Université Grenoble Alpes

Modern System On Chips Security Against Physical Attacks

Phd. defense

Slides available at `https://thomas.trouchkine.com/assets/pdf/defense.pdf`

Defended by Thomas Trouchkine on March 24, 2021

Introduction - Handling sensitive operations

Sensitive operations

Introduction - Handling sensitive operations

Sensitive operations

Payment

Healthcare

Identification

Introduction - Handling sensitive operations

Sensitive operations

Payment

Healthcare

Identification

Historically

- handled by smartcards
- security designed devices
- high level security evaluation

Introduction - Handling sensitive operations

Sensitive operations


Payment


Healthcare


Identification

Historically

- handled by smartcards
- security designed devices
- high level security evaluation

Nowadays

- handled by smartphones or laptops
- performance designed devices
- security added recently
- no security evaluation

Introduction - Handling sensitive operations

Sensitive operations

Payment Healthcare Identification

Historically

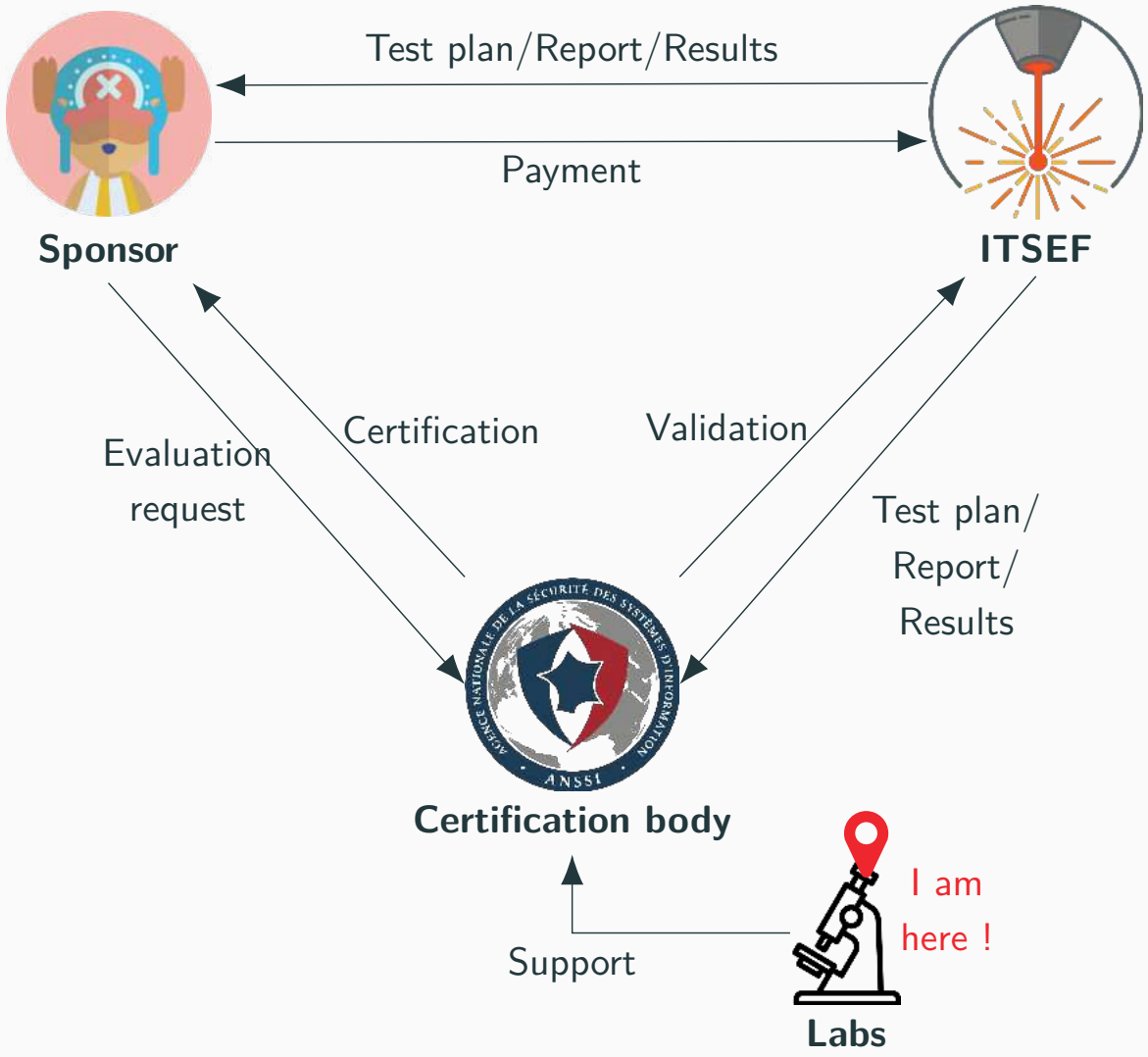
- handled by smartcards
- security designed devices
- high level security evaluation

Nowadays

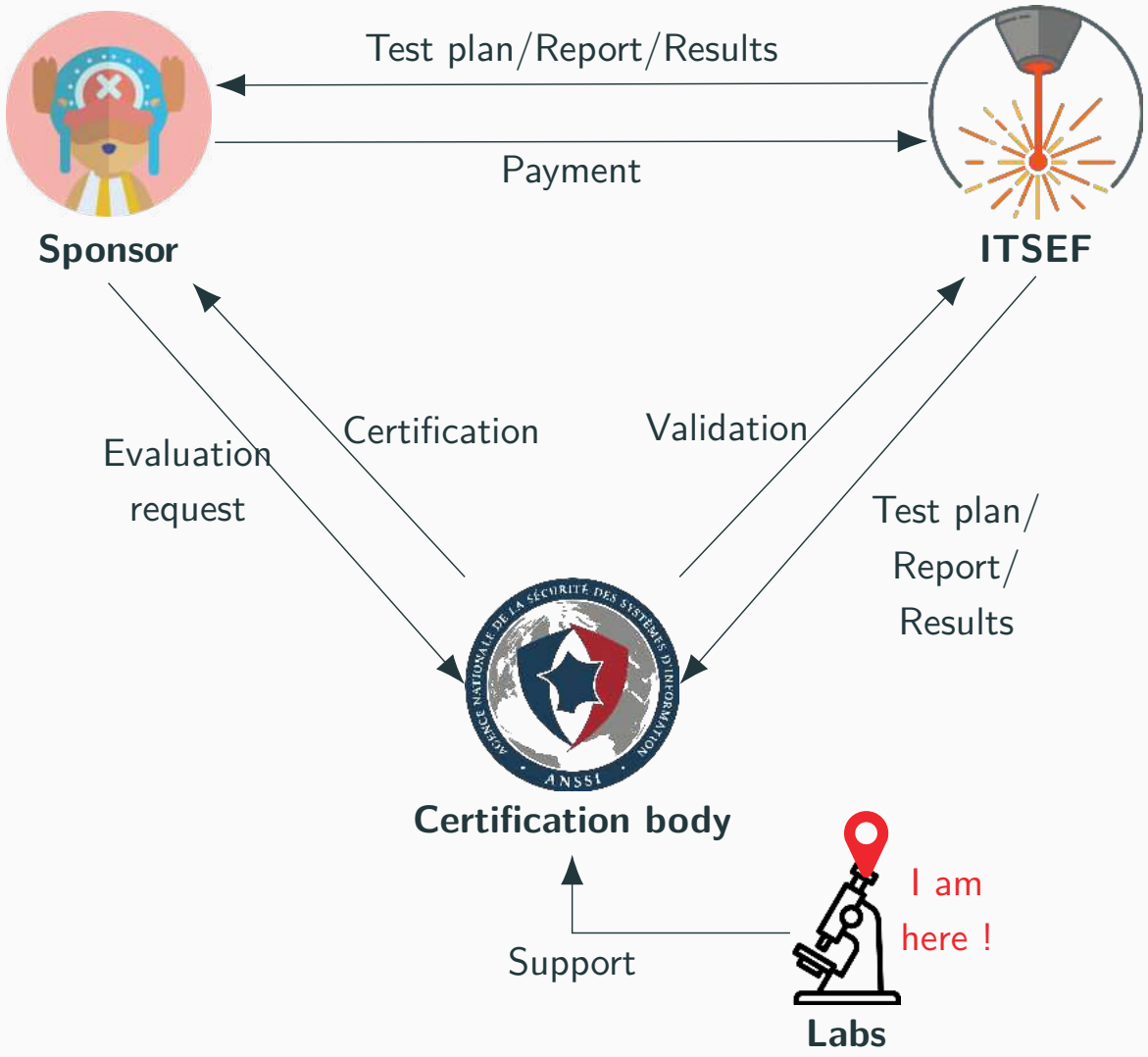
- handled by smartphones or laptops
- performance designed devices
- security added recently
- no security evaluation

We want to evaluate smartphones' security !

Introduction - Security Evaluation



Introduction - Security Evaluation



Hardware evaluations mainly focus on **smartcards**

Introduction - SEs vs SoCs

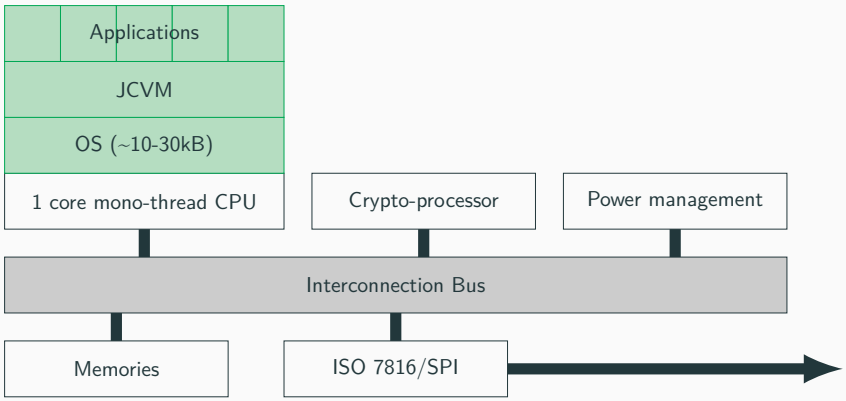
Smartcards

Smartphones

Introduction - SEs vs SoCs

Smartcards

- secure elements (SEs)

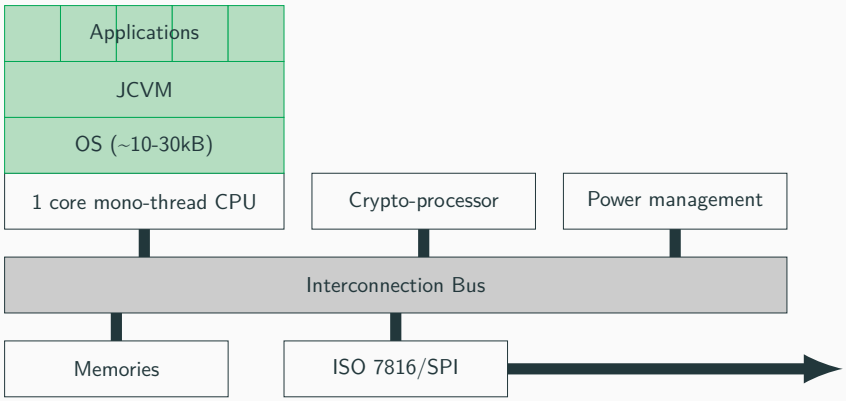


Smartphones

Introduction - SEs vs SoCs

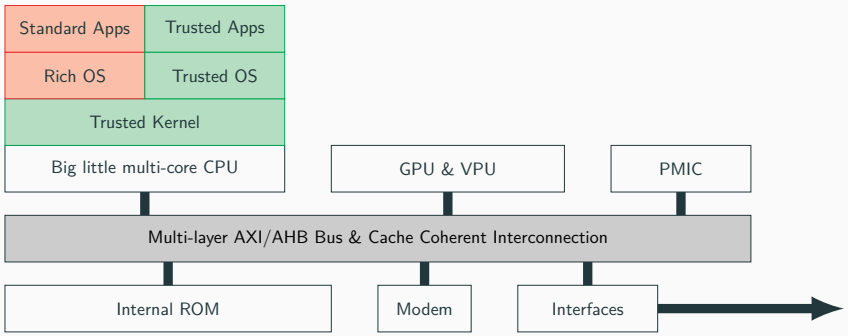
Smartcards

- secure elements (SEs)

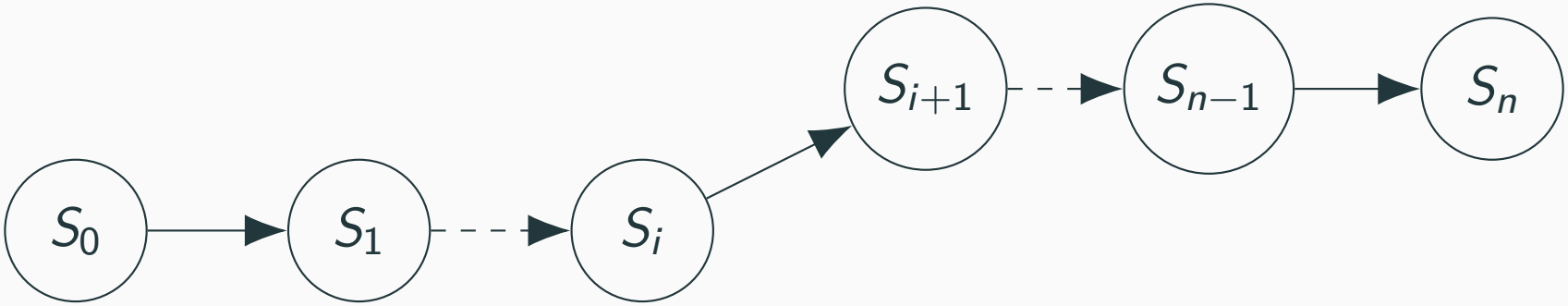


Smartphones

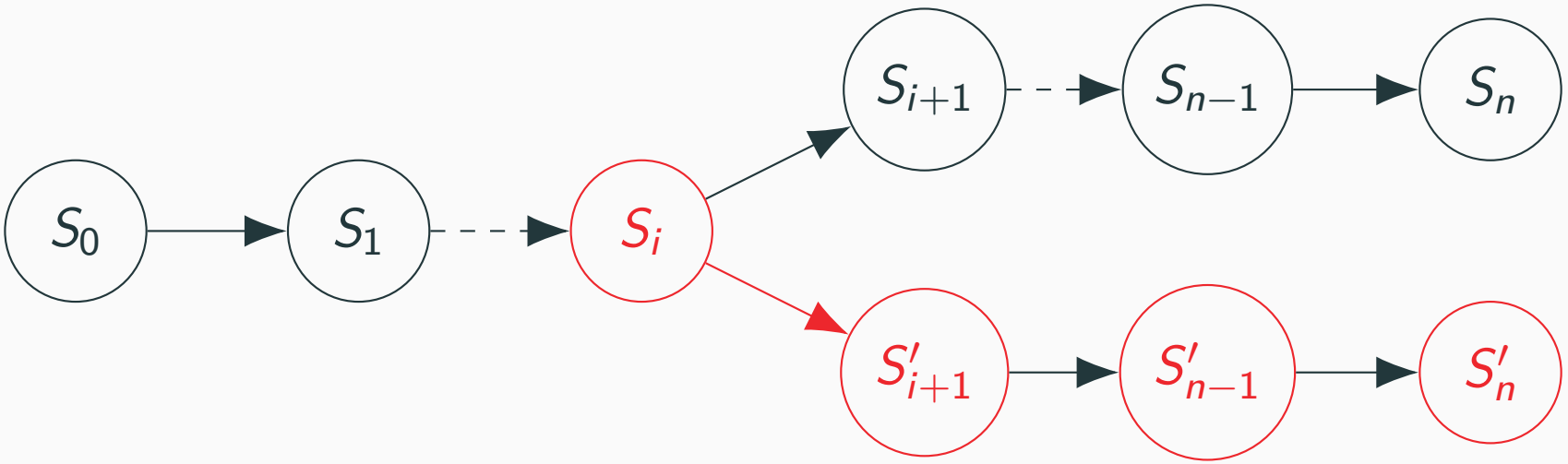
- complex systems on chip (SoCs)



Introduction - Perturbation attacks



Introduction - Perturbation attacks



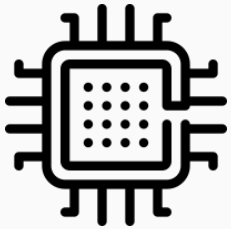
Introduction - Perturbation attacks

Electromagnetic waves

Temperature

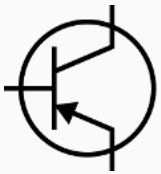
Voltage

Light



Clock

X-ray



Body biasing

Software

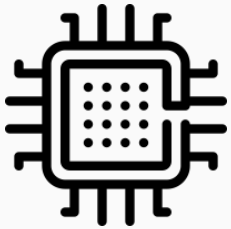
Introduction - Perturbation attacks

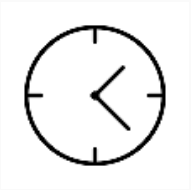
Electromagnetic waves [OGM15; DLM19]

Temperature

Voltage

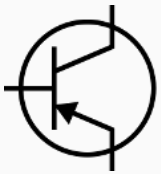
Light [Sam+02; SHP09]





Clock

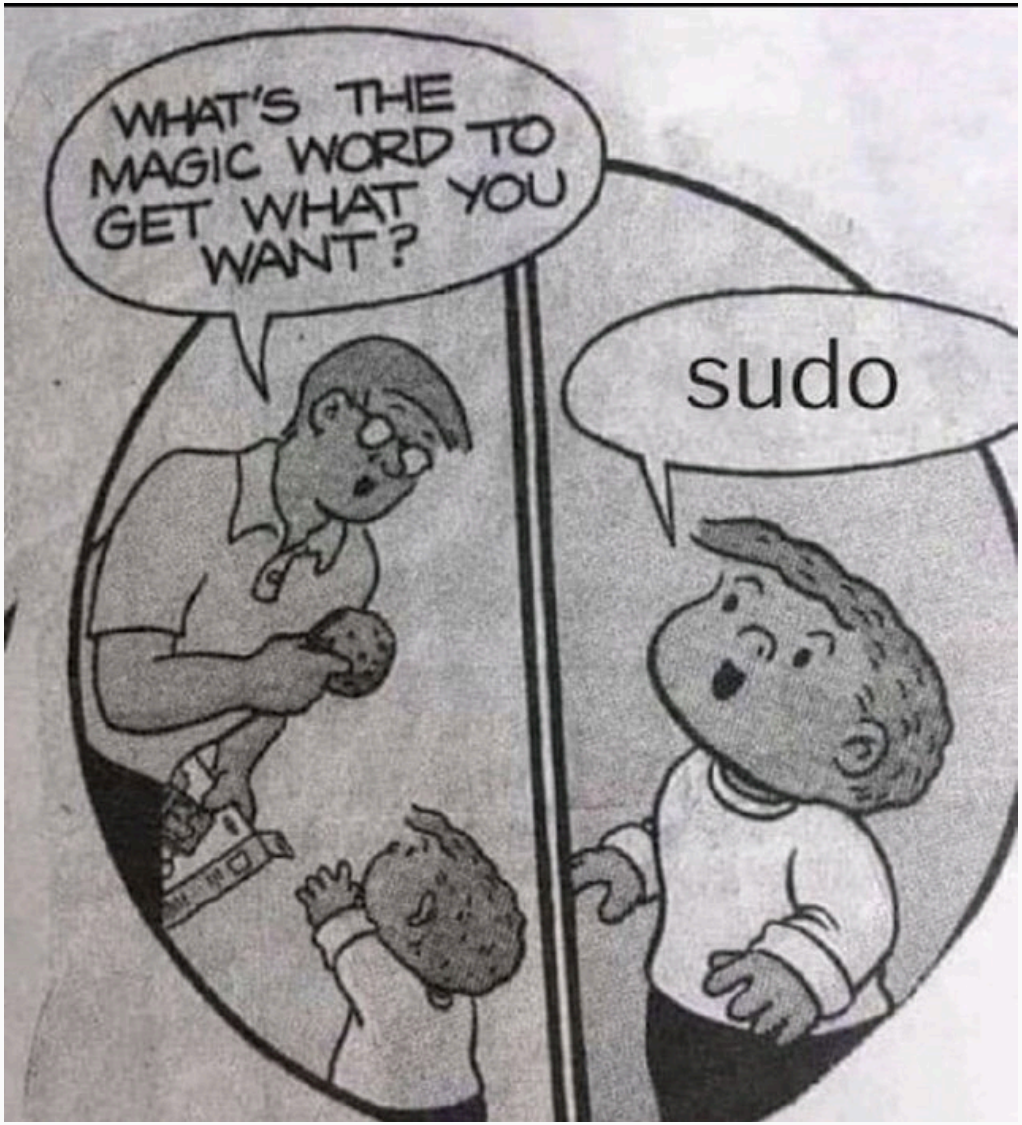
X-ray



Body biasing

Software

Case study - User authentication on Linux



Case study - User authentication on Linux

Password authentication of the `sudo` program on Debian 9

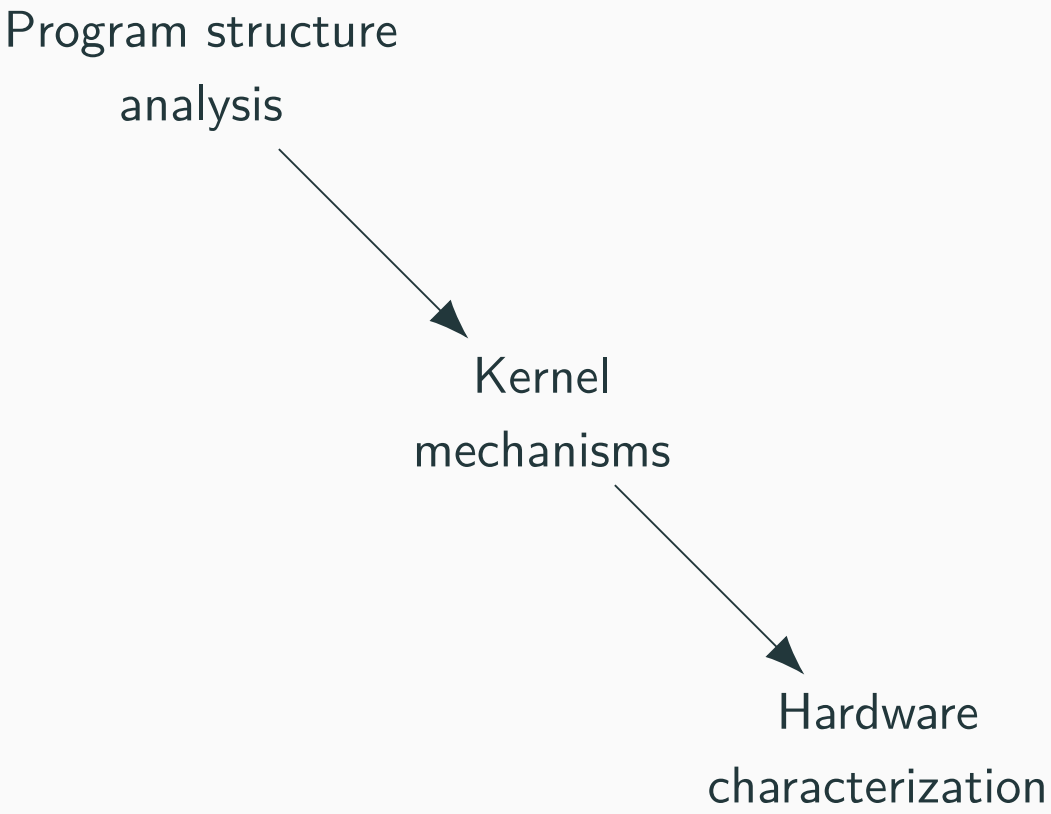
Targets

BCM2837
(Raspberry Pi 3 B)

