KASPER OSWALD
Ingenieure für innovative Sicherheitslösungen

RUB

Open, Sesame!

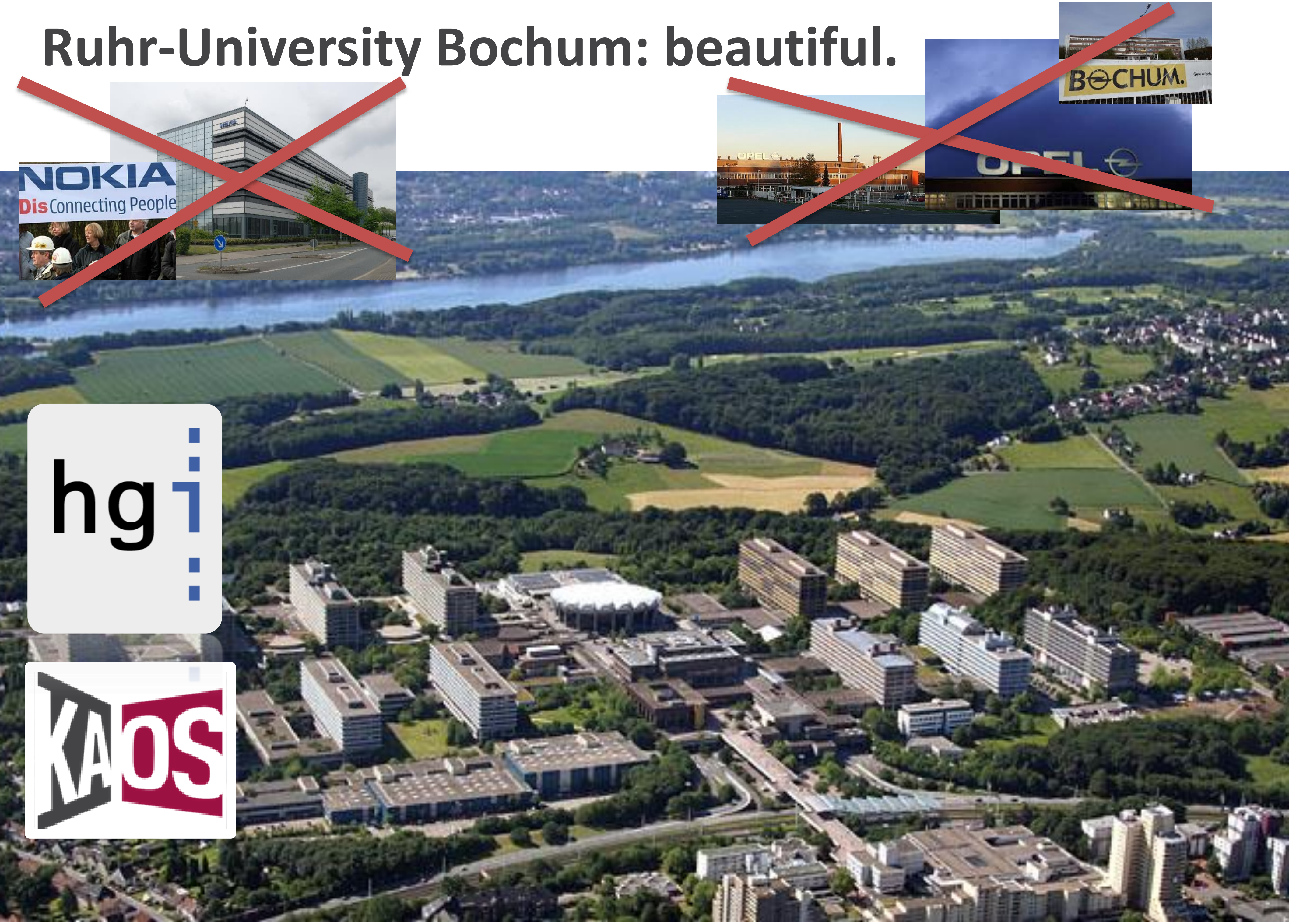
On the Security of Electronic Locks

David Oswald (david.oswald@rub.de)
Ruhr-Uni Bochum / Kasper & Oswald

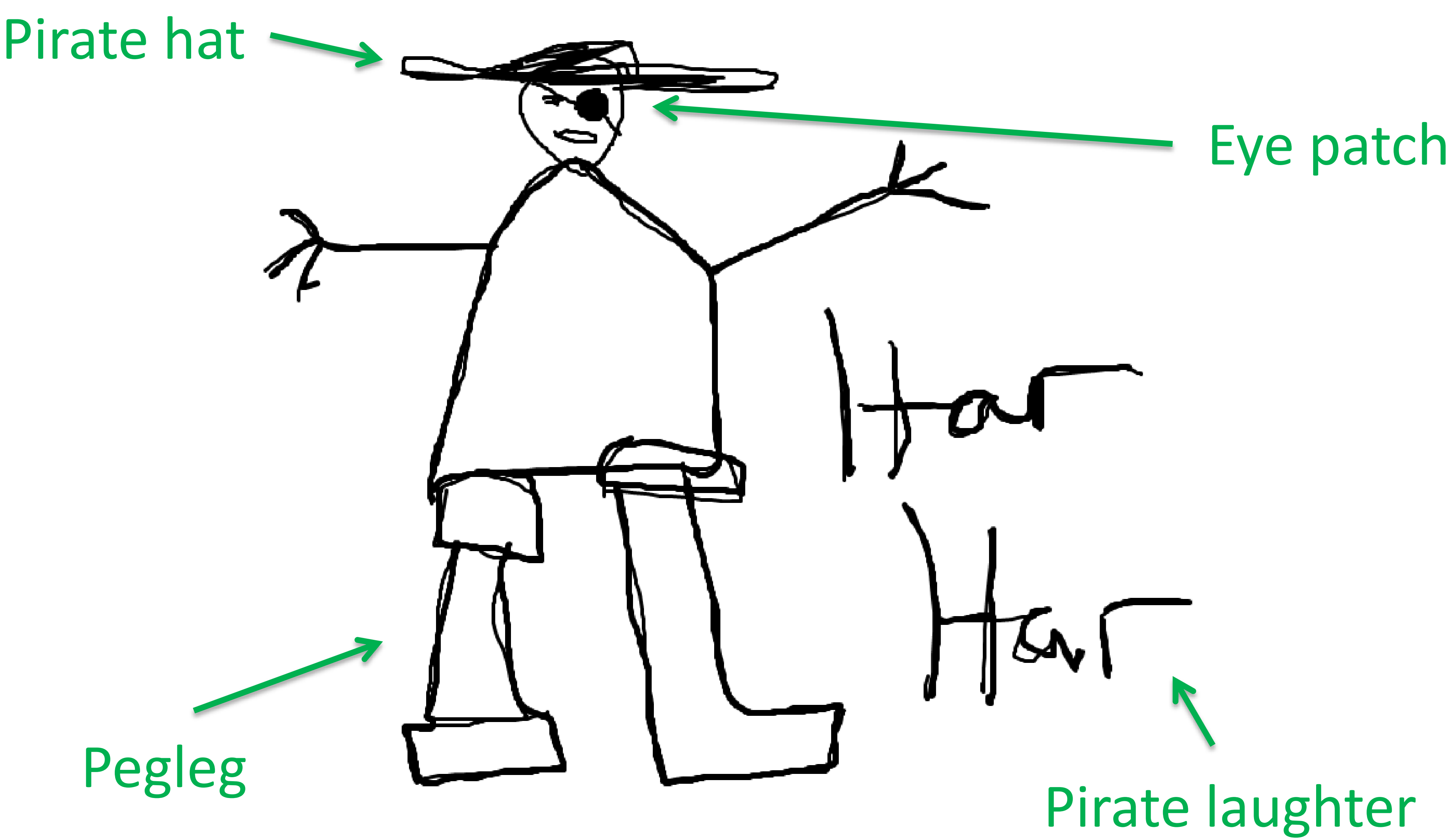
No, I did not do all this stuff alone

- Christof Paar
- Timo Kasper
- Benedikt Driessen
- Simon Küppers
- Gregor Leander
- Amir Moradi
- Ingo von Maurich
- Falk Schellenberg
- Daehyun Strobel

Ruhr-University Bochum: beautiful.



(The life of) a typical pirate



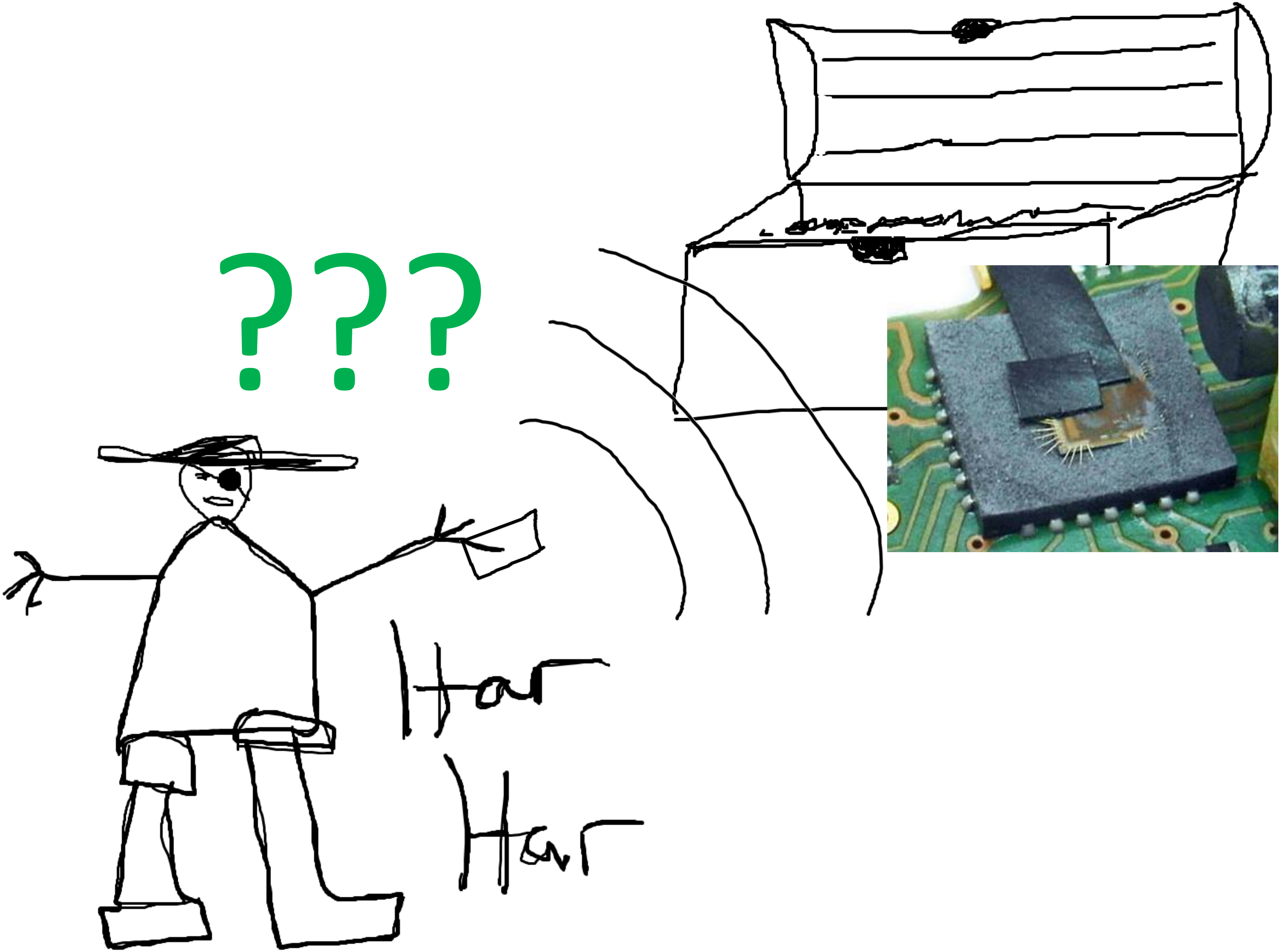
Pirate hat

Eye patch

Pegleg

Pirate laughter

Har
Har



???

