MASTER IN

CYBER–SECURITY

UNLOCK YOUR POTENTIAL

Lab 2: Fault Injection simulation and countermeasures

UNIVERSIDAD DE MURCIA

Facultad de Informática

Introduction

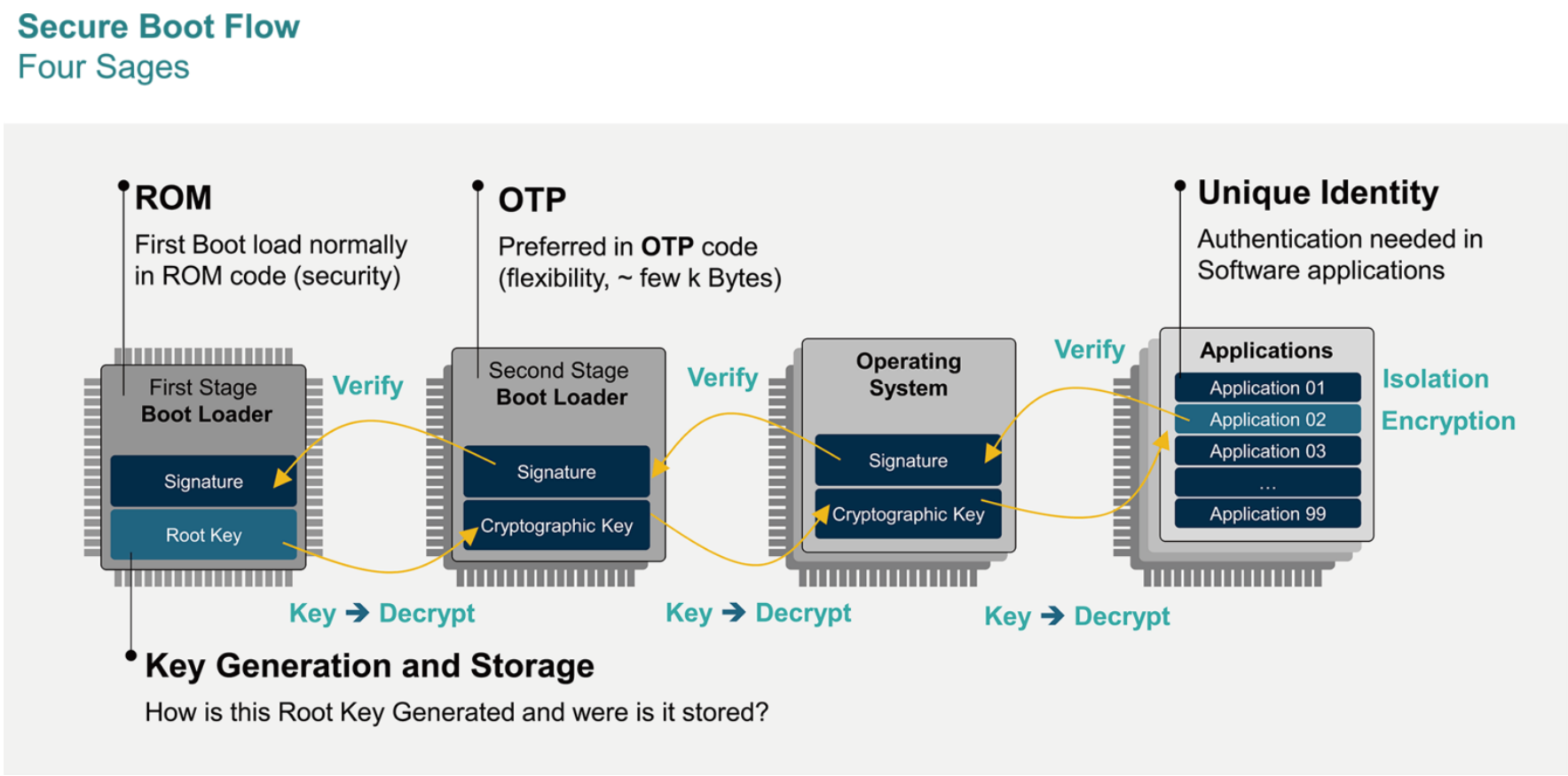
- The aim of this lab is to introduce fault injection and its countermeasures.

- Concretely we will use a simulator for fault injection attacks used with bootloader code

- You will modify the code to introduce countermeasures and analyse the results.