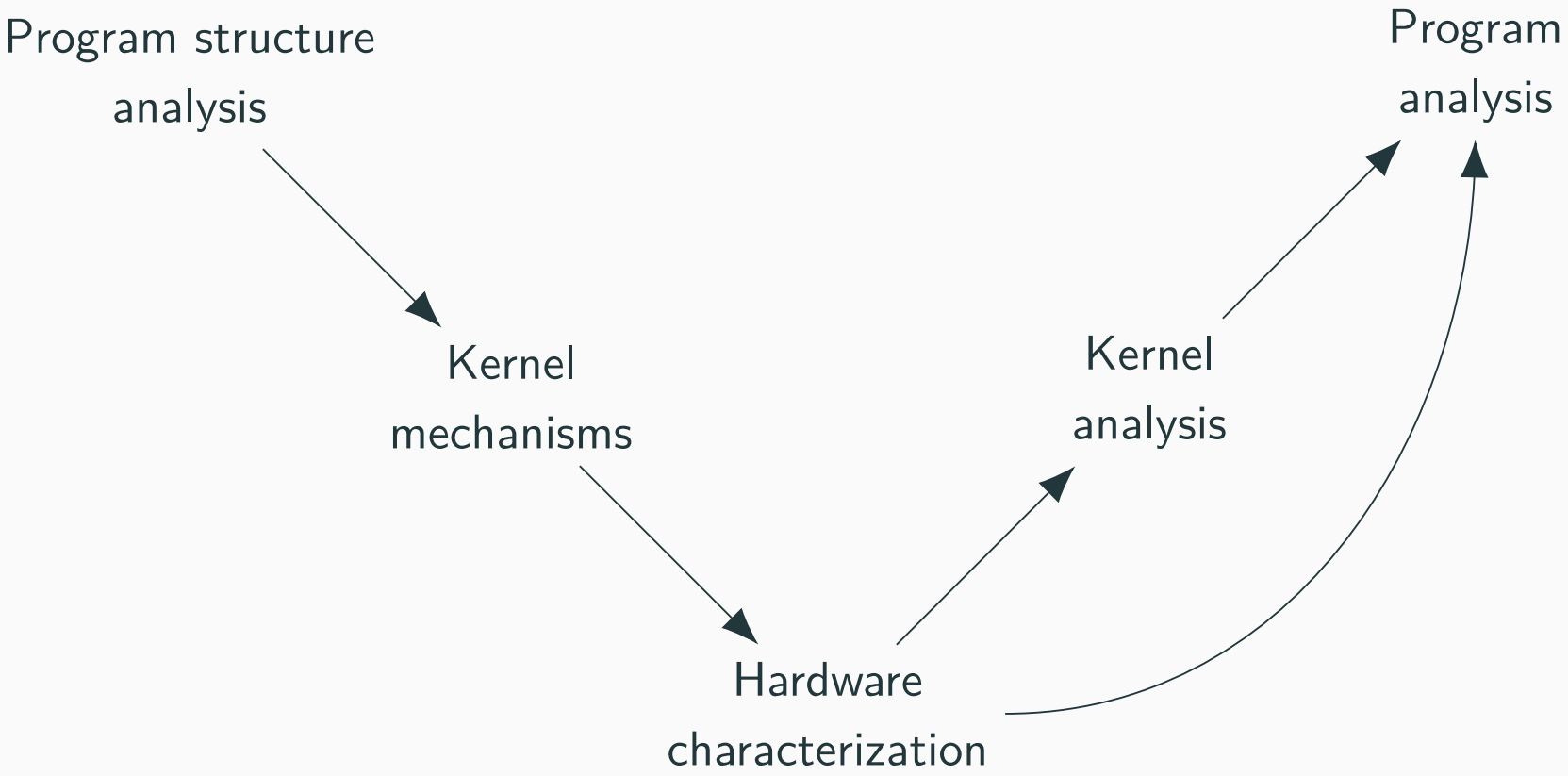
Intel Core i3-6100T
(Custom motherboard)

BCM2711b0
(Raspberry Pi 4)

Case study - Evaluation methodology



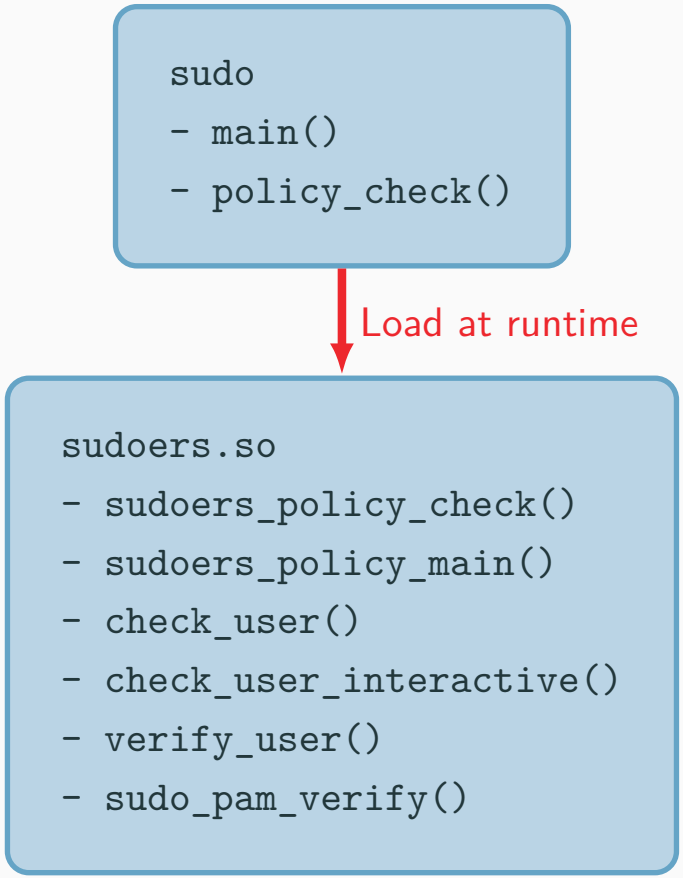
Case study - Evaluation methodology



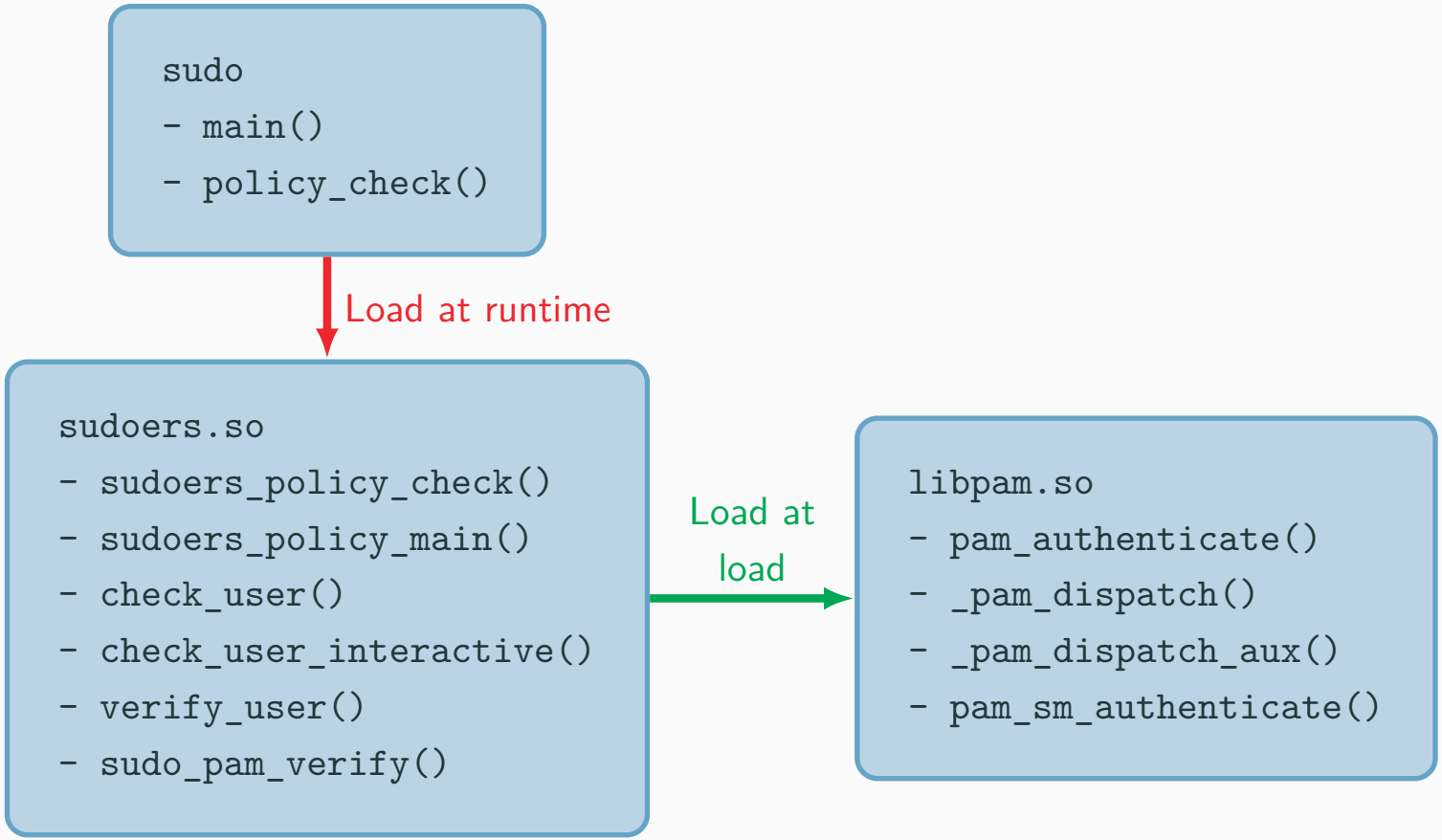
Case study - `sudo` program life

```
sudo
- main()
- policy_check()
```