Har
Har

„Opening“ doors – LEVEL 1

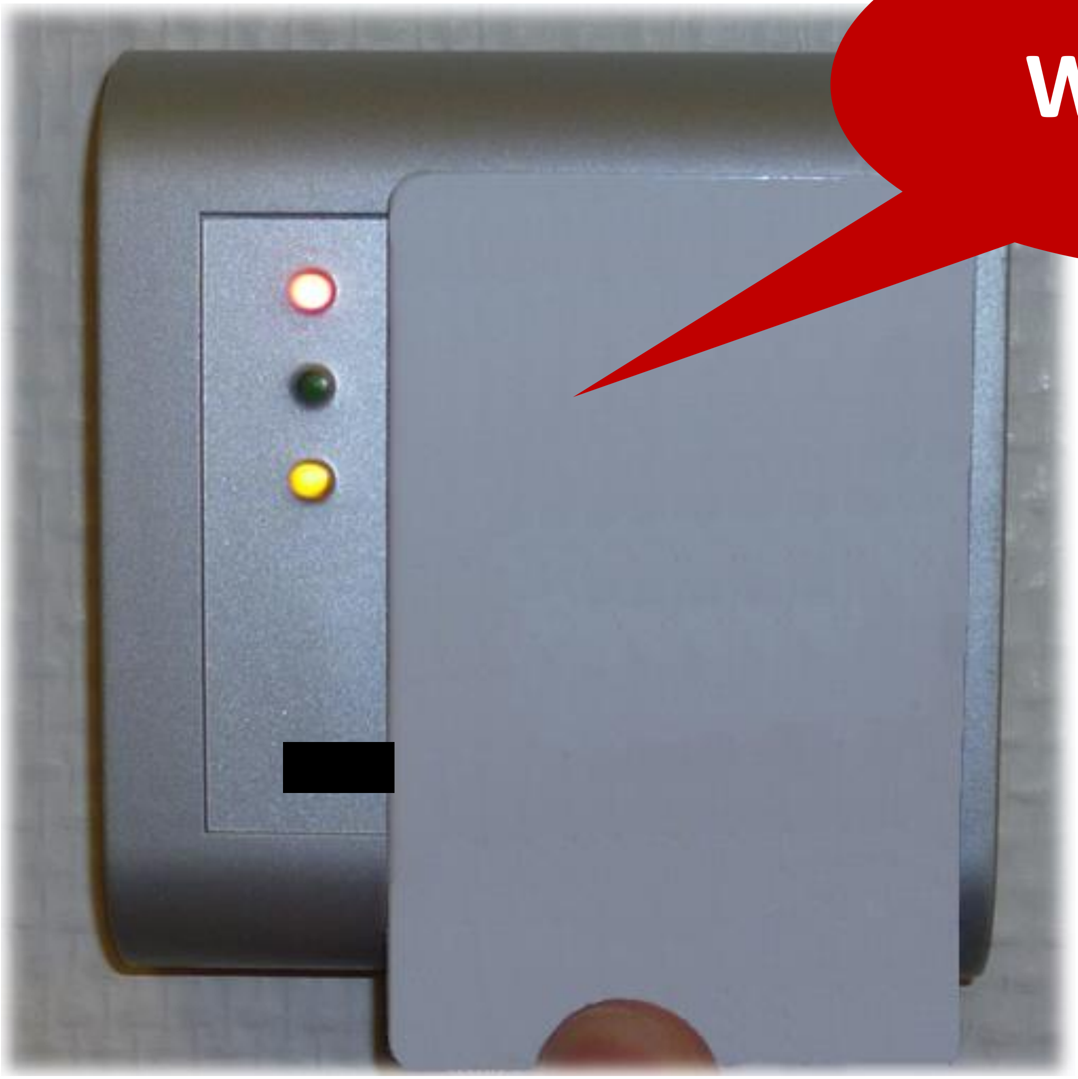
Opening doors – LEVEL 2

Access Control System

- Mifare Classic cards unlock doors and elevators
- Secret keys are default (0xA0A1A2A3A4A5)
- Identification by UID and 1st block of 1st sector
- **UID usually not changeable ...**

[redacted]

Clone on Blank Card Fails (wrong UID)



[redacted]

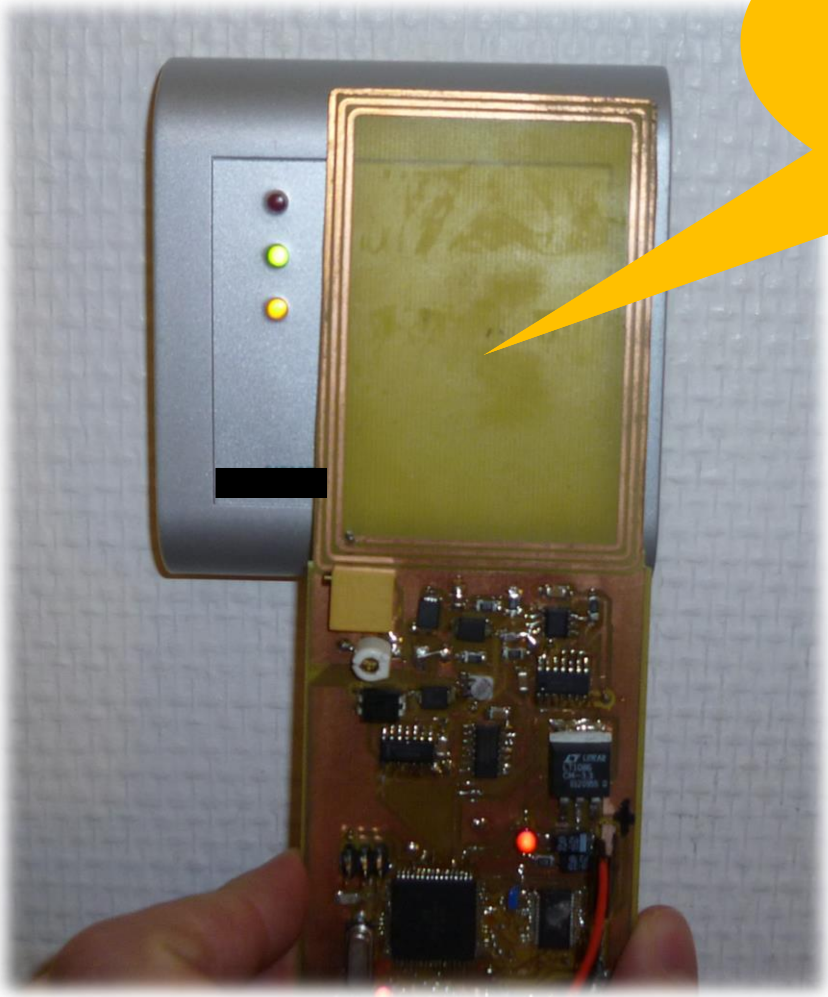
Wrong UID

ChameleonMini

- Chameleon emulates everything *including* UID
- **Open source project:** https://github.com/emsec/ChameleonMini
- **Buy / Kickstarter info:** http://kasper-oswald.de/gb/chameleonmini

Chameleon Succeeds

(emulates everything including UID)



[redacted]

Quite old prototype, was actually stolen …

Level 2: Summary

- Many locks still use UID only (from 125 kHz to DESFire EV1…)
- Mifare Ultralight (no crypto) e.g. used for hotel rooms
- Mifare Classic (broken in 2009) still wide-spread
- Backwards compatibility & mixed systems …

Opening doors – LEVEL 3

Electronic Locking System

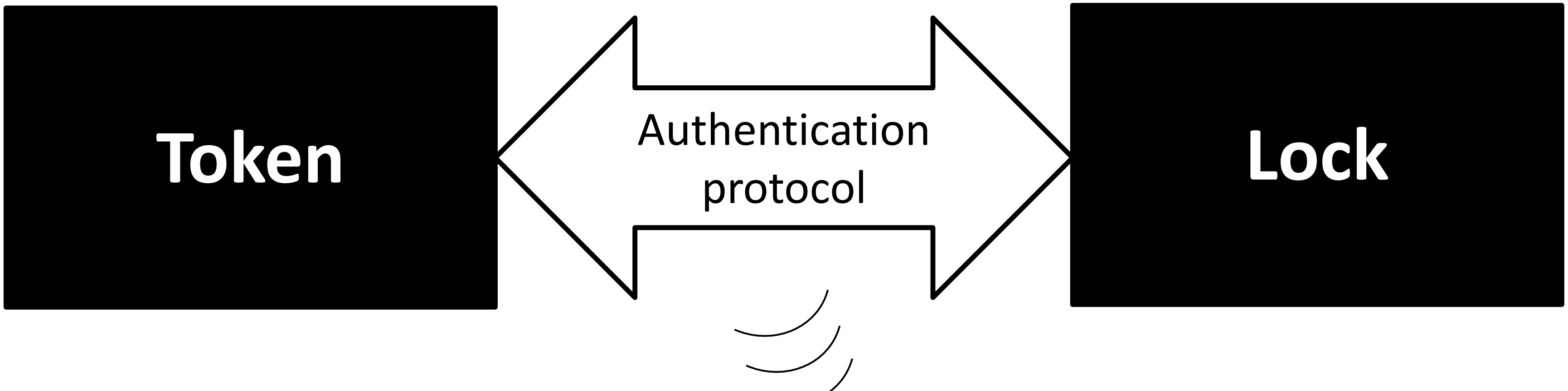
Token

Lock

Reverse-Engineering (1)

Black-box analysis:

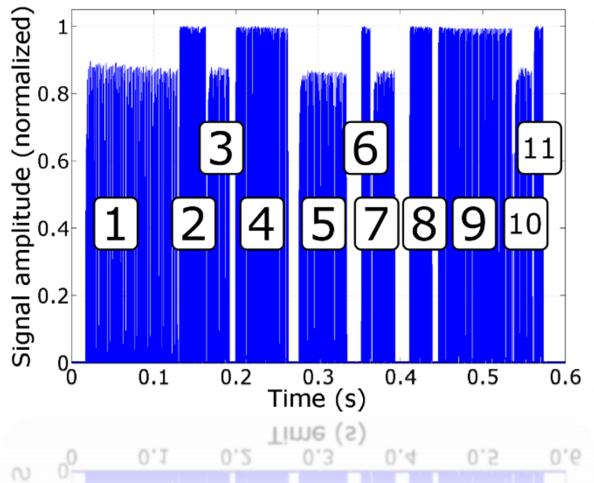
Token and lock perform authentication protocol



Token

Authentication protocol

Lock



???

