Perturbation attack on modern CPUs, from the fault model to the exploitation

Thomas TROUCHKINE[1], Guillaume BOUFFARD[1,2], Jessy CLÉDIÈRE[3]

[1] National Cybersecurity Agency of France (ANSSI)

[2] Information Security Group, École Normale Supérieure

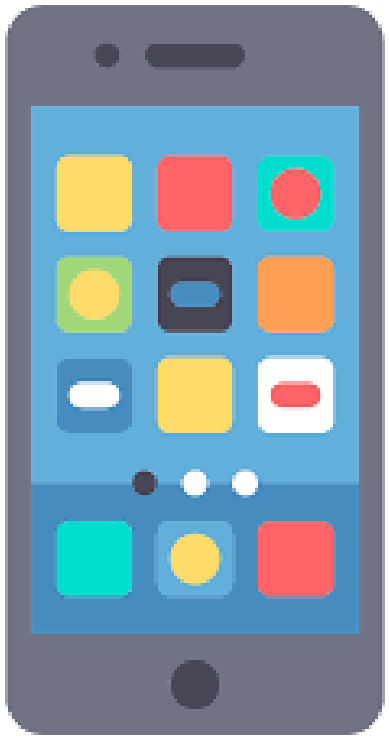
[3] CEA, LETI, MINATEC

September, 24th 2020

My thesis

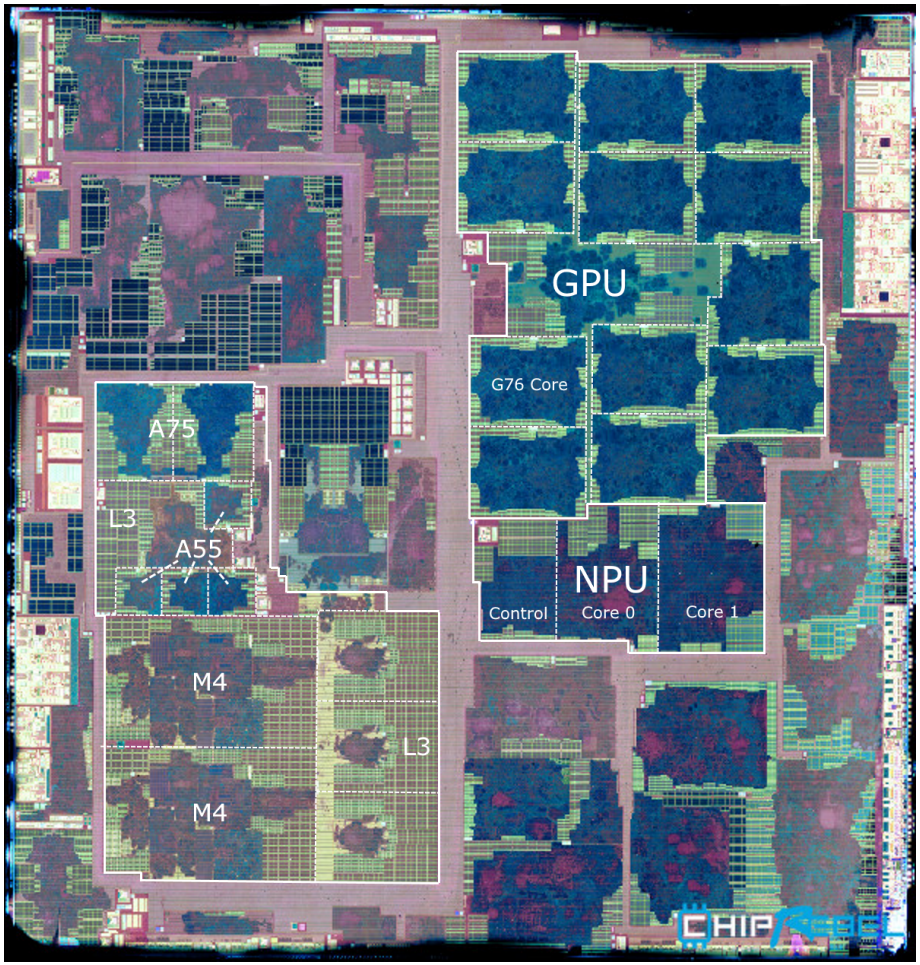
Evaluation of hardware attacks against System-On-Chip

- Jessy Clédière (Director)
- Guillaume Bouffard (Supervisor)



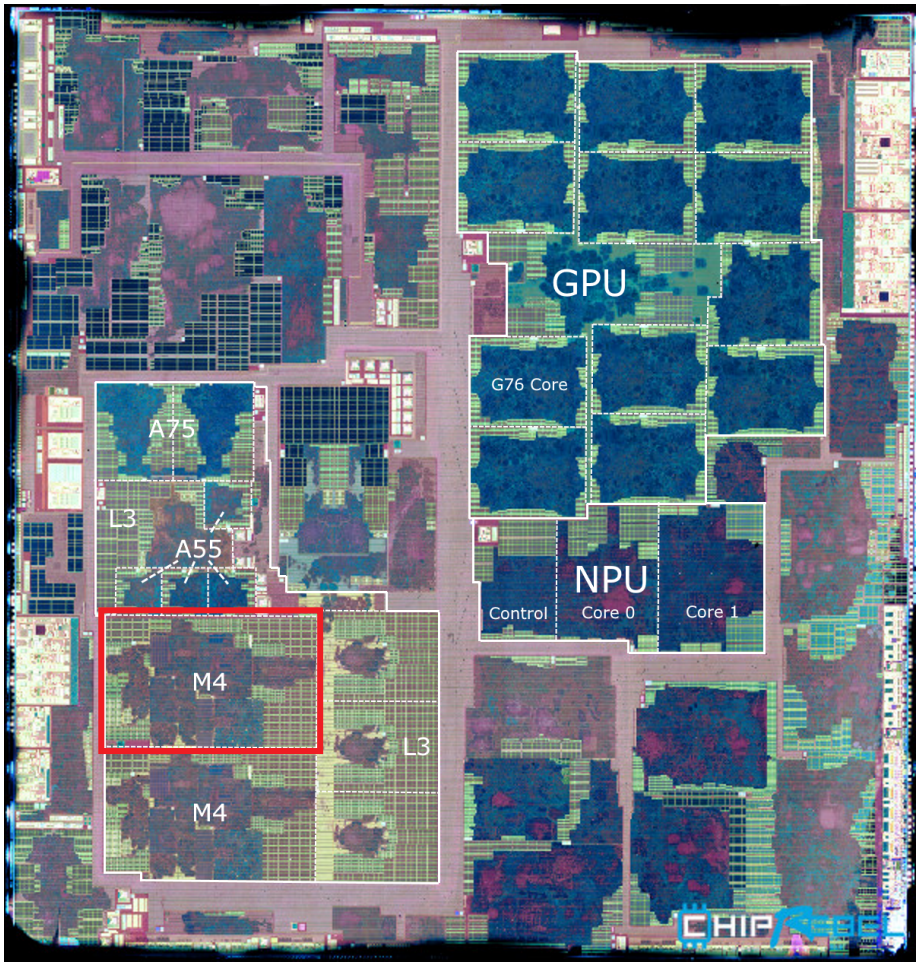
Focus on the perturbation of modern CPUs

Modern CPU ?

