From NOP to ADD and Beyond

A Novel Fault-Model Comprising Variable-Length Instruction Sets

Marvin Saß
TU-Berlin

Thomas Martin Johannes Lehrach
TU-Berlin

Jean-Pierre Seifert
TU-Berlin

SECT

Fault Diagnosis and Tolerance in Cryptography

From NOP to ADD

- *Central question of this work:*
  - *“What if, a single bit-flip could re-write your entire program?”*
- Implications of injecting faults are well understood for *embedded* devices, where we encounter:
  - simple architecture
  - reduced instruction set characteristics (RISC)
  - instructions of clearly-defined lengths
- For all x86-based systems we have:
  - highly complex micro-architecture
  - *complex* instruction set
  - *variable*-length instructions

Yet, we are considering their fault models to be equal!

Most-Related Work

Trouchkine, Thomas, Guillaume Bouffard, and Jessy Clédière.
"Fault injection characterization on modern cpus: From the isa to the micro-architecture."
IFIP International Conference on Information Security Theory and Practice.
Cham: Springer International Publishing, 2019.

- Methodology to map micro-architectural blocks (MABs)
  - By this making statements about their behavior under fault:
    - Incorrect arithmetic? ➡ ALU affected
    - Incorrect memory values? ➡ Load/Store unit affected
    - Instruction corrupted? ➡ Fetch/Decode unit affected

➢ Does *not* consider decoder misalignment

Most-Related Work

Trouchkine, Thomas, Guillaume Bouffard, and Jessy Clédière.
"EM fault model characterization on SoCs: from different architectures to the same fault model."
2021 Workshop on Fault Detection and Tolerance in Cryptography (FDTC).

- Analyzes ARM (BCM2837) and x86 (i3-6000u) instructions to exhibit same fault models
  - single instructions show similar bit flip pattern
  - hence, a similar fault model is assumed

➢Does not consider decoder misalignment

Most-Related Work

Alshaer, Ihab, et al.
"Variable-length instruction set: Feature or bug?."
2022 25th Euromicro Conference on Digital System Design (DSD).

- Fault Injection on variable length ISA (ARM with Thumb-II)
  - Skip-4B or Skip-4B and repeat previous instruction
  - Decoder always stays aligned
  - No hidden instructions executed

➢Does not consider decoder misalignment

Variable Length Instruction Sets

ARM (Thumb-II), Risc-V (RV32-C), …

- Considered variable-length ISAs as well
- Commonly *switchable* between *32*- and *16*- bits
  - ➢Saving program memory for embedded devices

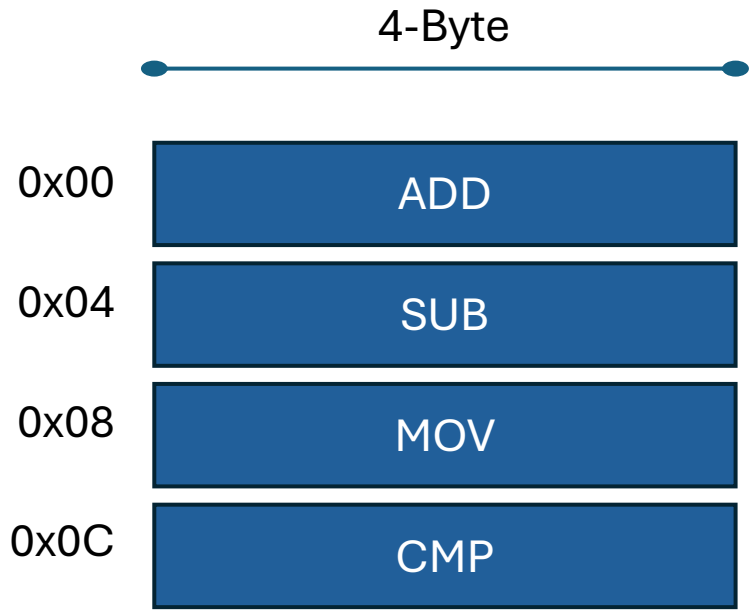
X86 (always)

- True variable instruction set
  - Common instructions range from 1 to 15 bytes!
- An x86 program is nothing more but a continuous sequence of bytes
  - Must be interpreted *on the fly!*

➢**One** correct interpretation

Injecting Faults on ARM32

ARM32



If we assume PC+4 fault model …

- Any of the instructions could be skipped
  - e.g., the ADD at 0x00
- New program would only execute SUB, MOV, CMP

How would that look on x86?