Reverse-Engineering (2)

Token

Lock

Reverse-Engineering (3)

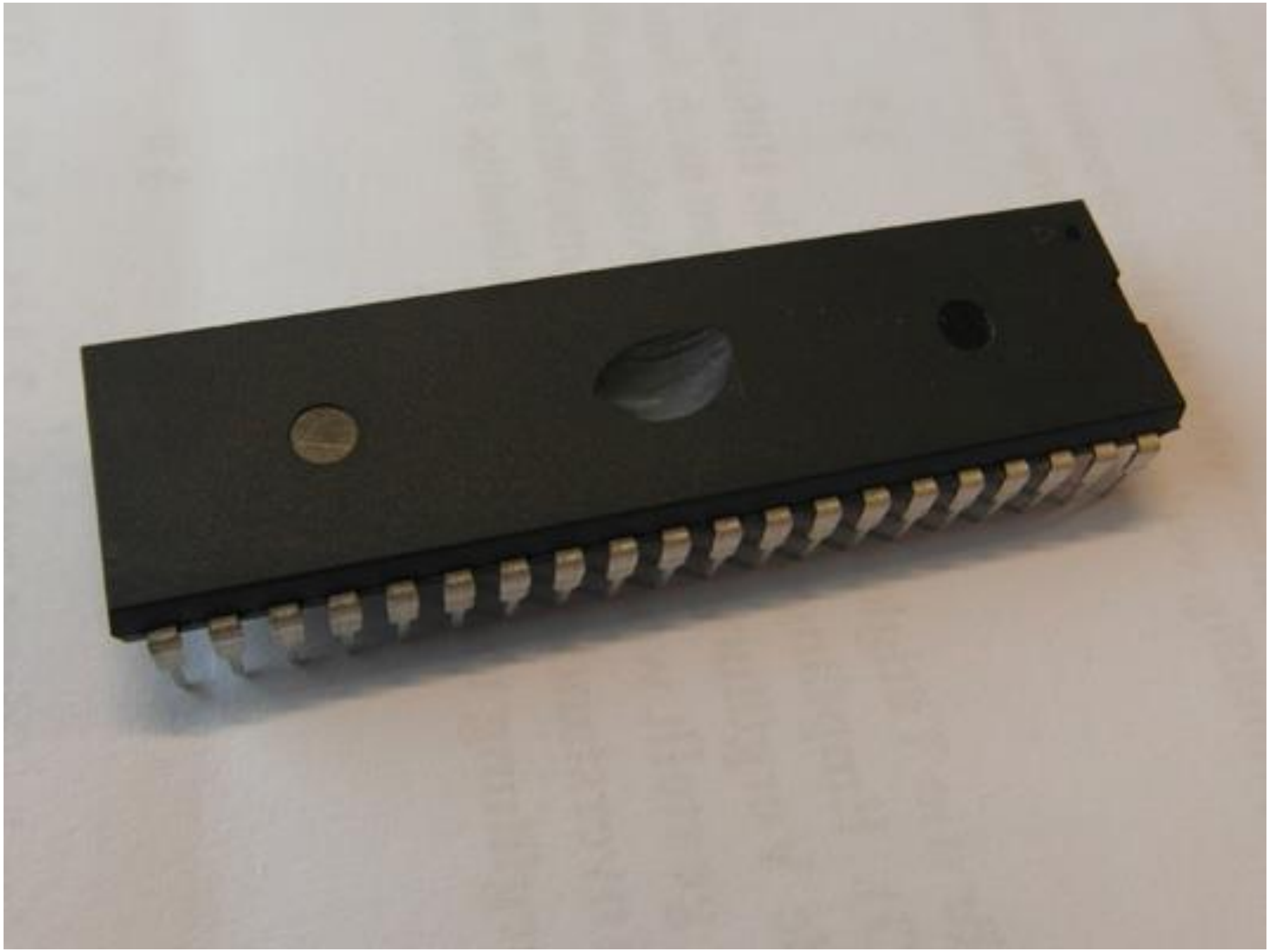
Embedded code?

Token

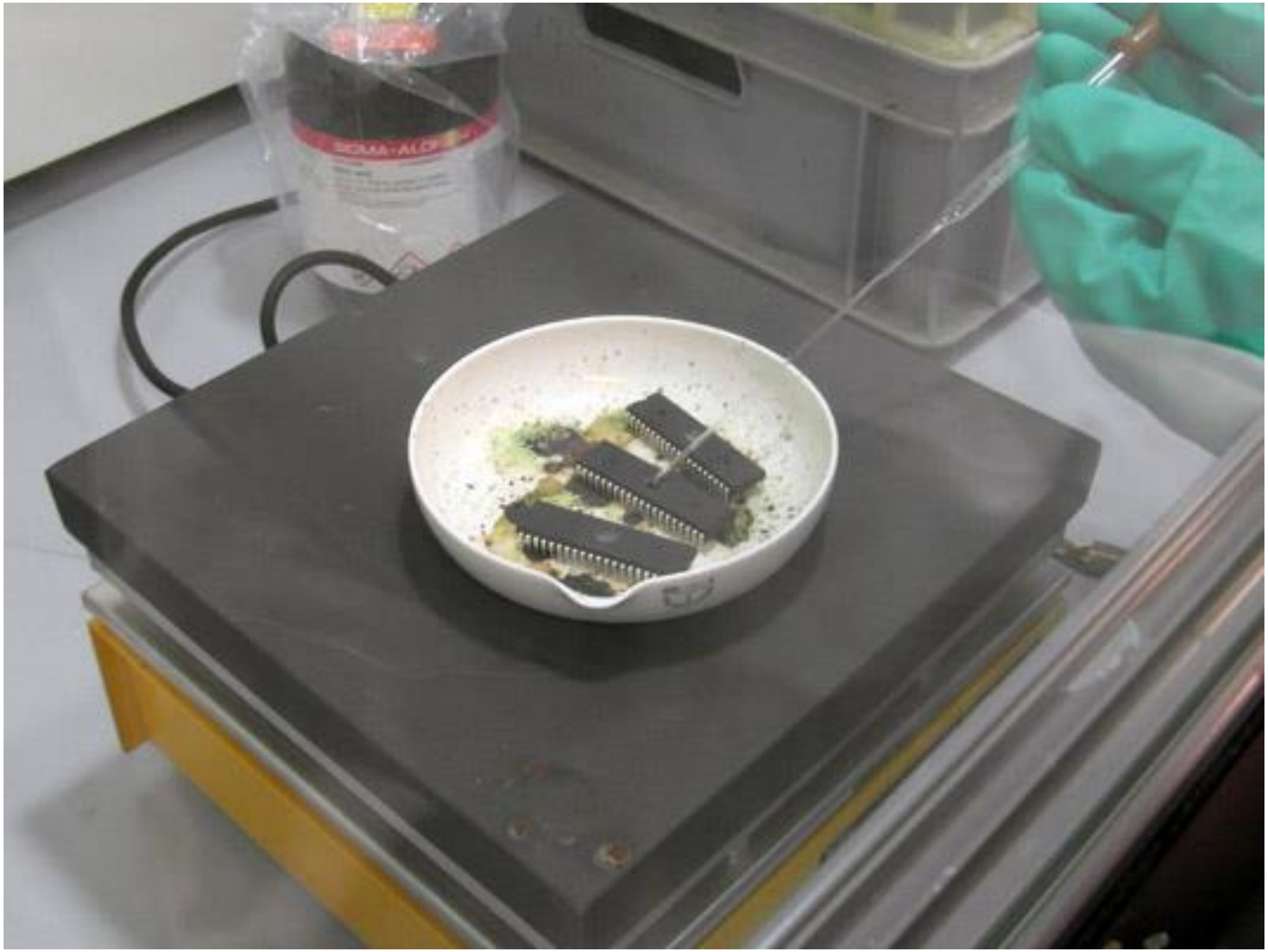
Lock

Read-out protection!

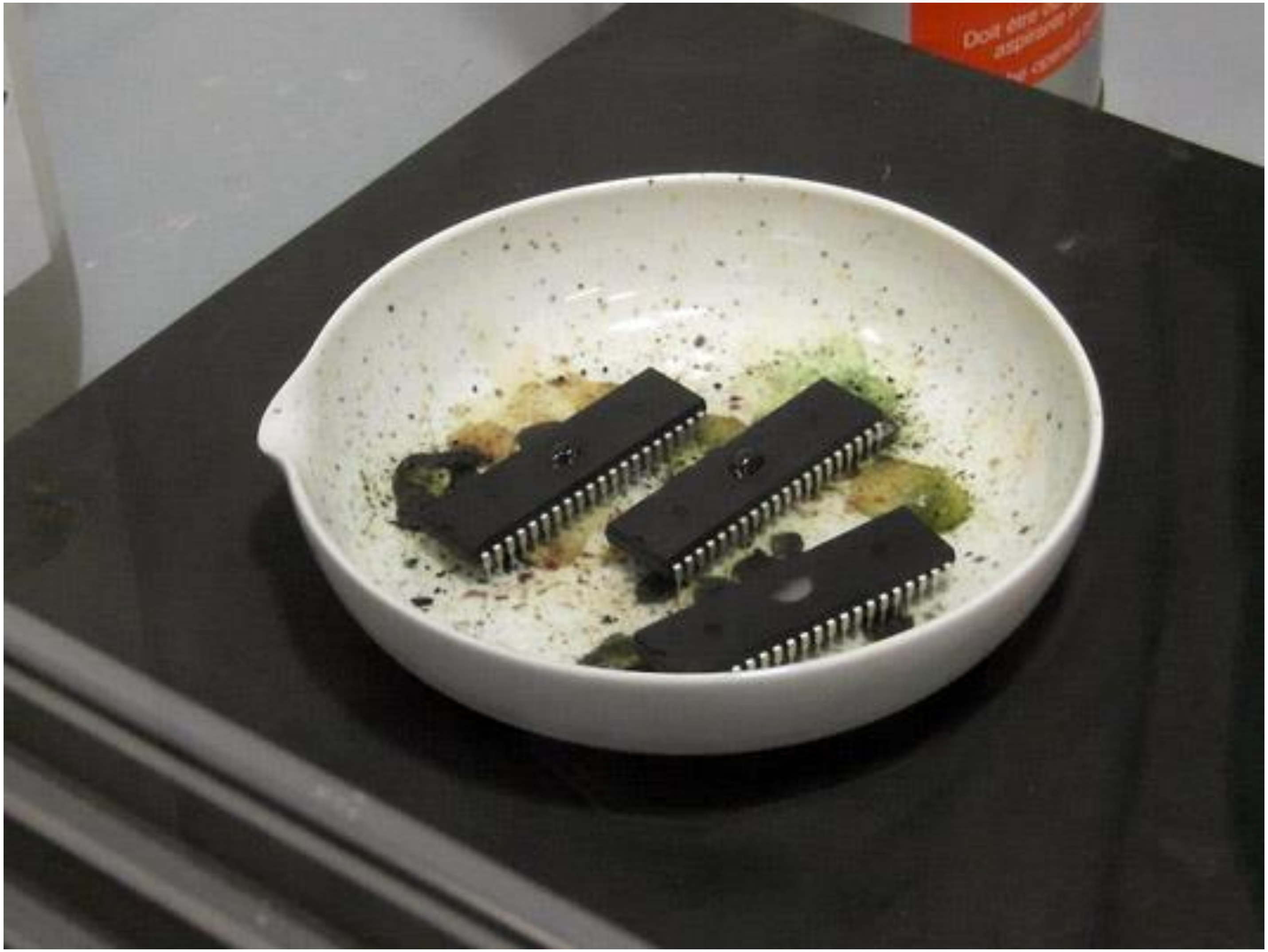
Decapping an IC (1)



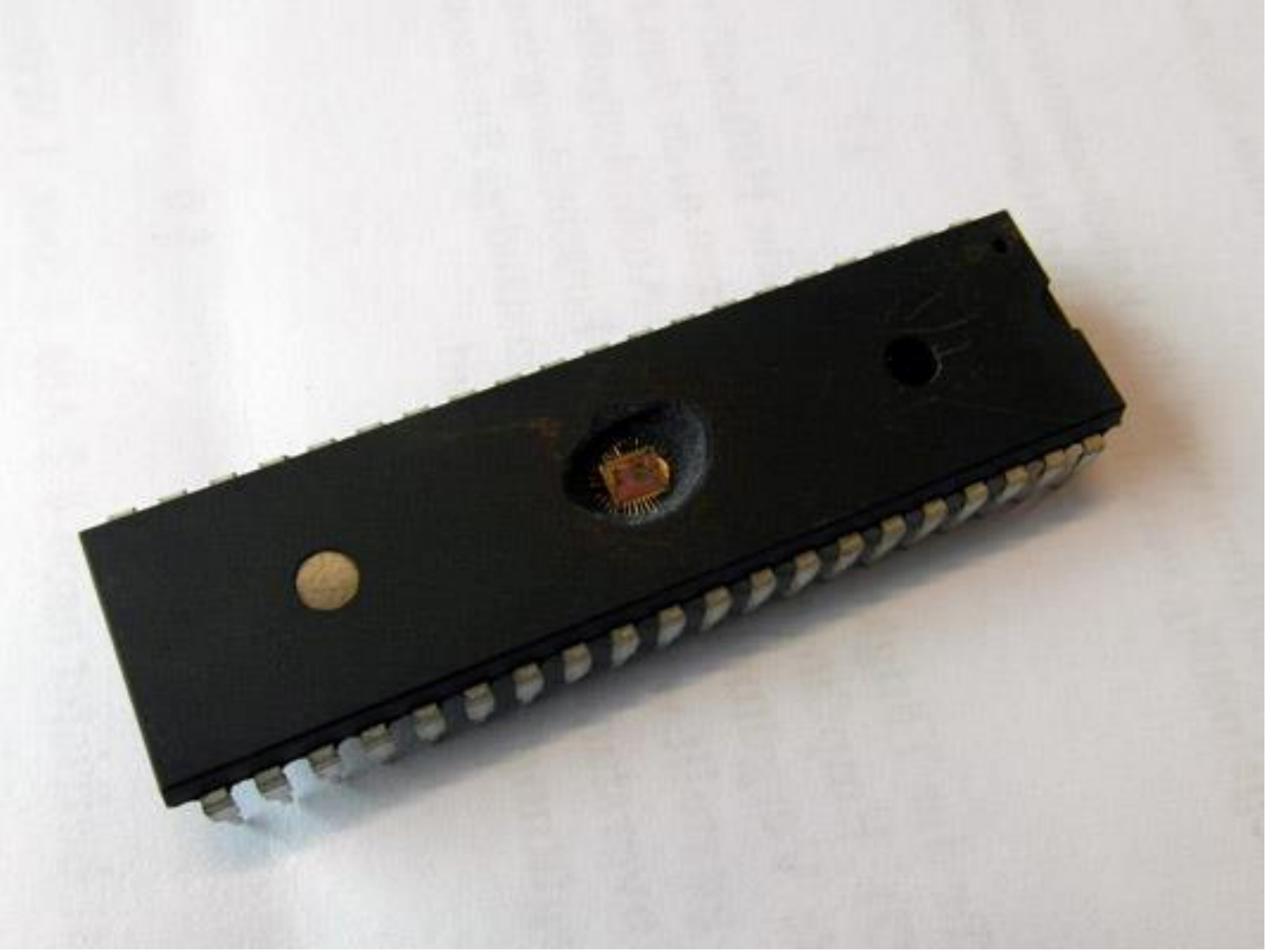
Decapping an IC (2)



