Attaques sur la micro-architecture

Clémentine Maurice, CNRS, CRIStAL
@BloodyTangerine

24 Novembre 2022—8ème édition de la journée Sécu Min2Rien

Attacks on micro-architecture

- hardware usually modeled as an abstract layer behaving correctly

Attacks on micro-architecture

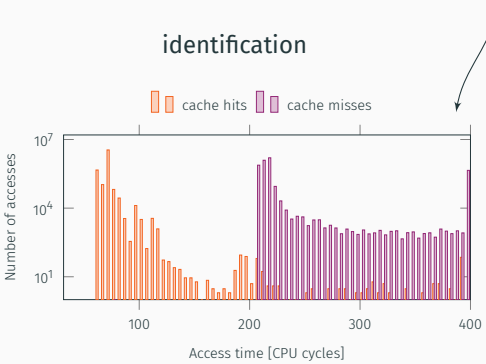
- hardware usually modeled as an abstract layer behaving correctly, but possible attacks

Attacks on micro-architecture

- hardware usually modeled as an abstract layer behaving correctly, but possible attacks
  - faults: bypassing software protections by causing hardware errors
  - side channels: observing side effects of hardware on computations

Attacks on micro-architecture

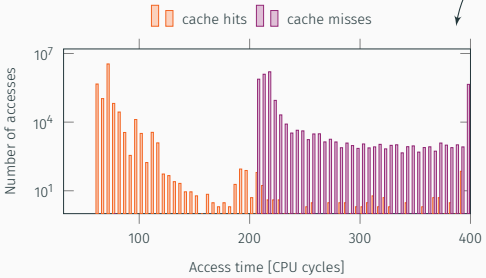
- hardware usually modeled as an abstract layer behaving correctly, but possible attacks
  - faults: bypassing software protections by causing hardware errors
  - side channels: observing side effects of hardware on computations



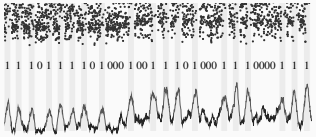
Attacks on micro-architecture

- hardware usually modeled as an abstract layer behaving correctly, but possible attacks
  - faults: bypassing software protections by causing hardware errors
  - side channels: observing side effects of hardware on computations

identification



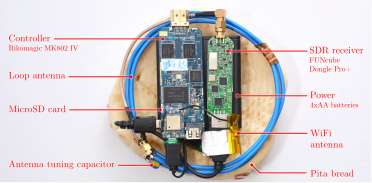
attack



- retrieving secret keys, keystroke timings
- bypassing OS security (ASLR)