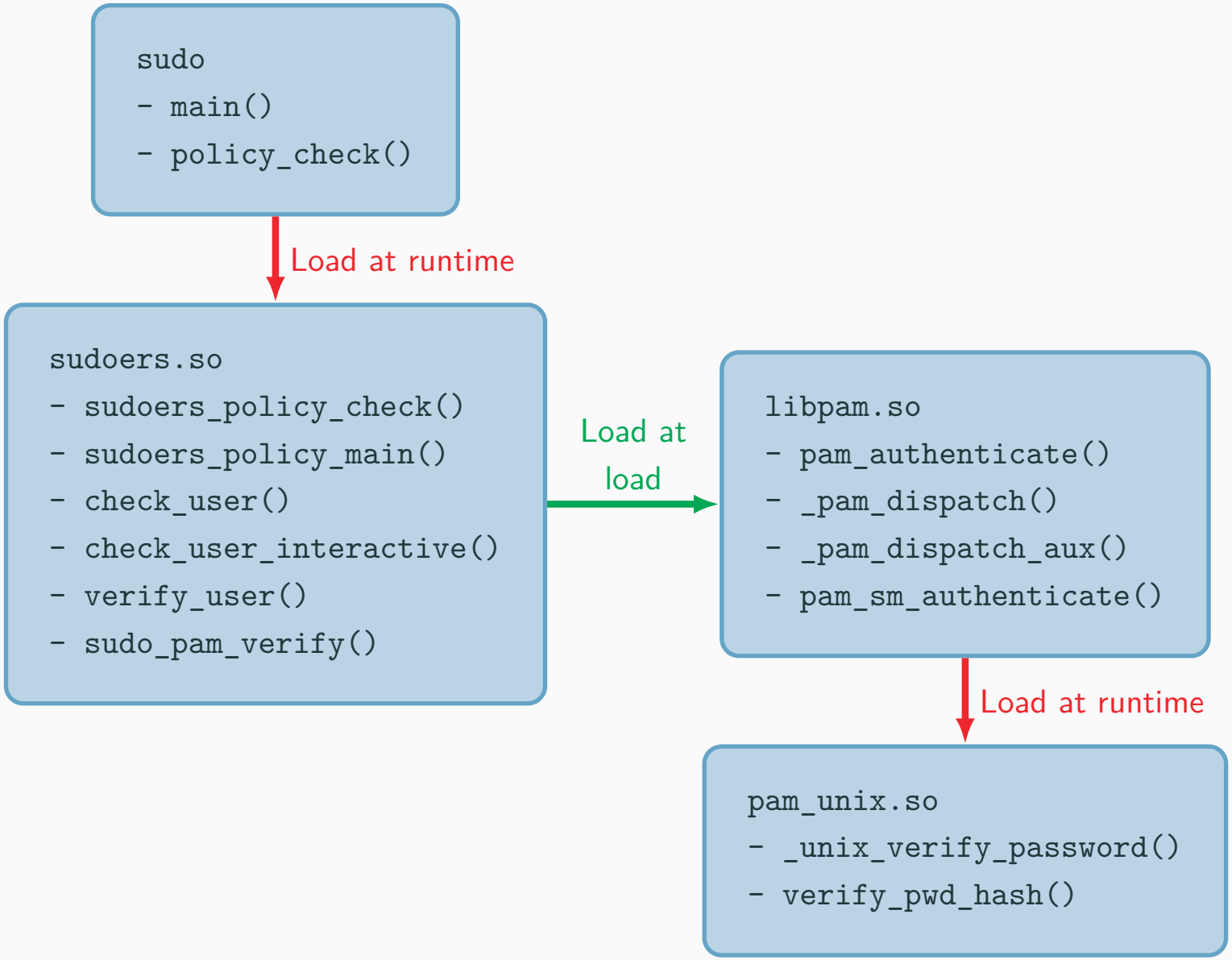
Case study - sudo program life



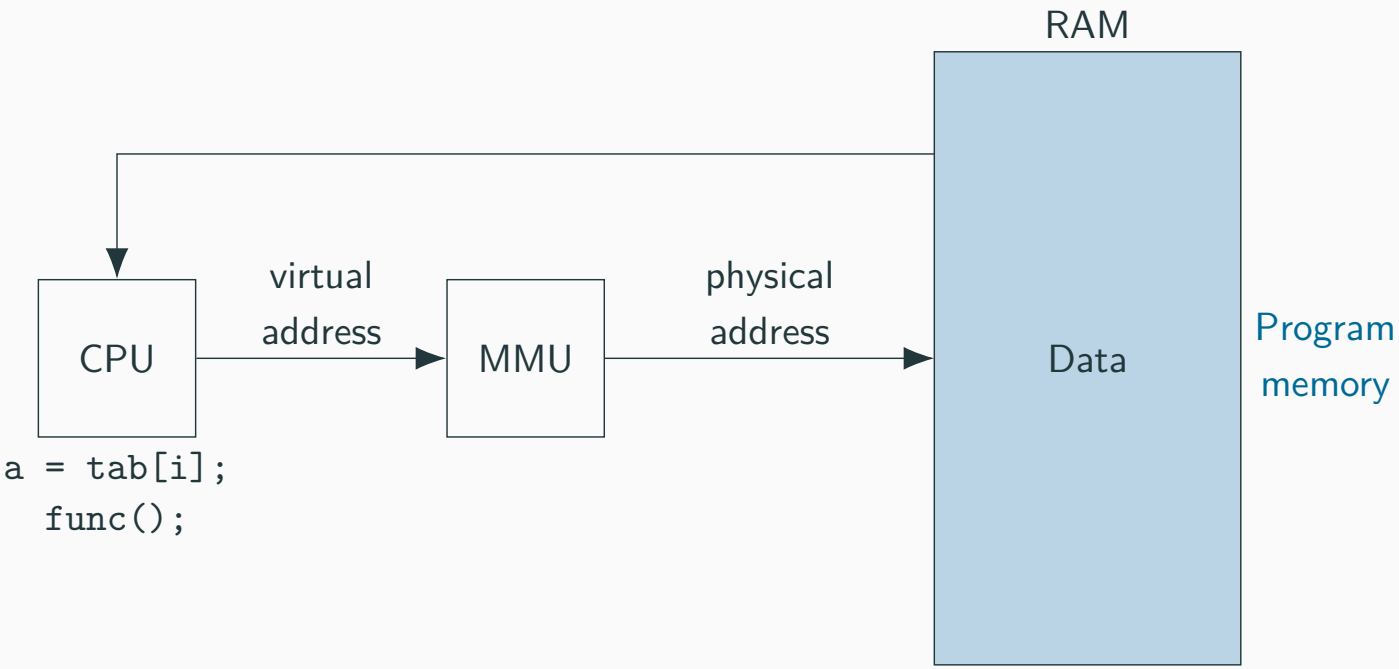
Case study - sudo program life



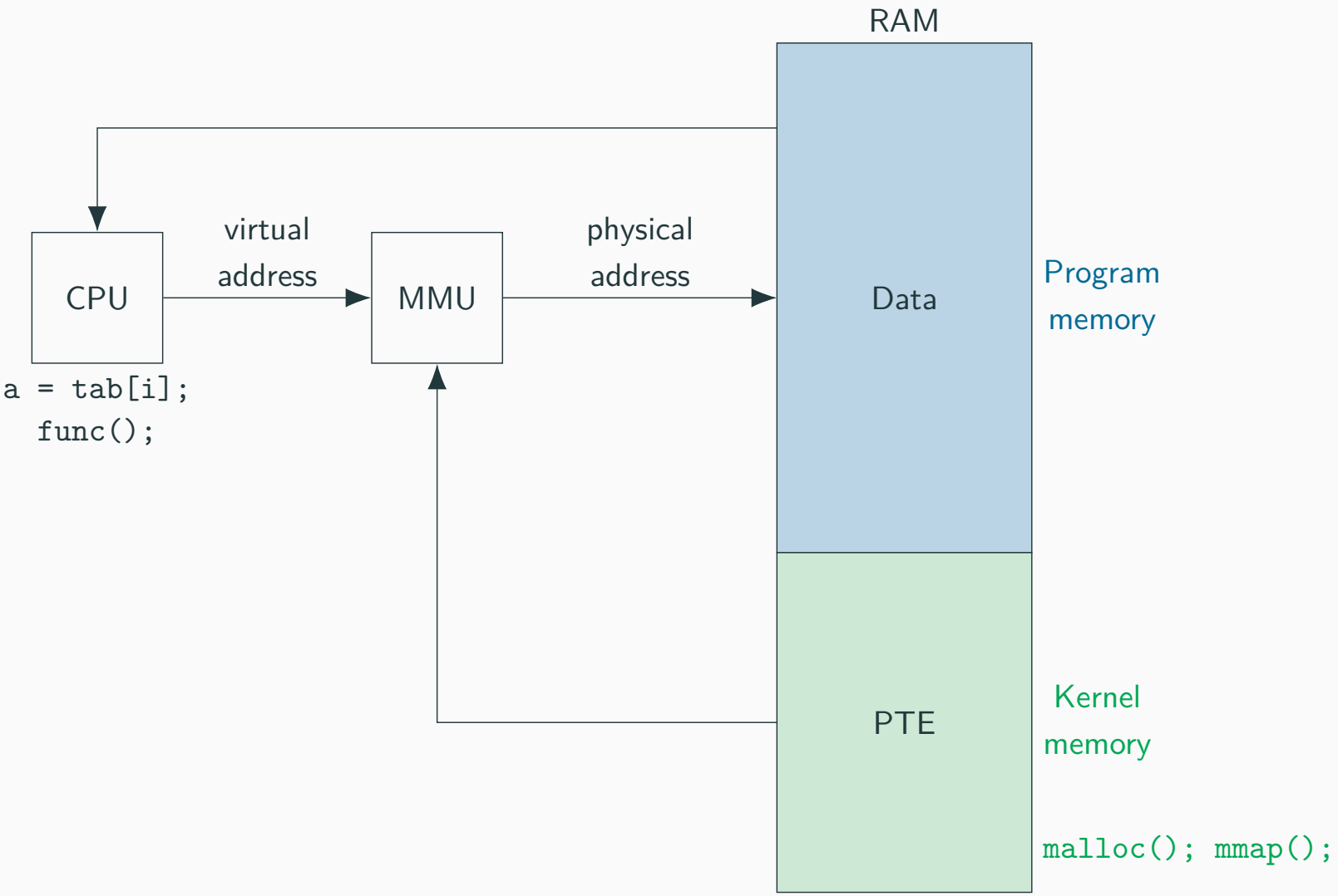
Case study - sudo program life



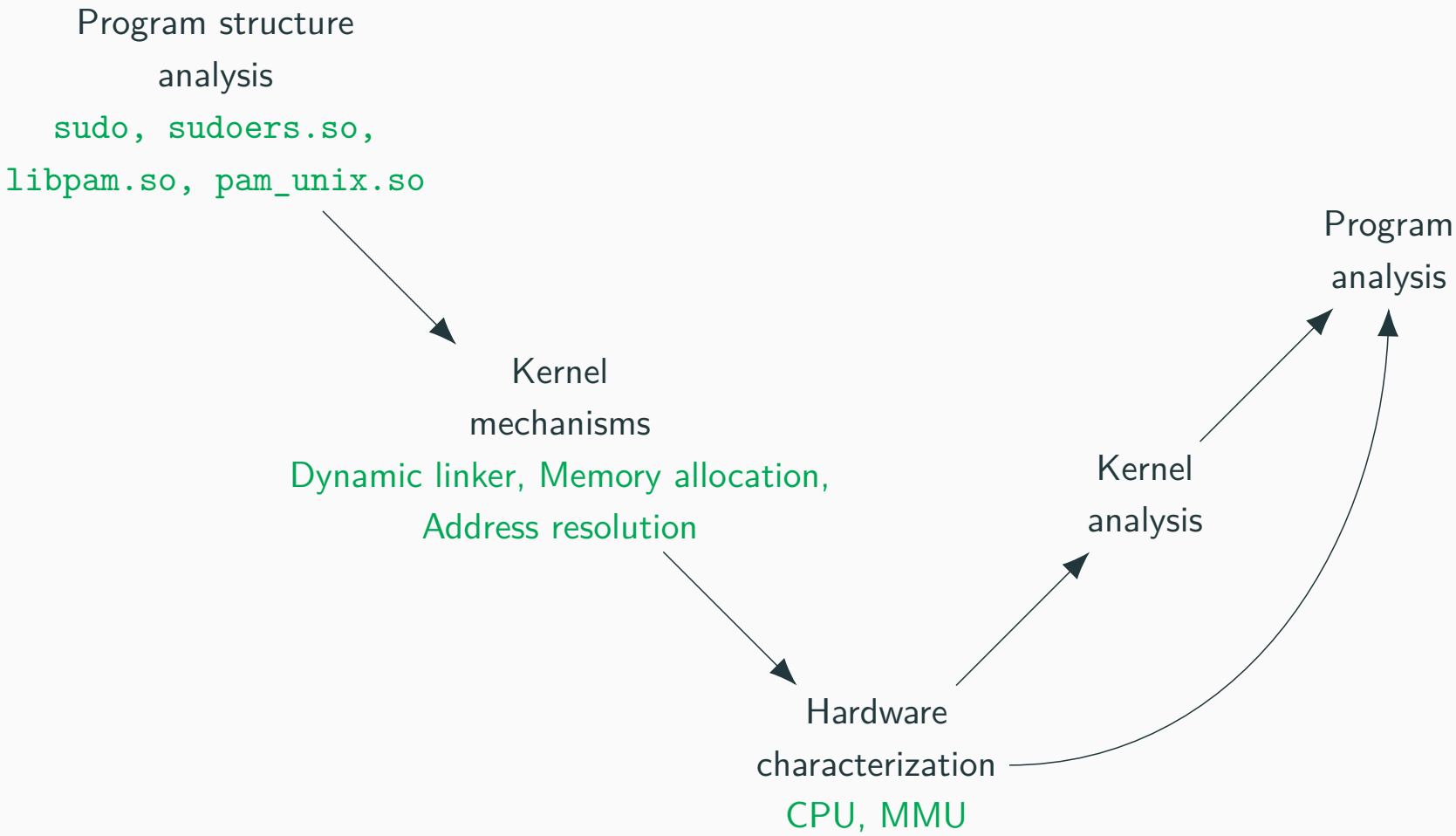
Case study - Address resolution



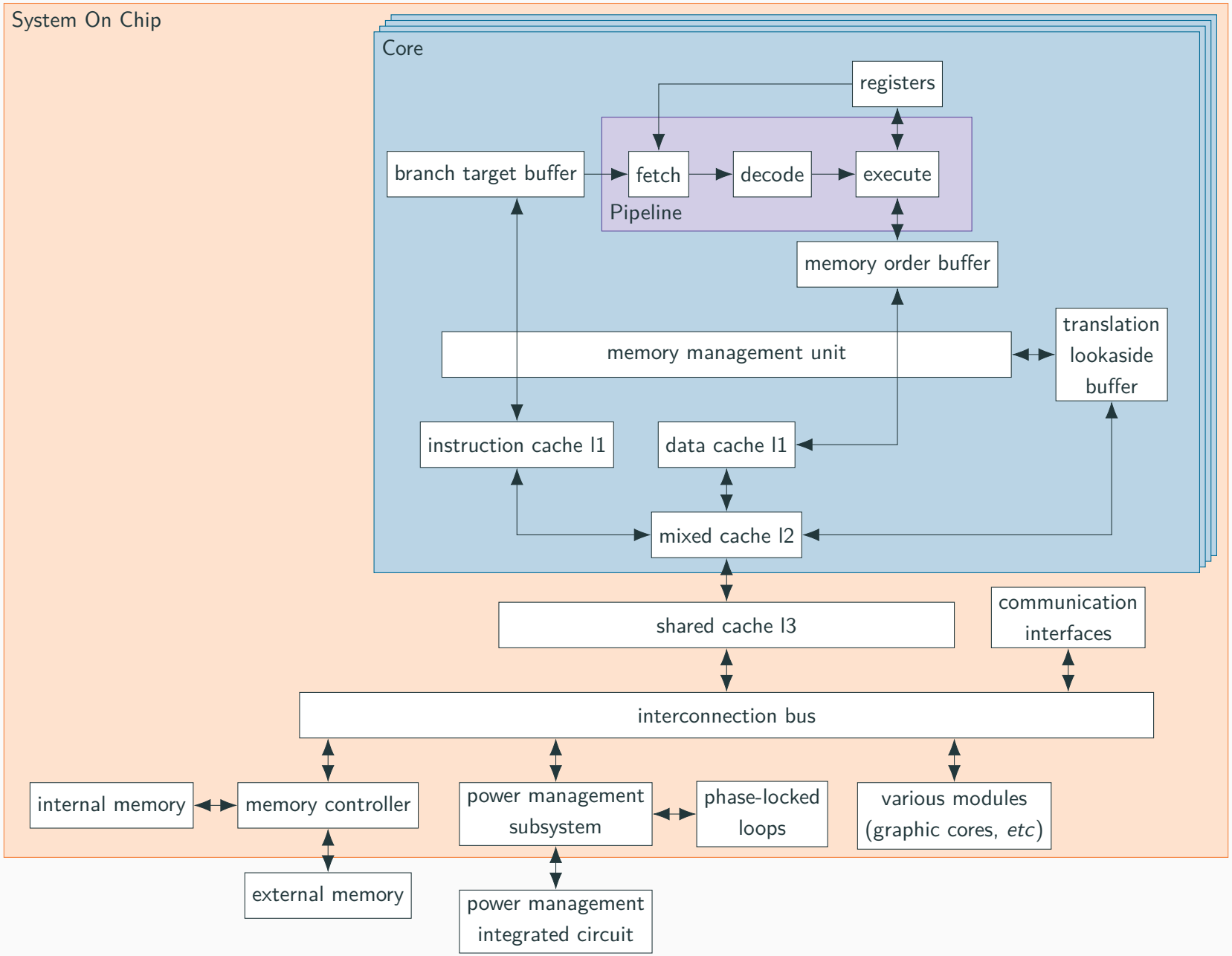
Case study - Address resolution



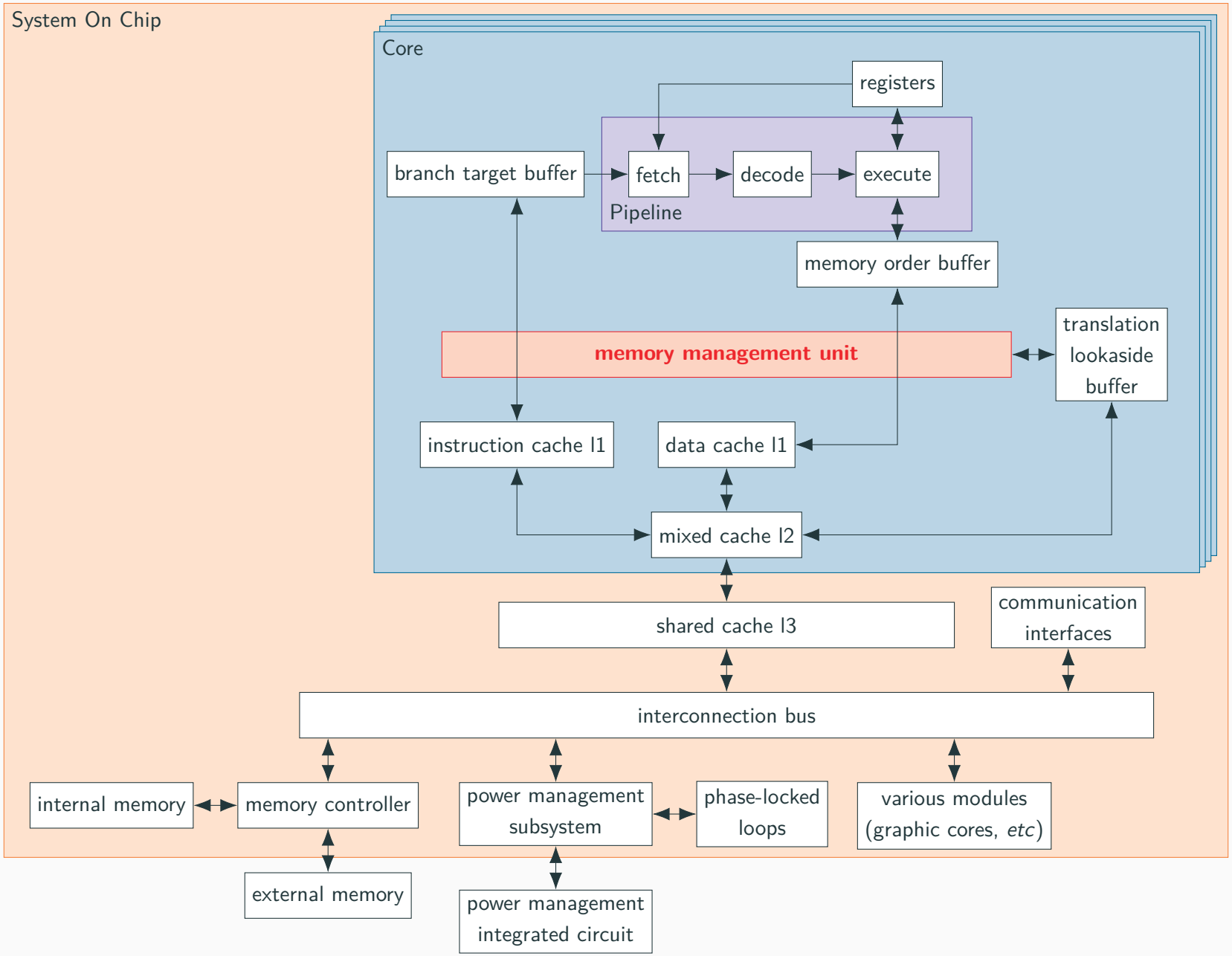
Case study - Evaluation methodology



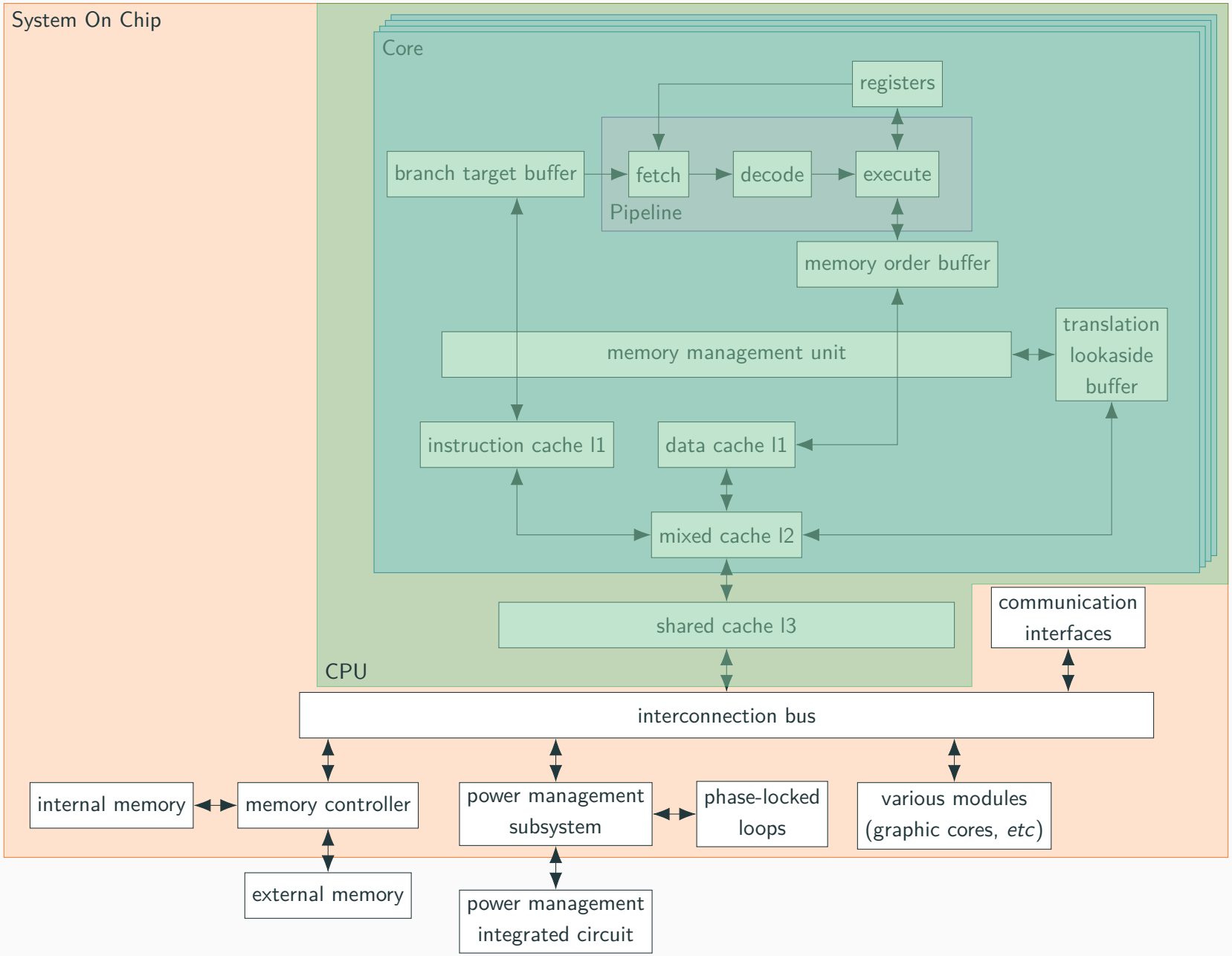
Case study - SoC architecture



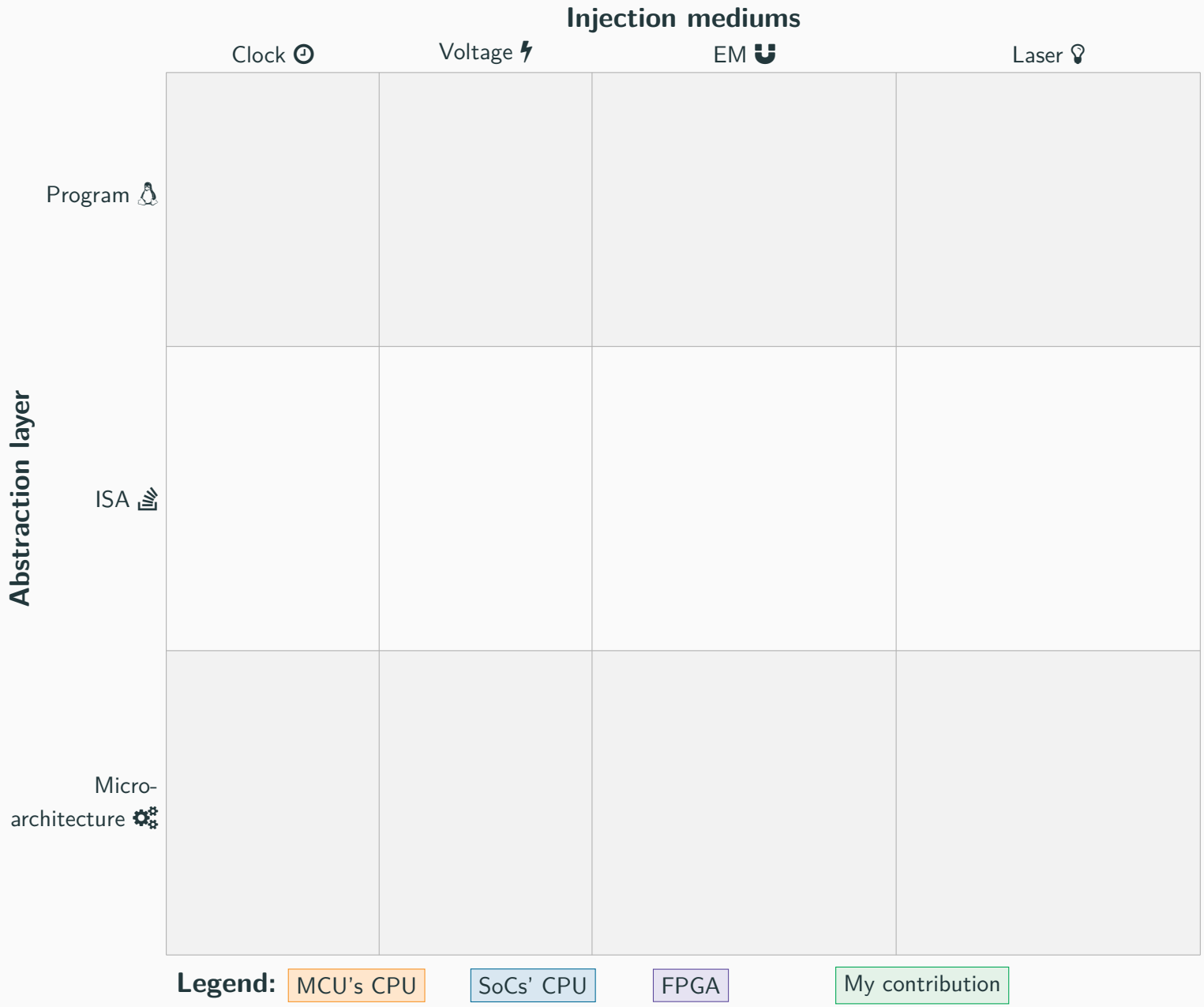
Case study - SoC architecture



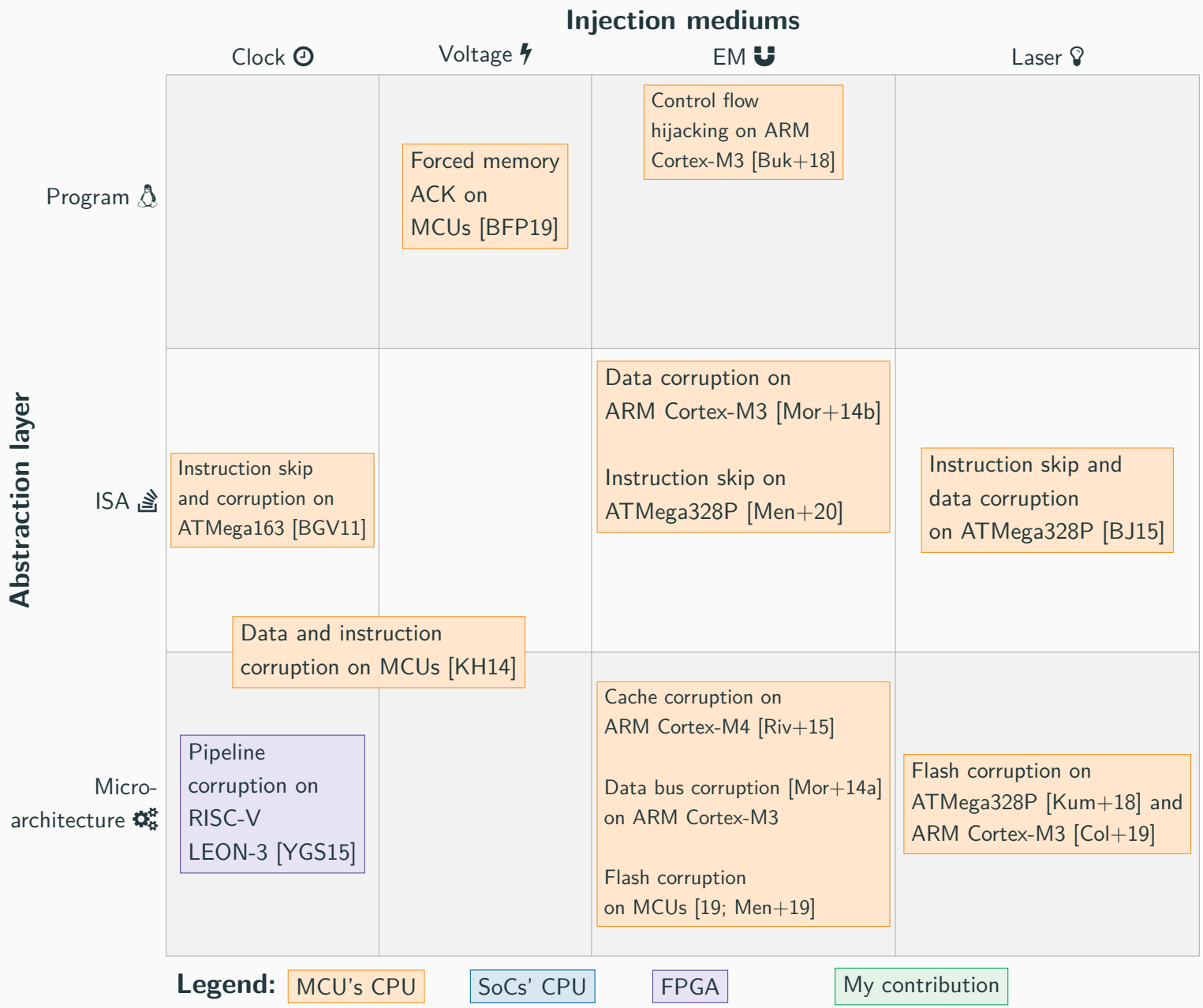
Case study - SoC architecture



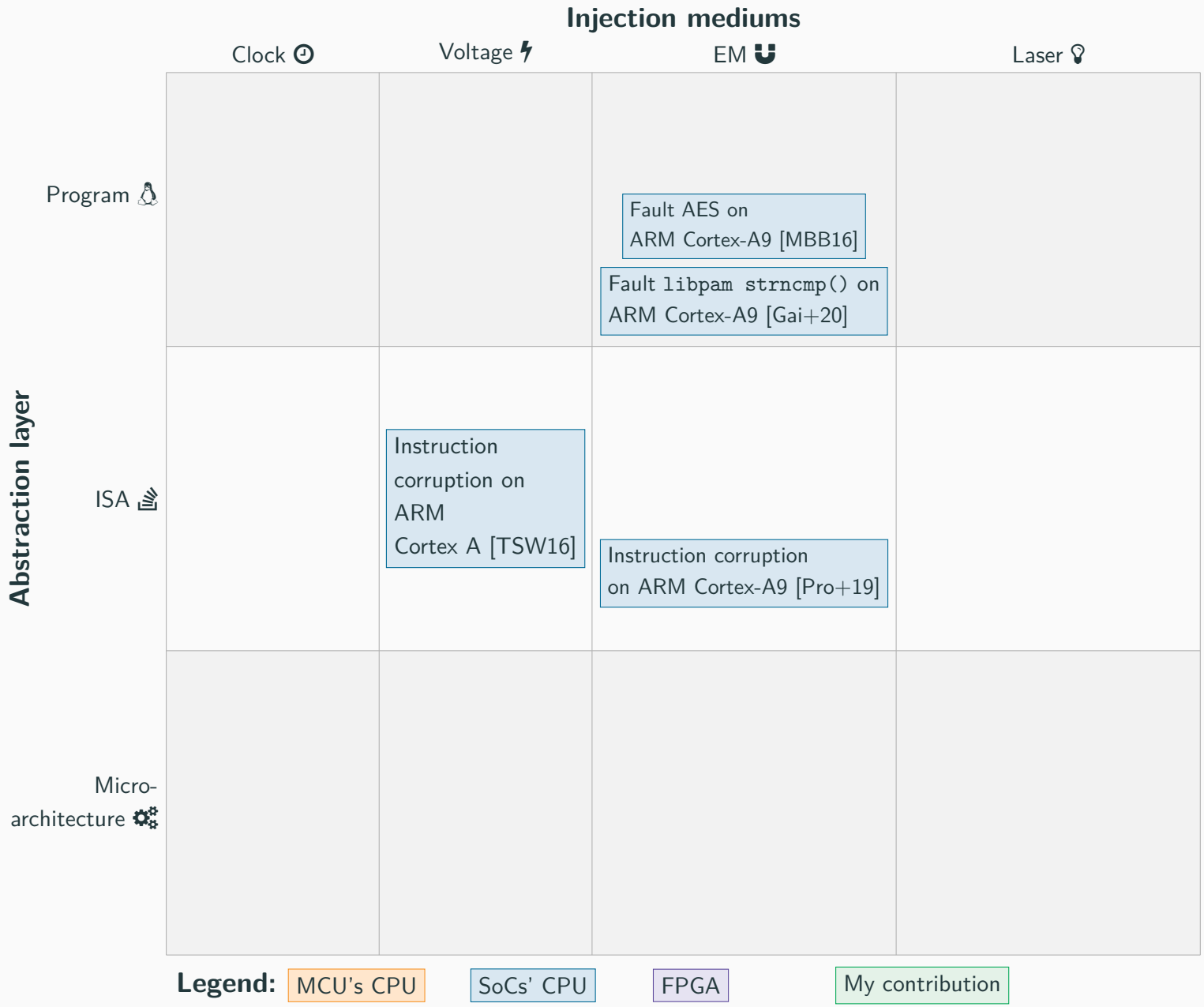
Characterization - State of the art



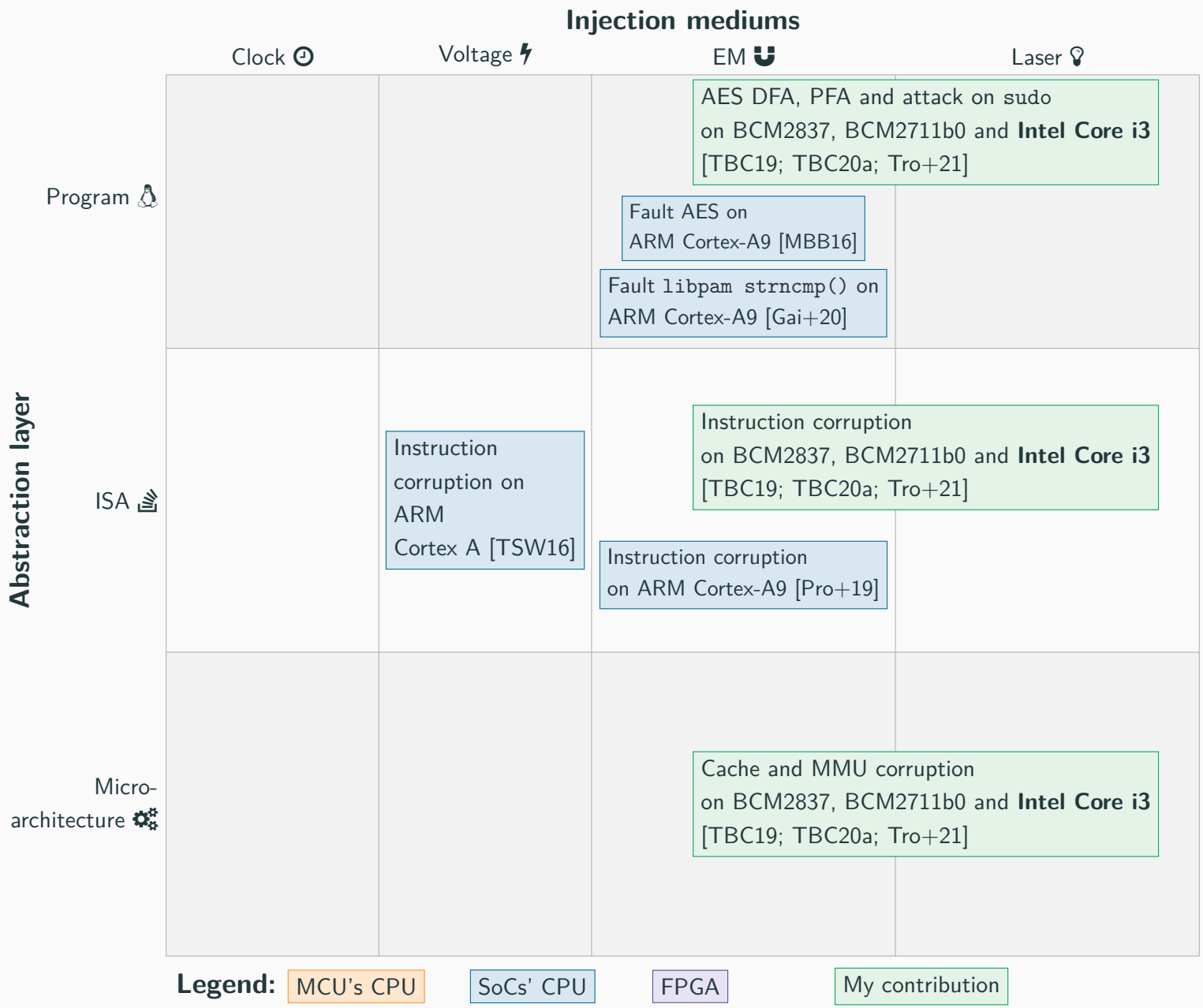
Characterization - State of the art



Characterization - State of the art



Characterization - State of the art



Case study - Characterization Method

Test program

```
orr r5, r5;
/*
 * Arbitrary number
 * of repetitions
 */
orr r5, r5;
```

