Binary level code analysis : Side channel, Fault injection & Library stubs

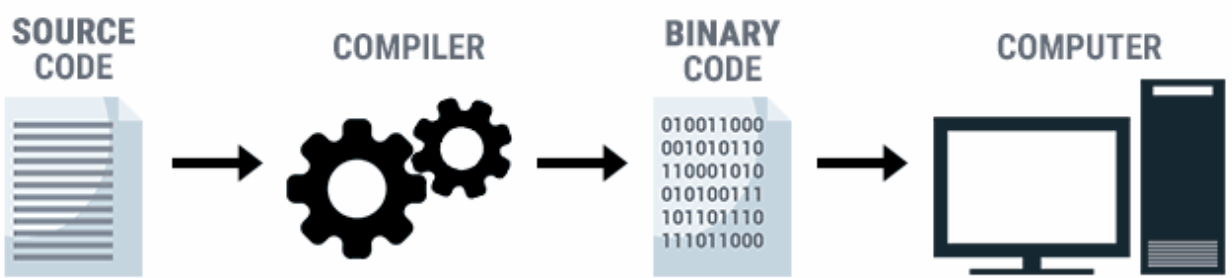
Frédéric Recoules

Yanis Sellami

Sébastien Bardin

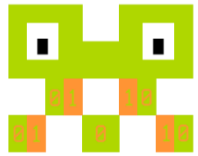
A need for binary level analysis

No source code



What You See Is Not What You Execute

BINSEC in a nutshell (since 2012)



binary lifting,
IR, CFG, call graph,
symbolic execution,
static analysis, ..

Solvers

Decoders

Symbolic engine

BINSEC

Generic IR

OCaml
56k LOC

Vulnerability Assessment

Security critical components

- Fault injection
- Side channel attack
- Attacker model

Bug finding

Supply chain

- Advanced fuzzing
- Test case generation

Reverse Engineering

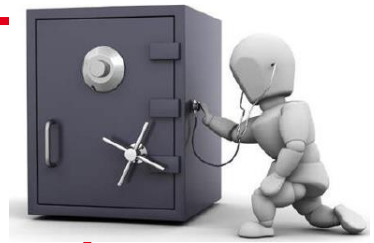
Malware comprehension

- Capture The Flag
- Deobuscation
- Decompilation

UNISIM Virtual Platforms .org

Research project topics

Side Channel Attacks



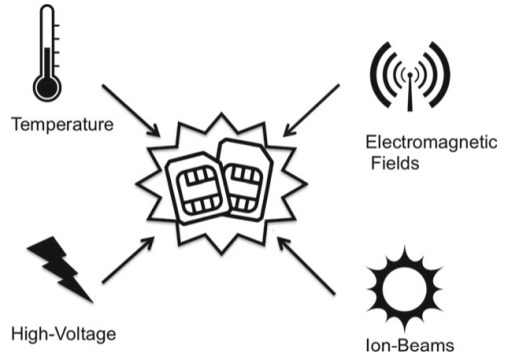
- **Leakages.** Timing information, power consumption, electromagnetic leaks and sound, etc.
- **Constant time verification.** IEEE S&P 2020, NDSS 2021, CCS 2023 ✓
- **Goal.** *Handling of new threat models (e.g. Power attacks, Ciphertext attacks)*

Library stubs

- **Missing code.** Dynamically linked library (e.g. libc), syscalls, etc.
- **High-level concepts.** File system, string, etc.
- **Goal.** *Improvement of the expressiveness, the genericity or the automation of the stubbing mechanism*

Fault Injection Attacks

- **Perturbations.** High voltage, extreme temperature, electromagnetic pulses, laser beam, etc.



- **Adversarial symbolic execution.** ESOP 2023 ✓
- **Goal.** *Handling of new threat models (e.g. RowHammer) or improvement of the analysis scalability*