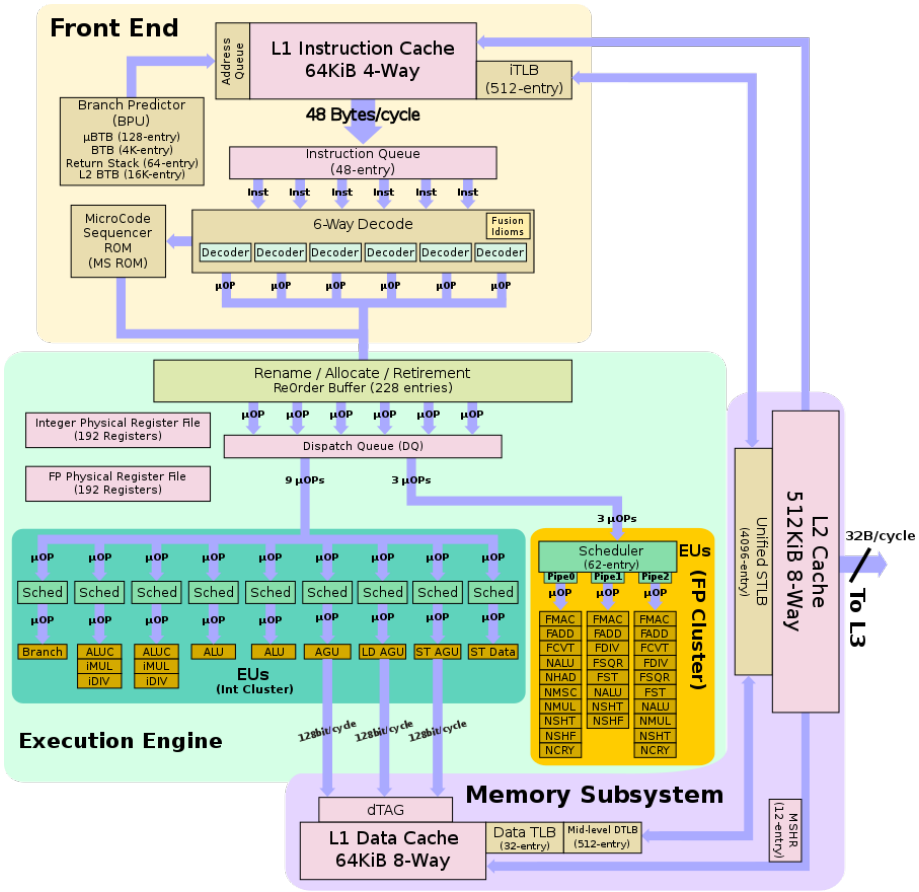


Exynos 9820 SoC (Samsung)

Modern CPU ?



Exynos 9820 SoC (Samsung)



Exynos M4 core

Targets

BCM2837
(Raspberry Pi 3 model B)

Intel Core i3-6100T
(Custom motherboard)

BMC2711b0
(Raspberry Pi 4)

Linux based OS (Raspbian Buster/Debian 9)

Fault injection mediums

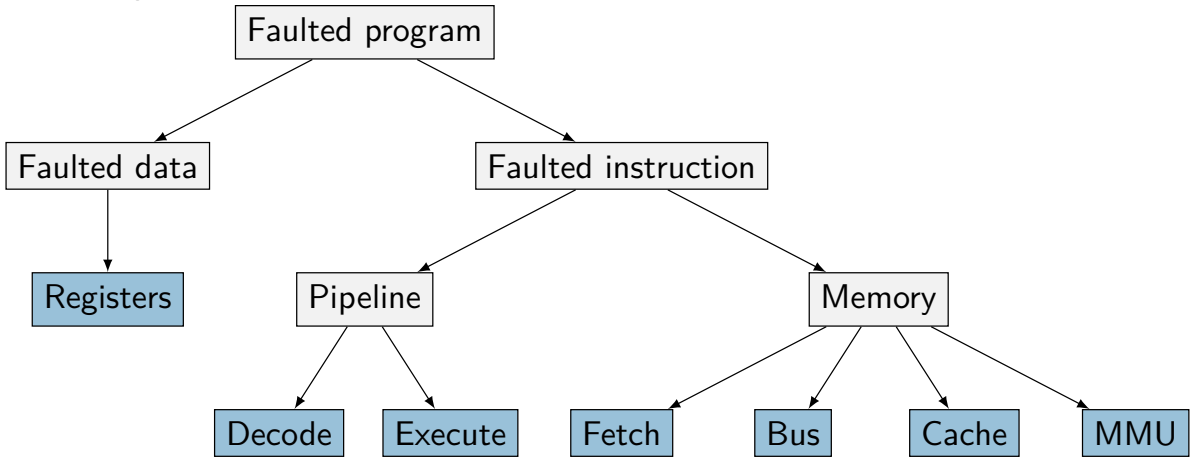
Device	EMFI	LFI
BCM2837 (RPi3)	✓	X
Intel Core i3	✓	X
BCM2711b0 (RPi4)	⏳	✓

Characterization method

Tested program

```
trigger_up();
orr r5, r5;
... # several times
orr r5, r5;
trigger_down();
```

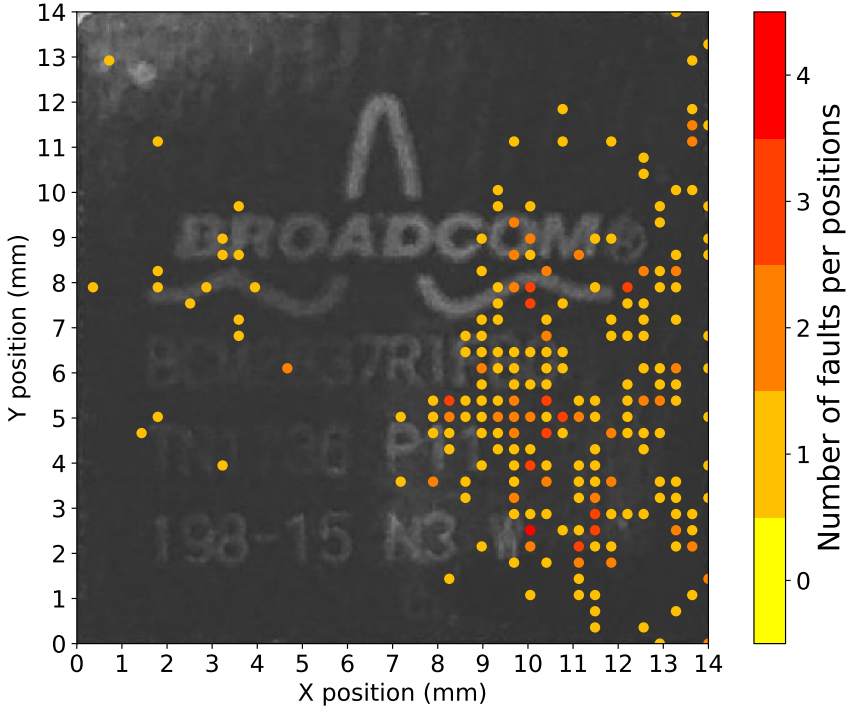
Analysis paths



Initial values

Register	Value
r0	0xfffe0001
r1	0xfffd0002
r2	0xfffb0004
r3	0xfff70008
r4	0xffef0010
r5	0xffdf0020
r6	0xffbf0040
r7	0xff7f0080
r8	0xfeff0100
r9	0xfdff0200

Characterization (BCM2837)



Positions of the probe over the chip leading to faults.

