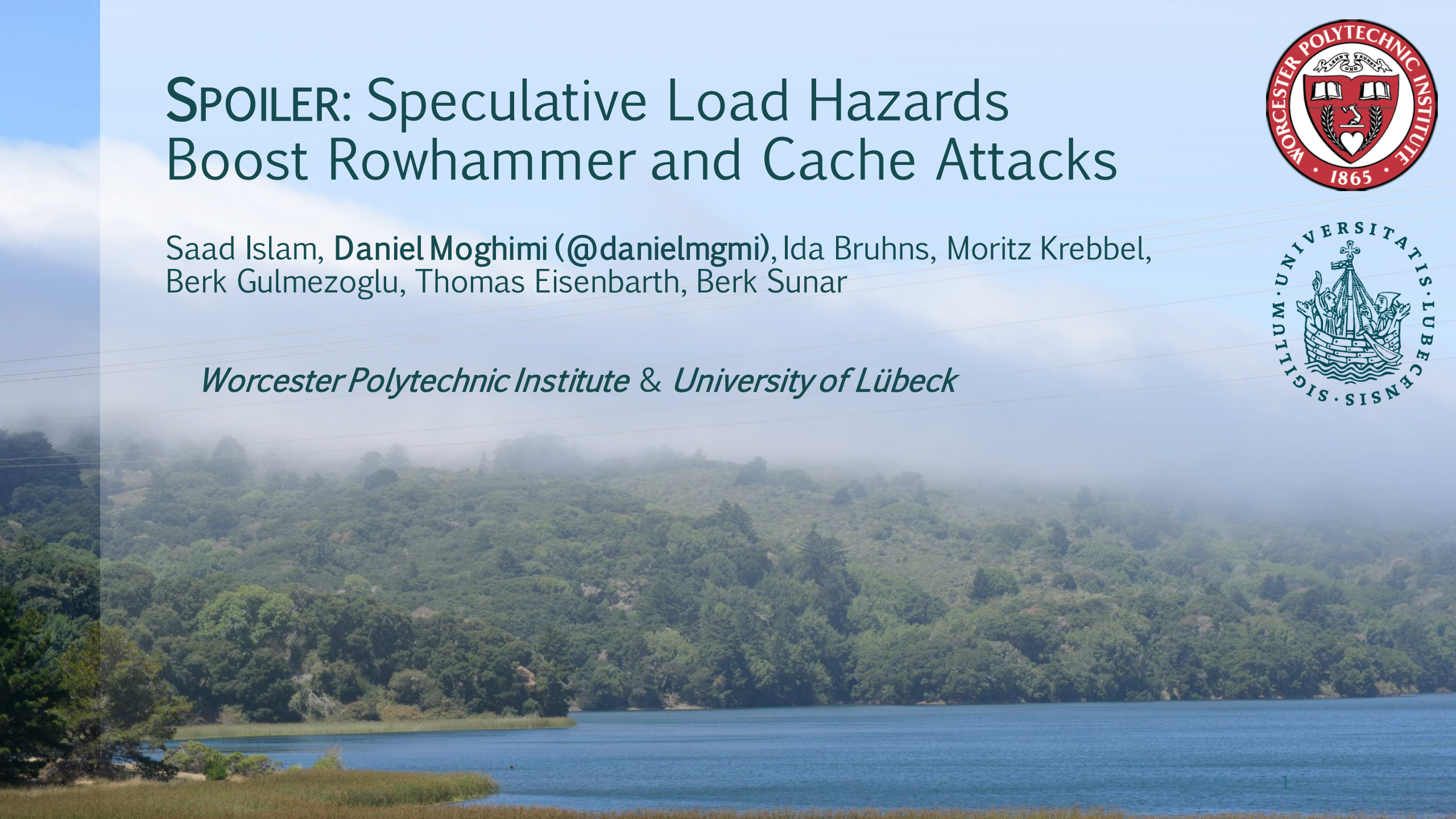
SPOILER: Speculative Load Hazards Boost Rowhammer and Cache Attacks

Saad Islam, **Daniel Moghimi (@danielmgmi)**, Ida Bruhns, Moritz Krebbel, Berk Gulmezoglu, Thomas Eisenbarth, Berk Sunar