Case study - Characterization Method

Test program

```
orr r5, r5;
/*
 * Arbitrary number
 * of repetitions
 */
orr r5, r5;
```

Initial values

Register	Initial values
r0	0xfffe0001
r1	0xfffd0002
r2	0xfffb0004
r3	0xfff70008
r4	0xffef0010

Case study - Characterization Method

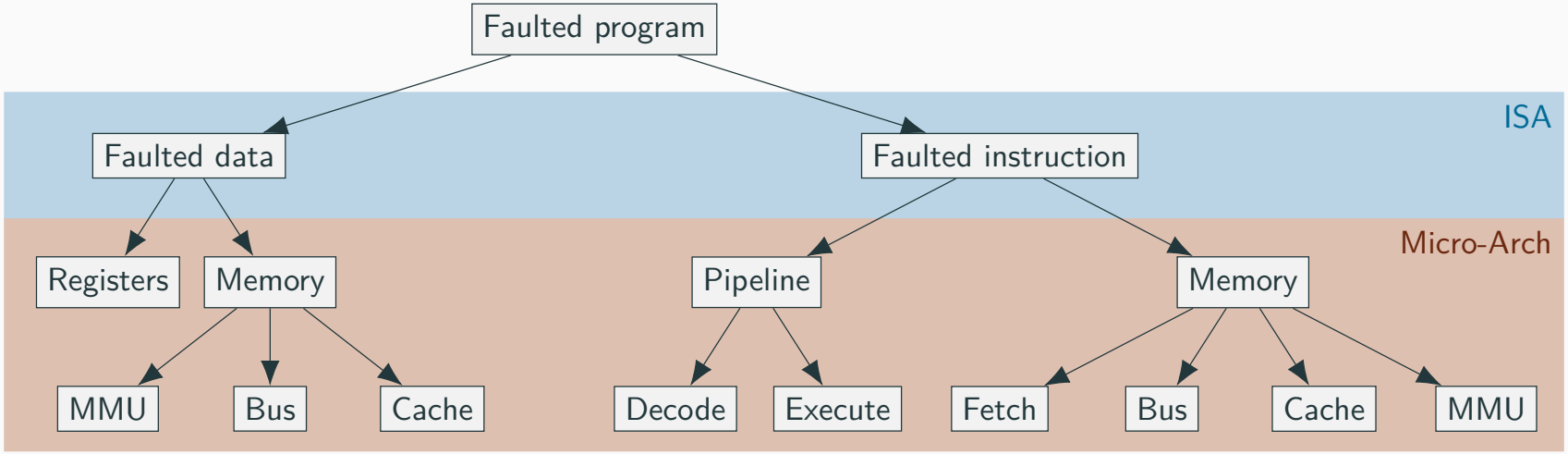
Test program

```
orr r5, r5;
/*
 * Arbitrary number
 * of repetitions
 */
orr r5, r5;
```

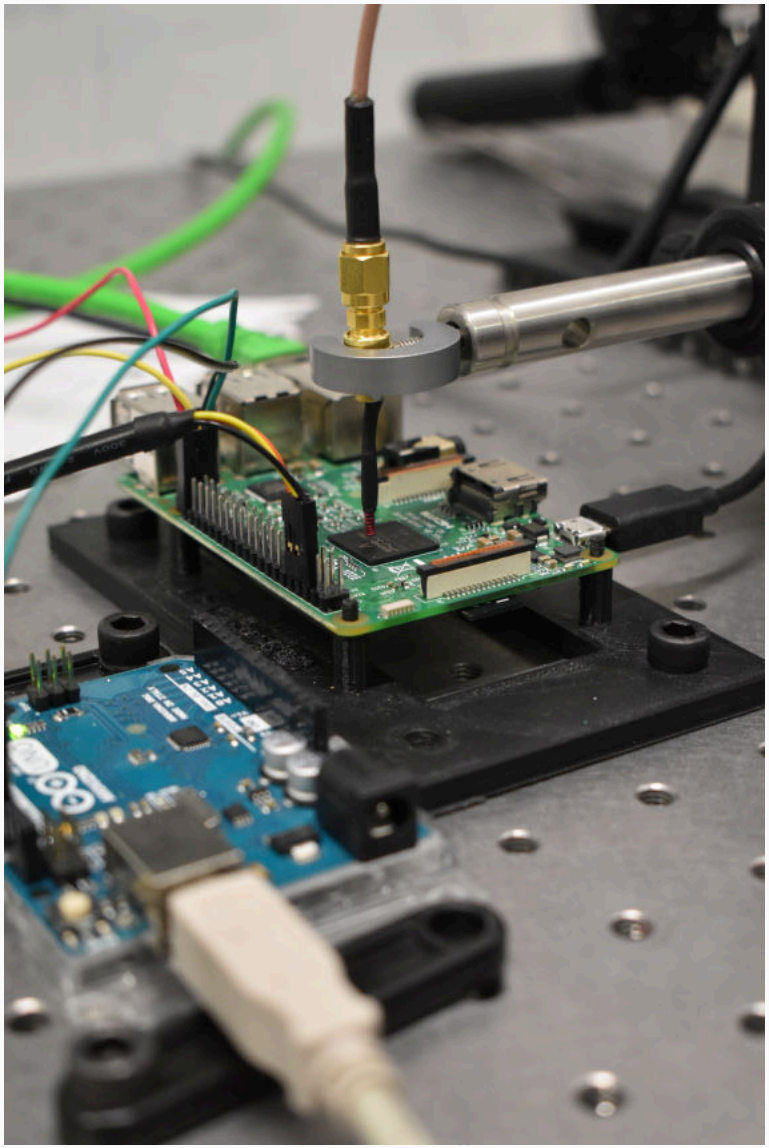
Initial values

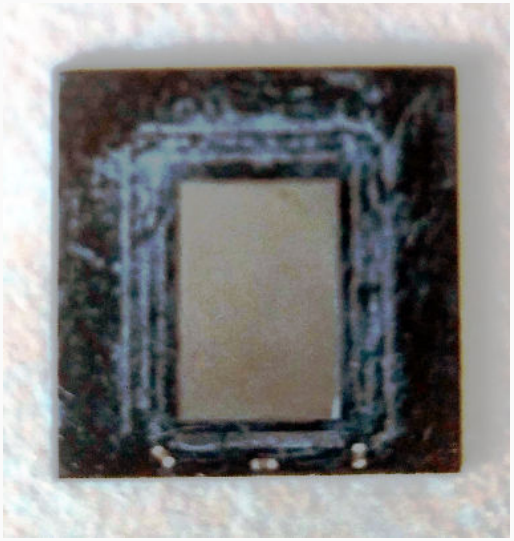
Register	Initial values
r0	0xfffe0001
r1	0xfffd0002
r2	0xfffb0004
r3	0xfff70008
r4	0xffef0010

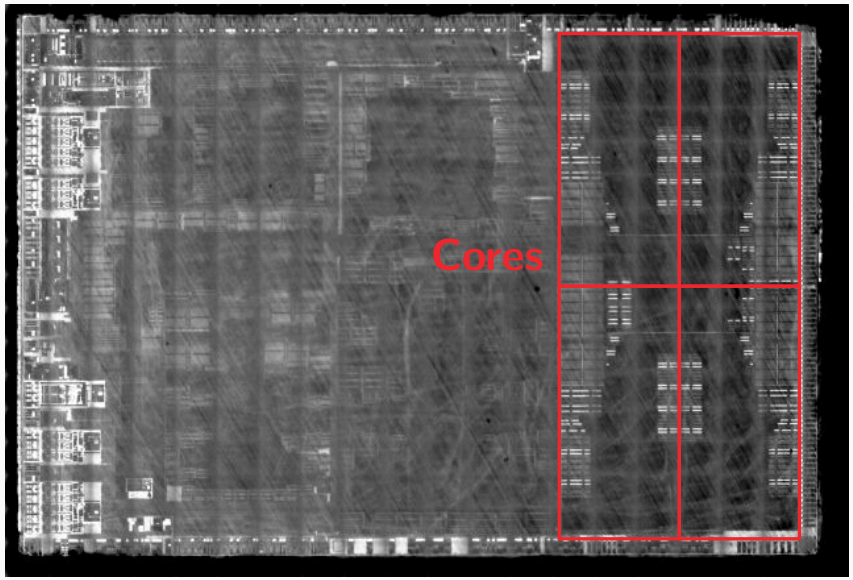
Top down approach



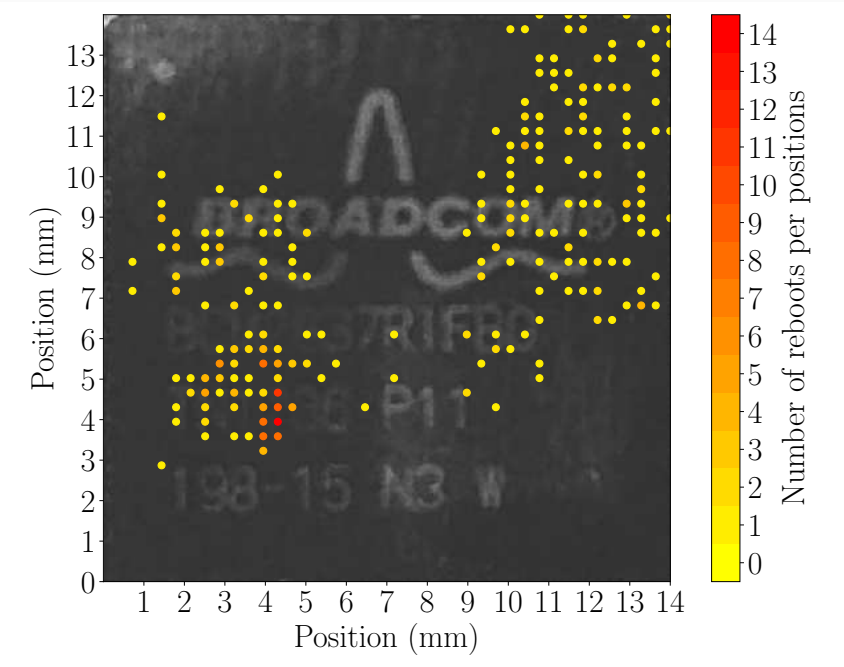
Characterization - BCM2837 (Raspberry Pi 3)



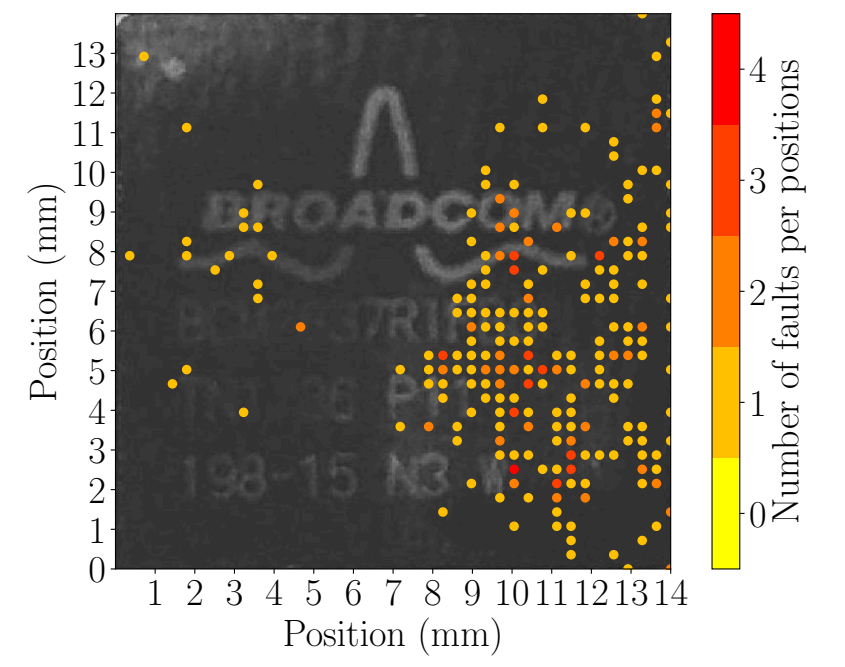




Characterization - BCM2837 (Raspberry Pi 3)



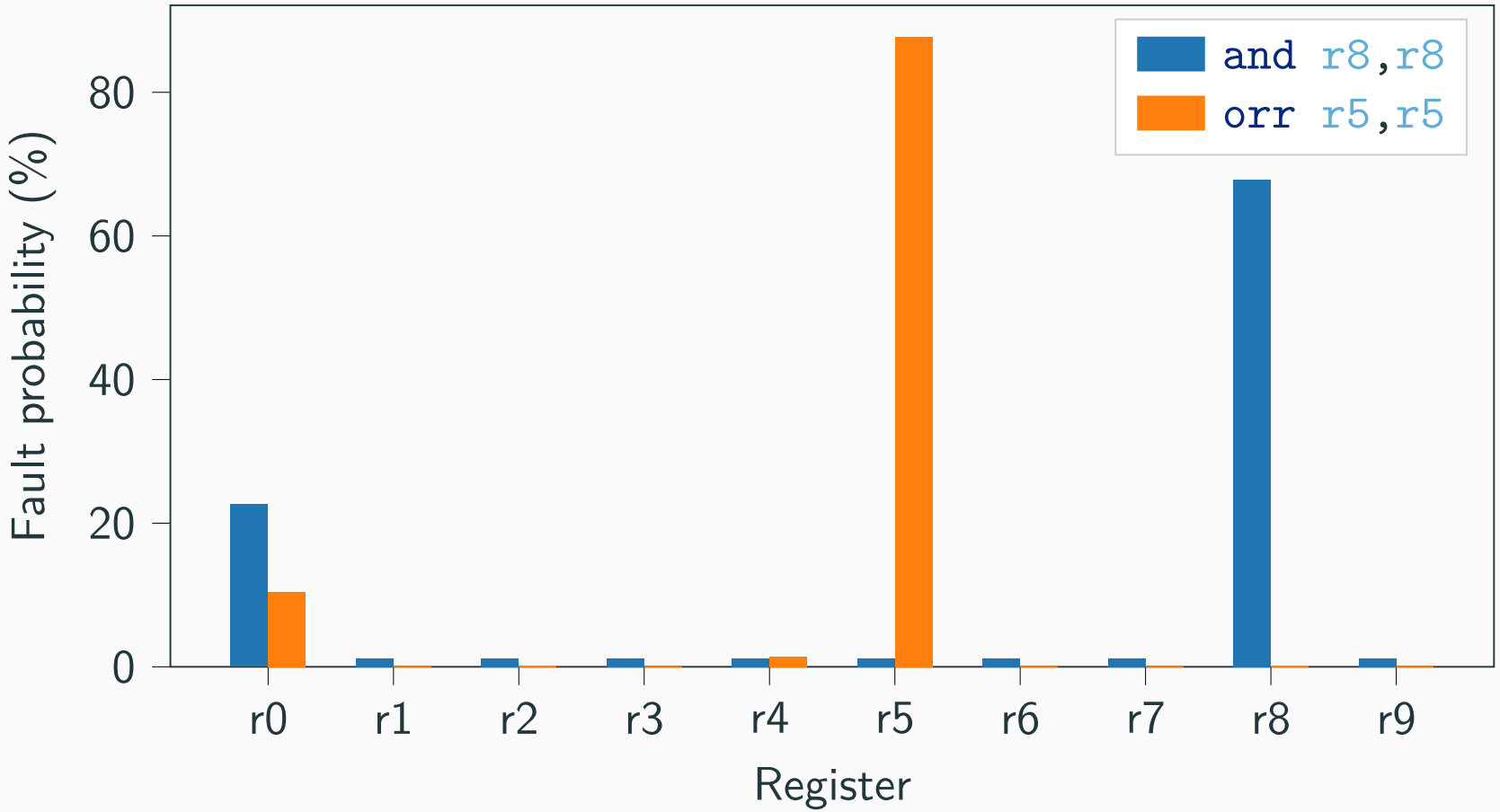
Spots leading to reboots



Spots leading to faults

Characterization - BCM2837

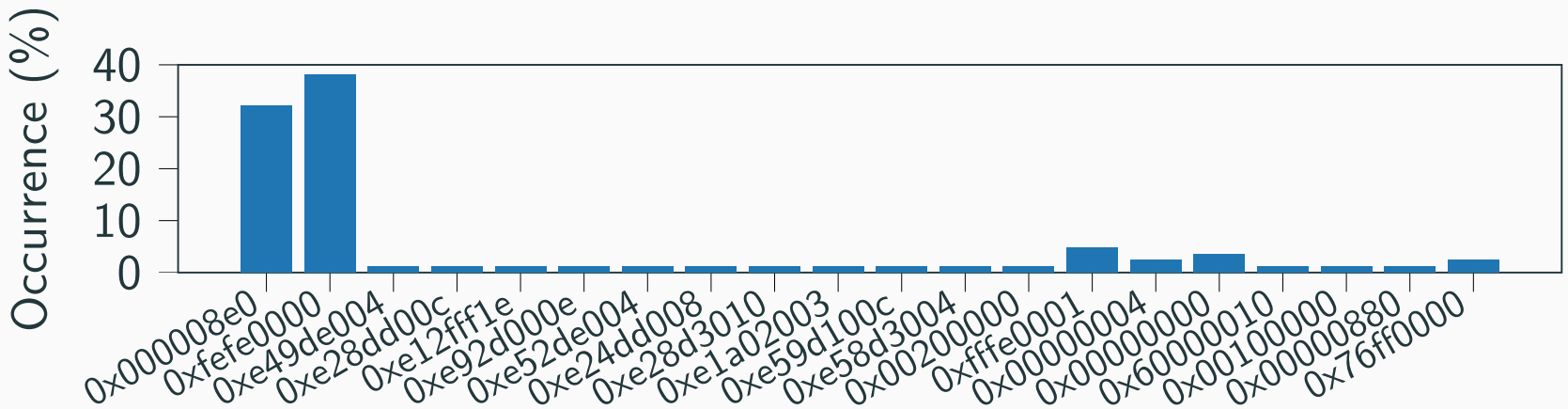
Faulted register distribution regarding the executed instruction



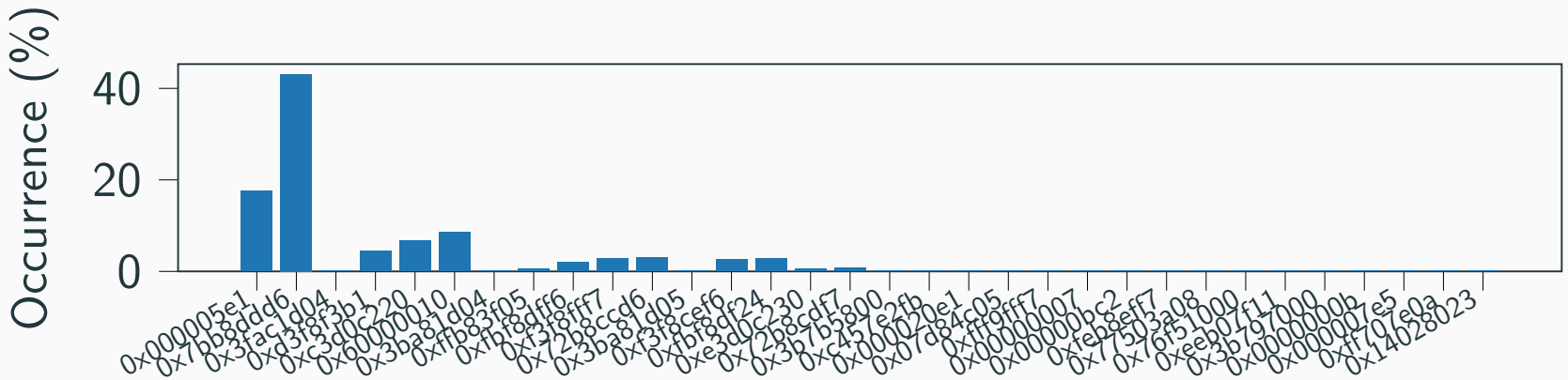
Characterization - BCM2837

Faulted value distribution regarding the executed instruction

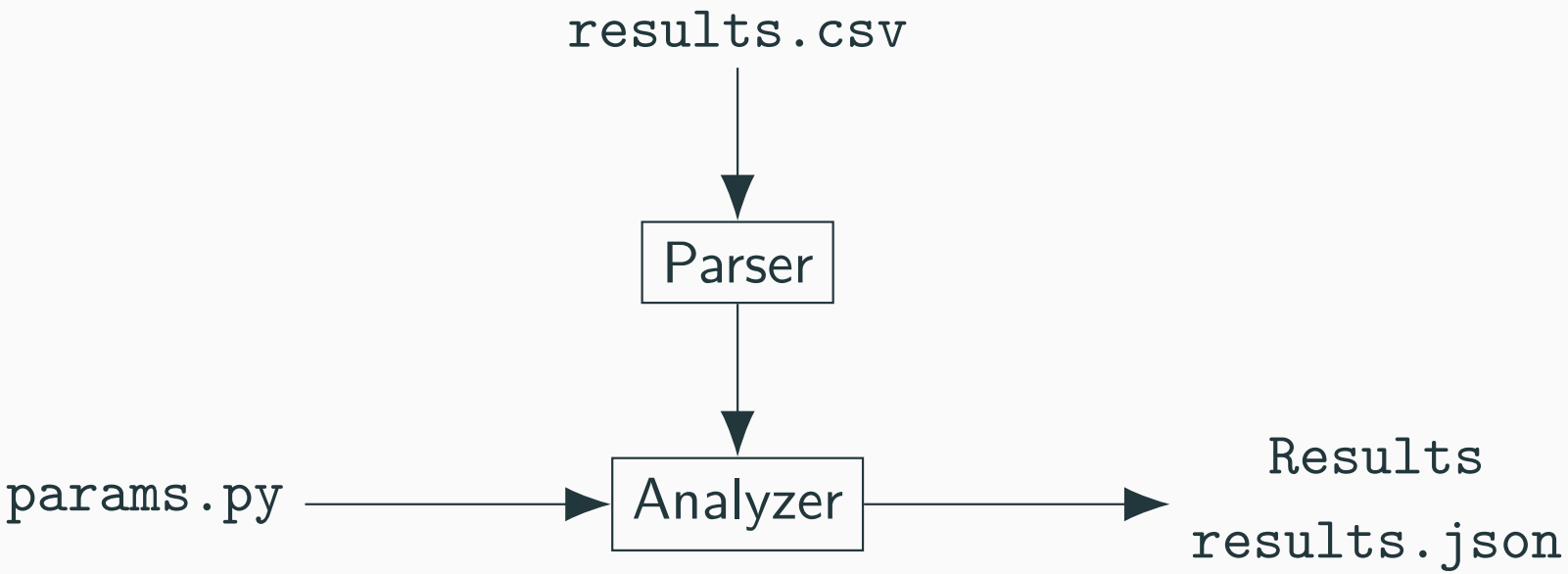
`and r8,r8`



`orr r5,r5`



Characterization - Analysis tool



Characterization - BCM2837

Fault model distribution regarding the executed instruction

Characterization - BCM2837

Instruction matching the "Or with other obs" fault model for the `orr r5,r5` instruction

