Motivation

- Fault injection (FI) [1] is a threat to embedded systems
- Security analysis under FI attacks is performed either using:
  - Tests on real platforms [2,3]
  - Simulation [4,5]
  - Formal methods [6]
- Common fault models considered at the ISA level:
  - Instruction replacement
  - Register corruption
- Many fault effects depend on the hardware implementation and are not directly addressable at the ISA level [7], e.g.,
  - Pipeline
  - Speculative execution
- A precise knowledge of the microarchitecture is essential for:
  - A better understanding of the effect of faults
  - A better security evaluation

Contributions

- Develop a workflow that encompasses the SH/HW to identify harmful faults
- Illustrate the approach on a use case
- Exhibit new fault effects that are difficult to model at the ISA level

Approach

- Yosys produces a formal model (SMT-Lib)
- Software program, FI effects and FI timings are modeled by adding constraints
- Security properties are expressed with assertions
- Yices SMT Solver uses bounded model checking to find vulnerabilities
- Returned counter-examples highlight the propagation of the faults in the microarchitecture

Workflow

1. SW Program
2. Fault Model
3. Vulnerability Property
4. RTL
5. Yosys
6. SMT
7. SW Design
8. Compiler
9. Formal Model
10. Yices
11. SAT / unSAT
12. Counter-Example

Use Case

- Investigate single-FI attacks on the logic gates and wires of the design during the VerifyPin execution
- About 6 CPU hours of calculation on an 80-core cluster were needed
- 50 faults injections were identified (2 of them are detailed below on the left)

Fault Vulnerability Results

- We identify vulnerabilities already known in the literature [7], e.g., faulting the forwarding mechanism by targeting the operand_a_fw_mux_sel_o signal
- We highlight that a fault injection on the prefetch buffer (PFB) (i.e., status_cnt_n) leads to effects that are difficult to model at the ISA level:
  1. The fault can force the execution of instructions speculatively fetched in the PFB
  2. Next instructions are potentially pushed in the pipeline in an incorrect order
  3. The program jumps to an incorrect address at the next branch instruction

Perspectives

- Study more complex processors, e.g., CVA6
- Determine the minimal design to model new fault effects at the ISA level

Bibliography