Worcester Polytechnic Institute & *University of Lübeck*

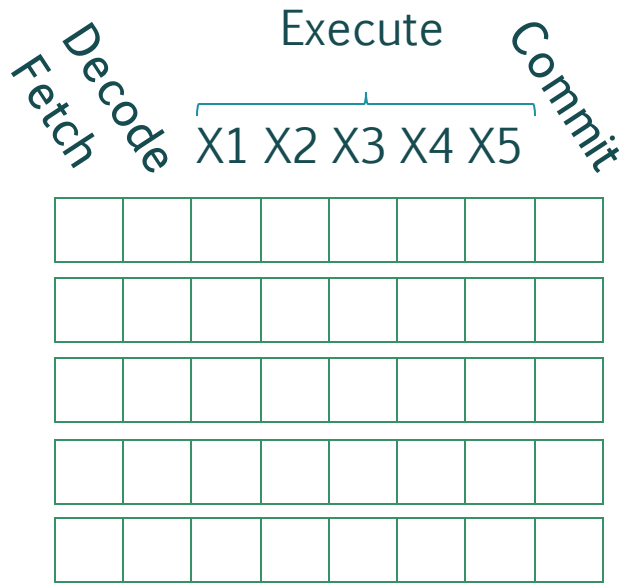


CPU Optimization?

- Branch Prediction
- Cache and internal buffers
- Speculate the Speculations??!
  - "speculative prefetching“
  - "speculatively scheduled operation“
  - "speculative execution event counter“
  - "speculative memory accesses“
  - "speculative load instruction"

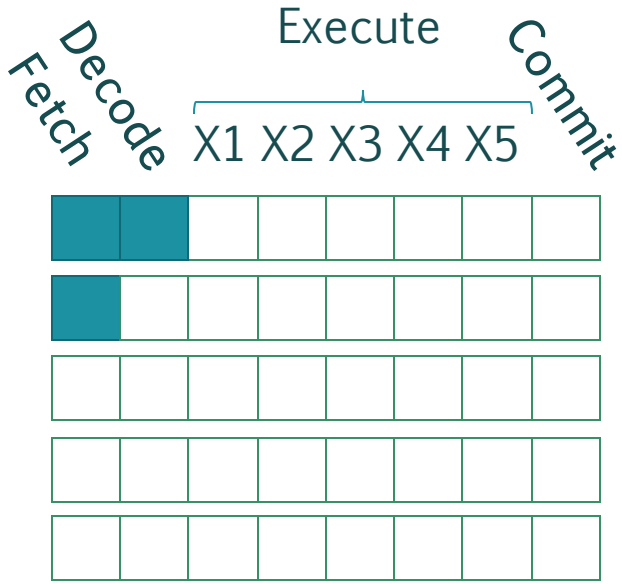
Speculative Load

```
store   a → X
store   b → Y
store   c → Z
load    d ← W
inc     d
```



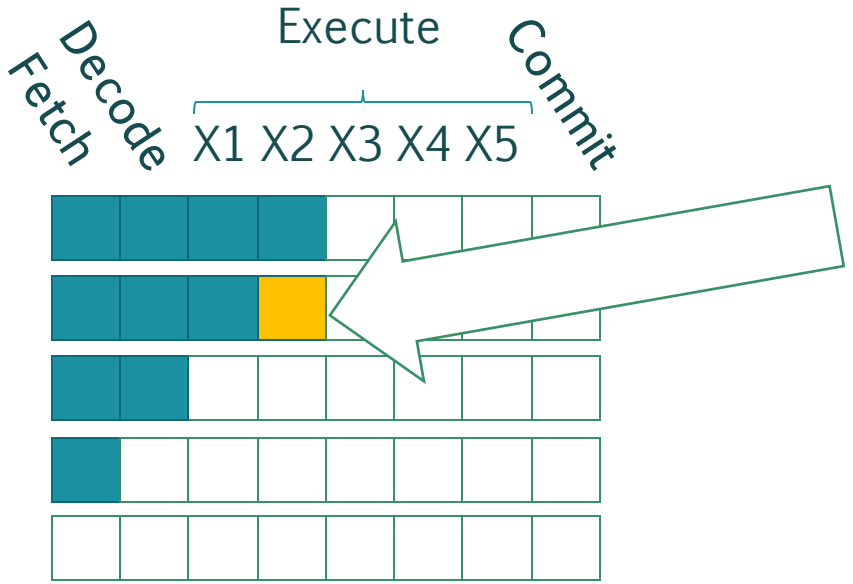
Speculative Load

```
store   a → X
store   b → Y
store   c → Z
load    d ← W
inc     d
```