Attacker model

Hardware-based attacks
a.k.a physical attacks



VS

Physical access to hardware
→ embedded devices

Co-located or remote attacker
→ complex systems

Outline

Focus on one fault attack: Rowhammer

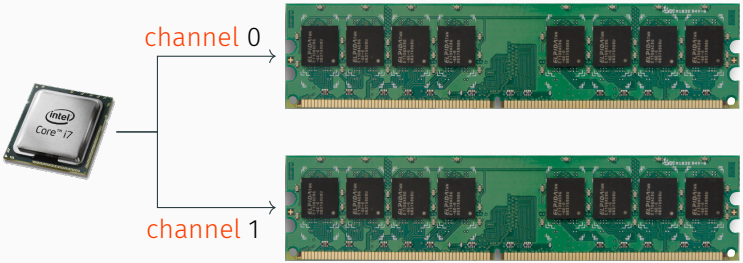
Overview of side-channel attacks

Conclusions

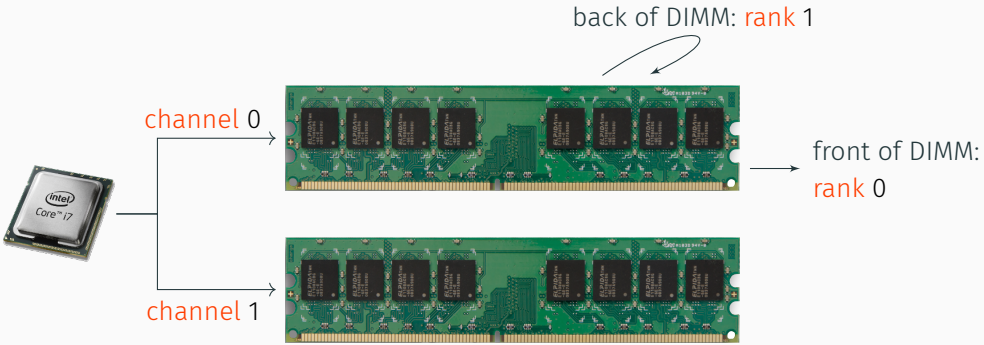
Focus on one fault attack: Rowhammer

DRAM organization

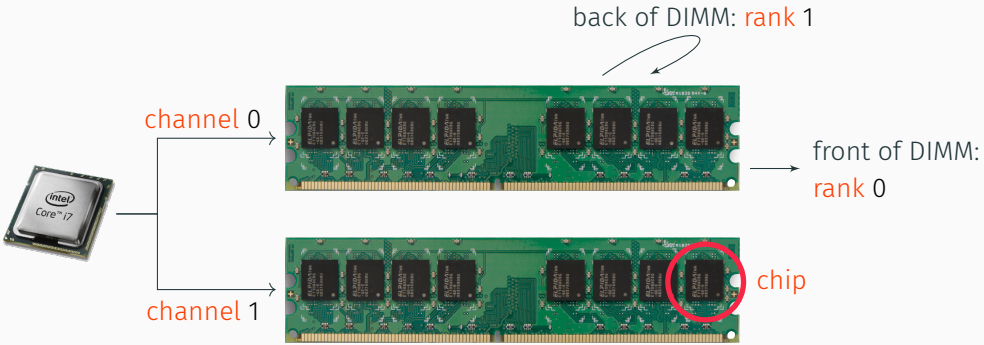
DRAM organization



DRAM organization



DRAM organization



DRAM organization

chip

bank 0

row 0
row 1
row 2
...
row 32767

row buffer

DRAM organization

chip

bank 0

row 0
row 1
row 2
...
row 32767

row buffer

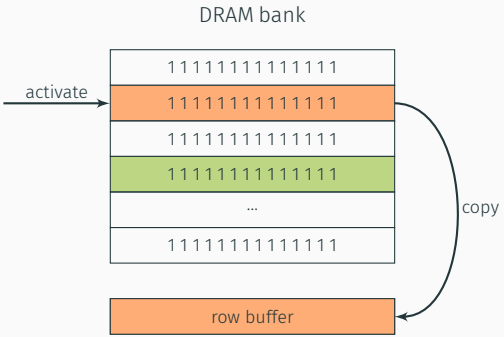
64k cells
1 capacitor,
1 transitor each

Rowhammer

"It's like breaking into an apartment by repeatedly slamming a neighbor's door until the vibrations open the door you were after" – Motherboard Vice

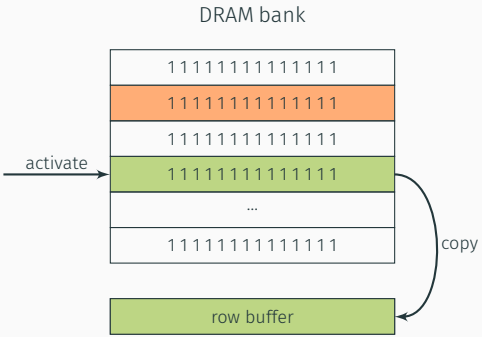
Rowhammer

"It's like breaking into an apartment by repeatedly slamming a neighbor's door until the vibrations open the door you were after" – Motherboard Vice



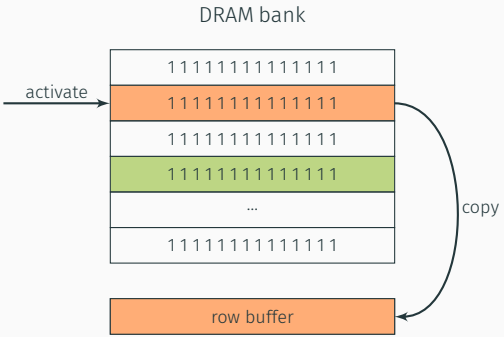
Rowhammer

"It's like breaking into an apartment by repeatedly slamming a neighbor's door until the vibrations open the door you were after" – Motherboard Vice