Secure Boot

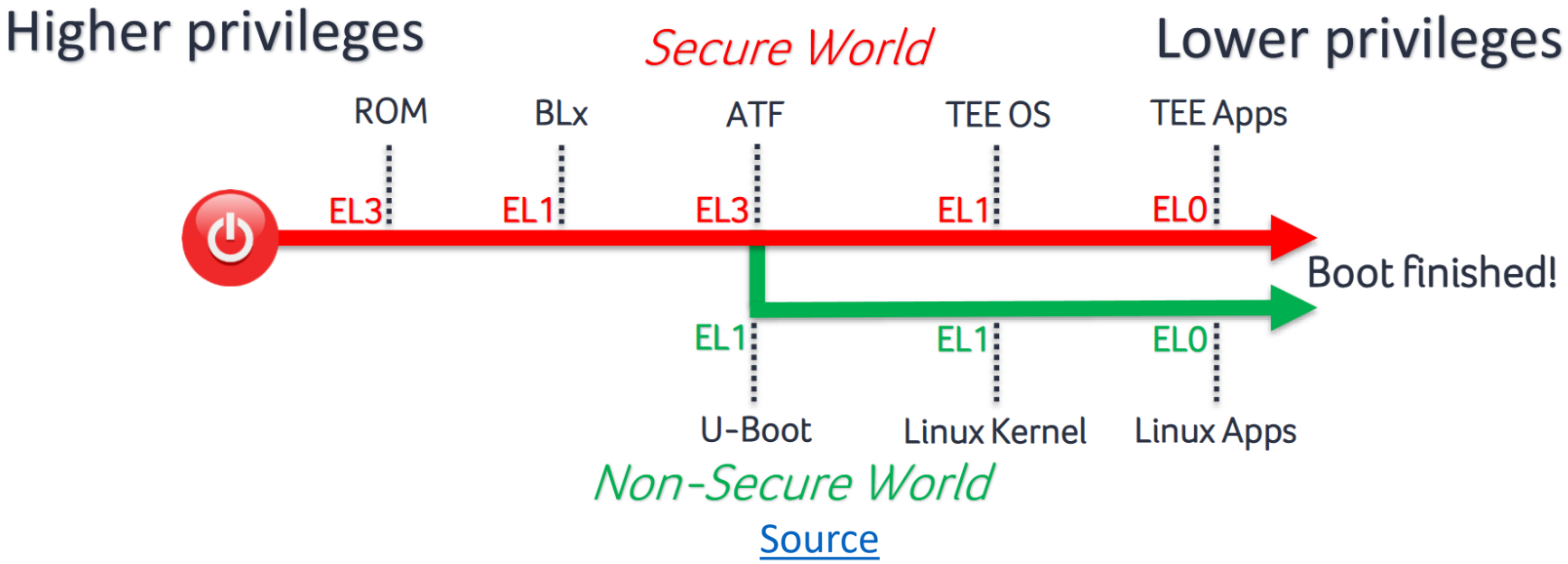
- The target code will be a simplified bootloader operating during a secure boot process



Source https://www.pufsecurity. com/technology/secure-boot/

Secure Boot

- The target code will be a simplified bootloader operating during a secure boot process



Source

Fault injection

- Intentionally cause errors in a system in order to compromise the security of the system.
  - Modifications on the contents of memory, registers or the executed instructions.
- The attack may result in glitches following different models, we will consider two:
  - A fault/glitch causes the micro-controller to skip an instruction.
    - E.g., `beq 1008f8` may be completely skipped (substituted for `nop`).
  - Flipping a bit in the encoding of a specific instruction.
    - E.g., `cmp r3, #0` may be changed into `cmp r3, \#1`.
    - Note this also may affect the instruction itself, not only constant/register values