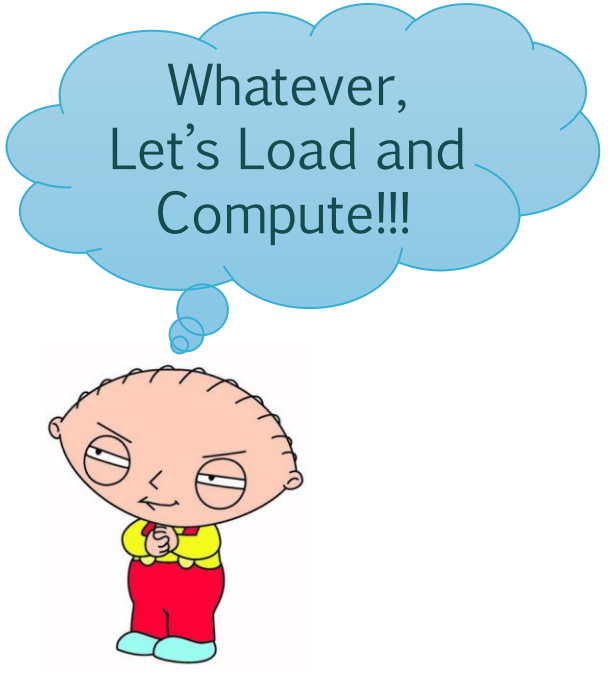


Speculative Load

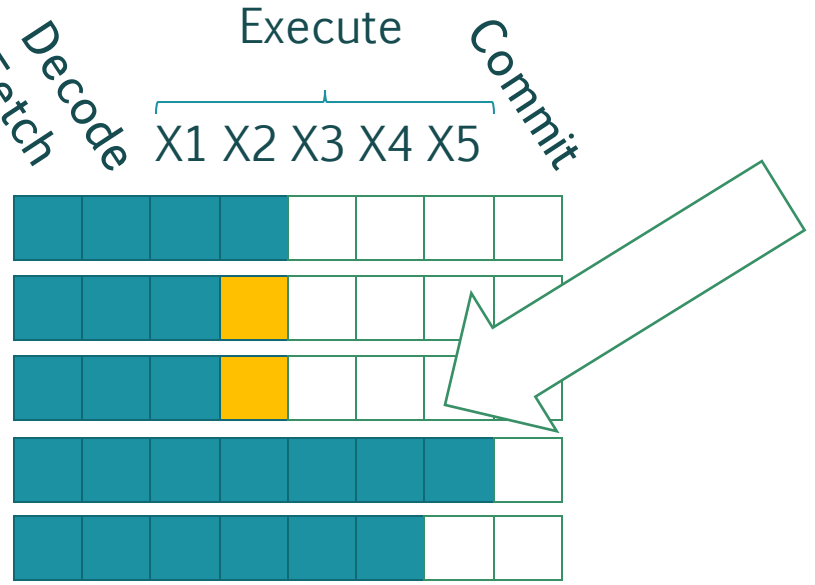

```
store   a → X
store   b → Y
store   c → Z
load    d ← W
inc     d
```



