Hochschule **RheinMain**
University of Applied Sciences
Wiesbaden Rüsselsheim

Mitigation of actual CPU attacks

A hare and hedgehog race not to win

09.08.2018

Jens Nazarenus

Conference 4th WAMOS 2018

TOC

RECAP '18

RECAP '18

01/04/18	Meltdown
	Spectre variant 1
	Spectre variant 2
01/25/18	Retpoline (Spectre variant 2)
01/28/18	KAISER / KPTI
02/07/18	Kernel patches (Spectre variant 1)
03/27/18	Branchscope
05/22/18	Spectre variant 3
	Spectre variant 4
07/10/18	Bounds check bypass store

RECAP '18

01/04/18	Meltdown
	Spectre variant 1
	Spectre variant 2
01/25/18	Retpoline (Spectre variant 2)
01/28/18	KAISER / KPTI
02/07/18	Kernel patches (Spectre variant 1)
03/27/18	Branchscope
05/22/18	Spectre variant 3
	Spectre variant 4
07/10/18	Bounds check bypass store

MELTDOWN

MELTDOWN

```
1 raise_exception();
2 // the next line is never reached
3 access(probe_array[data * 4096]);
```

→ Execute (3) out-of-order

→ Perform Cache-based side-channel attack

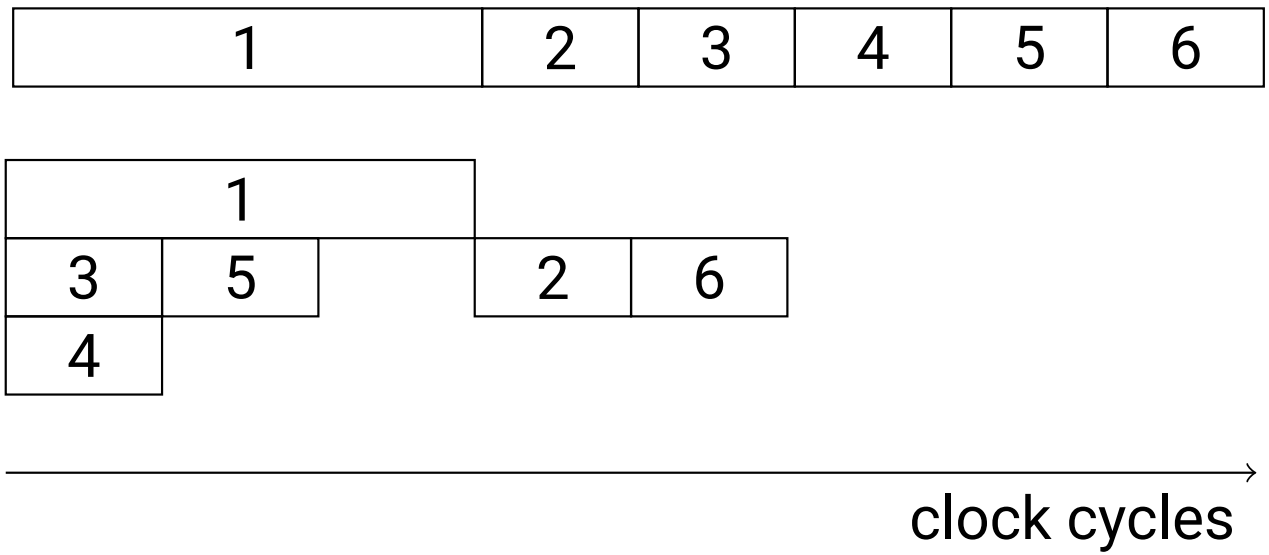
OUT-OF-ORDER EXECUTION

→ CPU design paradigm to increase performance

→ Increases “Instructions per clock cycle” (IPC)