Faulted instruction	Occurrence (%)
`orr r5,r1`	92.54 %
`orr r5,r0`	6.14 %
`orr r5,r7`	1.32 %

Characterization - BCM2837

Instruction matching the "Or with other obs" fault model for the `orr r5,r5` instruction

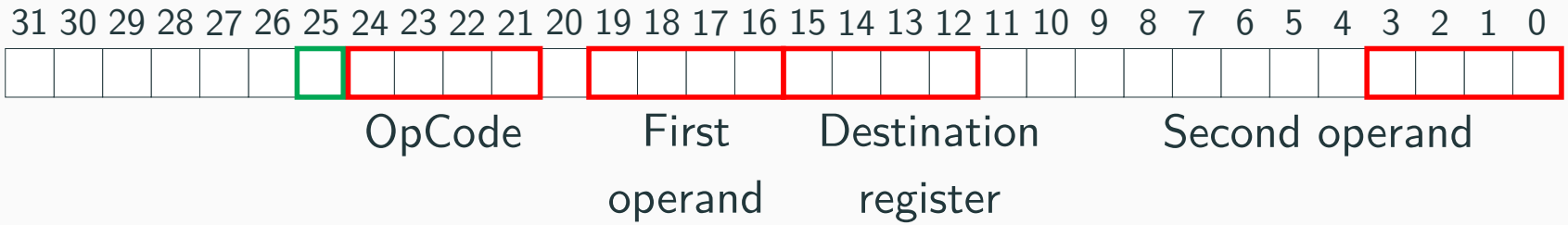
Faulted instruction	Occurrence (%)
`orr r5,r1`	92.54 %
`orr r5,r0`	6.14 %
`orr r5,r7`	1.32 %

Instruction matching the "And with other obs" fault model for the `and r8,r8` instruction

Faulted instruction	Occurrence (%)
`and r8,r0`	100 %

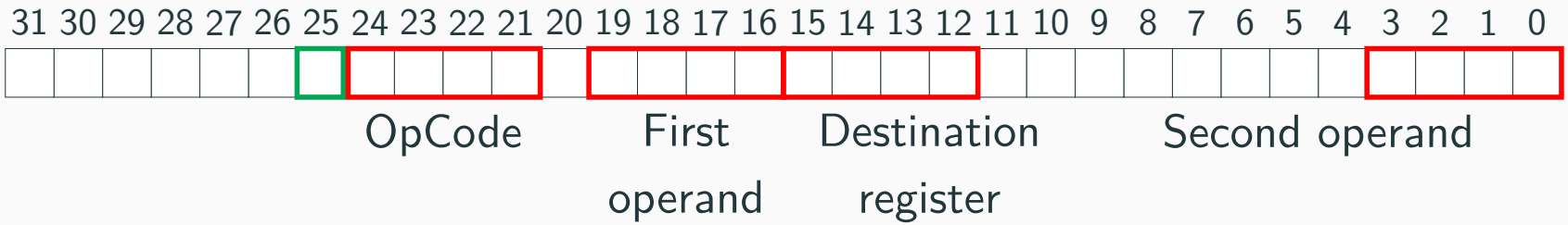
ARM data processing instruction encoding

If immediate value bit (25) is set to 0

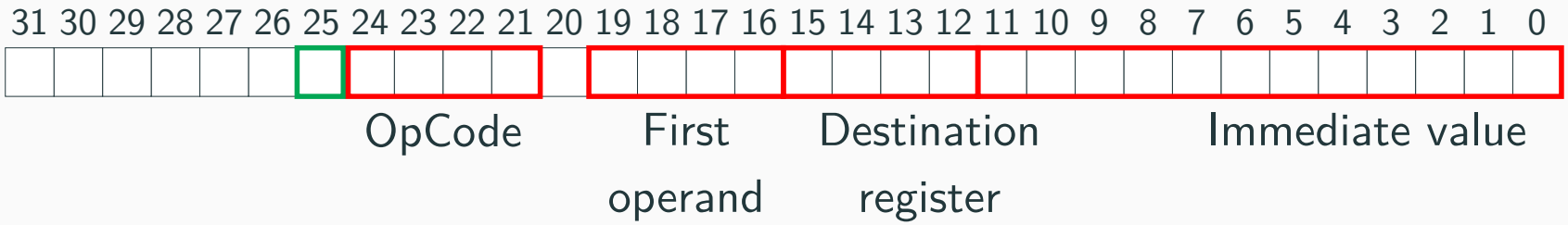


ARM data processing instruction encoding

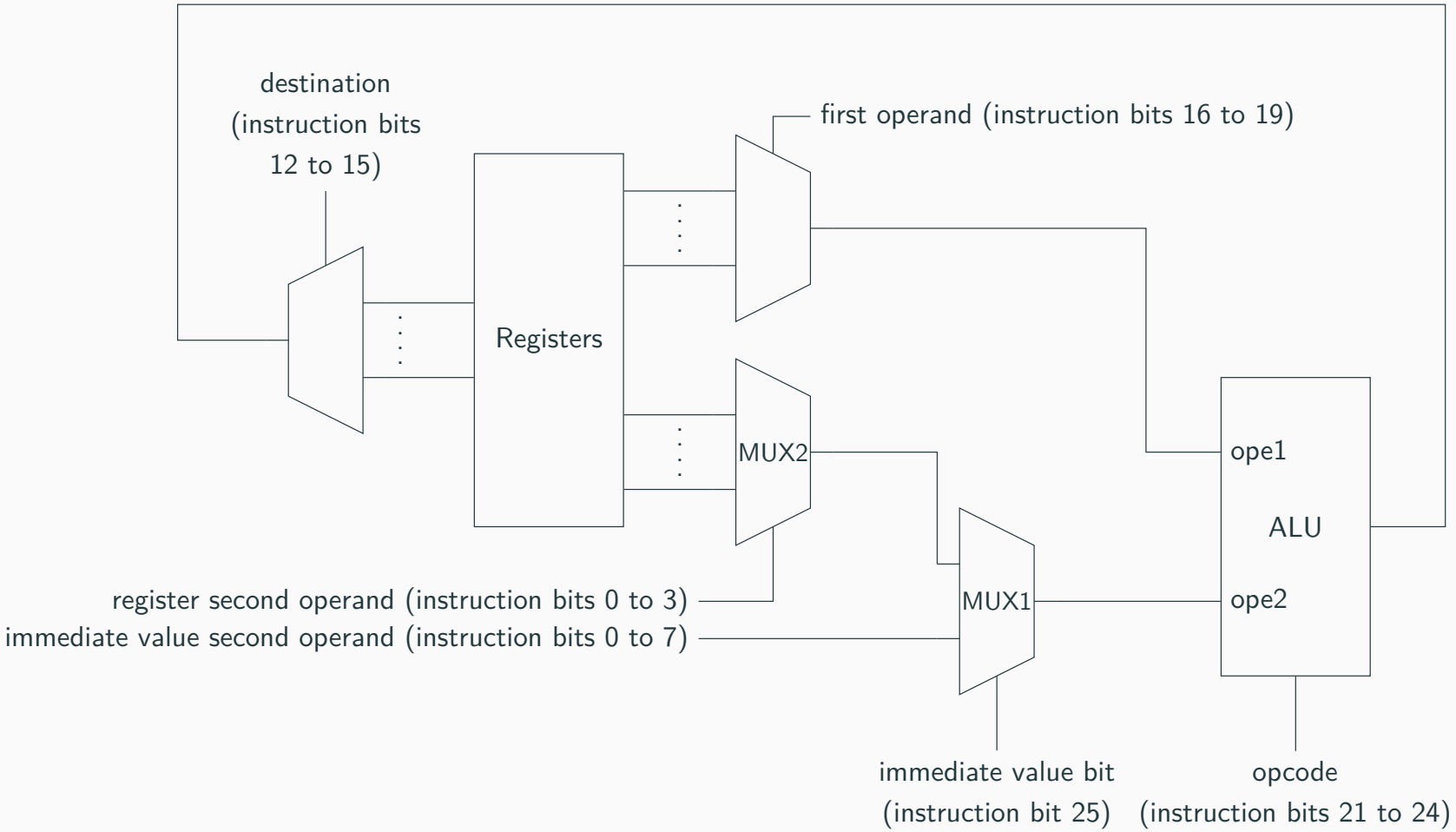
If immediate value bit (25) is set to 0



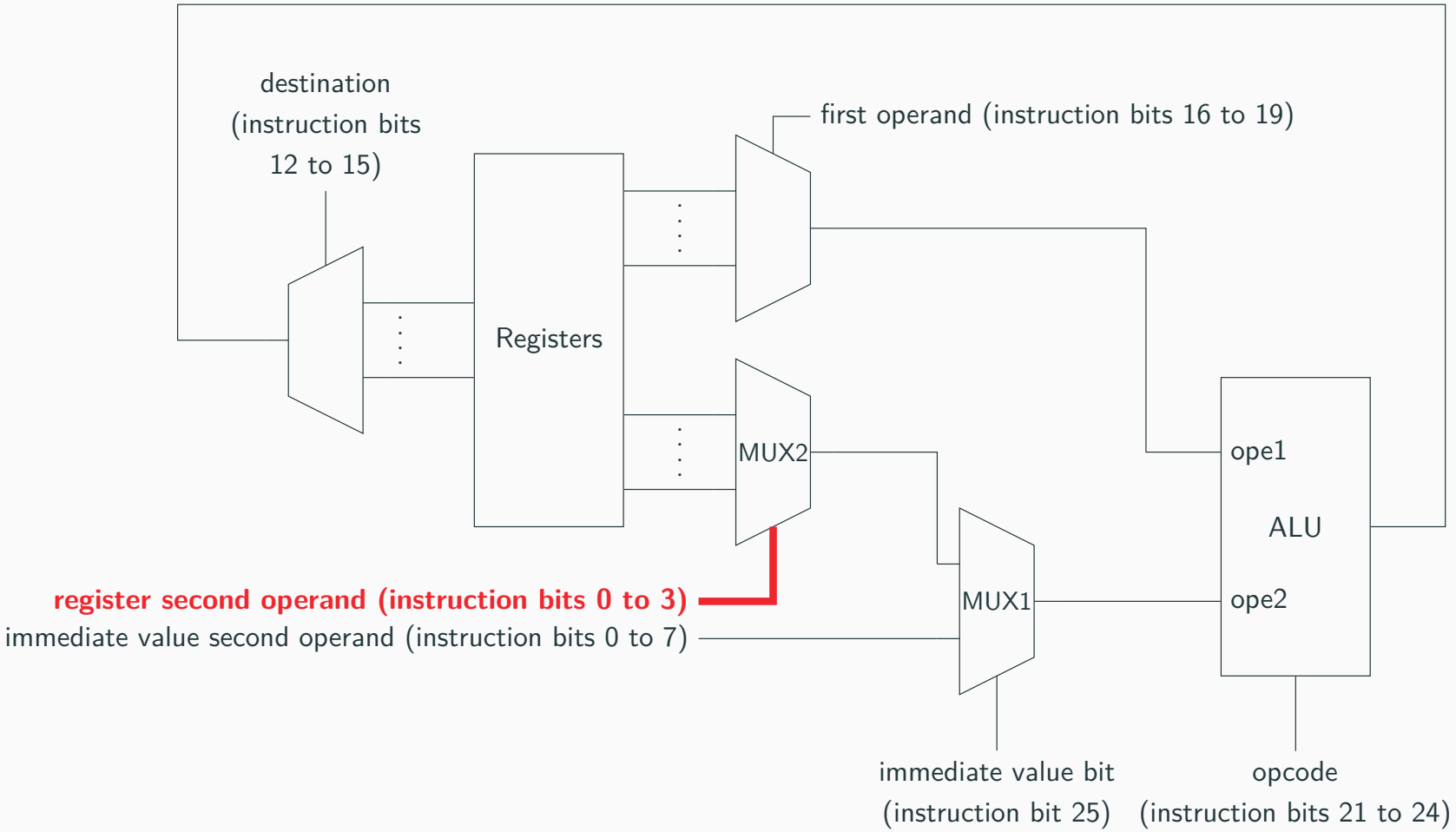
If immediate value bit (25) is set to 1



Characterization - BCM2837



Characterization - BCM2837



Characterization - BCM2837

Immediate value test code

```
          mov r3, #255
          cmp r3, #255
          bne fault
          b nofault
fault:    mov r9, #170
          b end
nofault:  mov r9, #85
end:      nop
```

Characterization - BCM2837

Immediate value test code

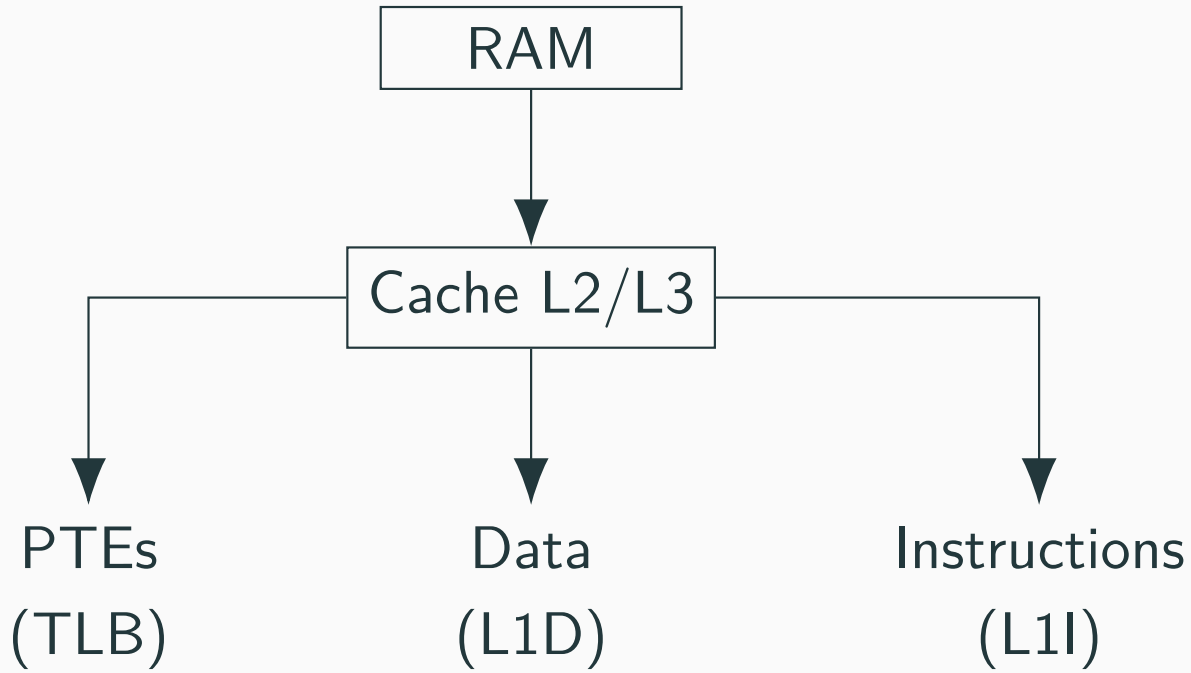
```
          mov r3, #255
          cmp r3, #255
          bne fault
          b nofault
fault:    mov r9, #170
          b end
nofault:  mov r9, #85
end:      nop
```

Results

Fault	r9 = 170	r9 = 0xfffcb924	Unknown
Rate	94%	4%	2%

Characterization - BCM2837

Memory subsystem pathing



Characterization - BCM2837

Memory test code

```
str r8, [r9] // Several
ldr r8, [r9] //      times
```

Initialization

- memory page allocated (4 kB)
- registered initialized to address in the page

Characterization - BCM2837

Memory test code

```
str r8, [r9] // Several
ldr r8, [r9] //     times
```

Initialization

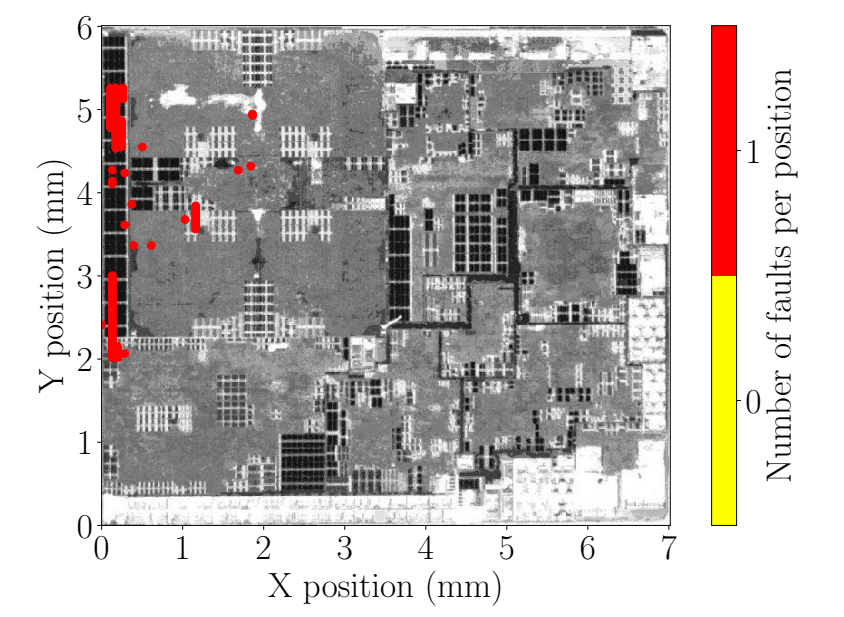
- memory page allocated (4 kB)
- registered initialized to address in the page

Results

- `ldr r8, [r9]` → `ldr r8, [PC]` (25 %)
- `ldr r8, [r9]` → `mov r8, r2` (74.4 %)
- no fault on fetched data

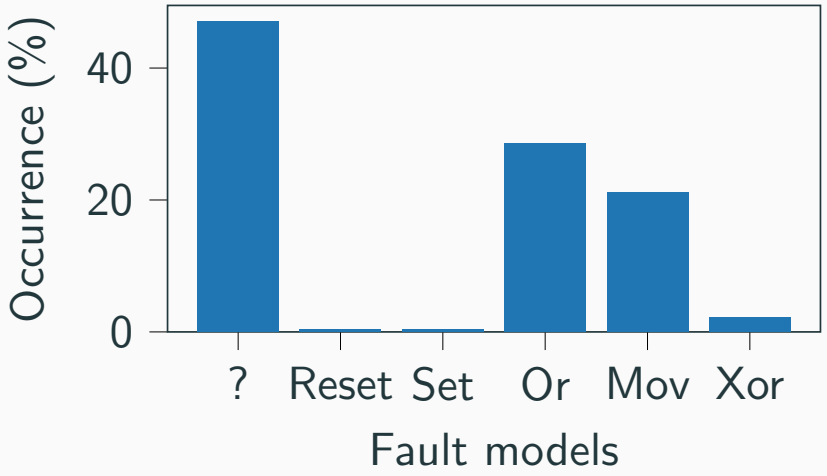
Characterization - BCM2711b0

```
orr r5,r5
```



Spots leading to faults

Fault models

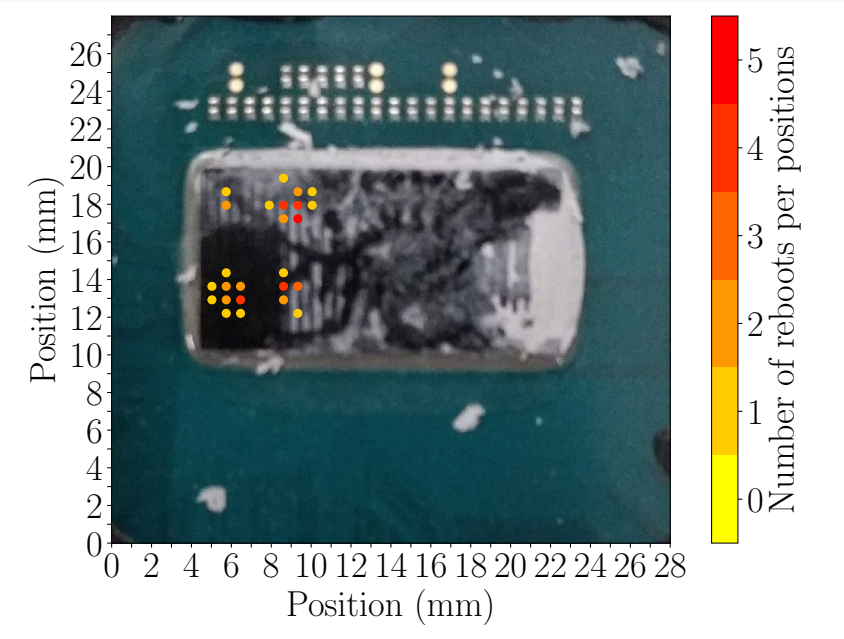


- Instruction corruption
  - Operands corruption
  - Opcode corruption

We mainly target the cache

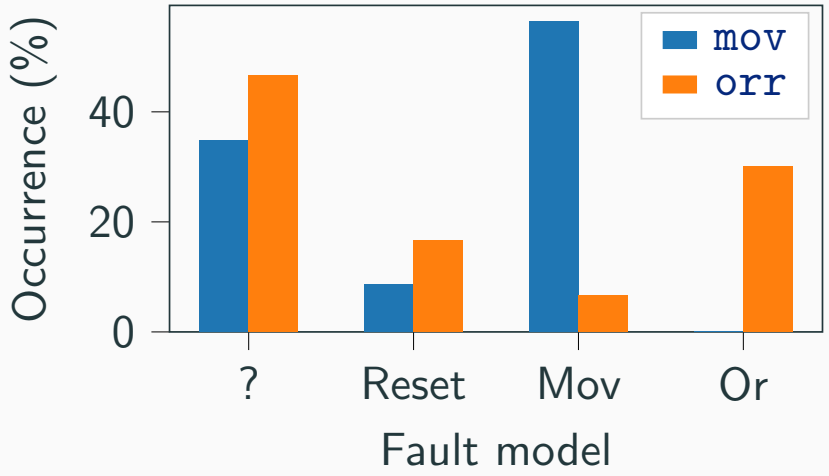
Characterization - Intel Core i3-6100T

`orr r5,r5`



Spots leading to reboots

Fault models



- Instruction corruption
  - Opcode corruption

Characterization - Conclusion

- Classical injection mediums (laser, EMFIs):
  - are efficient on complex CPUs,
  - can be characterized and are understandable

Characterization - Conclusion

- Classical injection mediums (laser, EMFIs):
  - are efficient on complex CPUs,
  - can be characterized and are understandable
- Different injection mediums have shown the similar fault models on different architecture (ARM, x86) and targets:
  - I suppose that there is an **underlying common mechanism** sensitive to perturbation,
  - this mechanism is faulted in the same way independently of the target and medium,
  - the **instruction cache** was identified as faulted on the BCM2837

Conclusion

- modern SoCs are more and more used for sensitive operations
- these SoCs are sensitive to faults
- evaluating modern SoCs is totally new
- security evaluation authorities (JHAS, ANSSI, *etc*) are looking forward modern SoCs evaluation (*cf.* JHAS SoC subgroup)

