From Hardware Vulnerabilities to Combined Hardware/Software Countermeasures

Journées Thématiques des Attaques par Injection de Fautes (Septembre 2023)

Ihab Alshaer and Sébastien Michelland – LCIS, Valence.

Hardware vulnerabilities: research questions

- **Fault injection attack**: Hardware disruptions (clock/power glitch, flip bits in transit...) causing the program to mis-execute and break its security.
- **Fault protection**: hardening software/hardware.

Research questions:

- How to design **accurate fault models**?
- How to integrate hardening with the **compilation process**?

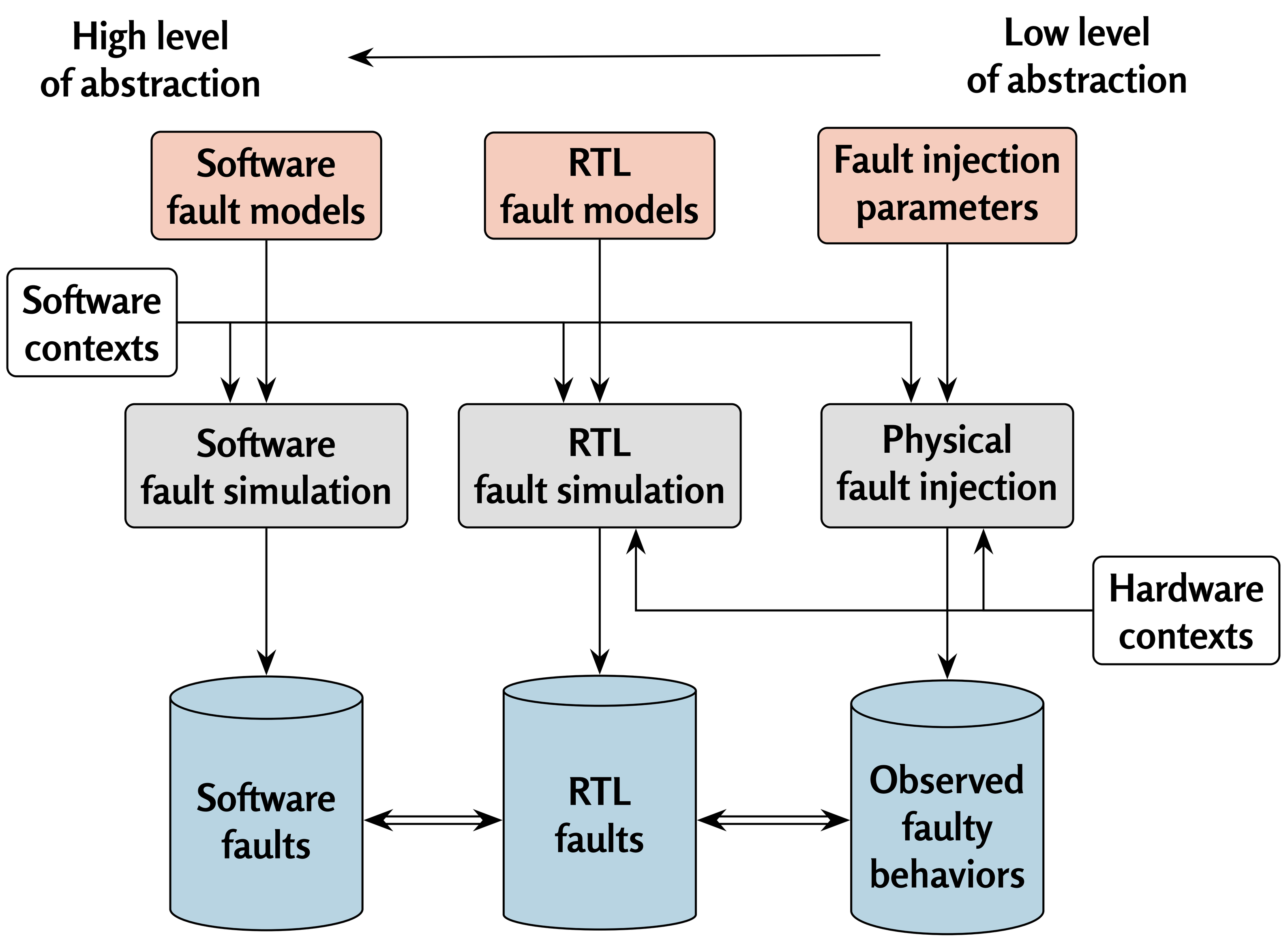
Accurate Fault Models?