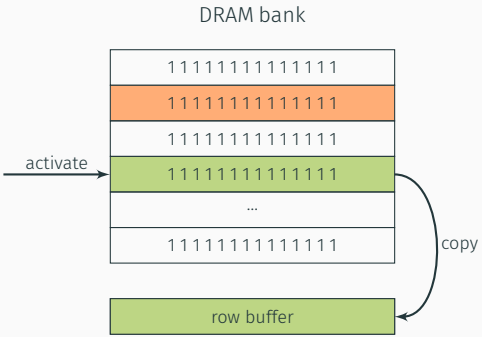
Rowhammer

"It's like breaking into an apartment by repeatedly slamming a neighbor's door until the vibrations open the door you were after" – Motherboard Vice



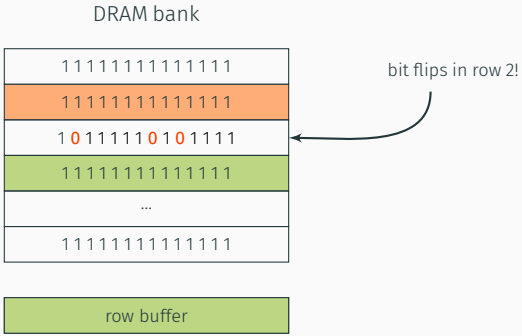
Rowhammer

"It's like breaking into an apartment by repeatedly slamming a neighbor's door until the vibrations open the door you were after" – Motherboard Vice



Rowhammer

"It's like breaking into an apartment by repeatedly slamming a neighbor's door until the vibrations open the door you were after" – Motherboard Vice



How to exploit random bit flips?

- Rowhammer was deemed non-exploitable and only a reliability issue

How to exploit random bit flips?

- Rowhammer was deemed non-exploitable and only a reliability issue
- bit flips are not random → highly reproducible flip pattern!
- ideas for exploitation
  1. bit flip in data structure, e.g., page table → privilege escalation
  2. bit flip in instruction opcode → privilege escalation
  3. bit flip in signature → fault-based cryptanalysis

Overview of side-channel attacks

From small optimizations...

- new microarchitectures yearly

From small optimizations...

- new microarchitectures yearly
- performance improvement $\approx 5\%$

From small optimizations...

- new microarchitectures yearly
- performance improvement $\approx 5\%$
- very small optimizations: caches, branch prediction...

... To microarchitectural side-channel attacks

- microarchitectural side channels come from these optimizations

... To microarchitectural side-channel attacks

- microarchitectural side channels come from these optimizations
- several processes are sharing microarchitectural components

… To microarchitectural side-channel attacks

- microarchitectural side channels come from these optimizations
- several processes are sharing microarchitectural components
- attacker infers information from a (vulnerable) victim process via hardware usage

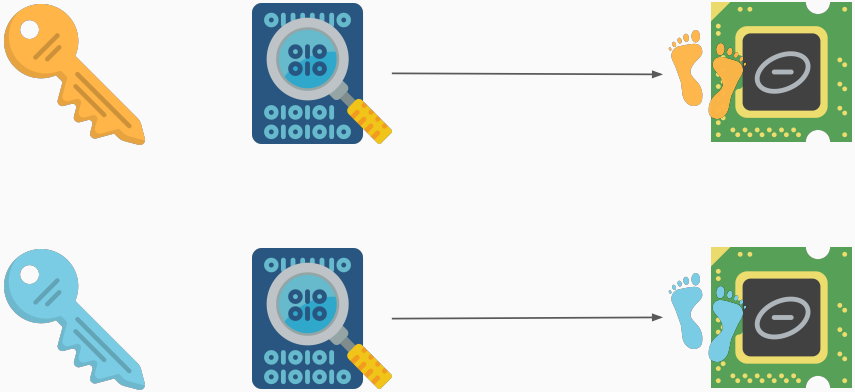
… To microarchitectural side-channel attacks

- microarchitectural side channels come from these optimizations
- several processes are sharing microarchitectural components
- attacker infers information from a (vulnerable) victim process via hardware usage
- pure-software attacks by unprivileged processes

... To microarchitectural side-channel attacks

- microarchitectural side channels come from these optimizations
- several processes are sharing microarchitectural components
- attacker infers information from a (vulnerable) victim process via hardware usage
- pure-software attacks by unprivileged processes
- sequences of benign-looking actions $\rightarrow$ hard to detect