Conclusion

In my thesis, I provide:

- a method to characterize faults on SoCs CPUs,
- a fault analyzer software,
- perturbation experiment results,
- a fault model on various targets (ARM and x86),
- attacks on AES on Linux,
- an attack path on `sudo`,
- a Linux software analyzer,
- a fault simulator adapted to Linux softwares

Identified research topics and future works

- improvement of the fault analyzer, the software analyzer and the fault simulator
- development of a pattern based trigger generator,
- micro-architectural characterization on remaining targets,
- development of countermeasures to protect the instructions,
- analyzing the Linux kernel and security programs against faults,
- confirming the `sudo` attack path with an actual attack

Contributions i

Publications

- Thomas Trouchkine, Guillaume Bouffard, and Jessy Clediere. "Fault Injection Characterization on modern CPUs – From the ISA to the Micro-Architecture". In: *WISTP 2019, Paris, France*. 2019
- Thomas Trouchkine, Guillaume Bouffard, and Jessy Clediere. "EM Injection Vs. Modern CPU - Fault Characterization And AES Differential Fault Analysis". In: *Comptatibilité électromagnétique France 2020*. 2020

Contributions ii

- Thomas Trouchkine et al. “Electromagnetic Fault Injection against a complex CPU, toward a new micro-architectural fault models”. In: *Journal of Cryptographic Engineering (JCEN)*. 2021

Workshops

- Thomas Trouchkine. “Problems and state of the art of faults injection on Systems on Chip”. In: *PHISIC 2018*. 2018
- Thomas Trouchkine. “EM fault injection on Raspberry Pi 3 and ZedBoard”. In: *GDR SoC2 research group*. 2018

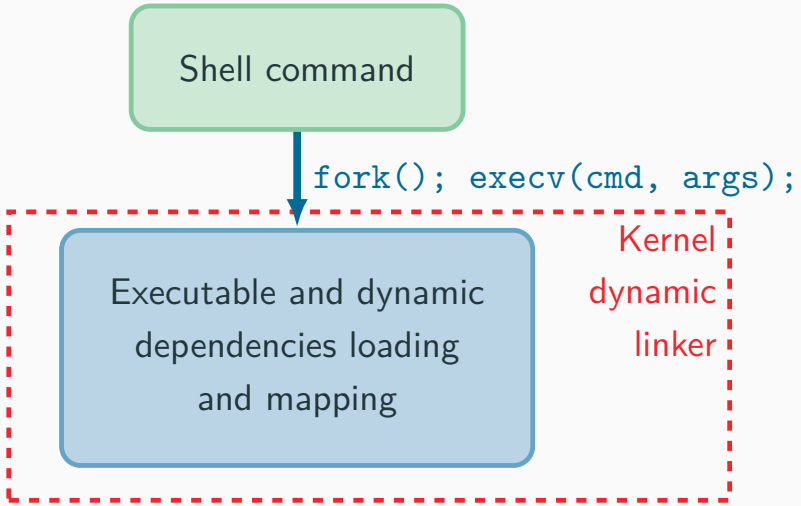
Contributions iii

- Thomas Trouchkine et al. “Do not trust modern System on Chip”. In: *Journée thématique sur les attaques par injection de fautes (JAIF)*. 2019
- Thomas Trouchkine et al. “Do not trust modern System-on-Chips”. In: *PHISIC 2019*. 2019
- Guillaume Bouffard et al. “Radically secure computing”. In: *SILM seminar 2020*. 2020
- Thomas Trouchkine, Guillaume Bouffard, and Jessy Clediere. “Perturbation attack on modern CPUs, from the fault model to the exploitation”. In: *Journée thématique sur les attaques par injection de fautes (JAIF)*. 2020

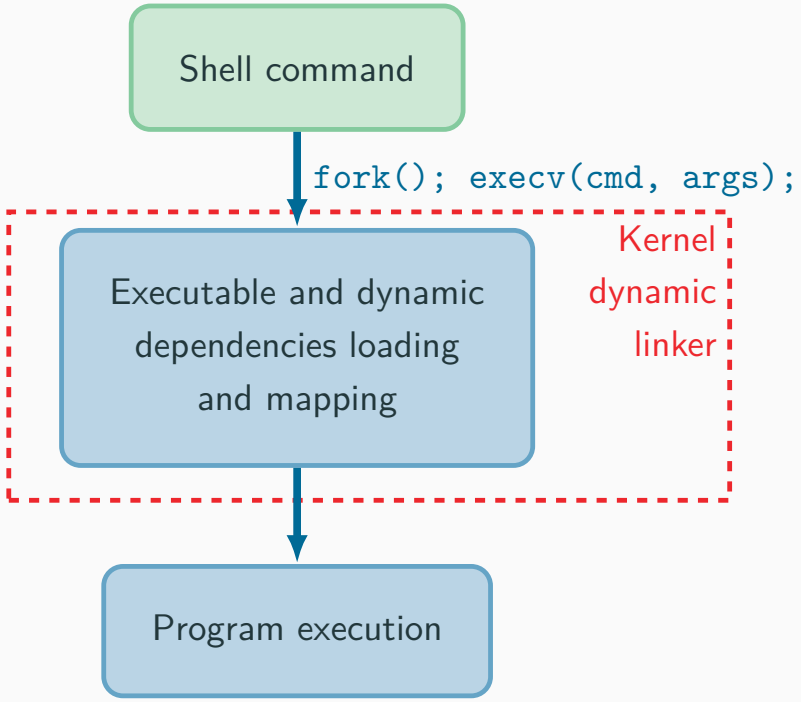
Questions ?

Appendice - Linux program life

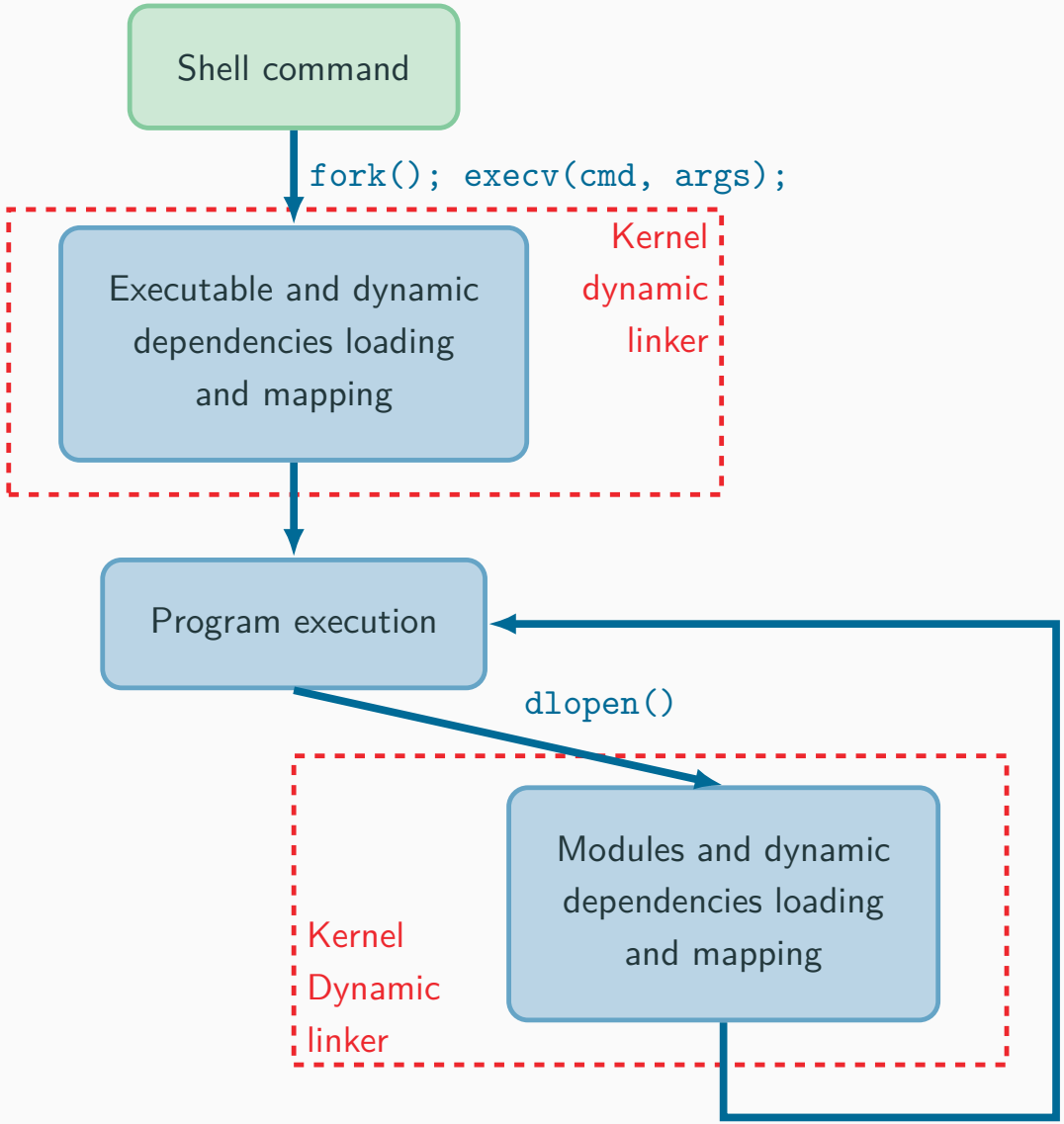
Appendice - Linux program life



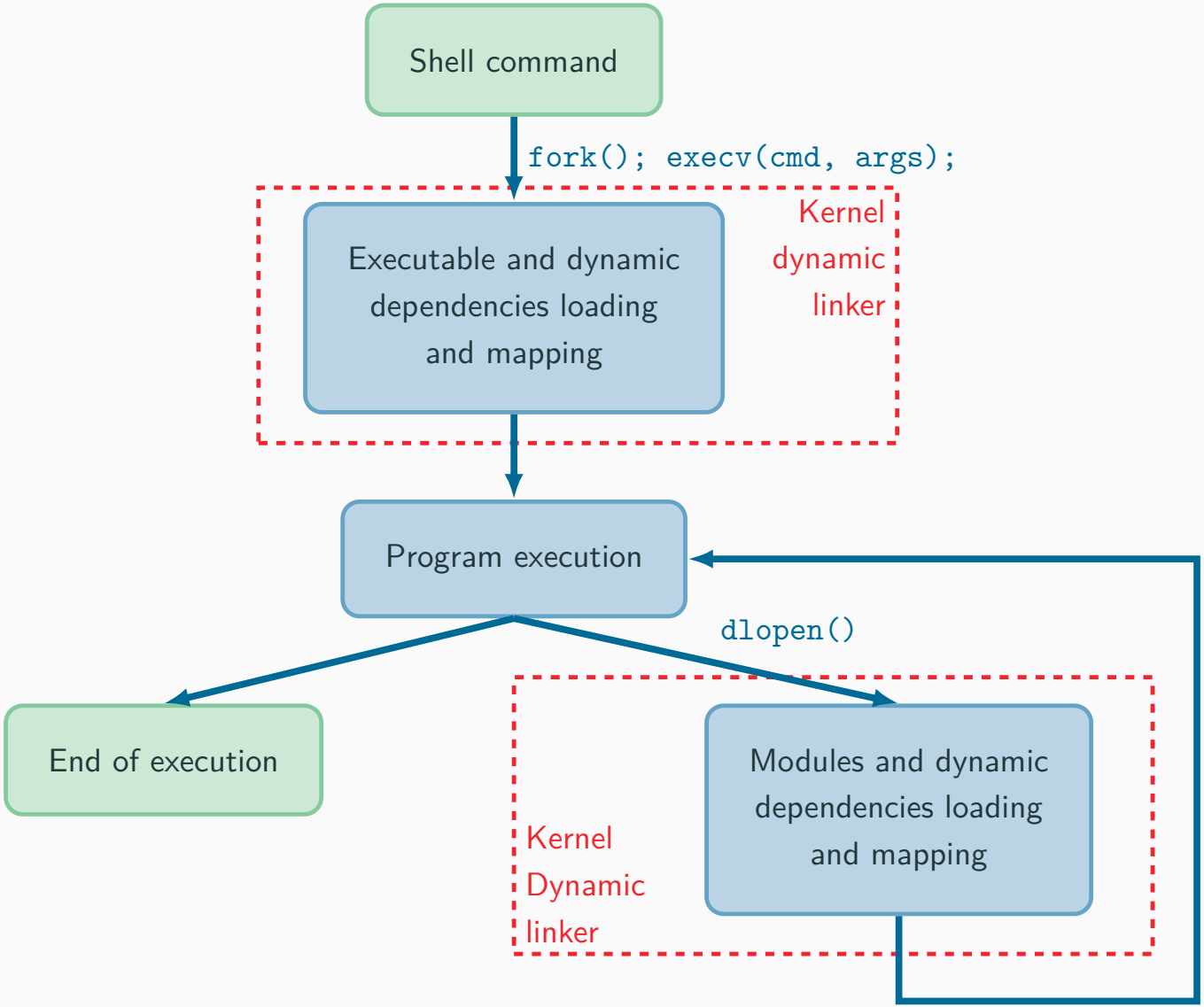
Appendice - Linux program life



Appendice - Linux program life



Appendice - Linux program life



Determining the number of faulted instructions

Test code

```
mov r0,r0
mov r1,r1
mov r2,r2
mov r3,r3
mov r4,r4
mov r5,r5
mov r6,r6
mov r7,r7
mov r8,r8
mov r9,r9
```

Result

On average:

- 1.45 faulted instructions

Appendice - BCM2837 MMU fault

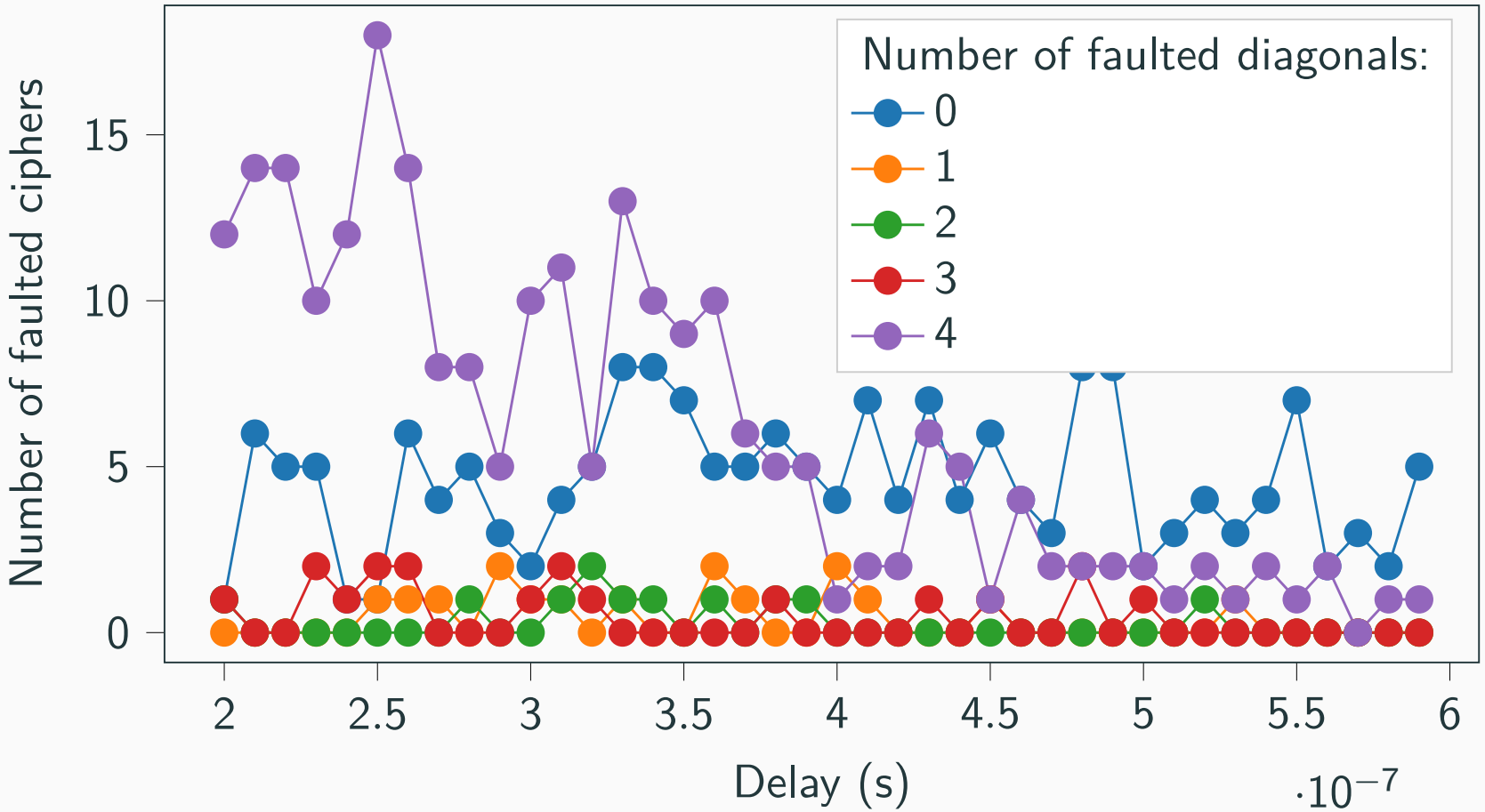
```
VA       -> PA
0x0      -> 0x0       0x80000 -> 0x80000
0x10000  -> 0x10000   0x90000 -> 0x90000
0x20000  -> 0x20000   0xa0000 -> 0xa0000
0x30000  -> 0x30000   0xb0000 -> 0xb0000
0x40000  -> 0x40000   0xc0000 -> 0xc0000
0x50000  -> 0x50000   0xd0000 -> 0xd0000
0x60000  -> 0x60000   0xe0000 -> 0xe0000
0x70000  -> 0x70000   0xf0000 -> 0xf0000
```

Appendice - BCM2837 MMU fault

```
VA       -> PA
0x0      -> 0x0       0x80000 -> 0x0
0x10000  -> 0x10000   0x90000 -> 0x0
0x20000  -> 0x20000   0xa0000 -> 0x0
0x30000  -> 0x30000   0xb0000 -> 0x0
0x40000  -> 0x40000   0xc0000 -> 0x80000
0x50000  -> 0x50000   0xd0000 -> 0x90000
0x60000  -> 0x60000   0xe0000 -> 0xa0000
0x70000  -> 0x70000   0xf0000 -> 0xb0000
```

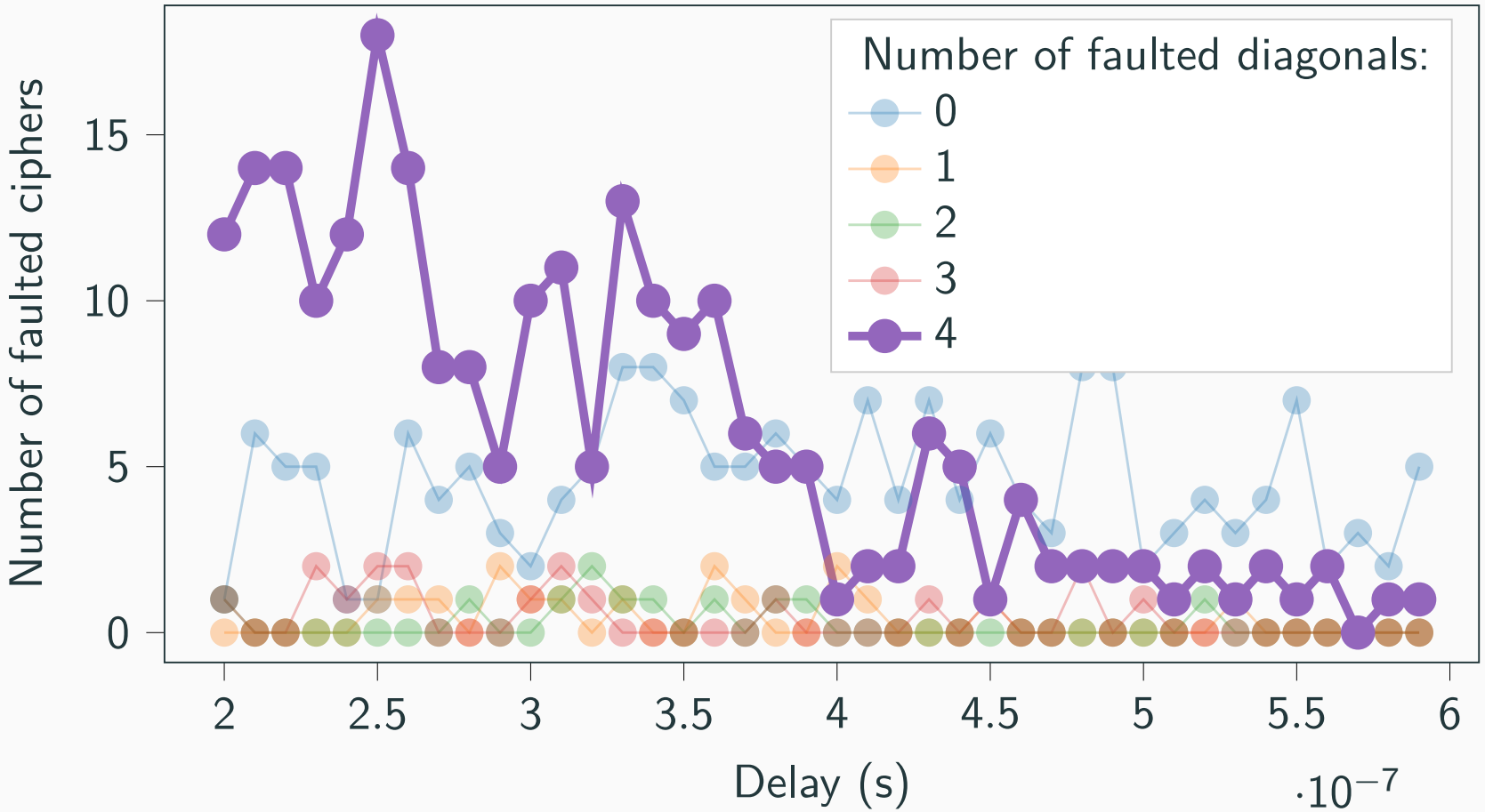
Appendice - OpenSSL AES

Number of faulted ciphers with a specific number of faulted diagonals regarding the delay of injection