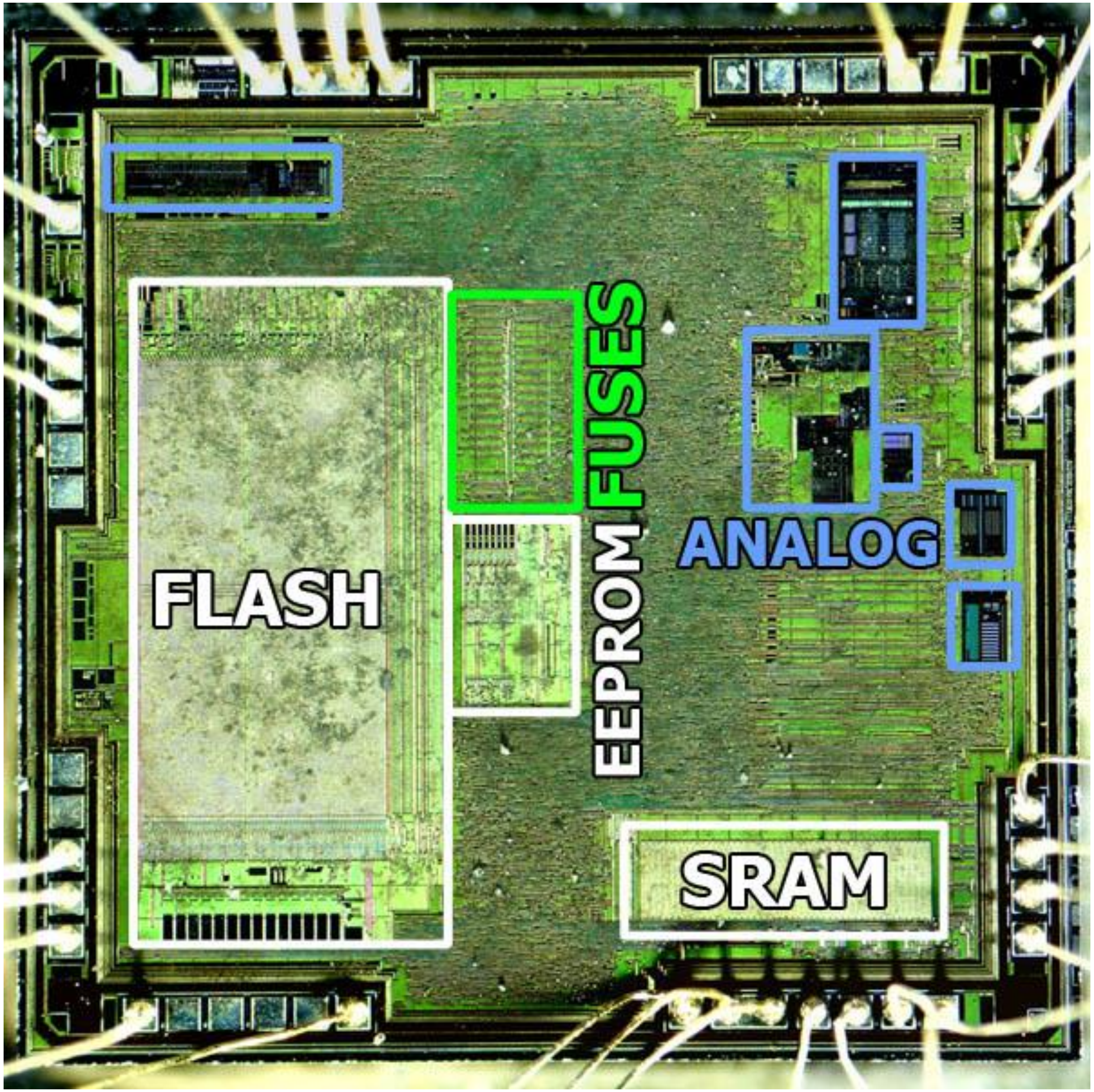
Decapping an IC (3)



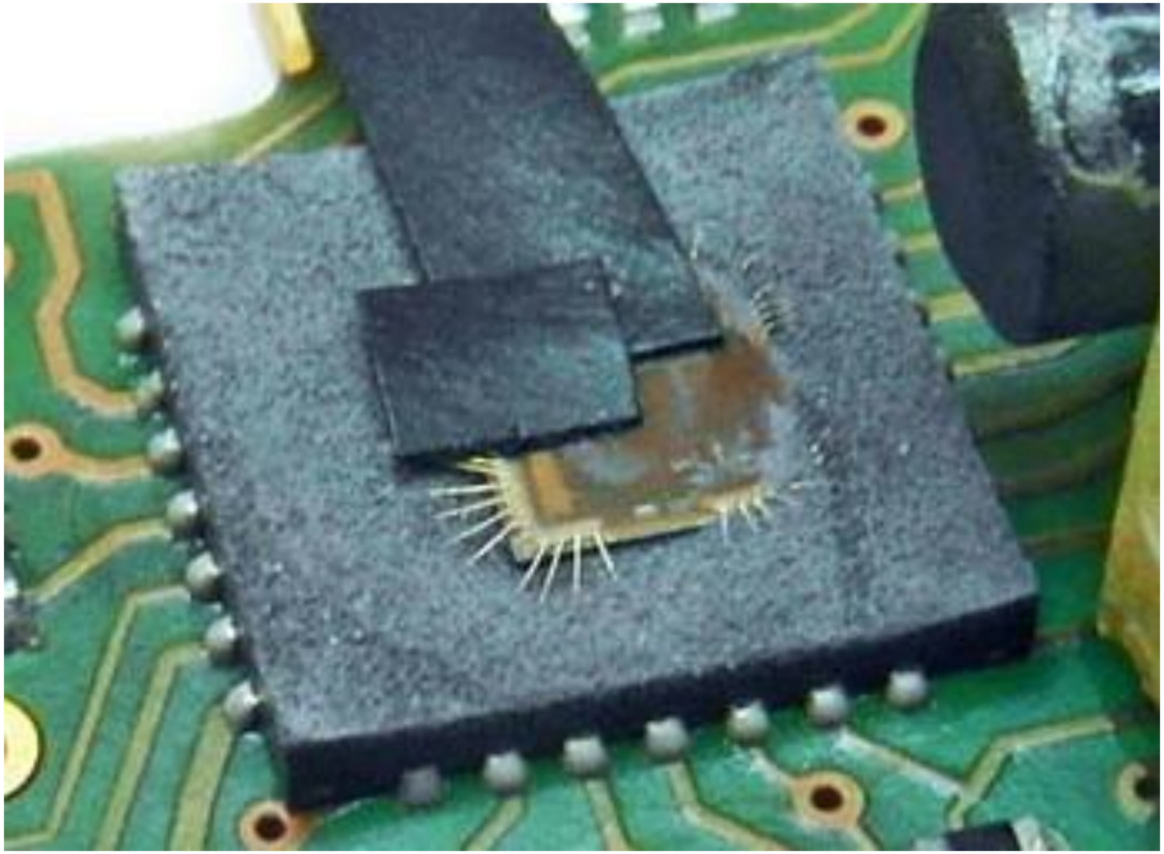
Decapping an IC (4)