Speculative Load

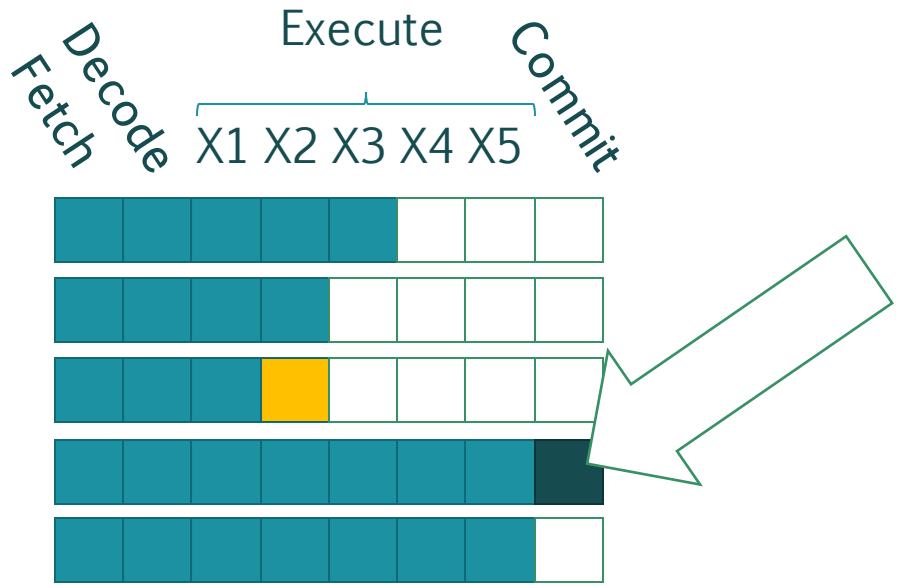


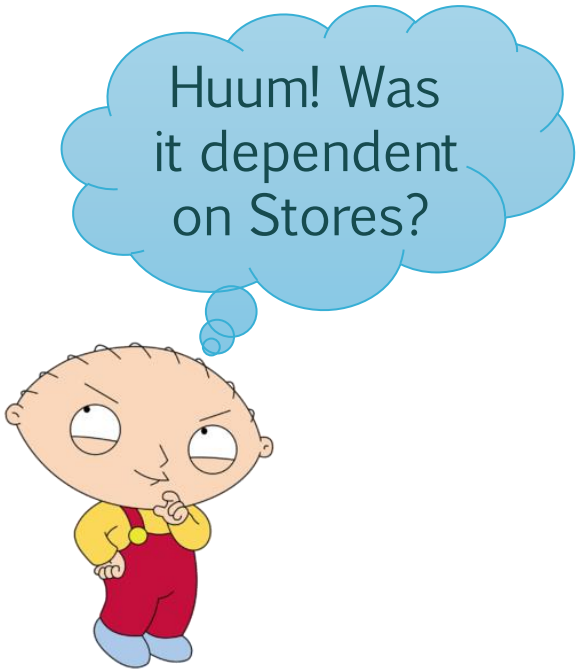

```
store   a → X
store   b → Y
store   c → Z
load    d ← W
inc     d
```



Speculative Load

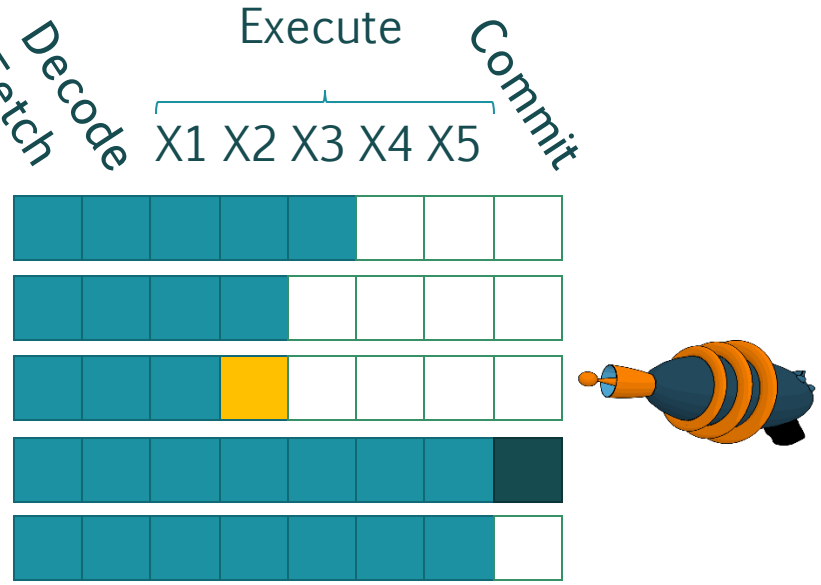
```
store   a → X
store   b → Y
store   c → Z
load    d ← W
inc     d
```



