Leadcore LC1813
  - Flash: 4GByte eMMC
  - RAM: 512 Mbyte DDR2
  - 2 Cameras
- Access via: Serial, ADB (after root)
- Data: Recorded videos on internal memory
  - Cant be deleted if device is broken





Other examples: Door bells

- Many models same design
  - SOC: HI3518
  - Flash: 8MByte SPI NOR Flash
- All devices use JFFS2/UBIFS
- Wi-Fi credential recoverable
- No video due to SD card



Good factory reset implementations

- Vacuum cleaner robot with usage of Trustzone for key storage
  - User data partitions encrypted using LUKS
  - Key managed by TEE and are device specific
  - Unlocking of configuration and user data at boot
  - At factory reset: deletion of key and recreation of partition

Conclusion

- The device “remembers”
- Secure and correct factory reset difficult to implement
  - Use of raw NAND defeats full wipe
  - There is no way to ensure that a device have been wiped
- Many vendors do not erase all user generated data
  - Usage data remains, Logfiles are not erased
  - Wi-Fi configuration files were overwritten, but information remained in other places
- Also: Missing knowledge from the user

Recommendations

- Do not sell or throw away your device
  - If you expect that it may contain sensitive information
  - If you cannot verify a full wipe
- Physically destroy the flash memory
- Use the device to practice soldering ;)
- Change your Wi-Fi credentials or use a separate IoT Wi-Fi

If its broken: why not break it more? ;)

Hint: Does not prevent leakage, but limits attackers access on your network

Acknowledgements

- Prof. Guevara Noubir (Khoury, Northeastern University)
- Prof. John L. Manferdelli (Khoury, Northeastern University)

Northeastern University
Khoury College of
Computer Sciences

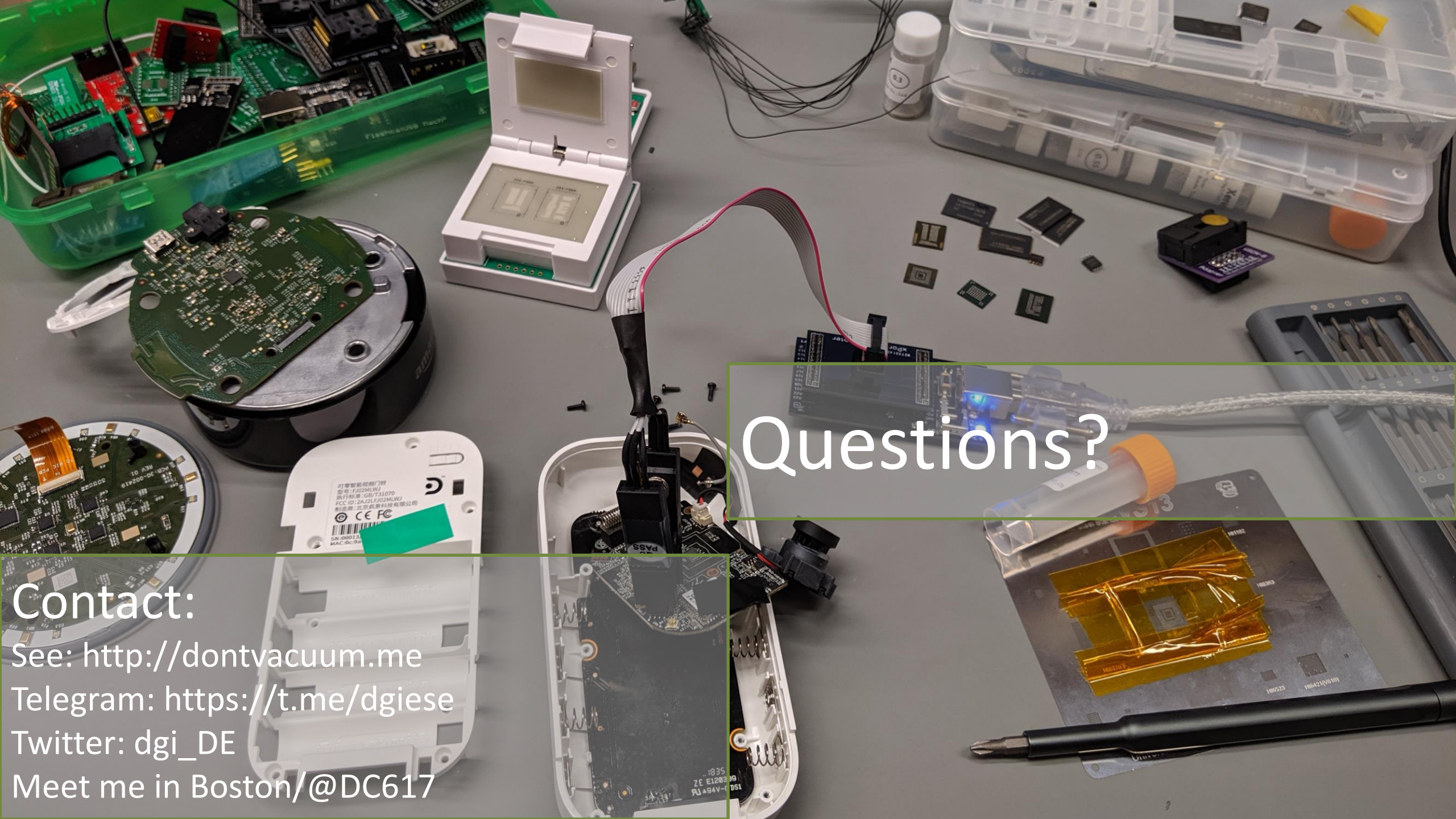
- Secure Mobile Networking (SEEMOO) Labs and CROSSING S1

SECURE MOBILE NETWORKING

CROSSING

TECHNISCHE
UNIVERSITÄT
DARMSTADT

http://www.ccs.neu.edu/home/noubir/
https://www.seemoo.informatik.tu-darmstadt.de/


Questions?
Contact:
See: http://dontvacuum.me
Telegram: https://t.me/dgiese
Twitter: dgi_DE
Meet me in Boston/@DC617


Northeastern University
College of Computer and Information Science