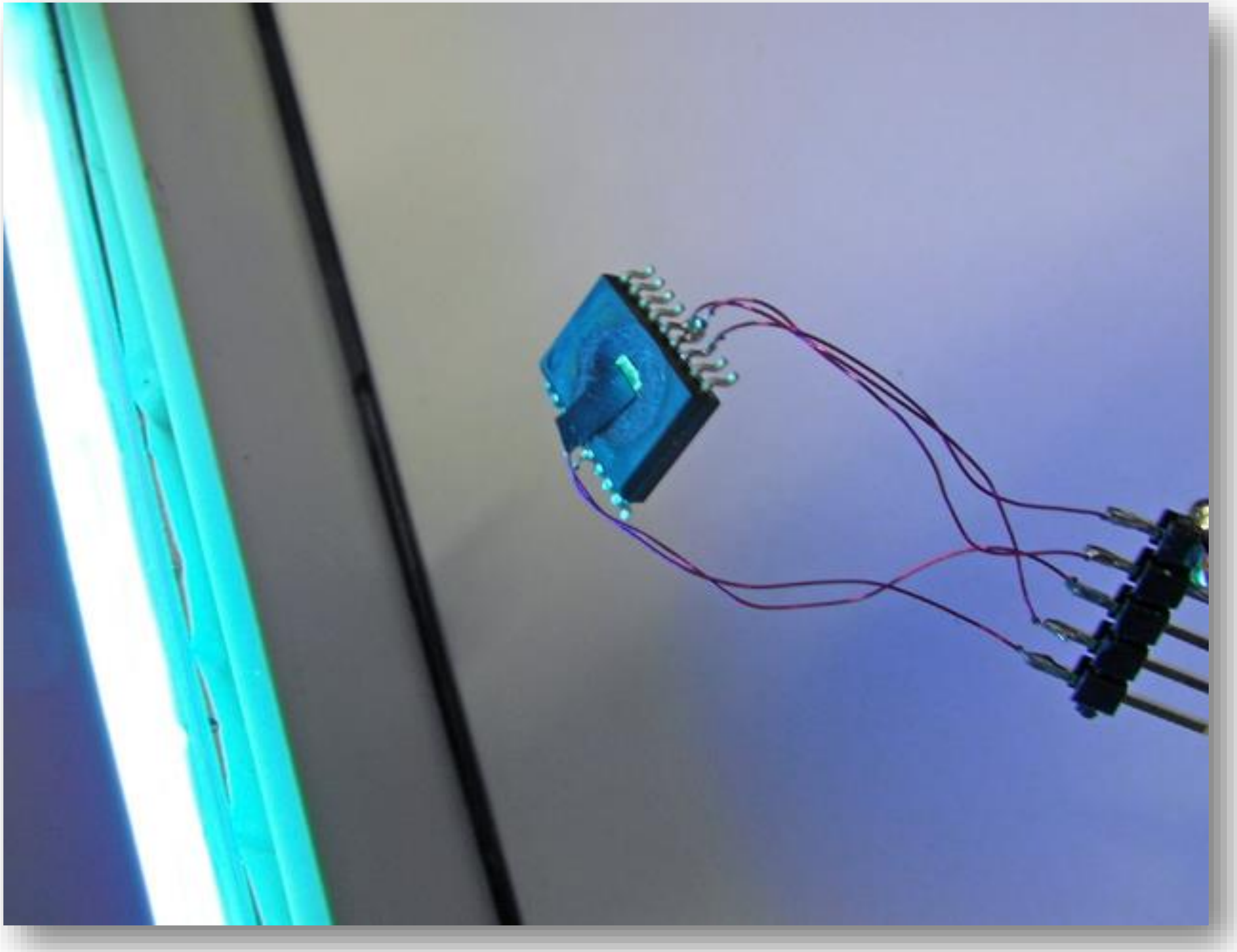
Microscopic View of the Silicon Die



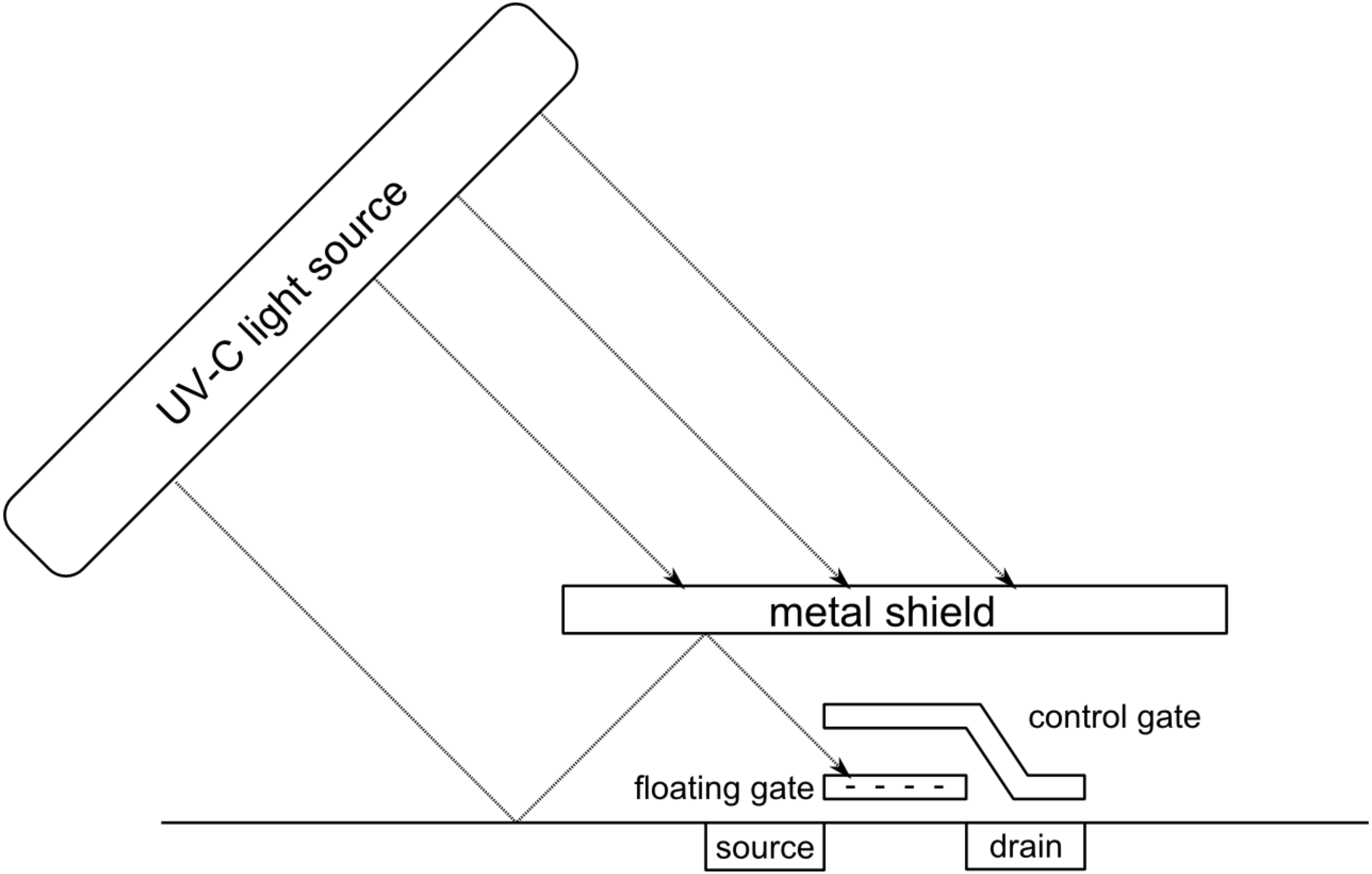
Exposure to UV-C: Disable Read-Out Protection (1)



Exposure to UV-C: Disable Read-Out Protection



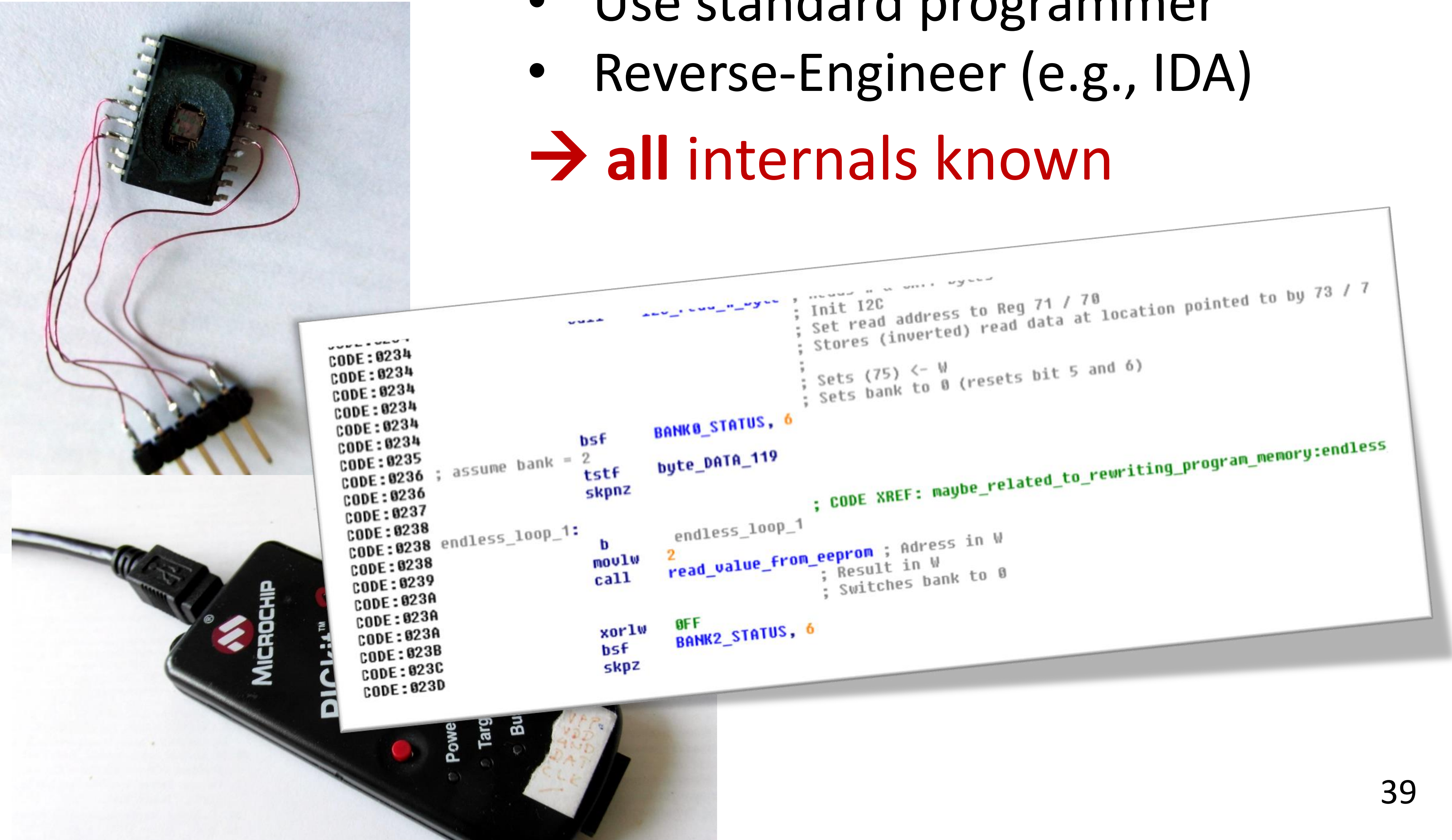
Exposure to UV-C: Why it works



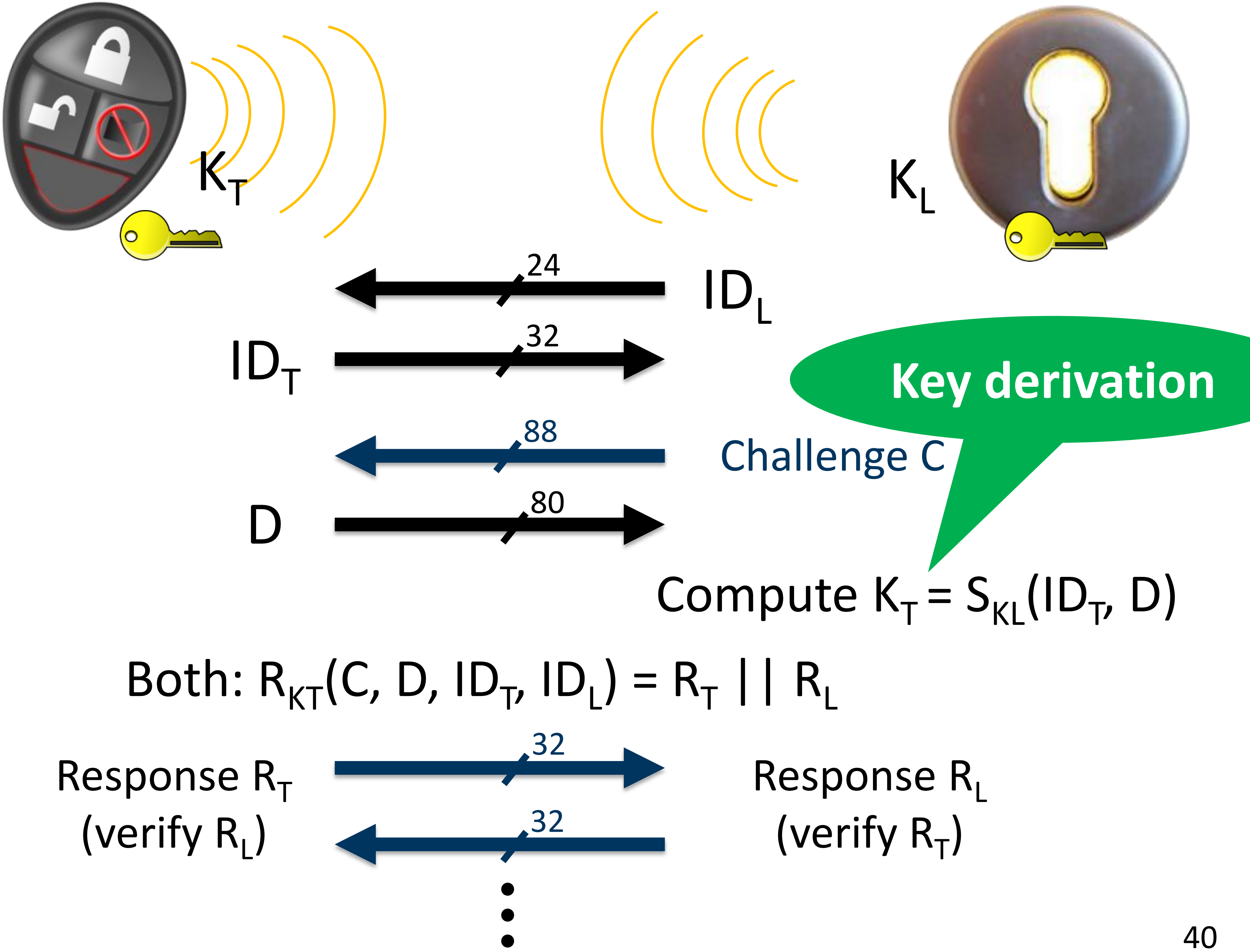
Reverse-Engineering continued

- Use standard programmer
- Reverse-Engineer (e.g., IDA)