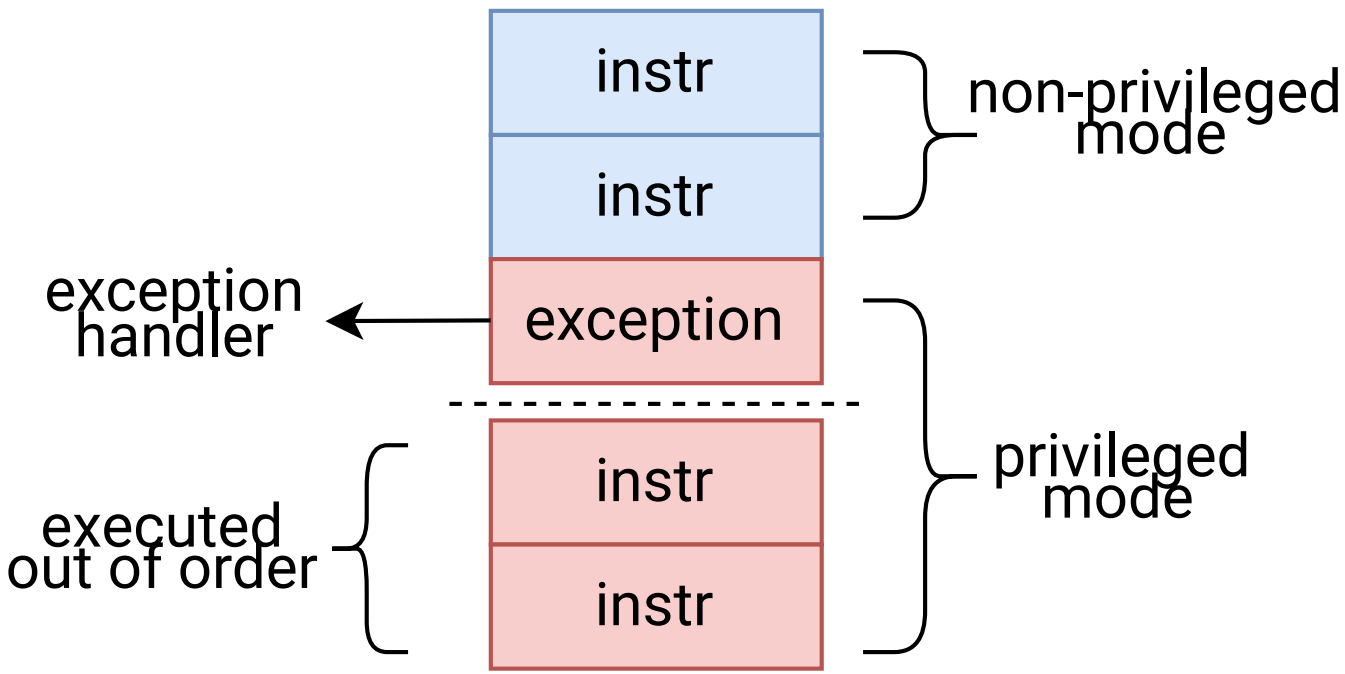
→ Does not preserve logical program order

OUT-OF-ORDER EXECUTION



MELTDOWN

```
1 raise_exception();
2 // the next line is never reached
3 access(probe_array[data * 4096]);
```