Fault injection software countermeasures

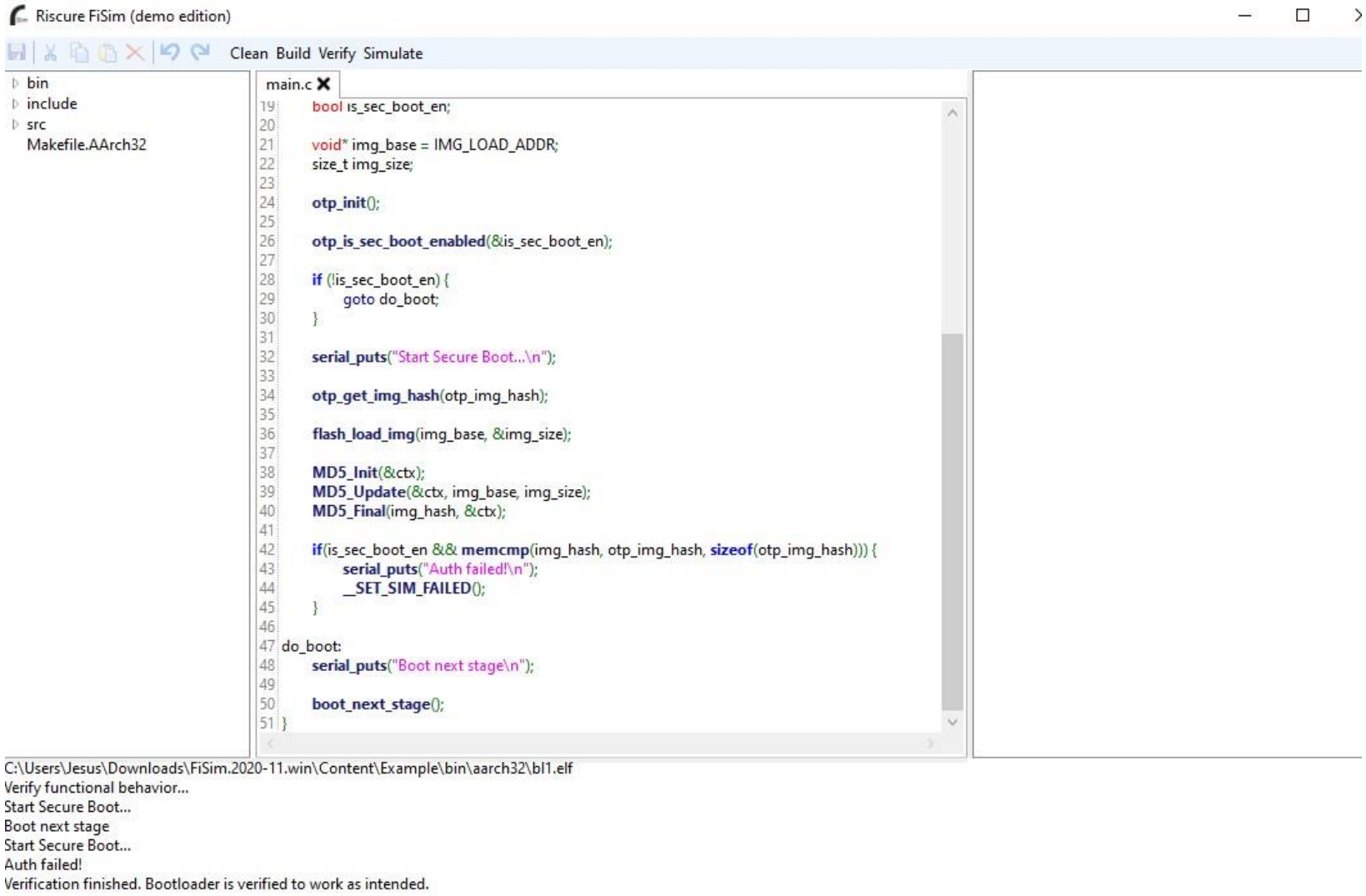
- **Redundant operations**: The main purpose of performing redundant operations (e.g., check) is to force an attacker to successfully perform multiple successive glitches.
- **Defensive coding**:
  - **Non-trivial constants**: Using ``magic numbers'' for constants, enums… (good practices are to include a good mix of 1s and 0s, high hamming distances…) and ``closed-on-fail'' principles.
  - **Initialize to error values**: The variables before a check is initially set to an error value, avoiding some simple attacks by e.g. jumping instructions.
  - **Error by default:** If an unexpected event happens, then a fault can be assumed and an error notified (default switch branch, "unreachable" branch…)
- **Timing countermeasures**: Not a direct countermeasure to the fault injection effect (will not be seen in the simulator). Fault attacks are timing critical, so introducing randomized access, random delays… makes it more difficult to target the glitch.

Fault injection software countermeasures

- **Fault detection test**: It attempts to detect fault injections by verifying that some arithmetic invariants are respected, and branch to return an error instead of the numerical result of the algorithm in case of invariant violation.
- **Infective computing**: It uses arithmetic invariants to compute a neutral element (that is not valid if fault happens) that is used to infect the result of the algorithm before returning it to make it unusable by the attacker.
- **Control flow integrity**: Advanced yet with limited applicability technique based on introducing checks that the correct execution flow of the program has occurred, e.g. through CRCs.