→ **all** internals known



```
CODE:0234                                          ; Init I2C
CODE:0234                                          ; Set read address to Reg 71 / 70
CODE:0234                                          ; Stores (inverted) read data at location pointed to by 73 / 7
CODE:0234                                          ;
CODE:0234                                          ; Sets (75) <- W
CODE:0234                                          ; Sets bank to 0 (resets bit 5 and 6)
CODE:0235                 bsf     BANK0_STATUS, 6
CODE:0236 ; assume bank = 2
CODE:0236                 tstf    byte_DATA_119
CODE:0237                 skpnz
CODE:0238                                          ; CODE XREF: maybe_related_to_rewriting_program_memory:endless
CODE:0238 endless_loop_1:
CODE:0238                 b       endless_loop_1
CODE:0239                 movlw   2
CODE:023A                 call    read_value_from_eeprom ; Adress in W
CODE:023A                                          ; Result in W
CODE:023A                                          ; Switches bank to 0
CODE:023B                 xorlw   0FF
CODE:023C                 bsf     BANK2_STATUS, 6
CODE:023D                 skpz
```



Compute $K_T = S_{KL}(ID_T, D)$

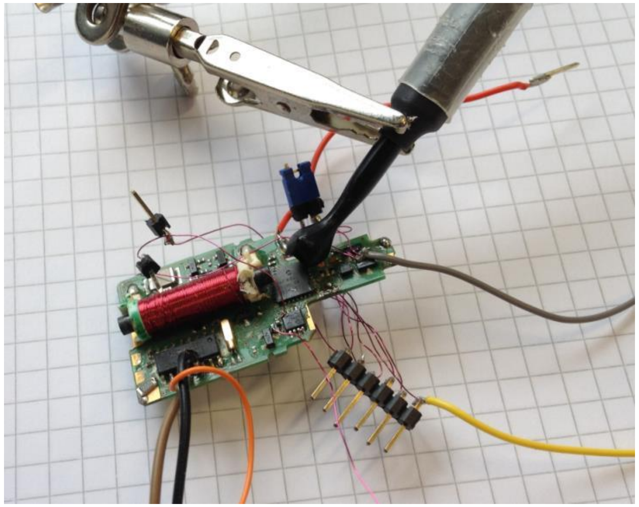
Both: $R_{KT}(C, D, ID_T, ID_L) = R_T \,||\, R_L$

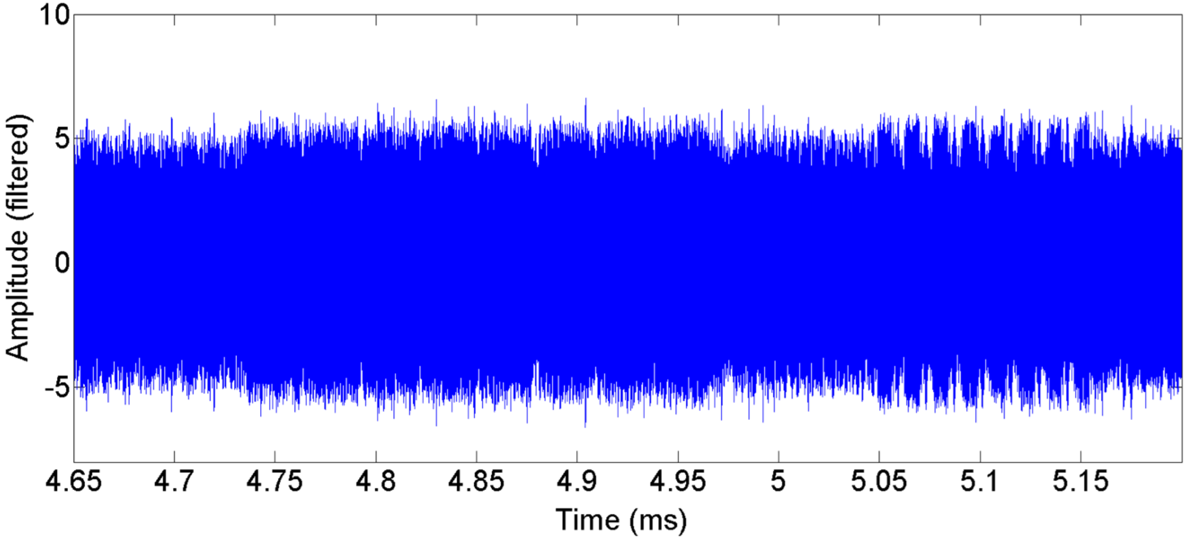
Response R_T
(verify R_L)

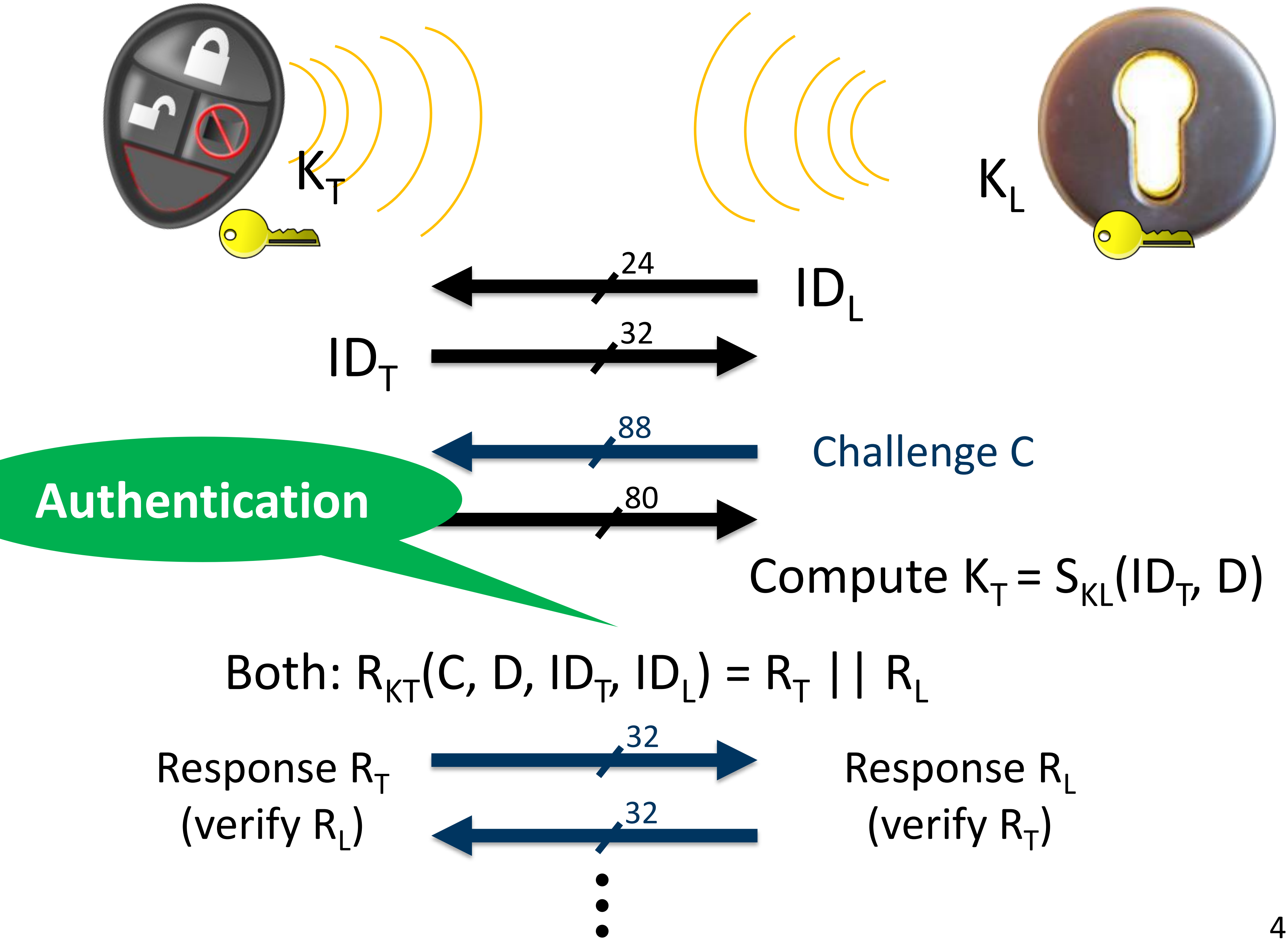
Response R_L
(verify R_T)

Weaknesses and Attacks (1)

- Each lock stores installation-wide cryptographic key
- UV-C attack in ~ 30 min (decap PIC)
- Side-channel attack in ~ 15 min (access to PIC)
- **Attacking one lock gives access to all doors**







ID_L

ID_T

Challenge C

Authentication

Compute $K_T = S_{KL}(ID_T, D)$

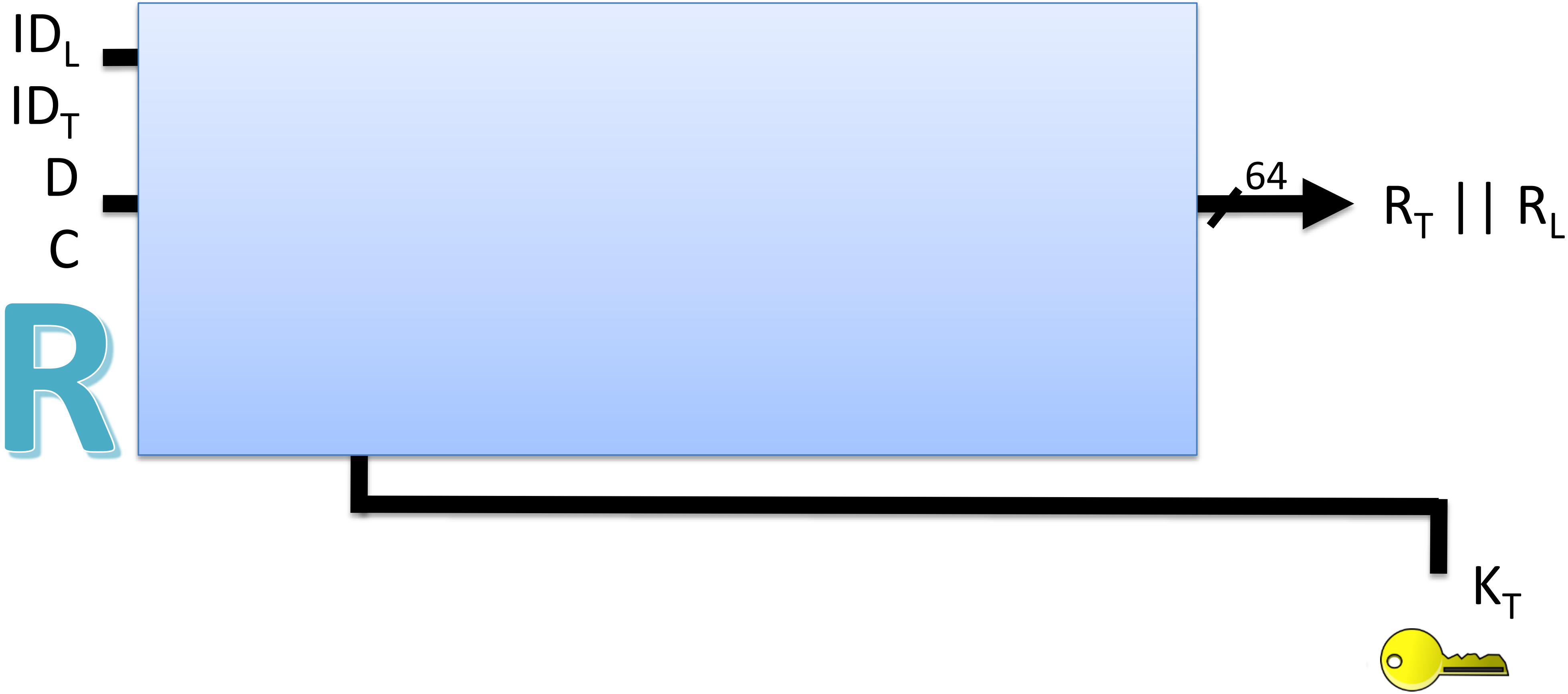
Both: $R_{KT}(C, D, ID_T, ID_L) = R_T \,||\, R_L$