Side-channel attacks



Side-channel attacks: Two faces of the same coin

Implementation

Hardware

Algorithm 1: Square-and-multiply exponentiation

Input: base b, exponent e, modulus n

Output: $b^e \mod n$

$X \leftarrow 1$

for $i \leftarrow bitlen(e)$ **downto** 0 **do**

  $X \leftarrow \text{multiply}(X, X)$

  if $e_i = 1$ **then**

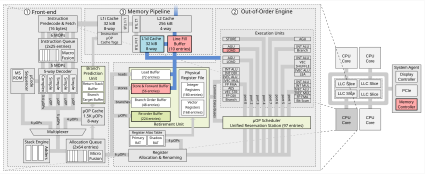
    $X \leftarrow \text{multiply}(X, b)$

  end

end

return X

&



Research questions

1. Which software implementation is vulnerable?

2. Which hardware component is vulnerable?

1. Which software implementation is vulnerable?

State of the art (more or less)

1. Spend too much time reading OpenSSL code
2. Find vulnerability
3. Exploit it manually using known side channel
   → e.g. CPU cache
4. Publish
5. goto step 1

For example: CVE-2016-0702, CVE-2016-2178, CVE-2016-7440, CVE-2016-7439, CVE-2016-7438, CVE-2018-0495, CVE-2018-0737, CVE-2018-10846, CVE-2019-9495, CVE-2019-13627, CVE-2019-13628, CVE-2019-13629, CVE-2020-16150

Canonical example: GnuPG 1.4.13 RSA square-and-multiply exponentiation

GnuPG version 1.4.13 (2013)

Algorithm 1: GnuPG 1.4.13 Square-and-multiply exponentiation

Input: base b, exponent e, modulus n

Output: $b^e \mod n$

$X \leftarrow 1$

for $i \leftarrow bitlen(e)$ **downto** 0 **do**

  $X \leftarrow \text{square}(X)$

  $X \leftarrow X \bmod n$

  if $e_i = 1$ **then**

    $X \leftarrow \text{multiply}(X, b)$

    $X \leftarrow X \bmod n$

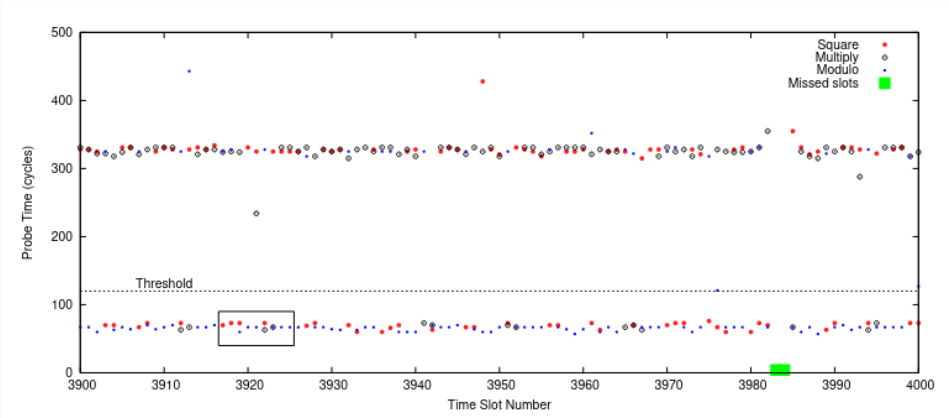
  end

end

return X

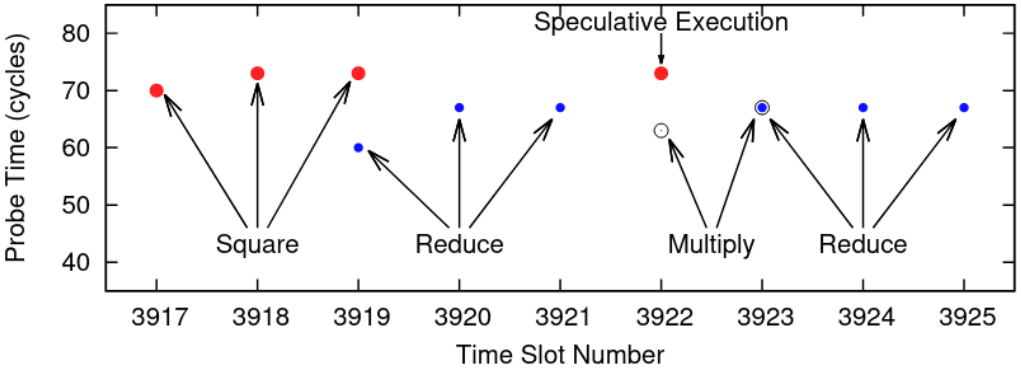
Attacking GnuPG 1.4.13 RSA exponentiation