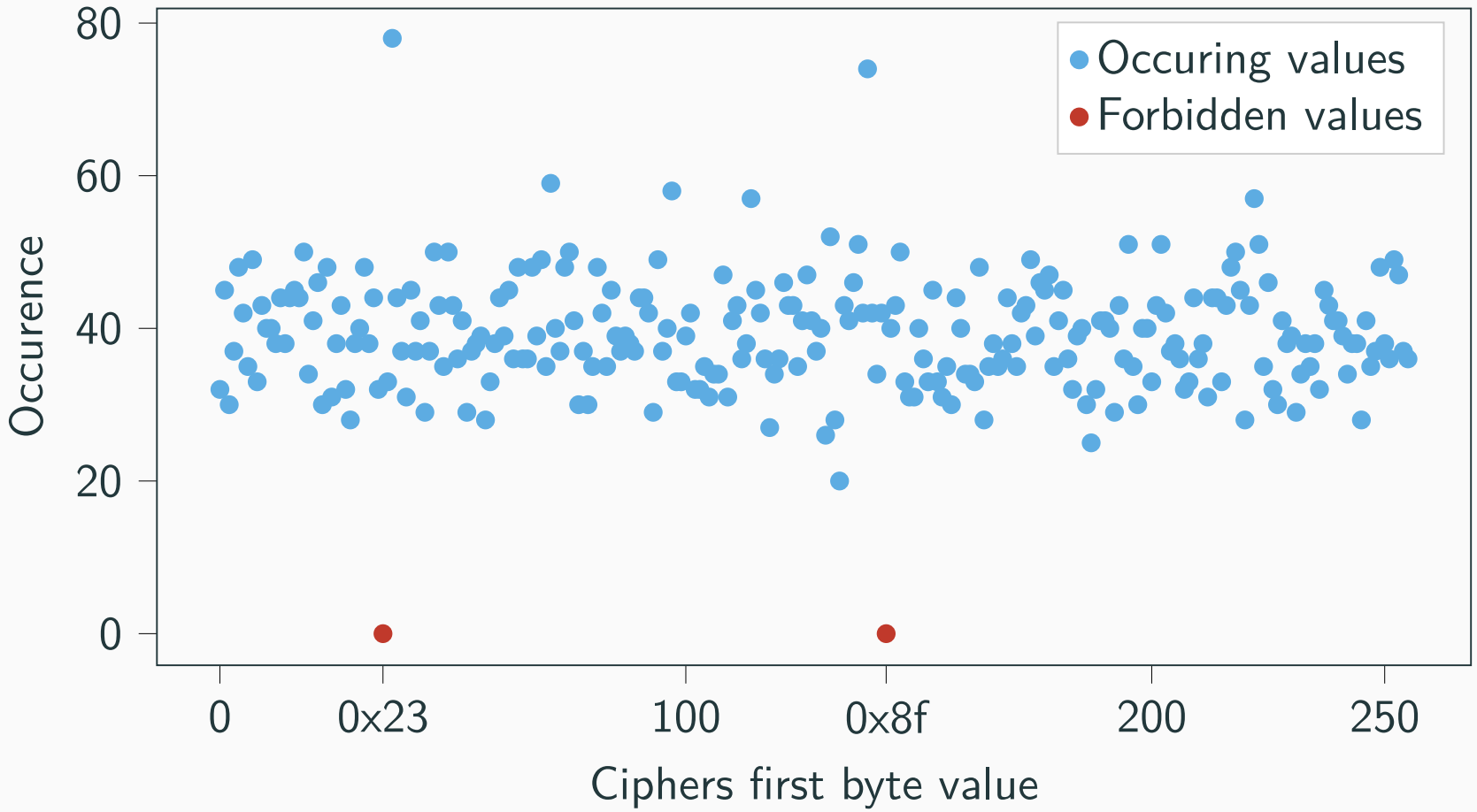


Appendice - OpenSSL AES

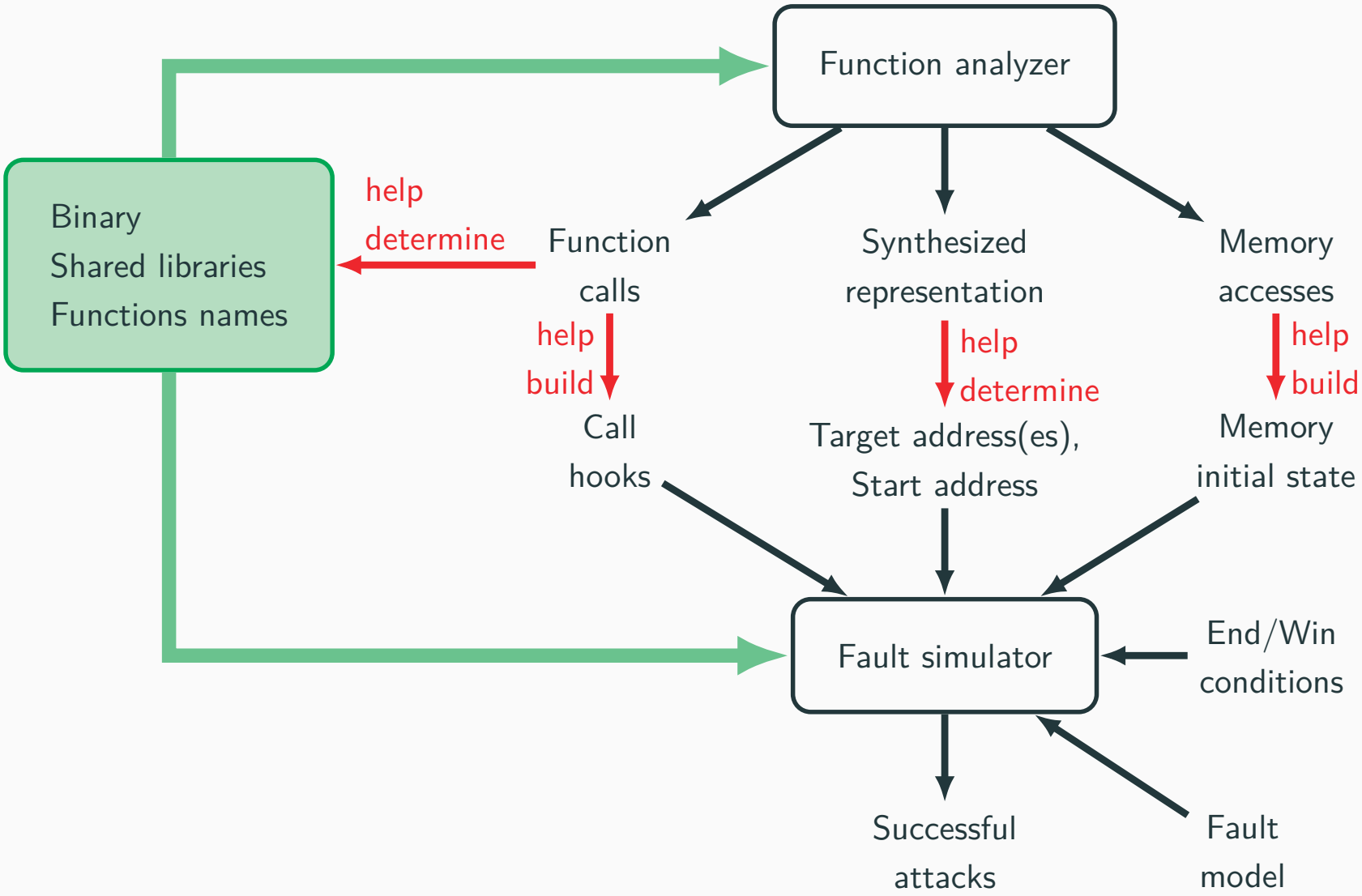
Number of faulted ciphers with a specific number of faulted diagonals regarding the delay of injection



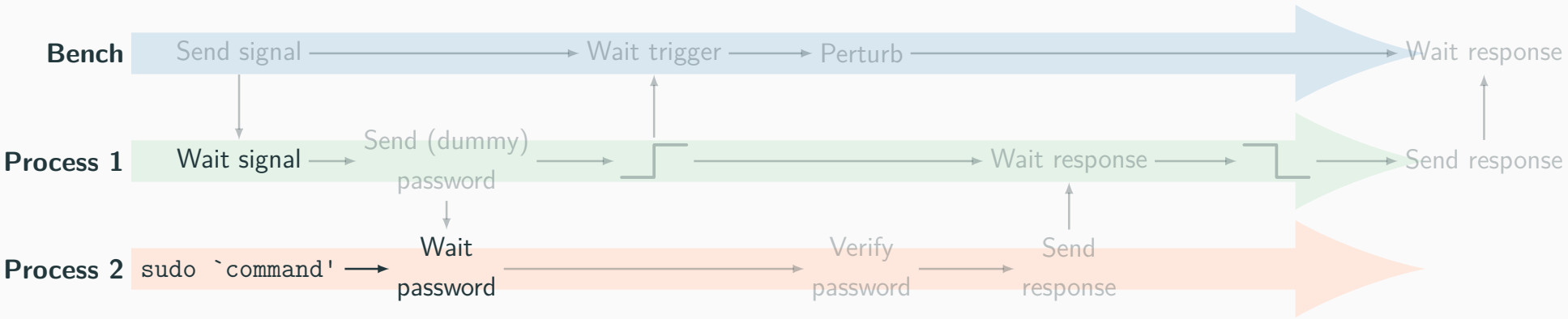
Appendice - AES PFA



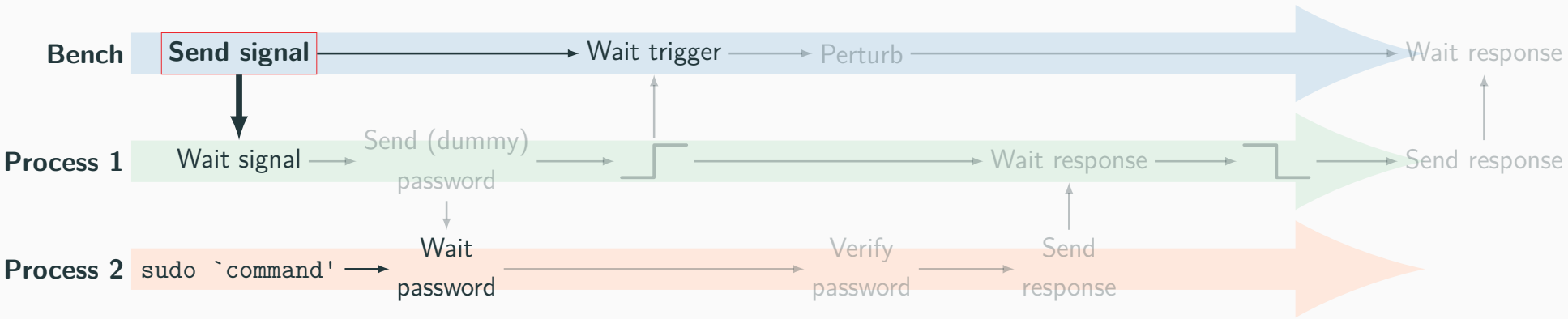
Appendice - Analyzer and simulator



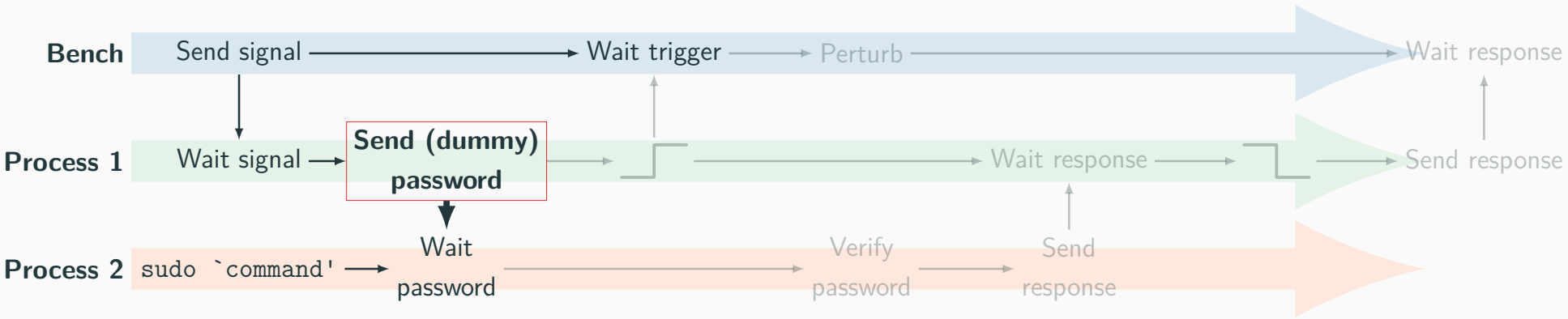
Appendice - `sudo` Forced authentication - User program



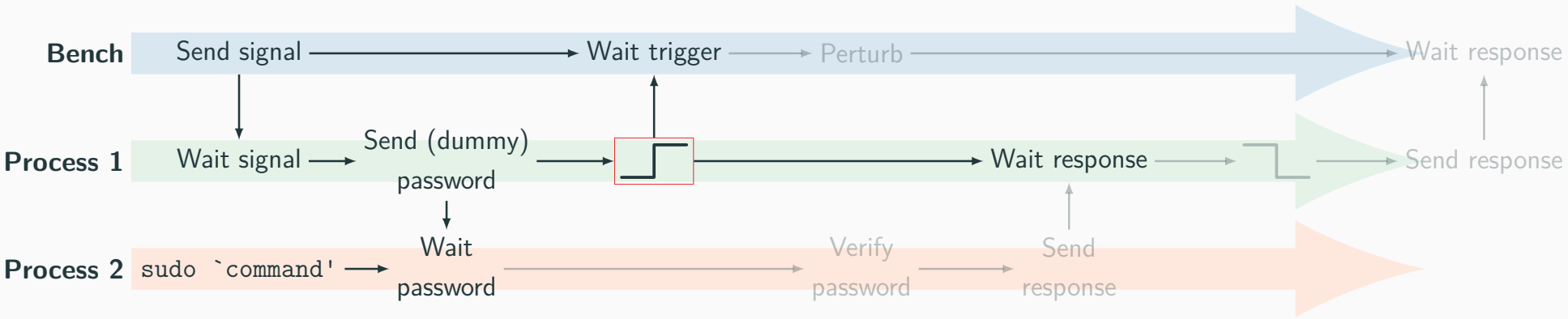
Appendice - `sudo` Forced authentication - User program



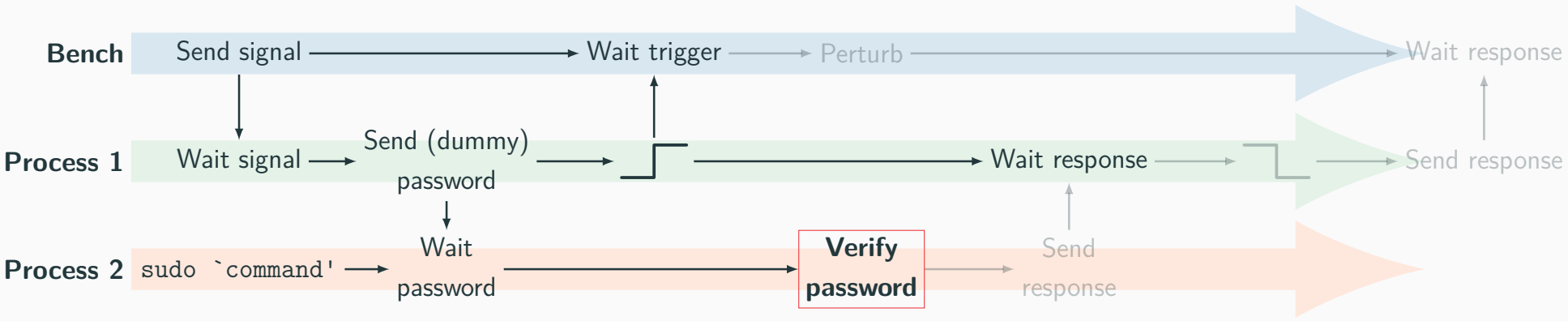
Appendice - `sudo` Forced authentication - User program



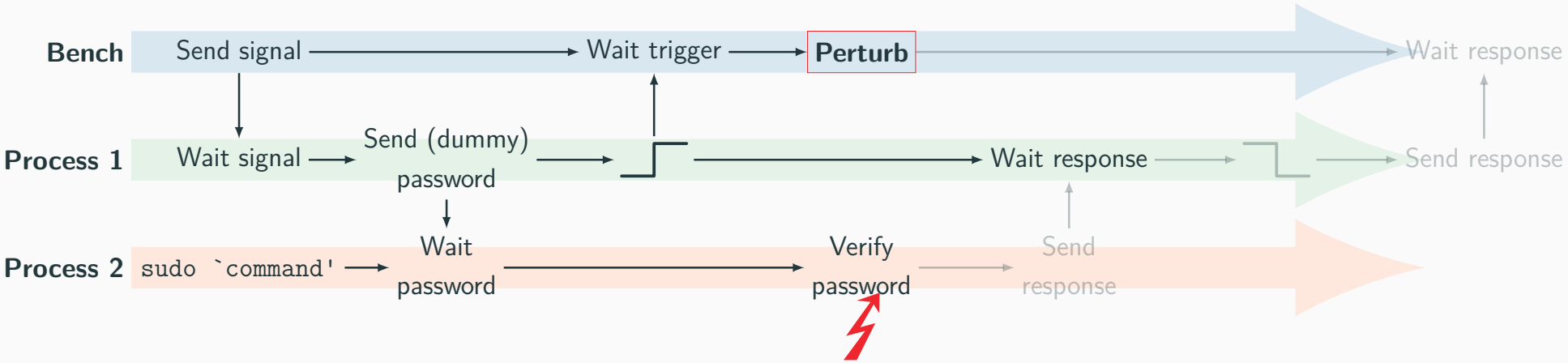
Appendice - `sudo` Forced authentication - User program



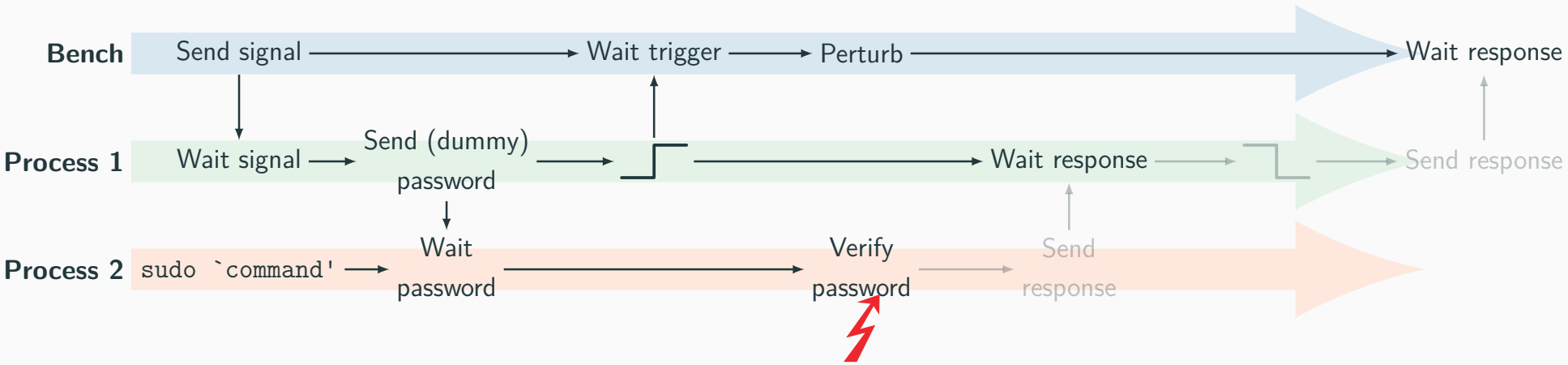
Appendice - `sudo` Forced authentication - User program



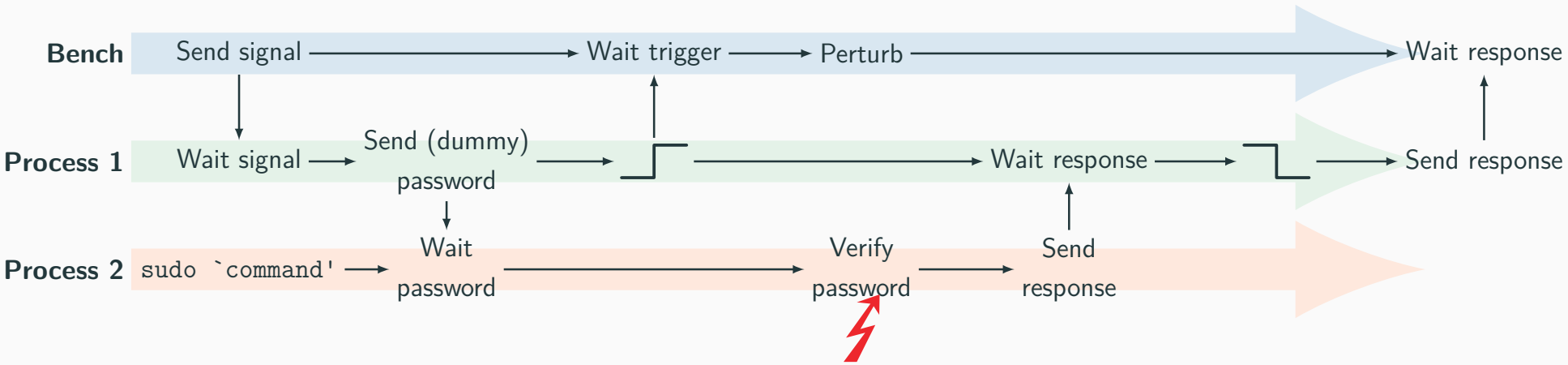
Appendice - `sudo` Forced authentication - User program



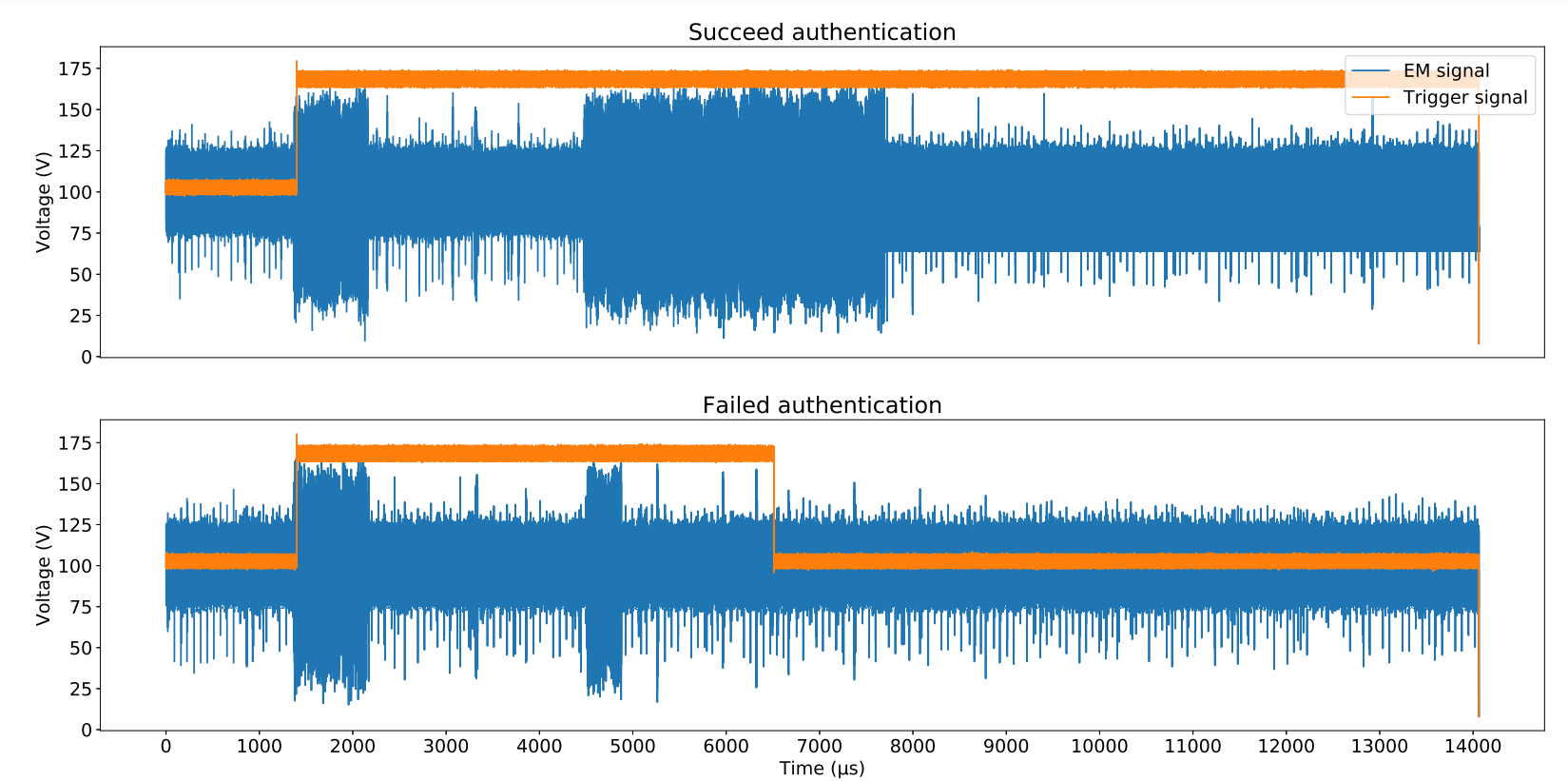
Appendice - `sudo` Forced authentication - User program