Response R_T
(verify R_L)

Response R_L
(verify R_T)

⋮

Cryptographic Functions **R** and **S**



ID_L

ID_T

D

C

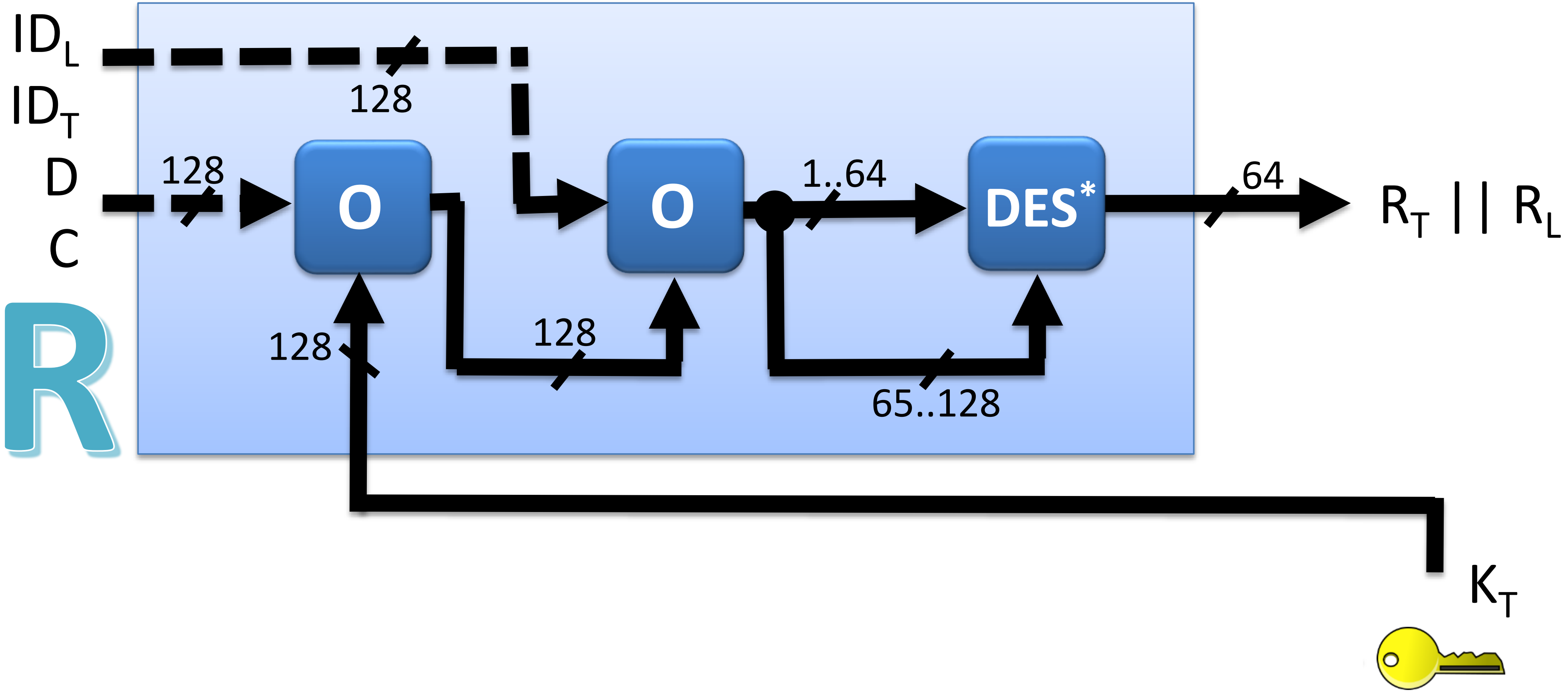
R

64

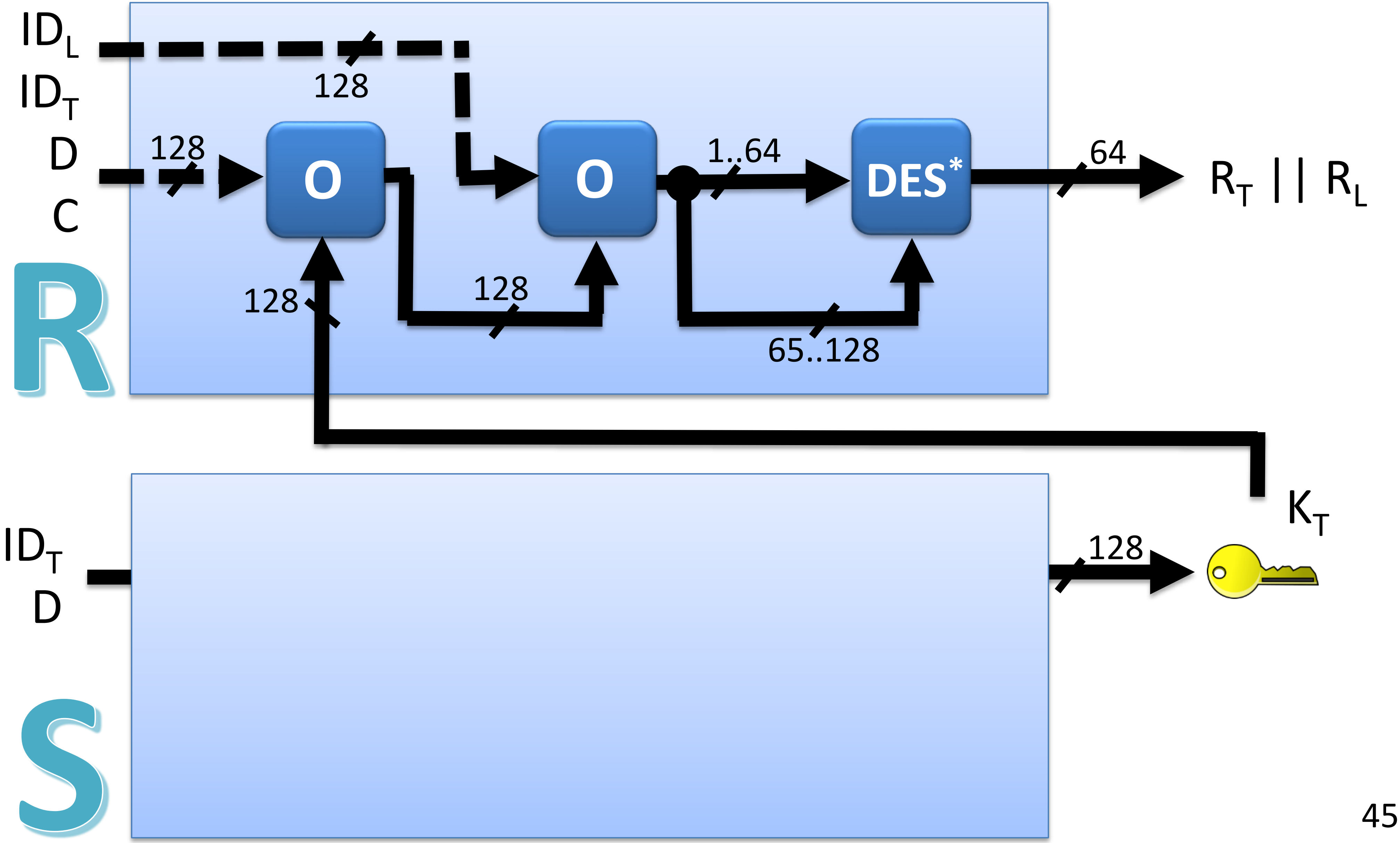
$R_T \,||\, R_L$

K_T

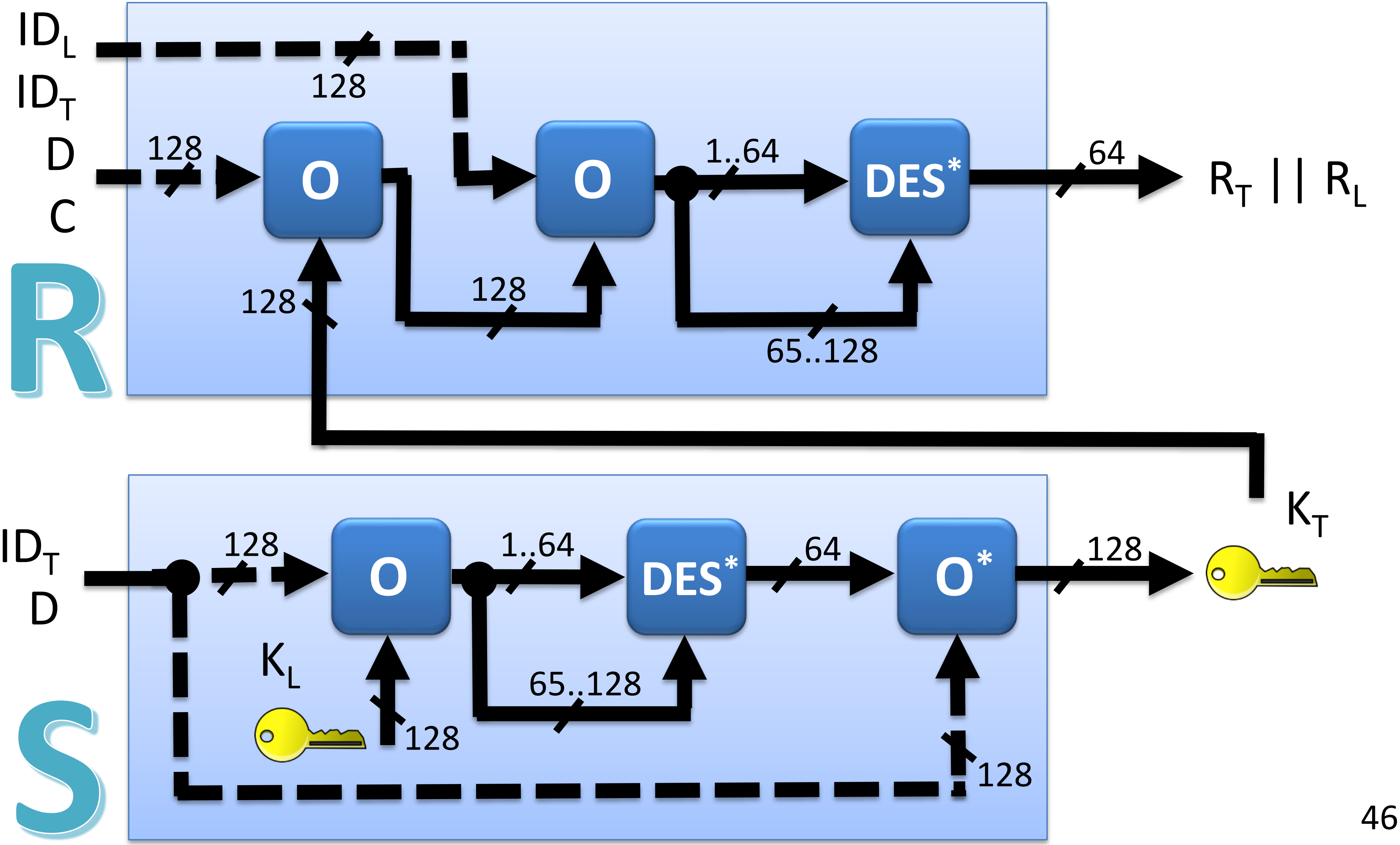
Cryptographic Functions **R** and **S**



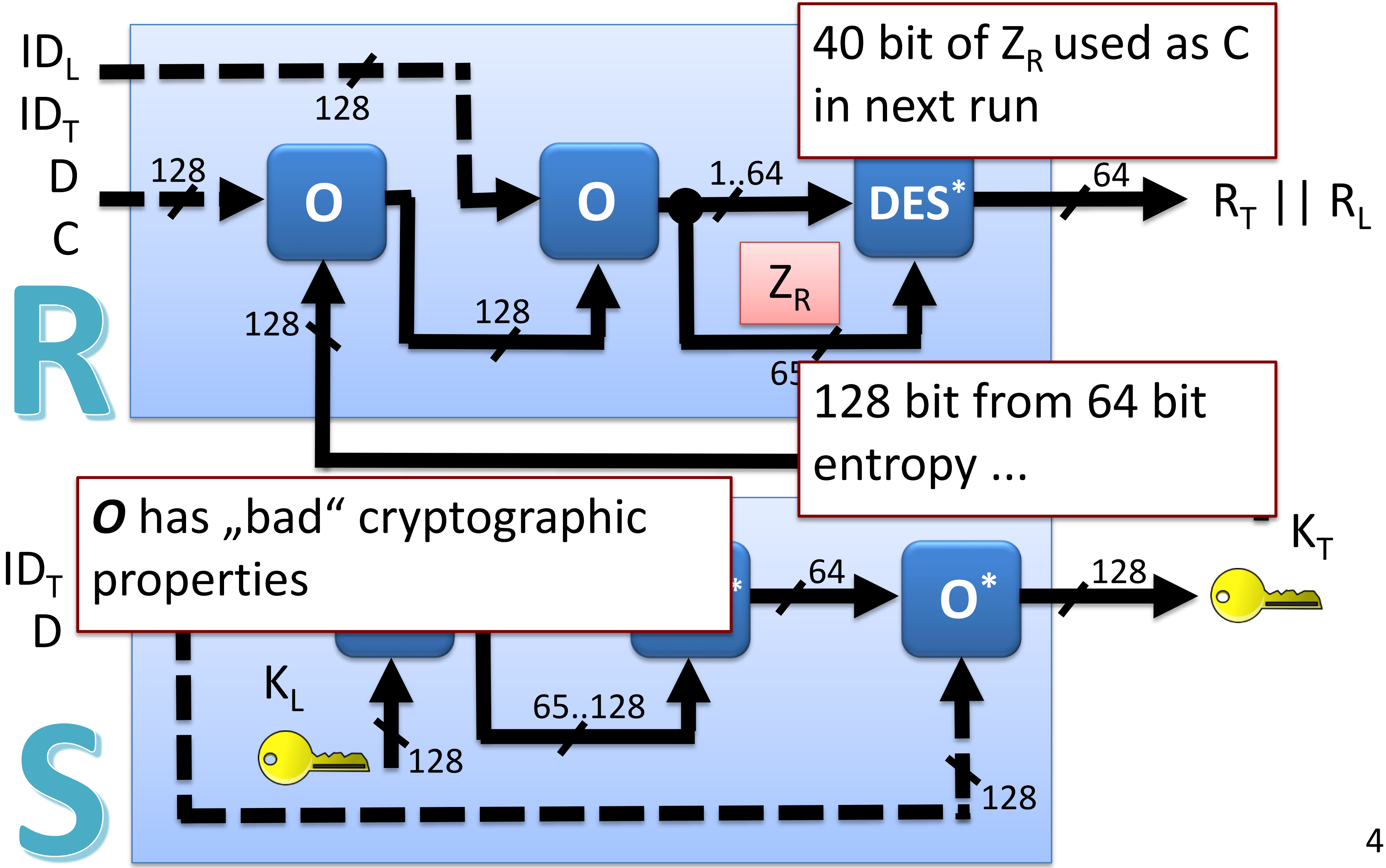
Cryptographic Functions **R** and **S**



Cryptographic Functions R and S



Cryptographic Functions **R** and **S**: Security Vulnerabilities



40 bit of Z_R used as C in next run

128 bit from 64 bit entropy ...

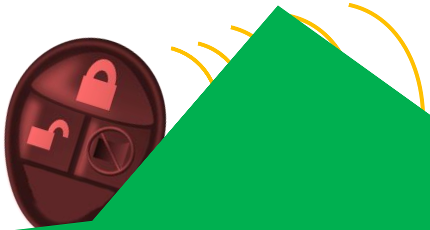
O has „bad“ cryptographic properties

Consequence: Wireless **Lock-only Attack**

- Initate some, not successful protocol runs
- Compute K_T (for known ID_T)

Protocol Runs	Run-Time	Key Candidates
3	3,36 min	21,34
4	11,5 s	1
5	1,2 s	1
6	650 ms	1

Consequence: Wireless **Lock-only Attack**

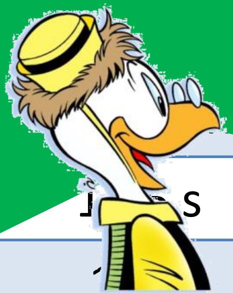


Firmware upgrade
of manufacturer:
Generate **random C**

→ Fix for this attack

- Initat ns
- Com

Protocol Ru		s
3		21,34
4	s	
5		1
6	650 ms	1



Report flaws

Improve

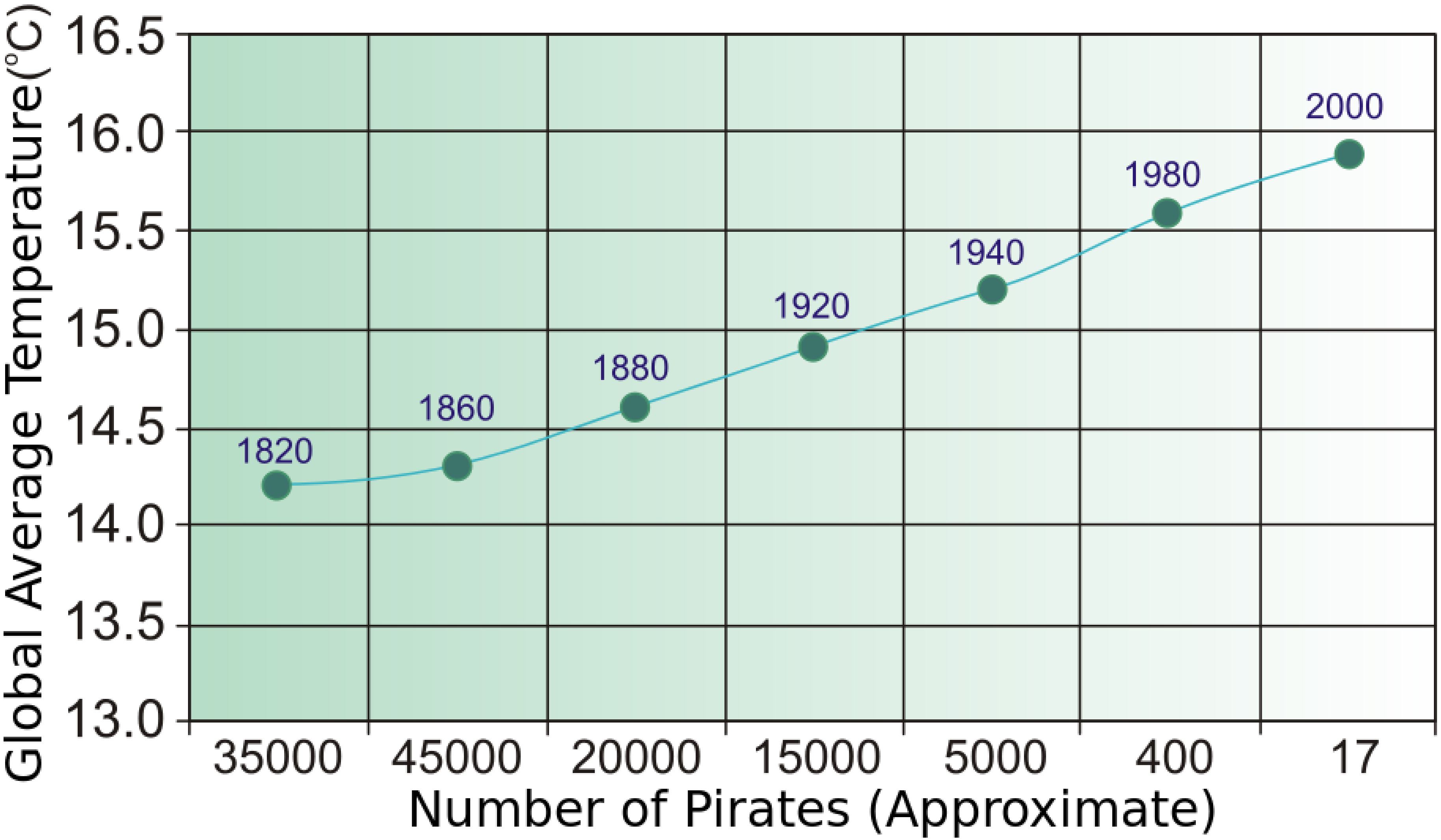
Level 3: Management Summary

- **Attacker can gain full access to any door**