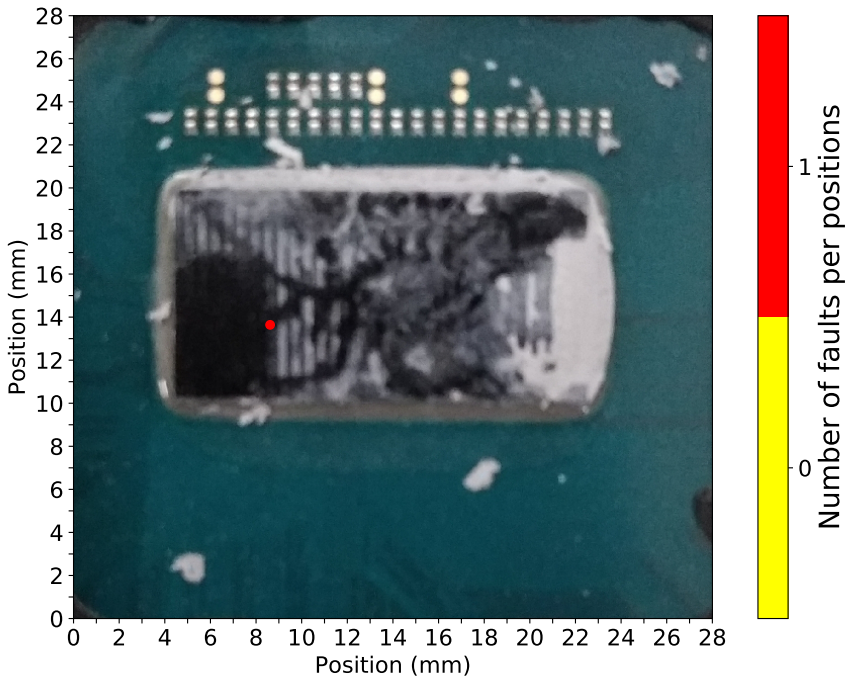
Fault models

- Register corruption
  - Bit reset
  - Instruction dependent value
- Instruction corruption
  - Operands corruption
  - Opcode corruption

Hypothesis

Fault targets cache

Characterization (Intel Core i3)



Positions of the probe over the die leading to faults.

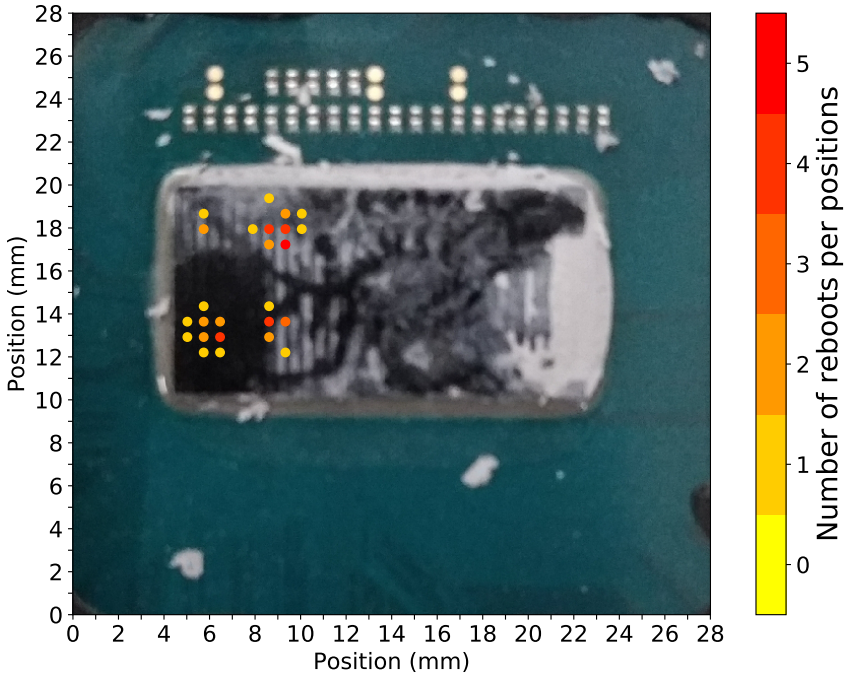
Fault models

- Register corruption
  - Bit reset
  - System values
- Instruction corruption
  - Operands corruption
  - Opcode corruption

Hypothesis

Fault targets cache

Characterization (Intel Core i3)



Positions of the probe over the die leading to reboots.

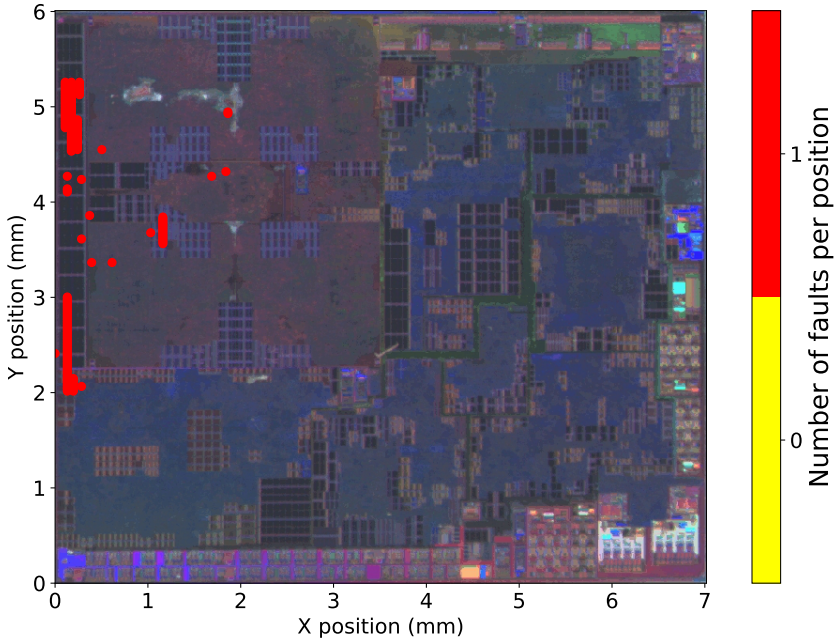
Fault models

- Register corruption
  - Bit reset
  - System values
- Instruction corruption
  - Operands corruption
  - Opcode corruption

Hypothesis

Fault targets cache

Characterization (BCM2711b0)



Positions of the laser spot over the die leading to faults.