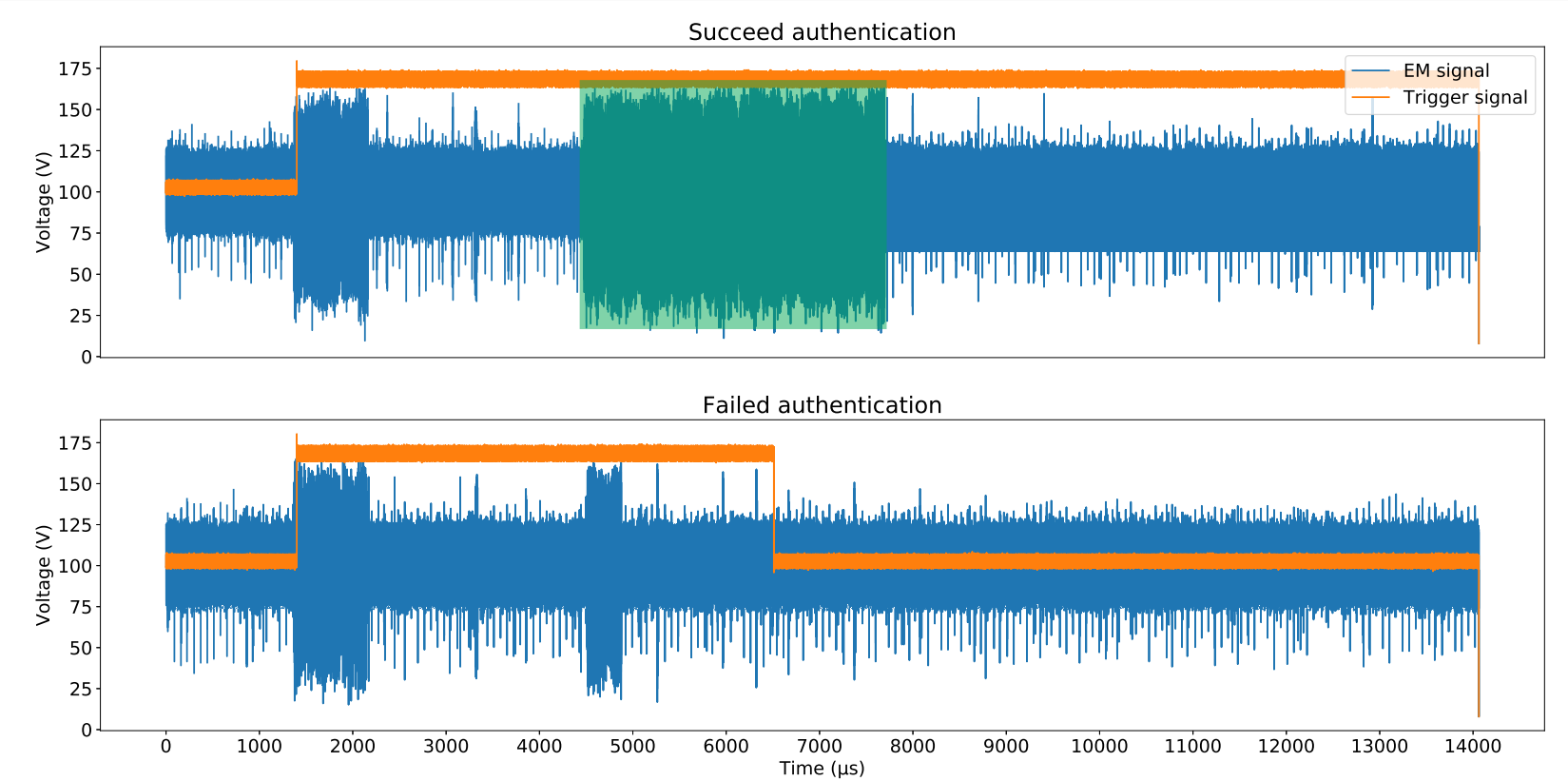
Appendice - `sudo` Forced authentication - User program



Appendice - `sudo` Forced authentication - EM analysis



Appendice - `sudo` Forced authentication - EM analysis



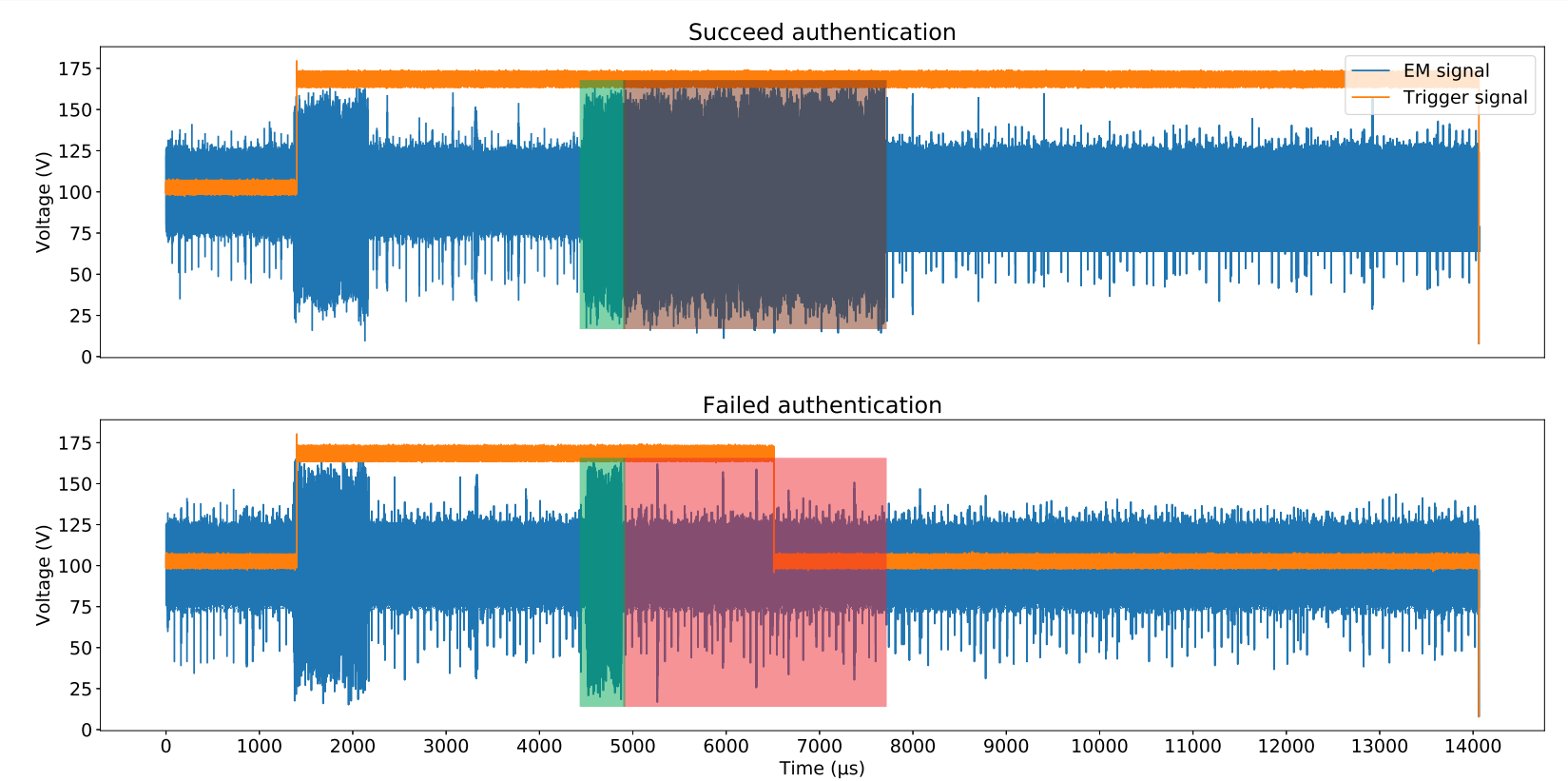

```
$6$hH.15uU5laaxuXHY$wtSOcCKWmY1JmyY2CWlVs/8ixyON36ZxQV2RpMJkITzqkIM18lyXNMICoYNIVDeUVXqHOFs39Onl6Lw8m5ArZ0
```

Appendice - `sudo` Forced authentication - EM analysis


```
$6$hH.15uU5laaxuXHY$wtSOcCKWmY1JmyY2CWlVs/8ixyON36ZxQV2RpMJkITzqkIM18lyXNMICoYNIVDeUVXqHOFs39Onl6Lw8m5ArZ0

$6$hH.15uU5laaxuXHY$4b7acwY3u21L9Wd8TxQeCIkpmasNufgDzIrScjXreP8oFQA4c.0nZmcYJB2zf5p6rDvPdBC0Fo6JWvquBKaVc.
```

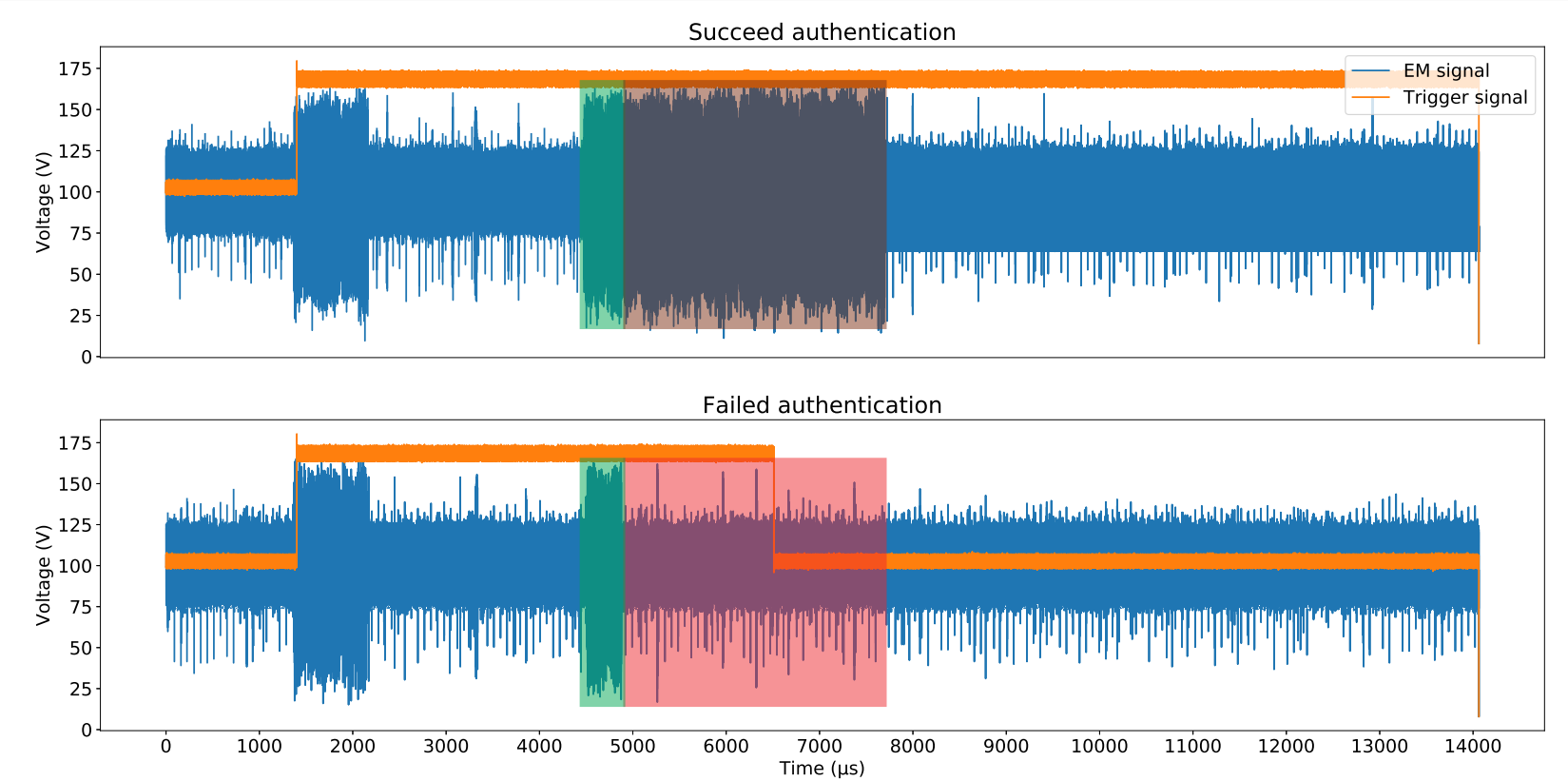
Appendice - `sudo` Forced authentication - EM analysis




```
$6$hH.15uU5laaxuXHY$wtSOcCKWmY1JmyY2CWlVs/8ixyON36ZxQV2RpMJkITzqkIM18lyXNMICoYNIVDeUVXqHOFs390nl6Lw8m5ArZ0

$6$hH.15uU5laaxuXHY$4b7acwY3u21L9Wd8TxQeCIkpmasNufgDzIrScjXreP8oFQA4c.0nZmcYJB2zf5p6rDvPdBC0Fo6JWvquBKaVc.
```

Appendice - `sudo` Forced authentication - EM analysis




```
$6$hH.15uU5laaxuXHY$wtSOcCKWmY1JmyY2CWlVs/8ixyON36ZxQV2RpMJkITzqkIM18lyXNMICoYNIVDeUVXqHOFs390nl6Lw8m5ArZ0

$6$hH.15uU5laaxuXHY$4b7acwY3u21L9Wd8TxQeCIkpmasNufgDzIrScjXreP8oFQA4c.0nZmcYJB2zf5p6rDvPdBC0Fo6JWvquBKaVc.
```

`strncmp()` in `verify_pwd_hash()` in `pam_unix.so`

PoC of forced authentication done in [Gai+20]

References

[19] *Characterization of EM faults on ATmega328p*. Zenodo, July 2019. DOI: 10.5281/zenodo.2647298. URL: https://doi.org/10.5281/zenodo.2647298.

Bibliography ii

[BFP19] Claudio Bozzato, Riccardo Focardi, and Francesco Palmarini. “Shaping the Glitch: Optimizing Voltage Fault Injection Attacks”. In: *IACR Trans. Cryptogr. Hardw. Embed. Syst.* 2019.2 (2019), pp. 199–224. DOI: 10.13154/tches.v2019.i2.199-224. URL: https://doi.org/10.13154/tches.v2019.i2.199-224.

Bibliography iii

[BGV11] Josep Balasch, Benedikt Gierlichs, and Ingrid Verbauwhede. “An In-depth and Black-box Characterization of the Effects of Clock Glitches on 8-bit MCUs”. In: *2011 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2011, Tokyo, Japan, September 29, 2011*. Ed. by Luca Breveglieri et al. IEEE Computer Society, 2011, pp. 105–114. DOI: 10.1109/FDTC.2011.9. URL: https://doi.org/10.1109/FDTC.2011.9.

Bibliography iv

[BJ15] Jakub Breier and Dirmanto Jap. “Testing Feasibility of Back-Side Laser Fault Injection on a Microcontroller”. In: *Proceedings of the 10th Workshop on Embedded Systems Security, WESS 2015, Amsterdam, The Netherlands, October 8, 2015*. Ed. by Stavros A. Koubias and Thilo Sauter. ACM, 2015, p. 5. DOI: 10.1145/2818362.2818367. URL: https://doi.org/10.1145/2818362.2818367.

[Bou+20] Guillaume Bouffard et al. “Radically secure computing”. In: *SILM seminar 2020*. 2020.

[Buk+18] Sébanjila Kevin Bukasa et al. “Let’s shock our IoT’s heart: ARMv7-M under (fault) attacks”. In: *Proceedings of the 13th International Conference on Availability, Reliability and Security, ARES 2018, Hamburg, Germany, August 27-30, 2018*. Ed. by Sebastian Doerr et al. ACM, 2018, 33:1–33:6. DOI: 10.1145/3230833.3230842. URL: https://doi.org/10.1145/3230833.3230842.

[Col+19] Brice Colombier et al. “Laser-induced Single-bit Faults in Flash Memory: Instructions Corruption on a 32-bit Microcontroller”. In: *IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2019, McLean, VA, USA, May 5-10, 2019*. IEEE, 2019, pp. 1–10. DOI: 10.1109/HST.2019.8741030. URL: https://doi.org/10.1109/HST.2019.8741030.

Bibliography vii

[DLM19] Mathieu Dumont, Mathieu Lisart, and Philippe Maurine. “Electromagnetic Fault Injection : How Faults Occur”. In: *2019 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2019, Atlanta, GA, USA, August 24, 2019*. IEEE, 2019, pp. 9–16. DOI: 10.1109/FDTC.2019.00010. URL: https://doi.org/10.1109/FDTC.2019.00010.

[Gai+20] Clément Gaine et al. “Electromagnetic Fault Injection as a New Forensic Approach for SoCs”. In: *IEEE WIFS 2020*. 2020.

[KH14] Thomas Korak and Michael Hoefler. “On the Effects of Clock and Power Supply Tampering on Two Microcontroller Platforms”. In: *2014 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2014, Busan, South Korea, September 23, 2014*. Ed. by Assia Tria and Dooho Choi. IEEE Computer Society, 2014, pp. 8–17. DOI: 10.1109/FDTC.2014.11. URL: https://doi.org/10.1109/FDTC.2014.11.

[Kum+18] Dilip S. V. Kumar et al. “An In-Depth and Black-Box Characterization of the Effects of Laser Pulses on ATmega328P”. In: *Smart Card Research and Advanced Applications, 17th International Conference, CARDIS 2018, Montpellier, France, November 12-14, 2018, Revised Selected Papers*. Ed. by Begül Bilgin and Jean-Bernard Fischer. Vol. 11389. Lecture Notes in Computer Science. Springer, 2018, pp. 156–170. DOI: 10.1007/978-3-030-15462-2_11. URL: https://doi.org/10.1007/978-3-030-15462-2%5C_11.

[MBB16] Fabien Majéric, Eric Bourbao, and Lilian Bossuet. "Electromagnetic security tests for SoC". In: *2016 IEEE International Conference on Electronics, Circuits and Systems, ICECS 2016, Monte Carlo, Monaco, December 11-14, 2016*. IEEE, 2016, pp. 265–268. DOI: 10.1109/ICECS.2016.7841183. URL: https://doi.org/10.1109/ICECS.2016.7841183.

[Men+19] Alexandre Menu et al. “Precise Spatio-Temporal Electromagnetic Fault Injections on Data Transfers”. In: *2019 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2019, Atlanta, GA, USA, August 24, 2019*. IEEE, 2019, pp. 1–8. DOI: 10.1109/FDTC.2019.00009. URL: https://doi.org/10.1109/FDTC.2019.00009.

[Men+20] Alexandre Menu et al. “Experimental Analysis of the Electromagnetic Instruction Skip Fault Model”. In: *15th Design & Technology of Integrated Systems in Nanoscale Era, DTIS 2020, Marrakech, Morocco, April 1-3, 2020*. IEEE, 2020, pp. 1–7. DOI: 10.1109/DTIS48698.2020.9081261. URL: https://doi.org/10.1109/DTIS48698.2020.9081261.

[Mor+14a] Nicolas Moro et al. “Electromagnetic fault injection: towards a fault model on a 32-bit microcontroller”. In: *CoRR* abs/1402.6421 (2014). arXiv: 1402.6421. URL: http://arxiv.org/abs/1402.6421.

[Mor+14b] Nicolas Moro et al. “Experimental evaluation of two software countermeasures against fault attacks”. In: *2014 IEEE International Symposium on Hardware-Oriented Security and Trust, HOST 2014, Arlington, VA, USA, May 6-7, 2014*. IEEE Computer Society, 2014, pp. 112–117. DOI: 10.1109/HST.2014.6855580. URL: https://doi.org/10.1109/HST.2014.6855580.

[OGM15] Sébastien Ordas, Ludovic Guillaume-Sage, and Philippe Maurine. “EM Injection: Fault Model and Locality”. In: *2015 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2015, Saint Malo, France, September 13, 2015*. Ed. by Naofumi Homma and Victor Lomné. IEEE Computer Society, 2015, pp. 3–13. DOI: 10.1109/FDTC.2015.9. URL: https://doi.org/10.1109/FDTC.2015.9.

[Pro+19] Julien Proy et al. “A First ISA-Level Characterization of EM Pulse Effects on Superscalar Microarchitectures: A Secure Software Perspective”. In: *Proceedings of the 14th International Conference on Availability, Reliability and Security, ARES 2019, Canterbury, UK, August 26-29, 2019*. ACM, 2019, 7:1–7:10. DOI: 10.1145/3339252.3339253. URL: https://doi.org/10.1145/3339252.3339253.

[Riv+15] Lionel Rivière et al. “High precision fault injections on the instruction cache of ARMv7-M architectures”. In: *IEEE International Symposium on Hardware Oriented Security and Trust, HOST 2015, Washington, DC, USA, 5-7 May, 2015*. IEEE Computer Society, 2015, pp. 62–67. DOI: 10.1109/HST.2015.7140238. URL: https://doi.org/10.1109/HST.2015.7140238.

[Sam+02] David Samyde et al. “On a New Way to Read Data from Memory”. In: *Proceedings of the First International IEEE Security in Storage Workshop, SISW 2002, Greenbelt, Maryland, USA, December 11, 2002*. IEEE Computer Society, 2002, pp. 65–69. DOI: 10.1109/SISW.2002.1183512. URL: https://doi.org/10.1109/SISW.2002.1183512.

[SHP09] Jörn-Marc Schmidt, Michael Hutter, and Thomas Plos. "Optical Fault Attacks on AES: A Threat in Violet". In: *Sixth International Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2009, Lausanne, Switzerland, 6 September 2009*. Ed. by Luca Breveglieri et al. IEEE Computer Society, 2009, pp. 13–22. DOI: 10.1109/FDTC.2009.37. URL: https://doi.org/10.1109/FDTC.2009.37.

[TBC19] Thomas Trouchkine, Guillaume Bouffard, and Jessy Clediere. “Fault Injection Characterization on modern CPUs – From the ISA to the Micro-Architecture”. In: *WISTP 2019, Paris, France*. 2019.

[TBC20a] Thomas Trouchkine, Guillaume Bouffard, and Jessy Clediere. “EM Injection Vs. Modern CPU - Fault Characterization And AES Differential Fault Analysis”. In: *Comptatibilité électromagnétique France 2020*. 2020.

Bibliography xx

[TBC20b] Thomas Trouchkine, Guillaume Bouffard, and Jessy Clediere. “Perturbation attack on modern CPUs, from the fault model to the exploitation”. In: *Journée thématique sur les attaques par injection de fautes (JAIF)*. 2020.

[Tro+19a] Thomas Trouchkine et al. “Do not trust modern System on Chip”. In: *Journée thématique sur les attaques par injection de fautes (JAIF)*. 2019.

[Tro+19b] Thomas Trouchkine et al. “Do not trust modern System-on-Chips”. In: *PHISIC 2019*. 2019.

[Tro+21] Thomas Trouchkine et al. “Electromagnetic Fault Injection against a complex CPU, toward a new micro-architectural fault models”. In: *Journal of Cryptographic Engineering (JCEN)*. 2021.

[Tro18a] Thomas Trouchkine. “EM fault injection on Raspberry Pi 3 and ZedBoard”. In: *GDR SoC2 research group*. 2018.

[Tro18b] Thomas Trouchkine. “Problems and state of the art of faults injection on Systems on Chip”. In: *PHISIC 2018*. 2018.

[TSW16] Niek Timmers, Albert Spruyt, and Marc Witteman. "Controlling PC on ARM Using Fault Injection". In: *2016 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2016, Santa Barbara, CA, USA, August 16, 2016*. IEEE Computer Society, 2016, pp. 25–35. DOI: 10.1109/FDTC.2016.18. URL: https://doi.org/10.1109/FDTC.2016.18.

[YGS15] Bilgiday Yuce, Nahid Farhady Ghalaty, and Patrick Schaumont. “Improving Fault Attacks on Embedded Software Using RISC Pipeline Characterization”. In: *2015 Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2015, Saint Malo, France, September 13, 2015*. Ed. by Naofumi Homma and Victor Lomné. IEEE Computer Society, 2015, pp. 97–108. DOI: 10.1109/FDTC.2015.16. URL: https://doi.org/10.1109/FDTC.2015.16.