Integrated Hardening?

Cross-layer inference methodology [2]

Faults' effects are varied and hard to predict. We propose a complete methodology to infer fault models at different levels of abstraction:

- For any device, program, injection method
- Progressively refine software/RTL models by comparing execution/simulation traces and **physical injection** effects.



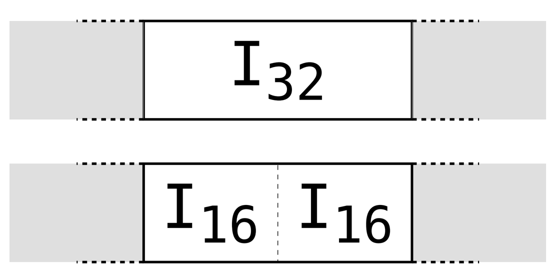
We used this methodology to infer **microarchitecture-aware** fault models for clock/voltage glitches on 32-bit MCUs.

Example of a fault

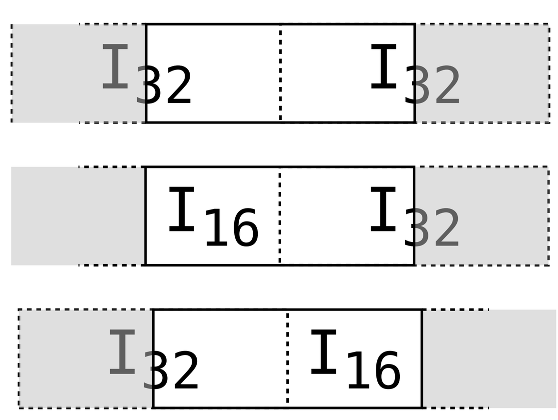
"Fetch skips" fault on 32-bit MCUs [1]

Classical ISA fault models (eg. instruction skip) ignore subtle **micro-architectural behaviors**.

- Clock/voltage glitches affect CPU fetches, which can result in *parts of* instructions being skipped or repeated.



Aligned instructions
(fetched at once)



Misaligned instructions

- Observed on ARM and RISC-V MCUs (using mixed 16/32-bit ISA: Thumb2, RVC).

Hardening during compilation [3]