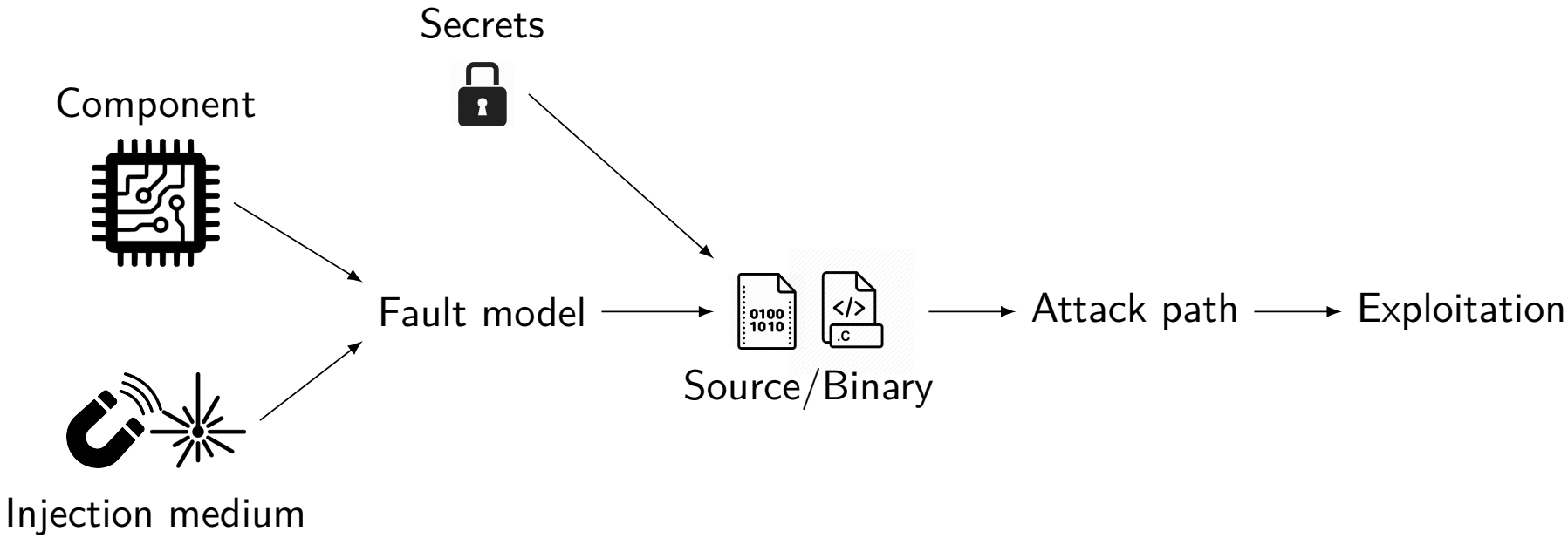
Fault models

- Register corruption
  - Bit set
  - Bit reset
- Instruction corruption
  - Operands corruption
  - Opcode corruption

Not an hypothesis

We mainly target the cache

Fault model exploitability



Characterization **Analysis** **Exploitation**

Exploitation

DFA on AES
(BCM2837)

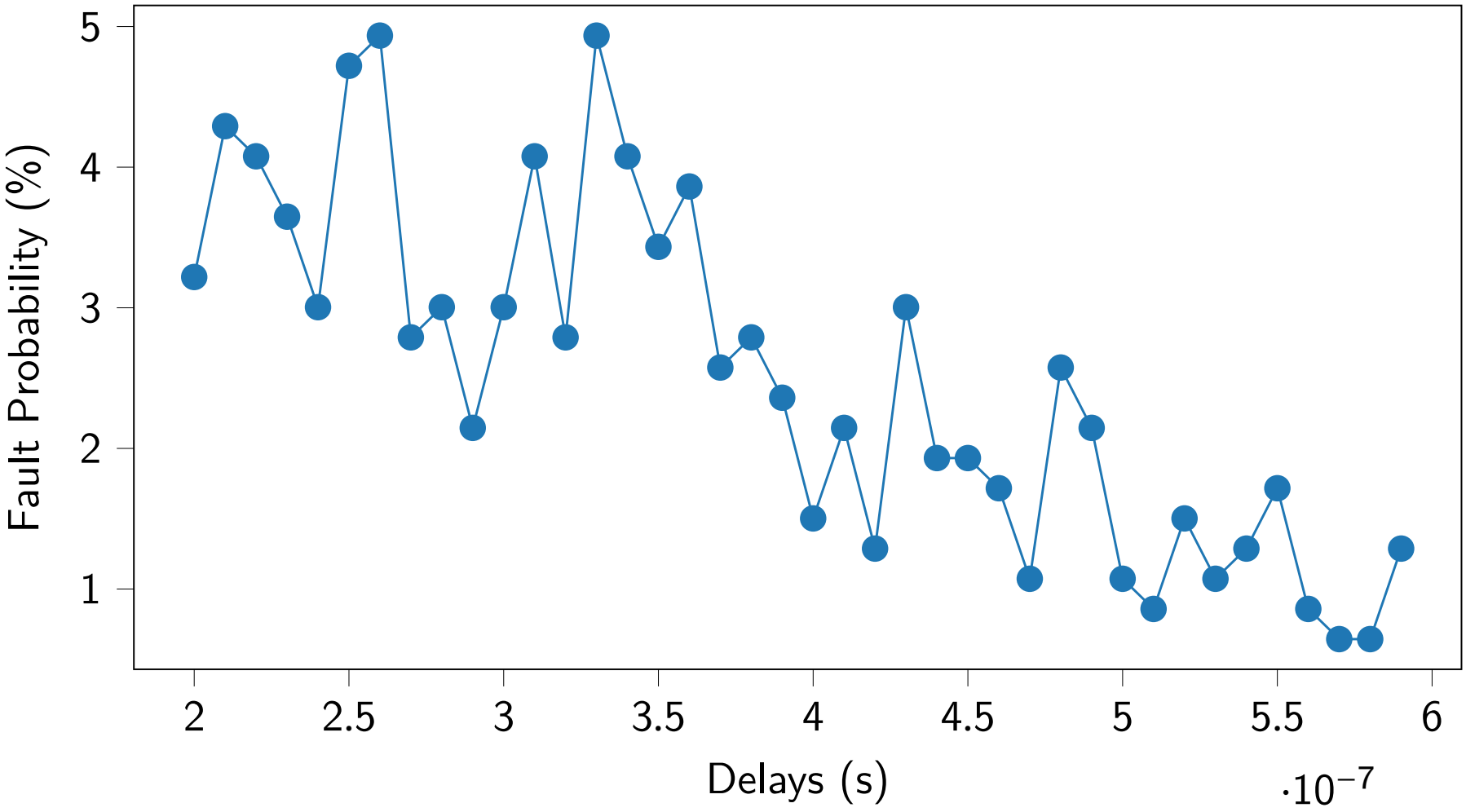
- target `MixColumns` 9^{th} round entry
- 1 useful cipher every 294 injection (0.34%)
- 1 useful cipher every 10 minutes
- 2 to 8 ciphers needed for the attack
- Up to 3 hours of fault injections