- monitor the square and multiply functions with a cache side channel to recover the bits of the secret exponent



Attacking GnuPG 1.4.13 RSA exponentiation

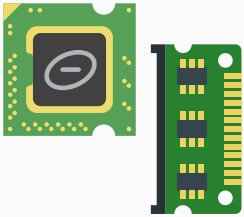
- monitor the square and multiply functions with a cache side channel to recover the bits of the secret exponent



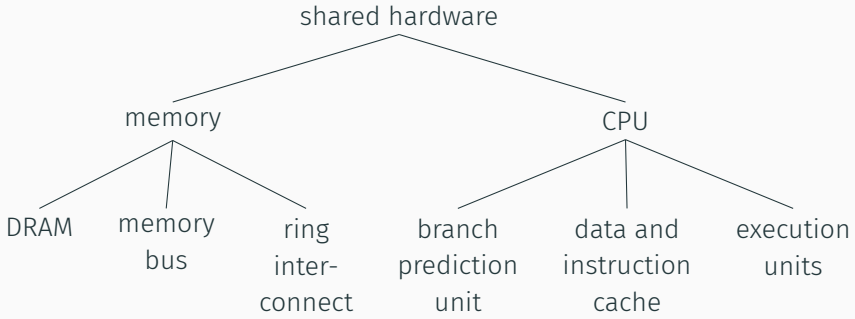
2. Which hardware component leaks information?

State of the art (more or less)

1. Spend too much time reading Intel manuals
2. Find weird behavior in corner cases
3. Exploit it using a known vulnerability
4. Publish
5. goto step 1



Shared hardware



Each component shared by two processes
is a potential micro-architectural side-channel vector

Hyper-threading: Same-core attacks

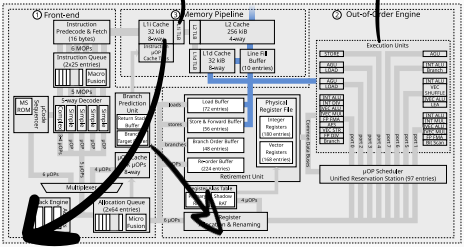
- threads sharing one core share resources: L1, L2 cache, branch predictor, TLB...

PortSmash

S&P'19

Translation leak-aside buffer

USENIX Sec'18



L1d, L1i, L2

cache attacks

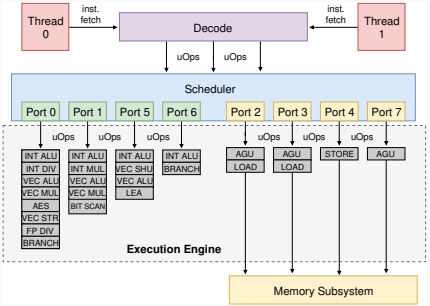
BSDCon'05, CT-RSA'06

Branch Prediction

CT-RSA'07

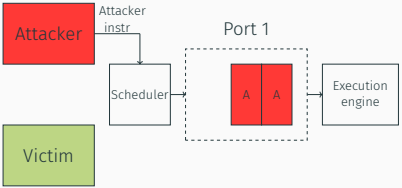
Background: Execution pipeline

- instructions are decomposed in uops to optimize Out-of-Order execution
- uops are dispatched to specialized execution units through CPU ports
- deterministic decomposition of instructions into uops



Port contention

No contention



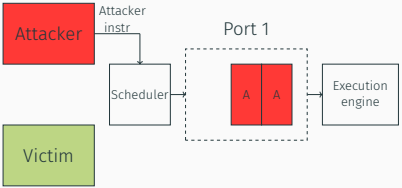
All attacker instructions are executed in a row