In this case, a totally new program would be executed, as the byte-stream would be re-interpreted from 0x04!

4-Byte alignment displayed for clarity only!

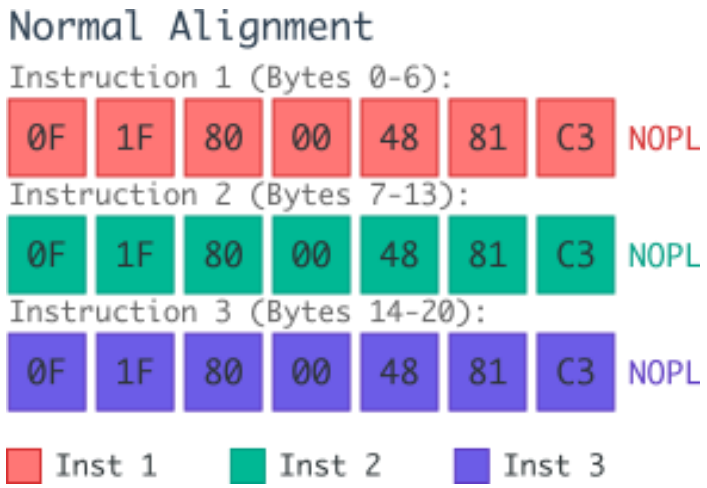
Verify misalignment in x86

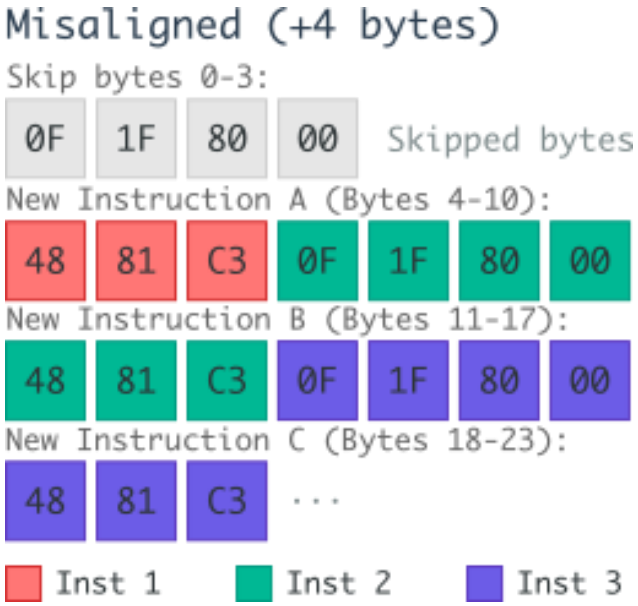
- How can we test such hypothesis?
  - A lot of *invalid* instructions
  - Misalignment *instable*
  - Hence, hard to observe!
- We craft a specific payload, hidden in regular instructions
  - Normally dormant
  - Activate by injecting faults
- x86 ISA defines multiple NOP encodings, such as:
  - 1-byte NOP:
    - 0x90
  - 7-byte NOP:
    - 0x0F1F8000000000
  - 9-byte NOP:
    - 0x660F1F840000000000

From 7-Byte NOP to 7-Byte ADD

- Note, that for longer NOPs the lower bytes are zero:
  - 7-byte NOP: 0F 1F 80 00 00 00 00
- These are ignored by the decoder
  - continues decoding at PC+7