Forced authentication
(On going)

- target password verification functions from `PAM` library
- use-case with `sudo` program
- 2 library dynamic loads and 12 functions involved in total

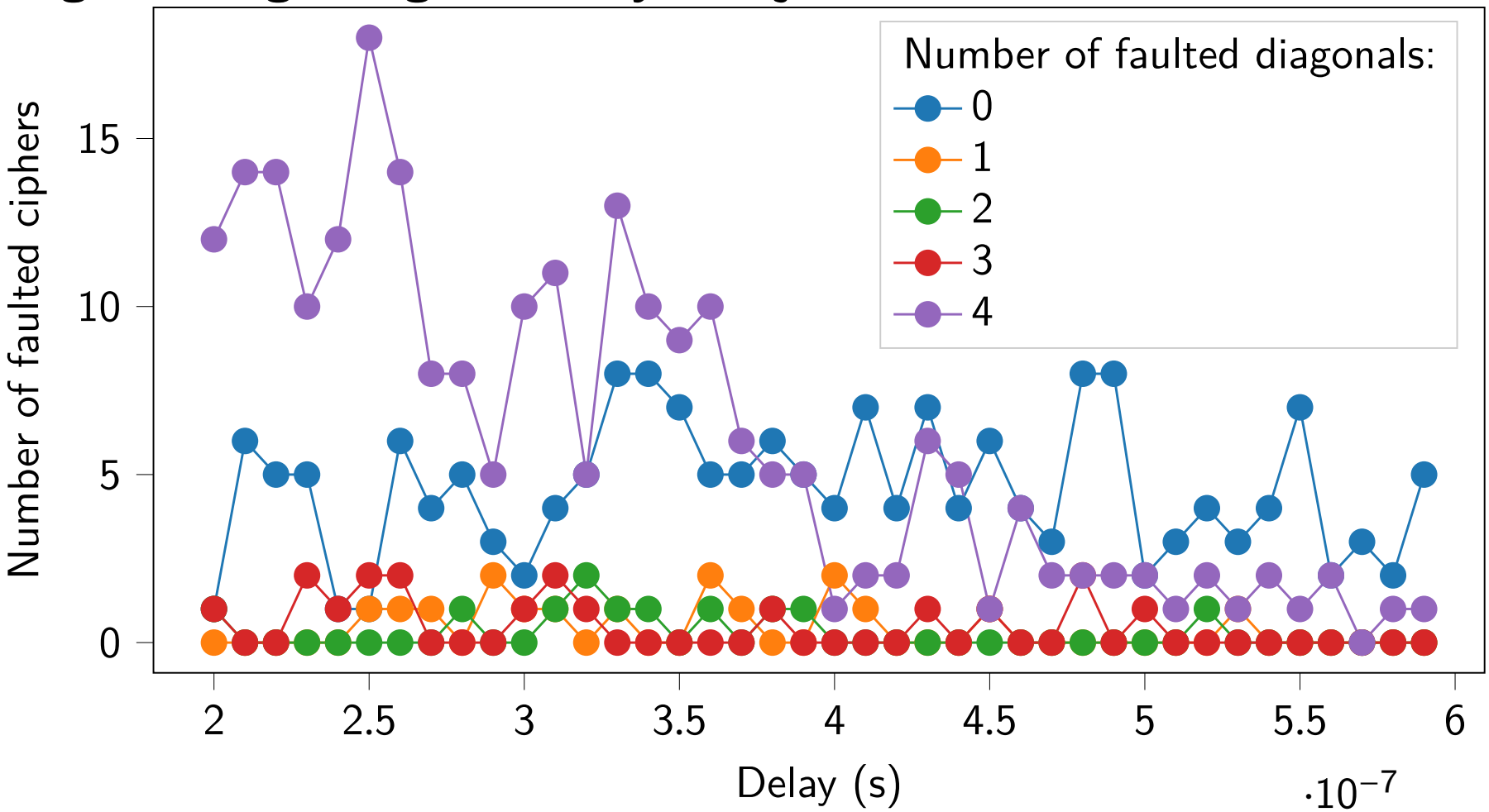
Exploitation - OpenSSL AES

Fault probability regarding the delay of injection



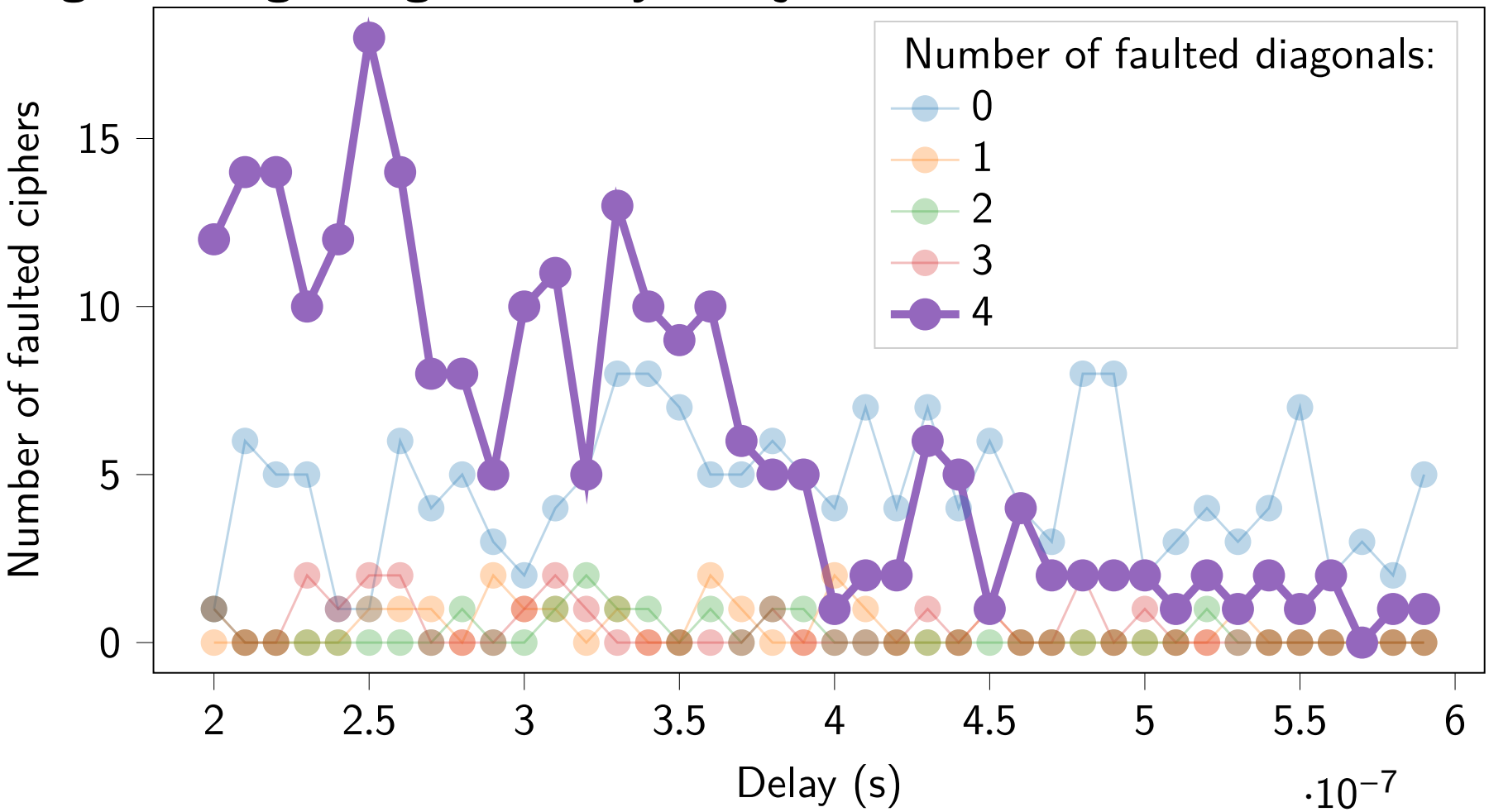
Exploitation - OpenSSL AES