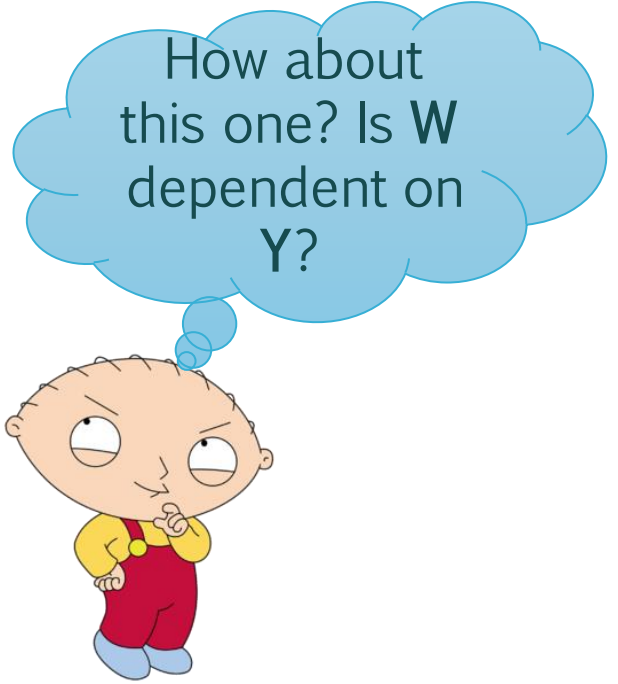


Speculative Load

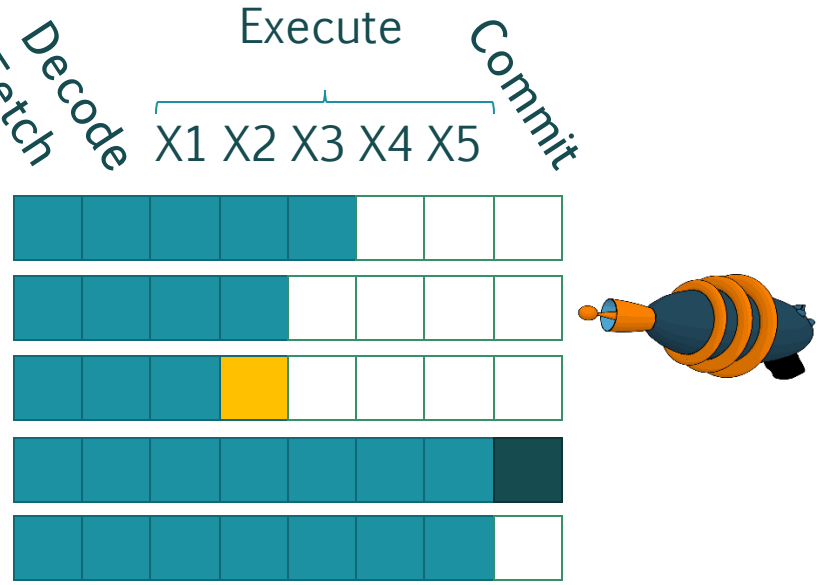

```
store   a → X
store   b → Y
store   c → Z
load    d ← W
inc     d
```



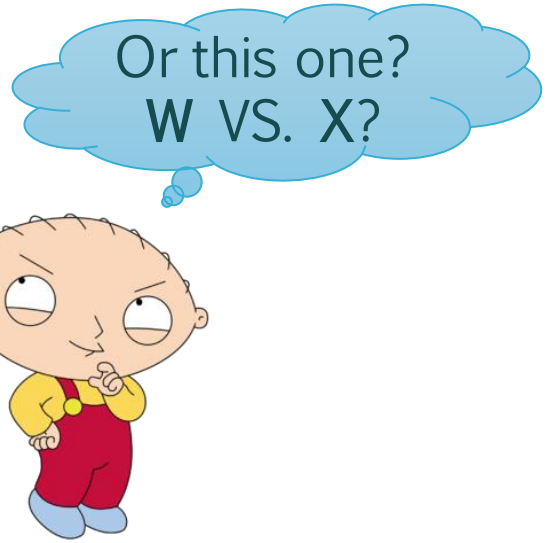
Speculative Load



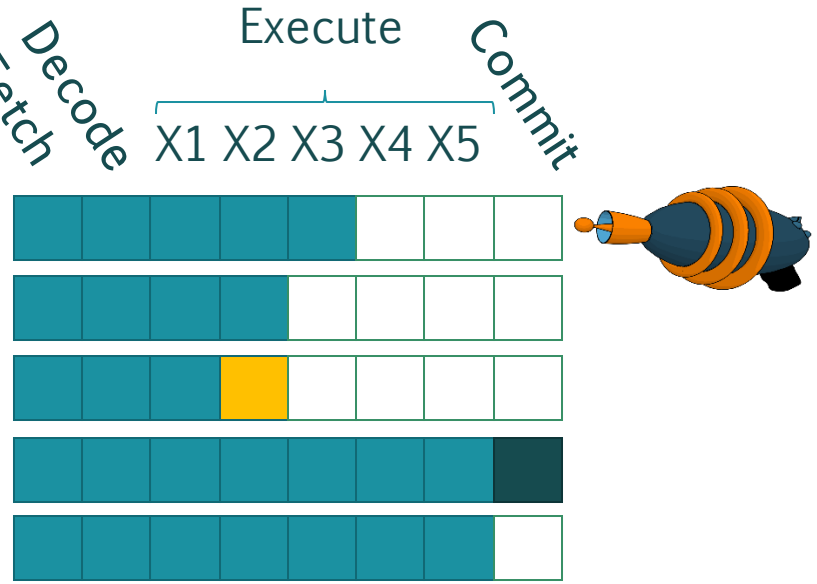

```
store   a → X
store   b → Y
store   c → Z
load    d ← W
inc     d
```