→ fast execution time

A. C. Aldaya et al. “Port Contention for Fun and Profit”. In: *S&P*. 2019.

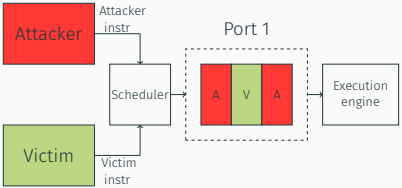
Port contention

No contention



All attacker instructions are executed in a row
→ fast execution time

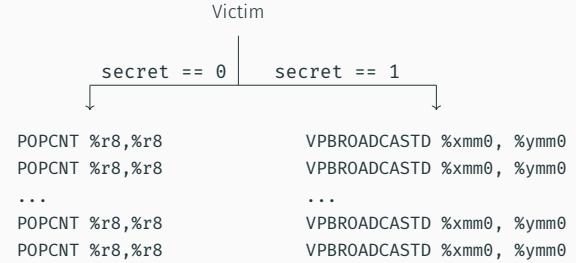
Contention



Victim instructions delay the attacker instructions
→ slow execution time

A. C. Aldaya et al. "Port Contention for Fun and Profit". In: *S&P*. 2019.

Port contention side-channel attack



Monitors port usage

Port contention side-channel attack

Victim

`secret == 0` `secret == 1`

```
POPCNT %r8,%r8              VPBROADCASTD %xmm0, %ymm0
POPCNT %r8,%r8              VPBROADCASTD %xmm0, %ymm0
...                         ...
POPCNT %r8,%r8              VPBROADCASTD %xmm0, %ymm0
POPCNT %r8,%r8              VPBROADCASTD %xmm0, %ymm0
```

Contention on Port 1

Secret is 0!

Port contention side-channel attack

Victim

`secret == 0`

```
POPCNT %r8,%r8
POPCNT %r8,%r8
...
POPCNT %r8,%r8
POPCNT %r8,%r8
```

`secret == 1`

```
VPBROADCASTD %xmm0, %ymm0
VPBROADCASTD %xmm0, %ymm0
...
VPBROADCASTD %xmm0, %ymm0
VPBROADCASTD %xmm0, %ymm0
```

Contention on Port 5

Secret is 1!

Port contention: applications

- end-to-end attack on a TLS server (OpenSSL 1.1.0h): recovers a P-384 ECDSA private key
  - → secret dependent on double-and-add operations of `ec_wNAF_mul` point multiplication
- SMoTherSpectre, a speculative code-reuse attack

A. C. Aldaya et al. "Port Contention for Fun and Profit". In: *S&P*. 2019.

A. Bhattacharyya et al. "SMoTherSpectre: Exploiting Speculative Execution through Port Contention". In: *CCS*. 2019.

Easy solution #1

Possible side channels using components shared by a core?

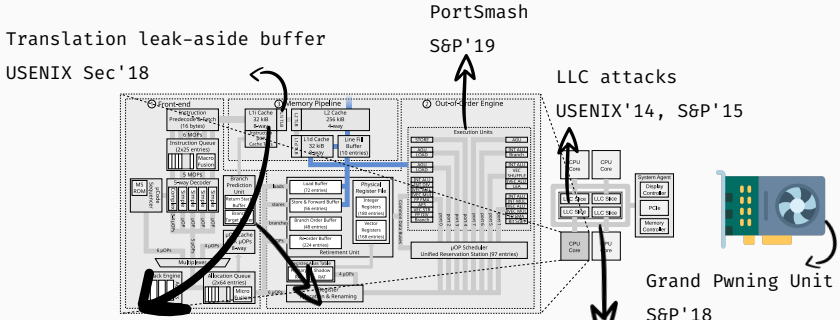
Easy solution #1

Possible side channels using components shared by a core?

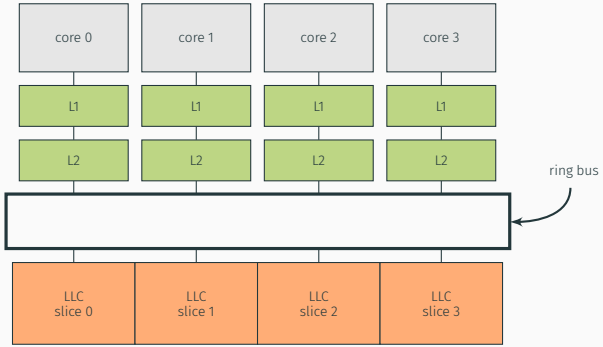
Stop sharing a core!

Cross-core attacks!

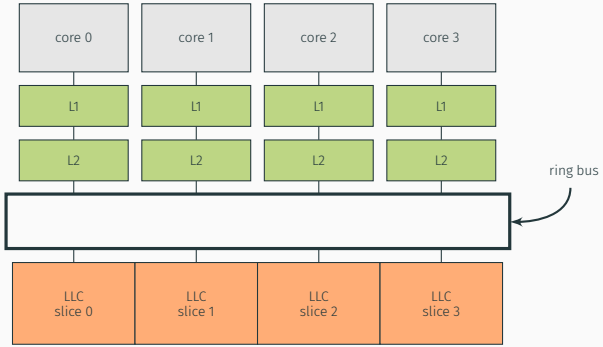
- cores also share resources: L3 cache, Ring Interconnect, GPU...