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



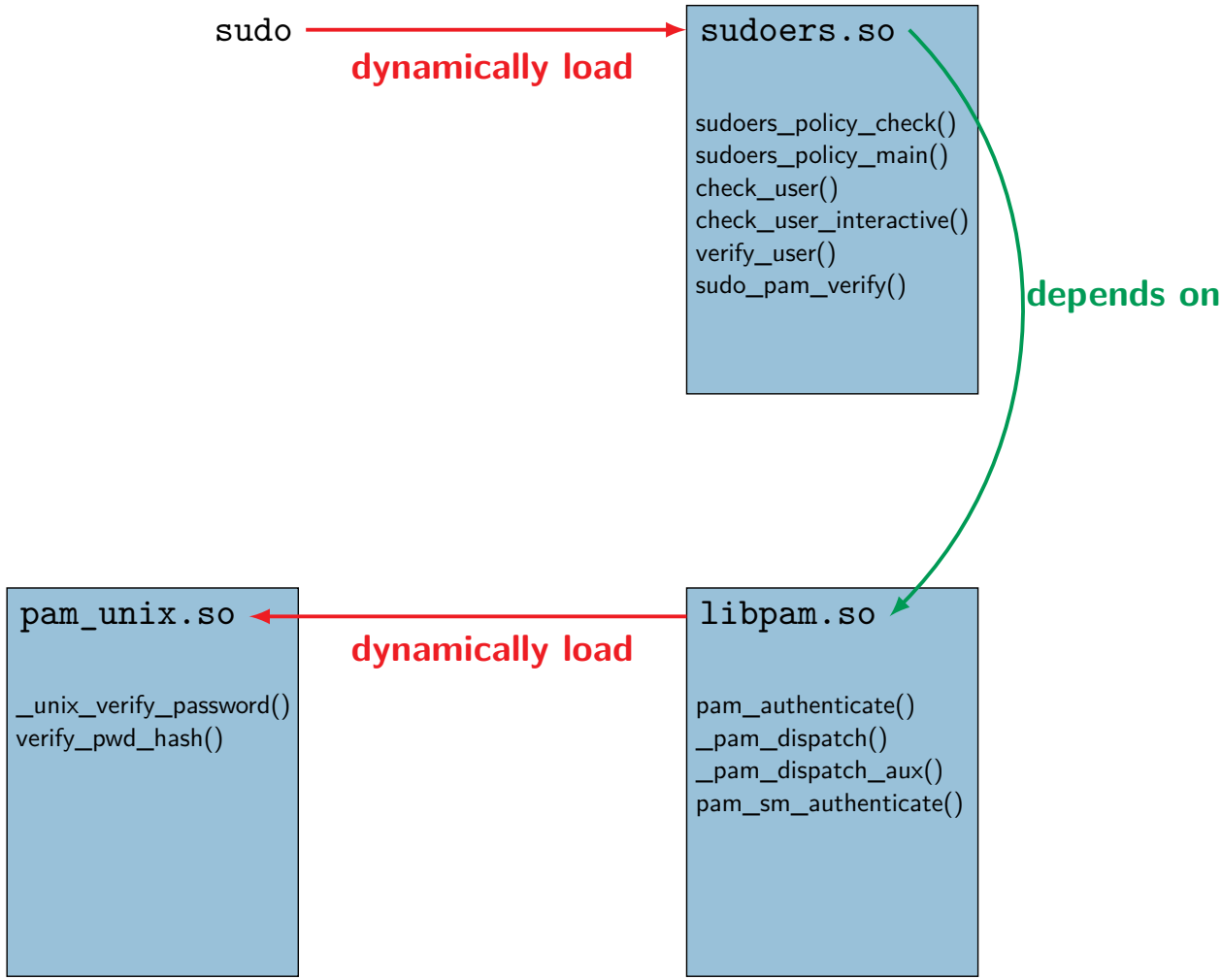
Exploitation - OpenSSL AES

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



Exploitation - Forced authentication

Default `sudo` behavior



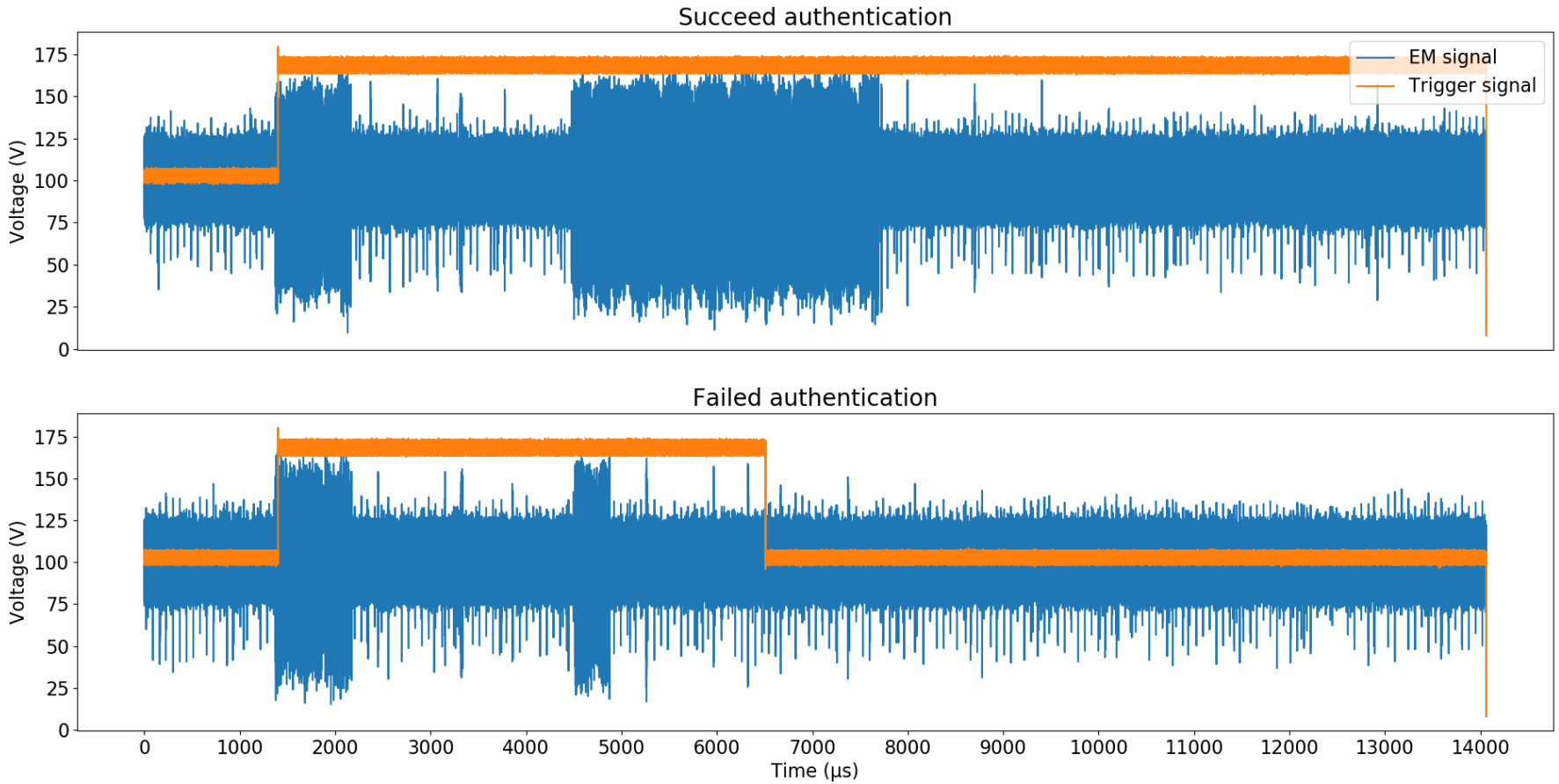
Exploitation - Forced authentication

sudo source code

```
/* Initialize plugin... */