➢We can *embed* another instruction

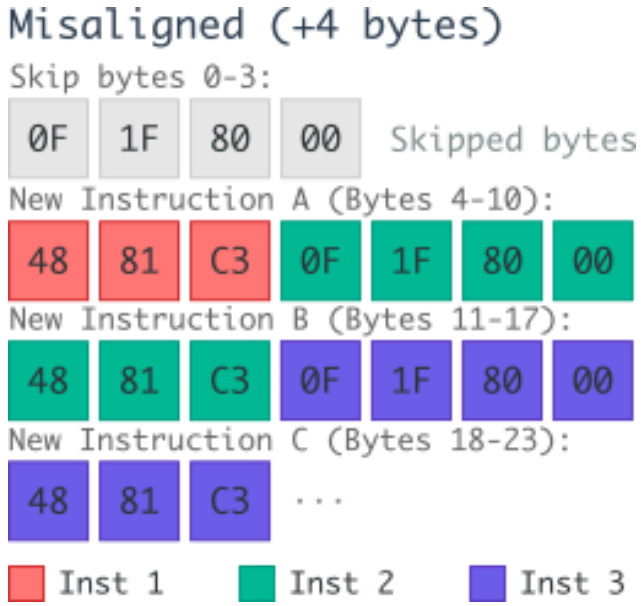




From 7-Byte NOP to 7-Byte ADD

- By this specifically crafted payload, we transform a stream of NOP instructions into a stream of **ADD RBX, imm32** instructions:
  - **RBX** is initialized to zero beginning of the program
  - **imm32** is 0x0F1F8000

- This is a target crafted to test our fault-model hypothesis (PoC)



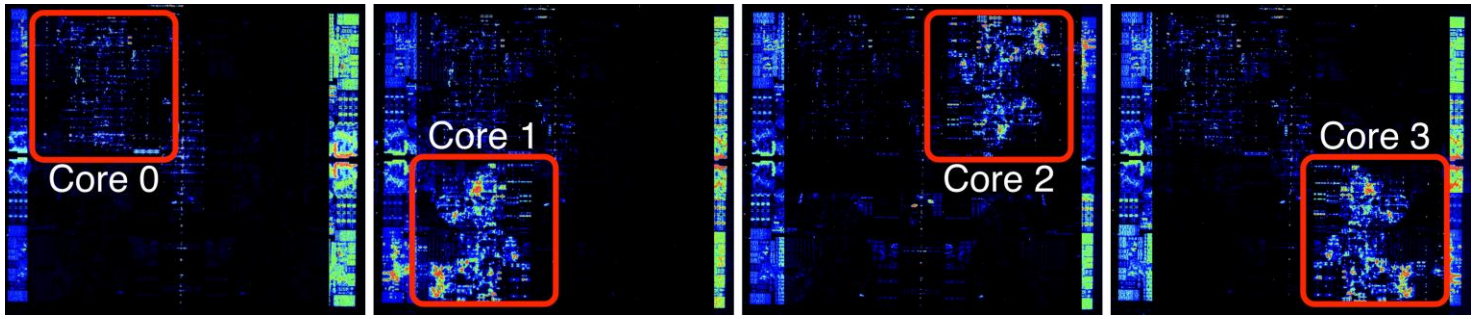
Device under Test

- We test our fault model hypothesis on
  - Latte-Panda Mu (N100)
  - Running an UEFI Module
    - UART based communication protocol (request / transmit commands and data)
    - Grants bare-metal access in absence of operating system
  - No DVFS, operating at constant 800 MHz
  - I-cache and D-cache deactivated
  - Gracemont micro-architecture (Alder Lake)
- Proposed attack also works with DVFS and caches being activated
  - However, results due to timing less accurate

Fault Injection Methodology

1. First, we perform photonic emission microscopy
   - by on-off modulation we discover the x86 core positions on the die

   ➢Drastically narrowing down XY search space! (weeks vs. hours)