Caches on Intel CPUs

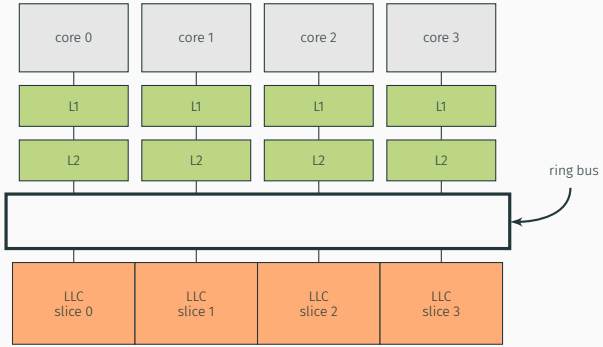


Caches on Intel CPUs



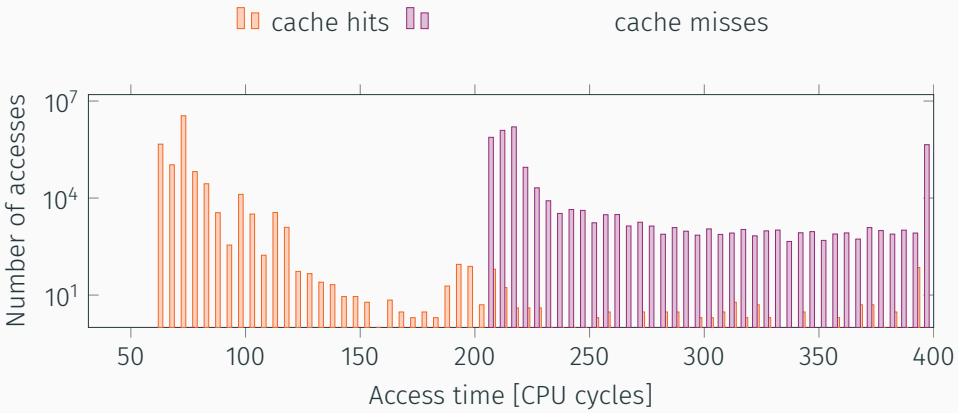
- L1 and L2 are private

Caches on Intel CPUs



- L1 and L2 are private
- last-level cache
  - divided in slices
  - shared across cores
  - inclusive

Cache timing differences



From theoretical to practical cache attacks

- first theoretical attack in 1996 by Kocher
- first practical attack on RSA in 2005 by Percival, on AES in 2006 by Osvik et al.
- renewed interest for the field in 2014 after Flush+Reload by Yarom and Falkner
- even more interest in 2018 after the disclosure of Spectre and Meltdown

---

P. C. Kocher. “Timing Attacks on Implementations of Diffe-Hellman, RSA, DSS, and Other Systems”. In: *Crypto’96*. 1996.

C. Percival. “Cache missing for fun and profit”. In: *Proceedings of BSDCan*. 2005.

D. A. Osvik, A. Shamir, and E. Tromer. “Cache Attacks and Countermeasures: the Case of AES”. In: *CT-RSA 2006*. 2006.

Y. Yarom and K. Falkner. “Flush+Reload: a High Resolution, Low Noise, L3 Cache Side-Channel Attack”. In: *USENIX Security Symposium*. 2014.

P. Kocher et al. “Spectre Attacks: Exploiting Speculative Execution”. In: *S&P*. 2019.

M. Lipp et al. “Meltdown: Reading Kernel Memory from User Space”. In: *USENIX Security Symposium*. 2018.

Cache attacks techniques

- two (main) techniques
  1. Flush+Reload (Gullasch et al., Osvik et al., Yarom et al.)
  2. Prime+Probe (Percival, Osvik et al., Liu et al.)
- exploitable on x86 and ARM

D. Gullasch, E. Bangerter, and S. Krenn. "Cache Games – Bringing Access-Based Cache Attacks on AES to Practice". In: *S&P'11*. 2011.

Y. Yarom and K. Falkner. "Flush+Reload: a High Resolution, Low Noise, L3 Cache Side-Channel Attack". In: *USENIX Security Symposium*. 2014.