ok = policy_check(&policy_plugin, nargc, nargv, env_add,
                  &command_info, &argv_out, &user_env_out);
if (ok != 1) {    /* Critical if comparison */
  if (ok == -2)
    usage(1);
  exit(EXIT_FAILURE);
}
/* Execute command as root... */
```

Exploitation - Forced authentication

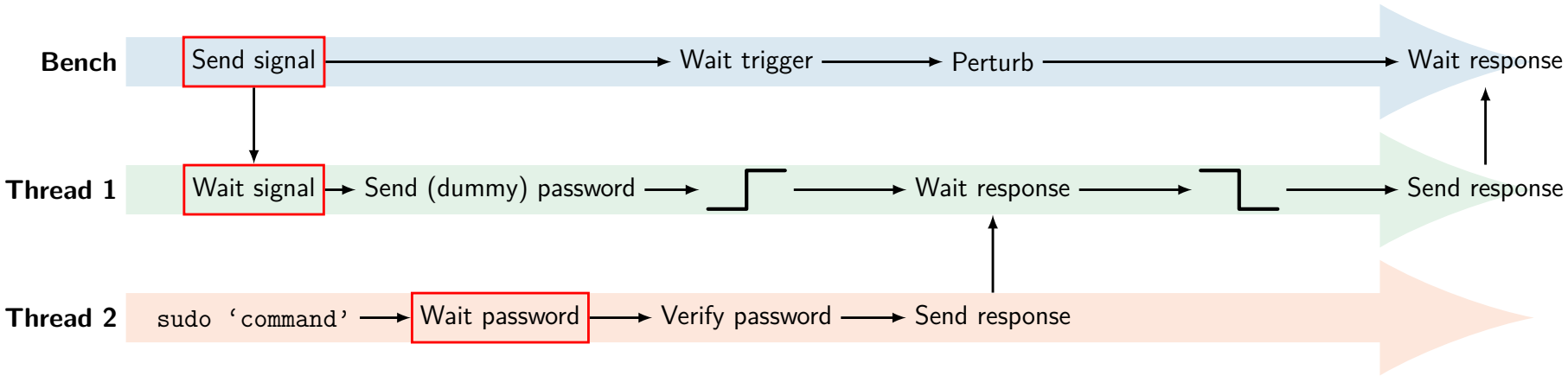
Traces acquired on BCM2711b0 (Laser Fault Injection)