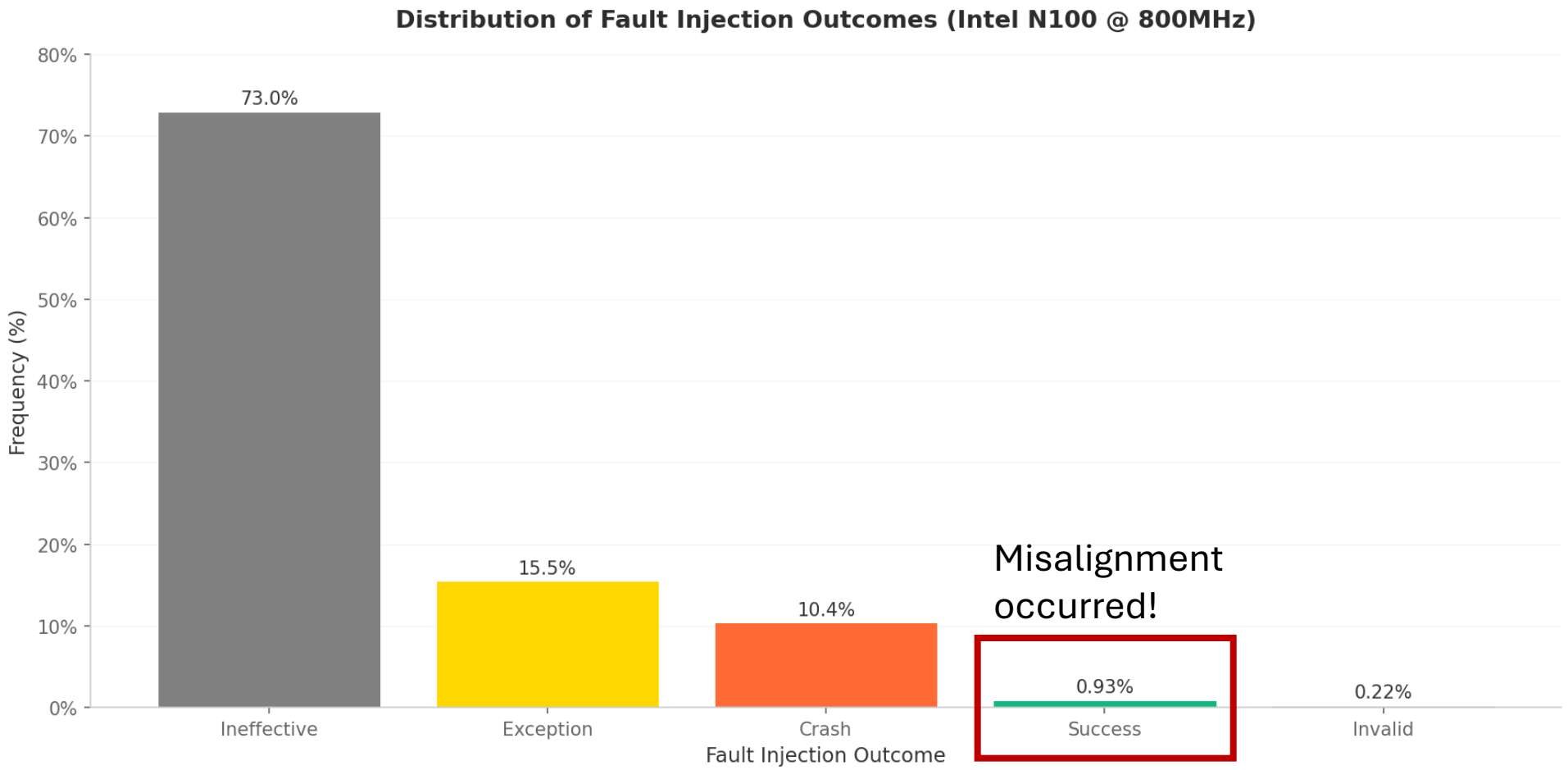
2. EMFI performed using NewAE's ChipShouter [4] in combination with OpenXYZ [5]
   - 150V, 80nS pulse width, 3mm NewAE EMFI coil (empirically most promising)

Do Misalignment fault happen?

- To evaluate our novel fault model, we execute an UEFI routine which:
  1. Generates a trigger
  2. Sets register values to clearly defined values
  3. Executes 10_000 of our specifically crafted NOPs
  4. Transmits processor state and performance counters

- The results show clear evidence of misalignment!