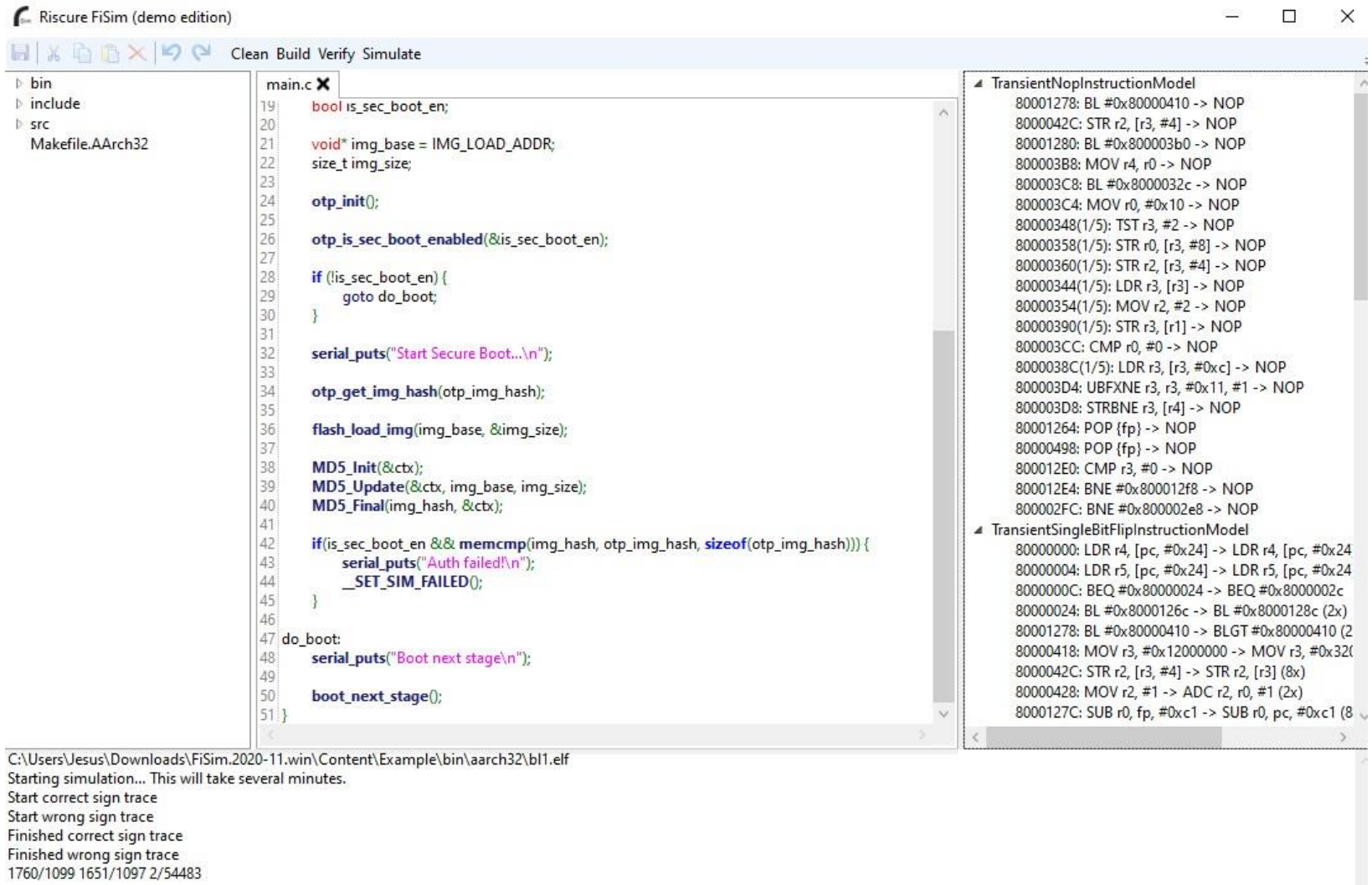
Tools

- The FiSim simulator



Tools

- The FiSim simulator



Exercise and report

- Modify the bootloader implementation to introduce countermeasures to the discovered fault attacks
  - At least 5
  - The countermeasures must be varied in their approach redundancy, constants and initialization, respond to faults detected/errors by default…
  - Tackling different parts of the code will also be valued.
- For each countermeasure implemented, the report must analyse:
  - The vulnerabilities tackled
  - The results of the application of the countermeasure
  - The trade-offs introduced
- Analyse the results obtained when all countermeasures are applied, including remaining vulnerable points detected by the simulator and some other potential fault scenarios and models that could bypass the mitigations.

Exercise and report

- Conducting the lab: Perform the specified tasks and document it as explained in the lab guide.
  - Go step-by-step carefully reading the guide.

- Reporting: Follow the style guide that can be found in the virtual classroom (some small things can be changed, e.g. using Latex fonts/formatting)

- Deadline: 18th May 2025

UNIVERSIDAD DE MURCIA

Facultad de Informática