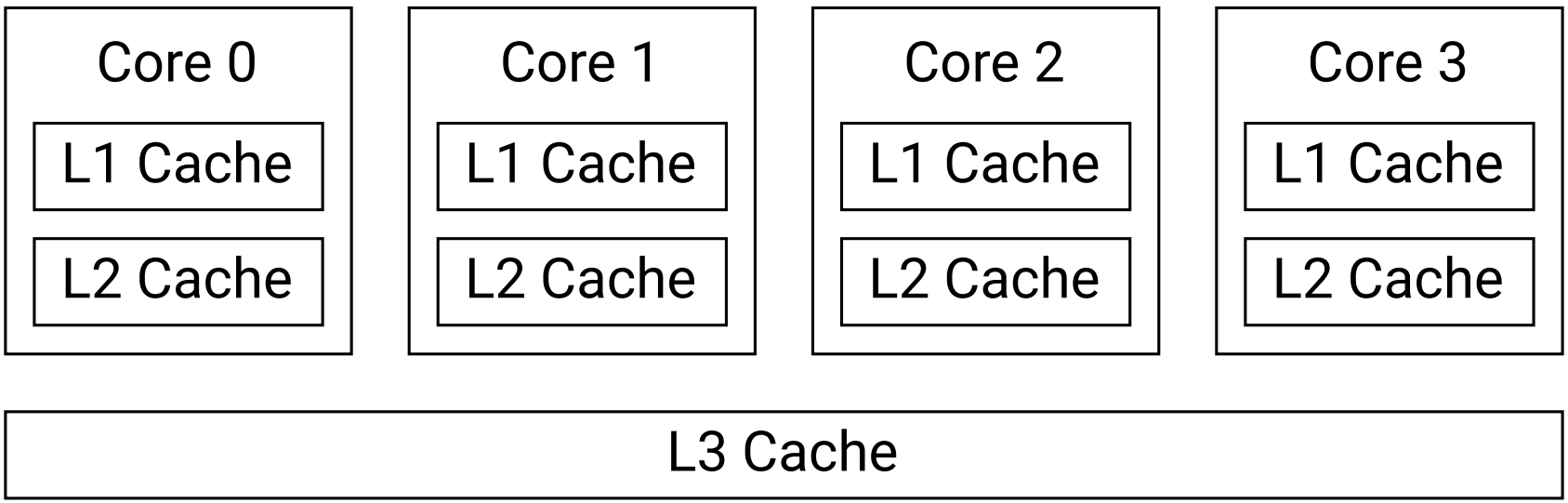


CACHE HIERARCHY

→ Small storages

→ Holds copies of recently used memory

→ Fast access time



CACHE-BASED SIDE-CHANNEL ATTACKS

- → Flush+Reload
- → Flush cache line in hierarchy
- → Wait for a specified time
- → Reload memory line
  - → Fast: Victim accessed memory
  - → Slow: Victim did not accessed memory
- → Spectre / Meltdown use Flush+Reload to access private data

MITIGATION: KAISER

→ Problem: Kernel mapped 1:1 into process page table

→ Solution: Split tables

→ It is not possible to access kernel space anymore

→ Merged with Linux kernel 4.15

SPECTRE

SPECTRE

→ Variant 1: bounds check bypass

→ Variant 2: branch target injection

SPECULATIVE EXECUTION

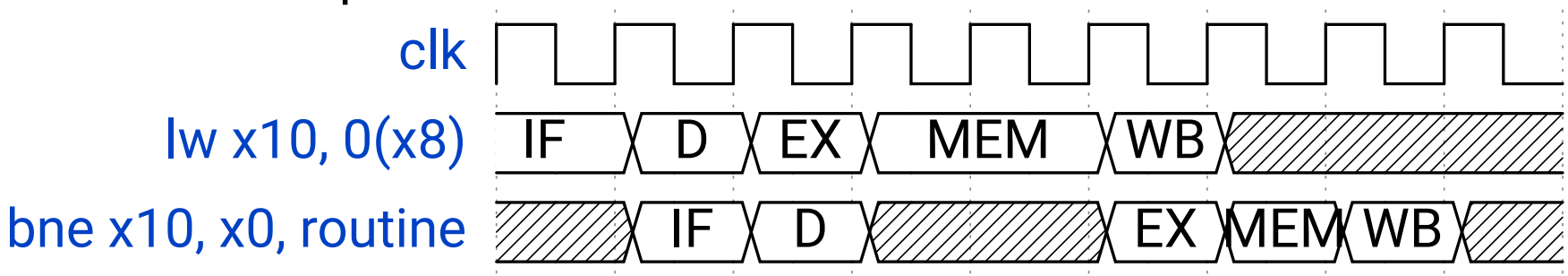
→ Branch prediction

→ Motivation?

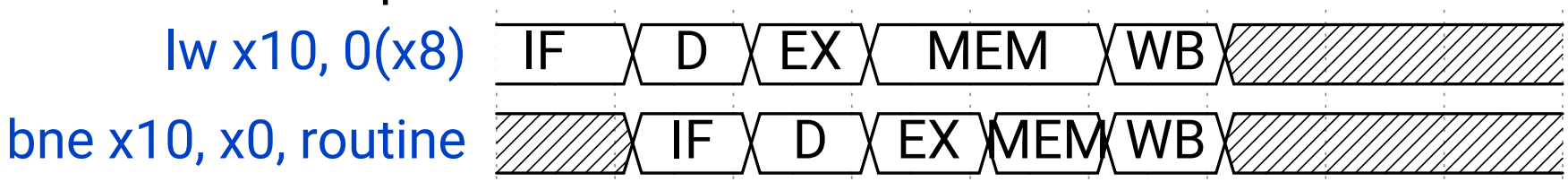
BRANCH PREDICTION

```
1 lw x10, 0(x8)
2 bne x10, x0, routine
3 j x1 // ra
```

no branch prediction:



with branch prediction:



BRANCH PREDICTION

→ If guessed wrong: Rollback instructions

→ But cache changes remain

SPECTRE

```
1 if (x < array1_size)
2     y = array2[array1[x] * 256];
```

→ Conditional jump gets mispredicted

→ `array1[x]` gets evaluated (because of condition)

→ Try to read `array2[`$\underbrace{\texttt{array1[x]}}_{\texttt{k}}$`* 256]`

→ Rollback instructions

→ Flush+Reload: Timing differences of `array2`.