hash comparison based on `strncmp()` function

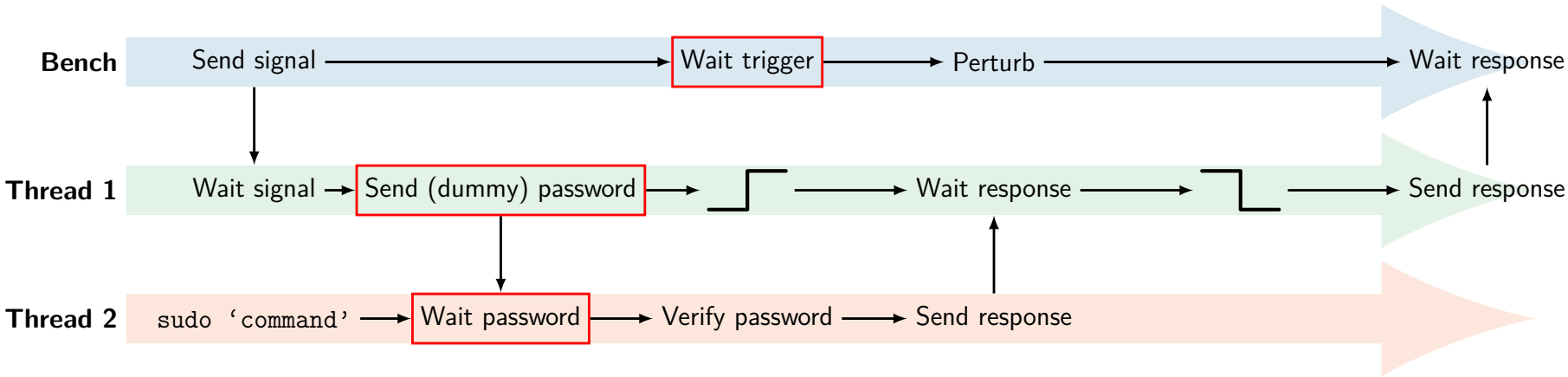
Exploitation - Forced authentication

Target program execution flow



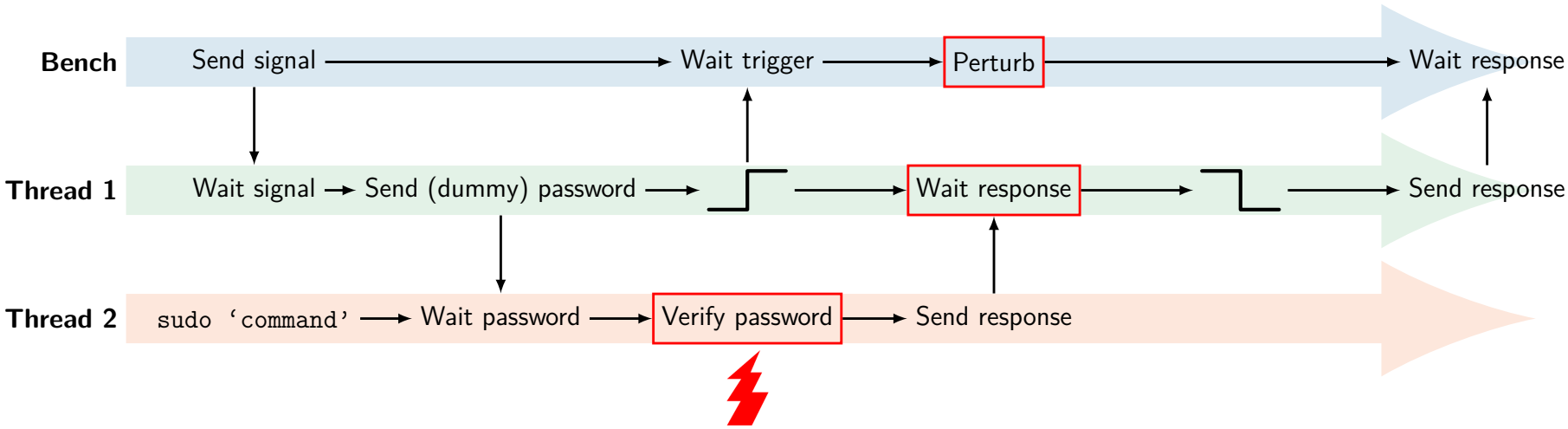
Exploitation - Forced authentication

Target program execution flow



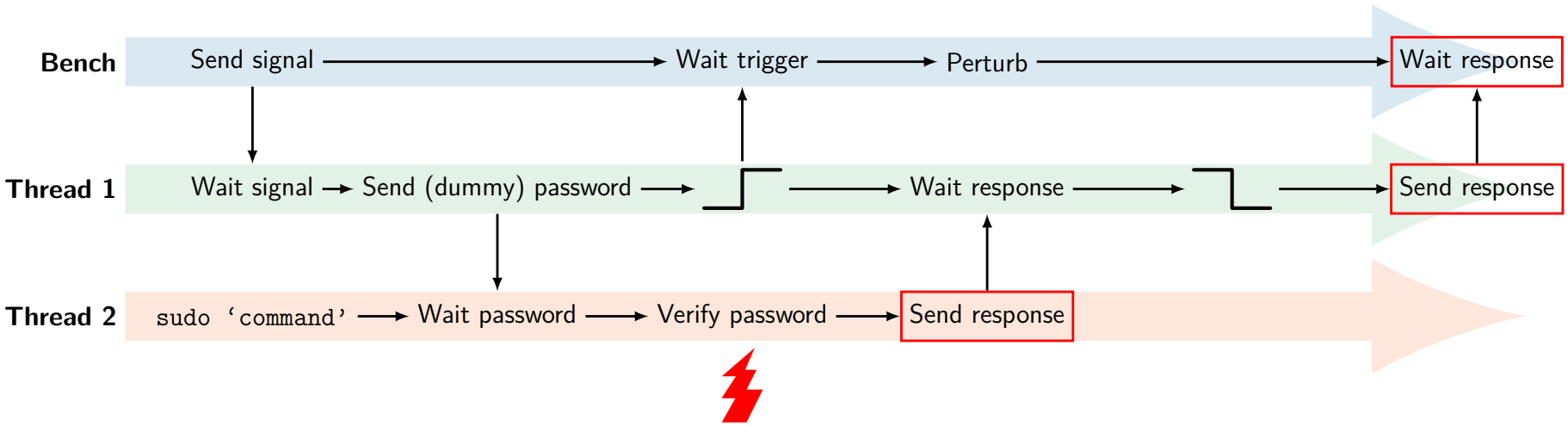
Exploitation - Forced authentication

Target program execution flow



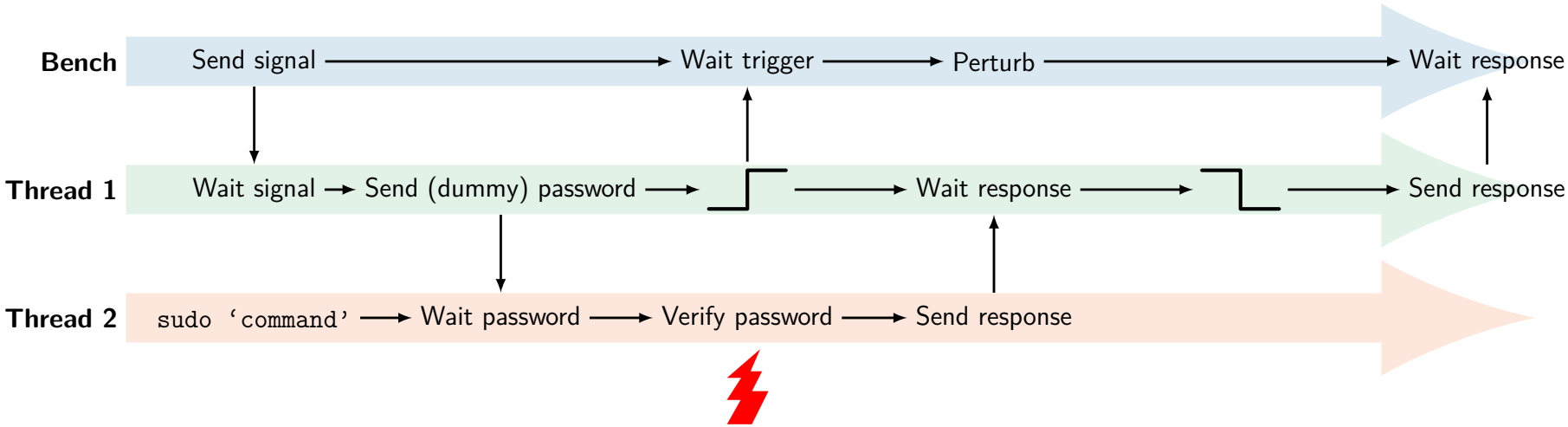
Exploitation - Forced authentication

Target program execution flow



Exploitation - Forced authentication

Target program execution flow



Conclusion

- Classical fault injection mediums (EMFI, Laser) are:
  - efficient on modern CPUs
  - characterizable and understandable
- Modern CPUs have shown sensitive to faults elements, in particular the cache memory
- Modern CPUs asynchronous behavior and high frequencies does not protect against timing precision demanding attacks like DFA

Future works

- Achieve a forced authentication on the targets
- Link side-channel activity with chip activity
- Realize tests on in production chips (embedded in smartphones for instance)
- Determine how the cache is faulted and design an adapted countermeasure

Questions?