(M)Wait For It:

Bringing the Gap Between Architectural and Microarchitectural Side Channels

USENIX Security ‘23

Ruiyi Zhang, Taehyun Kim, Daniel Weber, Michael Schwarz
Motivation

- Timing
- Power

Blind Spot
• Can we replace measurements with an architecturally-defined interface to leak side-channel information?

• Can such an interface also reduce the blind spot of existing side channels?
UMONITOR & UMWAIT

- Introduced with Intel Tremont and Alder Lake microarchitecture

- *umonitor* - Arm the hardware with a specified memory range
- *umwait* - Put the CPU into a sleep state
Different Wakeup Triggers

- The default timeout is **100,000 cycles on Linux**

OS-defined Timeout

User-defined Timeout

Interrupts

Memory Modification

Carry Flag: 1

Carry Flag: 0
- Although undocumented, **transient** writes also wake up the CPU
Transient Write Monitor

Timeline

Victim

Attacker

~175 cycles

100,000 cycles
Timerless Timing Measurement

Attacker – Core #X
- UMWAIT
  - Sleep
  - OS-defined Timeout

Attacker – Core #Y
- Padding (NOPs..)
- Event
- Write

Cache line
Interrupt Monitor

- Count the number of retired *umwait* within a coarse-grained time bucket (~10ms)
- Also available for AMD and ARM
  - *monitorx / mwaitx*
  - *wfi*
Case Studies

- Spectral
  - Up to 200 kbit/s

- AES T-table Attack

- Website Fingerprinting
  - 78 % on AMD
  - 71 % on Intel
  - 67 % on Arm
Takeaway

• First architectural side channel on Intel microarchitecture

• Minimal blind spot, High-precision, Low-noise

• Interrupt-timing attacks on Intel, AMD, and ARM with unprivileged instructions

Contact:
https://twitter.com/Rayiizzz
ruiyi.zhang@cispa.de

Artifact:
https://github.com/cispa/mwait