MITIGATION: RETPOLINE

→ Problem: Indirect branches

→ Look in register `x` and jump to this address

```
1 jmp *%rax
```

```
1 call load_label
2 capture_ret_spec:
3     pause ; lfence
4     jmp capture_ret_spec
5 load_label:
6     mov %rax, (%rsp)
7     ret
```

MITIGATION: RETPOLINE

→ Recompilation necessary

→ Merged with GCC 7.3

→ „007“ - improved Retpoline with minimal overhead

RISC-V

THE HARE AND THE HEDGEHOG

Gustav Süs, 1855, gemeinfrei

RECAP '18

01/04/18	Meltdown
	Spectre variant 1
	Spectre variant 2
01/25/18	Retpoline (Spectre variant 2)
01/28/18	KAISER / KPTI
02/07/18	Kernel patches (Spectre variant 1)
03/27/18	Branchscope
05/22/18	Spectre variant 3
	Spectre variant 4
07/10/18	Bounds check bypass store

MITIGATION ≠ FIX

- → CPU is an integrated circuit:
  - → Only semiconductors can fix them
- → While there are no hardware fixes:
  - → Software mitigation to protect data

MITIGATION ≠ FIX

→ Developers chase the same hedgehog again and again

→ How can the hare win the race?

RISC-V

- → Open source Instruction set architecture (BSD license)
- → Developed at the University of California, Berkeley
- → Free Software implementations available
  - → https://github.com/freechipsproject/rocket-chip
  - → https://github.com/SpinalHDL/VexRiscv

RISC-V

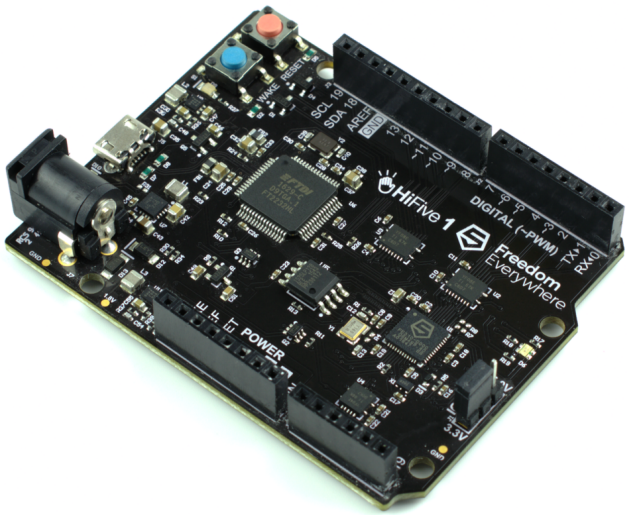
→ Open-source development at GitHub

→ Frameworks for formal verification (RVFI)

HIFIVE1

→ RISC-V based SoC

→ RISC-V CPU – rocket-chip, which is free software



© SiFive, Inc.

CONCLUSION

CONCLUSION

→ More and more CPU vulnerabilities

→ Huge time investment for mitigations

→ Free software RISC-V implementations as an alternative

LITERATURE


D. Gruss, M. Lipp, M. Schwarz, R. Fellner, C. Maurice, and S. Mangard.
KASLR is Dead: Long Live KASLR, volume 10379 LNCS of Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 161–176.
Springer-Verlag Italia, Italy, 2017.


P. Kocher, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz, and Y. Yarom.
Spectre attacks: Exploiting speculative execution.
ArXiv e-prints, Jan. 2018.


M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, S. Mangard, P. Kocher, D. Genkin, Y. Yarom, and M. Hamburg.
Meltdown.
ArXiv e-prints, Jan. 2018.


D. A. Patterson and J. L. Hennessy.
Computer Architecture: A Quantitative Approach.
Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1990.


Y. Yarom and K. Falkner.
Flush+reload: A high resolution, low noise, l3 cache side-channel attack.
In 23rd USENIX Security Symposium (USENIX Security 14), pages 719–732, San Diego, CA, 2014. USENIX Association.

TY

Thank you for listening.