- Reasons for security flaws
  - Insecure hardware
  - Proprietary cryptography
  - „Bad“ system design
- Can the system be „saved“?
  - **Cryptanalytical attacks:** Firmware update (cheap)
  - **HW attacks:** Require replacing all devices (expensive)

Responsible Disclosure

When pirates do good ...

Global Average Temperature vs. Number of Pirates



By RedAndr, Wikimedia Commons

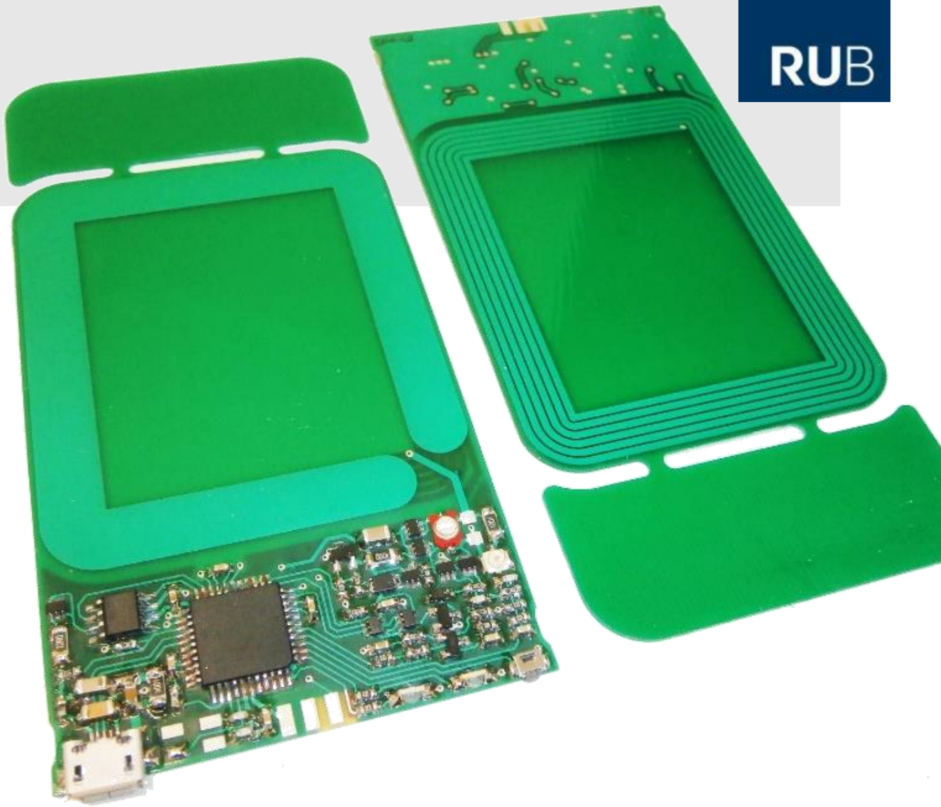
Responsible Disclosure

- **Locking system:**
  - Vendor informed ~ 1 year before
  - Discussion of found flaws
  - Deployed patch to fix mathematical attacks
- **Other examples:**
  - **Altera FPGAs:** Informed ~ 6 months before
  - **Yubikey**: Informed ~ 9 months before

Countermeasures

Countermeasures

- Implementation attacks: Practical threat, but:
- **First line of defense:** Classical countermeasures
  - Secure hardware (certified devices)
  - Algorithmic level
- **Second line of defense:** System level
  - **Detect**: Shadow accounts, logging
  - **Minimize impact** (where possible): Key diversification



Live Demo

„Everything that can go wrong, will go wrong“

Expect the unexpected.

Thanks!

Questions now?

or later:

david.oswald@rub.de
@sublevado