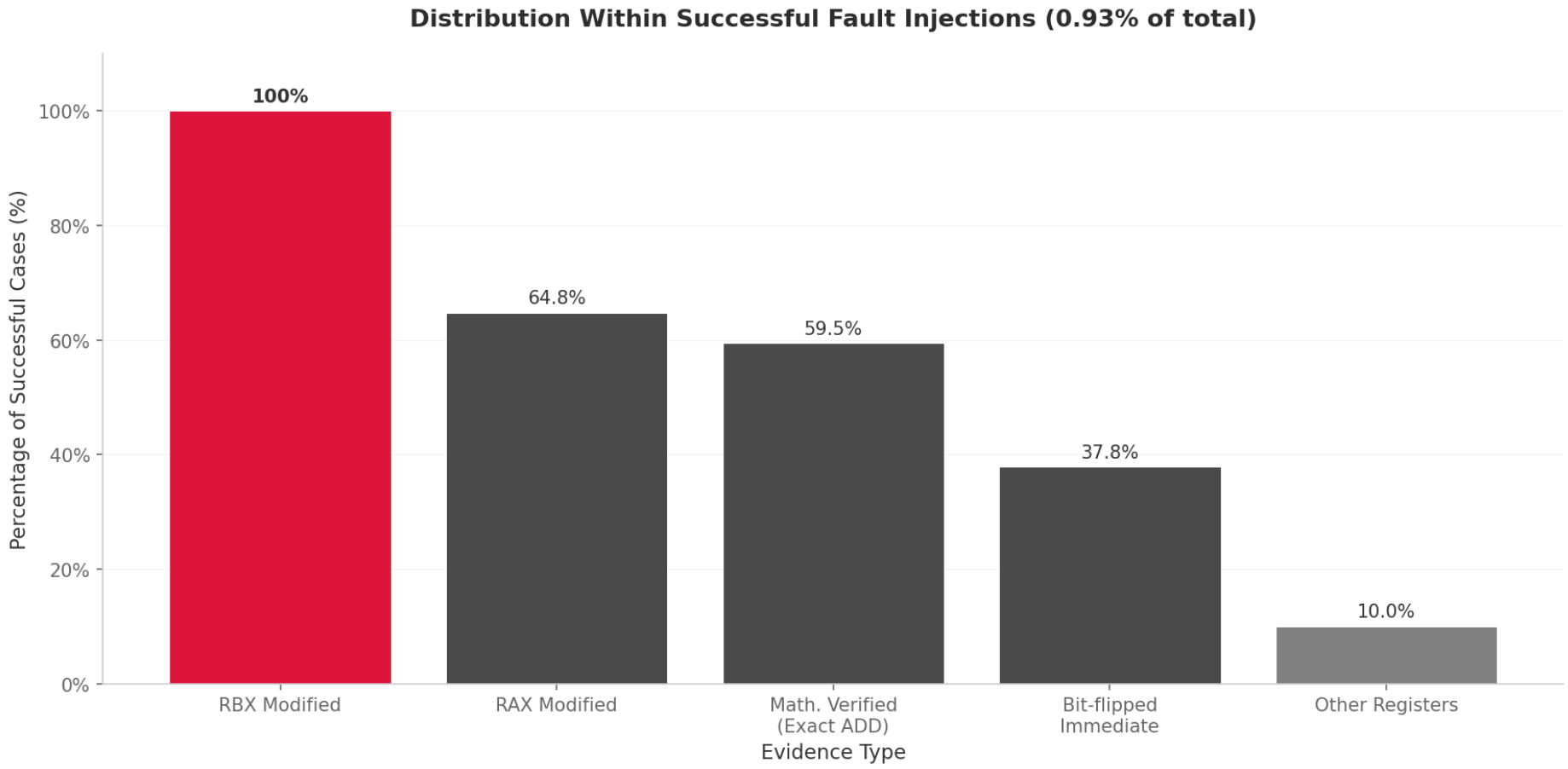
Distribution of Results



SECT

Fault Diagnosis and Tolerance in Cryptography

Distribution within the Successes



SECT

Fault Diagnosis and Tolerance in Cryptography

And Beyond

- Since acceptance of this work at FTDC we continue working on this project:
  - We created a QEMU-based Fault Injection simulator for x86
  - Simulator analyzes any provided x86 binary for misalignment vulnerability
    - Currently scanning open-source software
    - Discovered severe vulnerabilities
- Preliminary results confirm that out novel misalignment faults bypass fault-tolerant firmware!

Take-Aways 

- The fault model encountered on x86 is fundamentally different to those we know!
  - *Could* also explain results obtained in past work

- In this work we turned a stream of NOPs into a stream of ADDs with one electromagnetic pulse

Every x86 program hides a second program waiting to be revealed!

SECT Fault Diagnosis and Tolerance in Cryptography

References

- Trouchkine, Thomas, Guillaume Bouffard, and Jessy Clédière. "Fault injection characterization on modern cpus: From the isa to the micro-architecture." *IFIP International Conference on Information Security Theory and Practice*. Cham: Springer International Publishing, 2019.
- Trouchkine, Thomas, Guillaume Bouffard, and Jessy Clédière. "EM fault model characterization on SoCs: from different architectures to the same fault model." *2021 Workshop on Fault Detection and Tolerance in Cryptography (FDTC)*.

References

- Alshaer, Ihab, et al.
  "Variable-length instruction set: Feature or bug?."
  2022 25th Euromicro Conference on Digital System Design (DSD).
- https://github.com/newaetech/ChipSHOUTER
- https://github.com/HWS-XMS/OpenXYZ.git