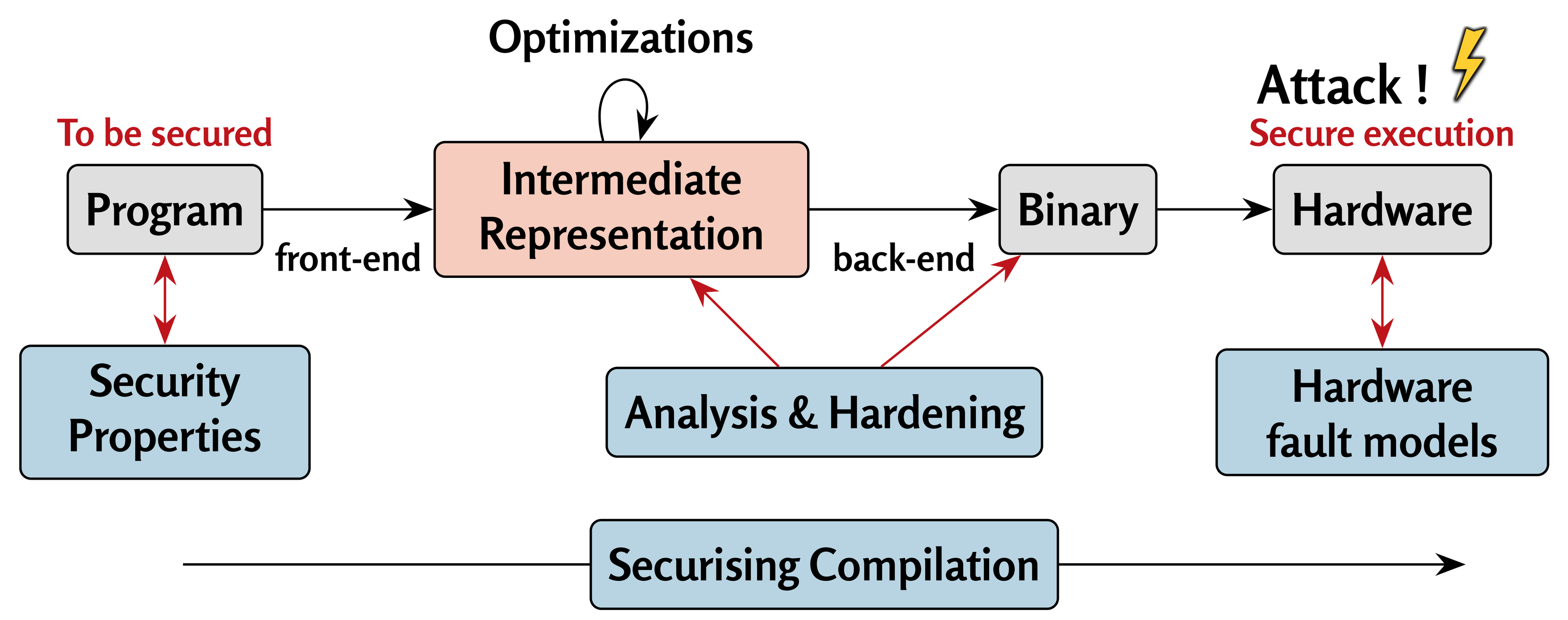
Central issue: **abstraction gap!**

- Desired security property expressed at source level (C)
- But attack targets assembly code/microarchitecture

"Security" doesn't even make sense until compilation finishes.

Thus, specific challenges:

1. The hardening transformation starts on "high-level" code
2. Proving security is hard: this transformation and attack don't coexist



The case of fetch skips

In [3], hardware computes a checksum of lines fetched from memory and software compares it to the expected value for a faultless execution.

- Comparison code added in compiler back-end
- Checksum values precomputed by linker

→ Multi-stage hardening is required.

Formal validation

Modeling and proving security guarantees

Security must be proven, not argued!

Method: *extend the semantics of a language* to incorporate a fault, and *prove* the security property

The case of fetch skips

Fetch in normal assembly: address $\Rightarrow$ value

Fetch in fetch-skip assembly: address, $\rho \Rightarrow \Delta$PC, value, ρ'

New semantics capture:

- ρ: Last line fetched (used in skip-and-repeat attack)
- ΔPC: Change in PC value (used in skip attack)

Future work: Connect internal compiler languages by *observable behavior* to study compiler-preserved security properties.

LCIS Laboratoire de Conception et d'Intégration des Systèmes https://lcis.grenoble-inp.fr

GRENOBLE INP UGA https://www.grenoble-inp.fr

UGA Université Grenoble Alpes https://www.univ-grenoble-alpes.fr

FRANCE 2030 https://www.pepr-cyber-arsene.fr

Emails

ihab.alshaer@lcis.grenoble-inp.fr

sebastien.michelland@lcis.grenoble-inp.fr

This work was partially funded by the France2030 ARSENE Project(ANR-22-PECY-0006)

[1] Ihab Alshaer, Brice Colombier, Christophe Deleuze, Vincent Beroulle, and Paolo Maistri. Variable-Length Instruction Set: Feature or Bug? In *25th Euromicro Conference on Digital System Design (DSD 2022)*. IEEE, 2022.

[2] Ihab Alshaer, Brice Colombier, Christophe Deleuze, Paolo Maistri, and Vincent Beroulle. Cross-layer inference methodology for microarchitecture-aware fault models. *Microelectronics Reliability*, 139:114841, 2022.

[3] Sébastien Michelland, Christophe Deleuze, and Laure Gonnord. Basic Block Jails: A Combined Software/Hardware Countermeasure Against Fetch Skip Attacks, 2023. Submitted at CGO'24 in Sept 2023.