D. A. Osvik, A. Shamir, and E. Tromer. "Cache Attacks and Countermeasures: the Case of AES". In: *CT-RSA 2006*. 2006.

C. Percival. "Cache missing for fun and profit". In: *Proceedings of BSDCan*. 2005.

F. Liu et al. "Last-Level Cache Side-Channel Attacks are Practical". In: *S&P'15*. 2015.

Flush+Reload: Applications

- side channel attacks on cryptographic primitives:
  - RSA: 96.7% of secret key bits in a single signature
  - AES: full key recovery in 30000 dec. (a few seconds)
- attacks against pseudorandom number generators
- attacks against RSA key generation
- revival of Bleichenbacher attacks on TLS

Y. Yarom and K. Falkner. "Flush+Reload: a High Resolution, Low Noise, L3 Cache Side-Channel Attack". In: *USENIX Security Symposium*. 2014.

B. Gülmezoglu et al. "A Faster and More Realistic Flush+Reload Attack on AES". In: *COSADE*. 2015.

S. Cohney et al. "Pseudorandom Black Swans: Cache Attacks on CTR_DRBG". In: *S&P*. 2020.

A. C. Aldaya et al. "Cache-Timing Attacks on RSA Key Generation". In: *TCHES* (2019).

E. Ronen et al. "The 9 Lives of Bleichenbacher's CAT: New Cache ATtacks on TLS Implementations". In: *S&P*. 2019.

Prime+Probe: Applications

- cross-VM side channel attacks on crypto implementations:
  - El Gamal (sliding window): full key recovery in 12 min.
- covert channels between virtual machines in the cloud
- tracking user behavior in the browser, in JavaScript

F. Liu et al. “Last-Level Cache Side-Channel Attacks are Practical”. In: *S&P'15*. 2015.

C. Maurice et al. “Hello from the Other Side: SSH over Robust Cache Covert Channels in the Cloud”. In: *NDSS'17*. 2017.

Y. Oren et al. “The Spy in the Sandbox: Practical Cache Attacks in JavaScript and their Implications”. In: *CCS'15*. 2015.

Easy solution #2

Possible side channels using components shared by a CPU?

Easy solution #2

Possible side channels using components shared by a CPU?

Stop sharing a CPU!?

Cross-CPU attacks!

- CPUs also share resources: DRAM

Translation leak-aside buffer

USENIX Sec'18

PortSmash

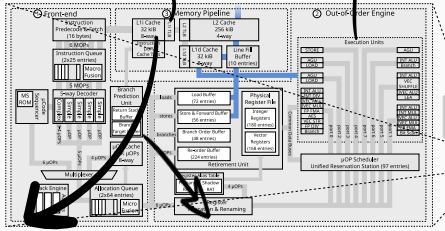
S&P'19

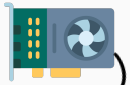
LLC attacks

USENIX'14, S&P'15

DRAMA

USENIX Sec'16





Grand Pwning Unit

S&P'18

L1d, L1i, L2

cache attacks

BSDCon'05, CT-RSA'06

Branch Prediction

CT-RSA'07

Lord of the Ring(s)

USENIX Sec'21

Conclusions

Conclusions

- first paper by Kocher in 1996: 25 years of research in this area

Conclusions

- first paper by Kocher in 1996: 25 years of research in this area
- domain still in expansion: increasing number of papers published since 2015

Conclusions

- first paper by Kocher in 1996: 25 years of research in this area
- domain still in expansion: increasing number of papers published since 2015
- any shared component is a potential side-channel vector

Conclusions

- first paper by Kocher in 1996: 25 years of research in this area
- domain still in expansion: increasing number of papers published since 2015
- any shared component is a potential side-channel vector
- it's **really** hard not to share a component

Conclusions

- first paper by Kocher in 1996: 25 years of research in this area
- domain still in expansion: increasing number of papers published since 2015
- any shared component is a potential side-channel vector
- it's **really** hard not to share a component
- micro-architectural attacks require a low-level understanding and control over the components

Conclusions

- first paper by Kocher in 1996: 25 years of research in this area
- domain still in expansion: increasing number of papers published since 2015
- any shared component is a potential side-channel vector
- it's **really** hard not to share a component
- micro-architectural attacks require a low-level understanding and control over the components
- how to prevent attacks based on performance optimizations without removing performance?

Thank you!

Contact

clementine.maurice@inria.fr

@BloodyTangerine

Attaques sur la micro-architecture

Clémentine Maurice, CNRS, CRIStAL
@BloodyTangerine

24 Novembre 2022—8ème édition de la journée Sécu Min2Rien