Speculative Load

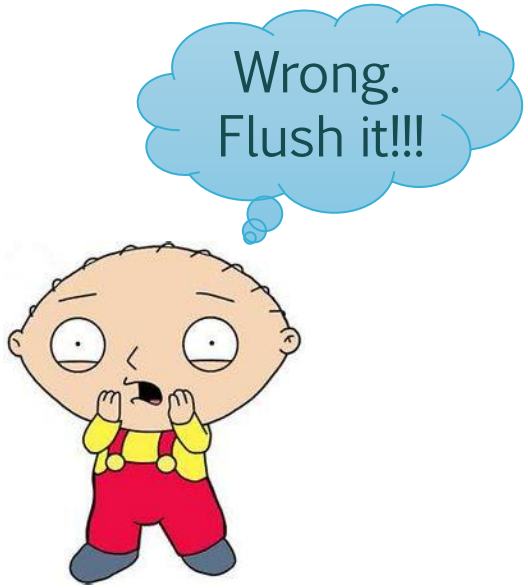



```
store   a → X
store   b → Y
store   c → Z
load    d ← W
inc     d
```



Speculative Load

```
store   a → X
store   b → Y
store   c → Z
load    d ← W
inc     d
```

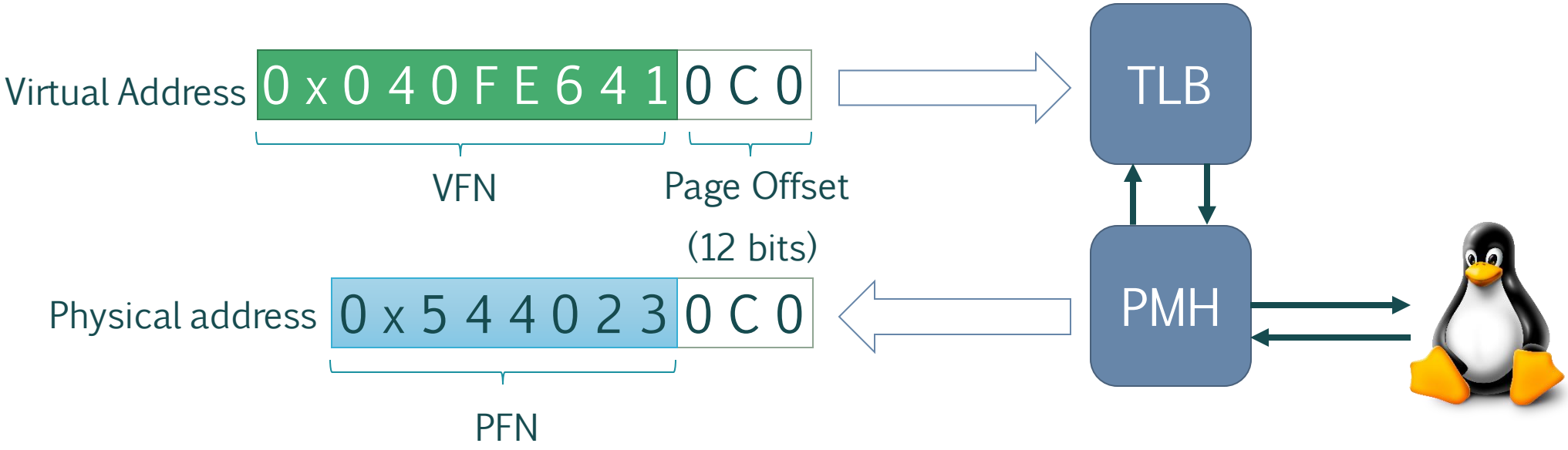


Virtual & Physical Addresses
What are store and load addresses?

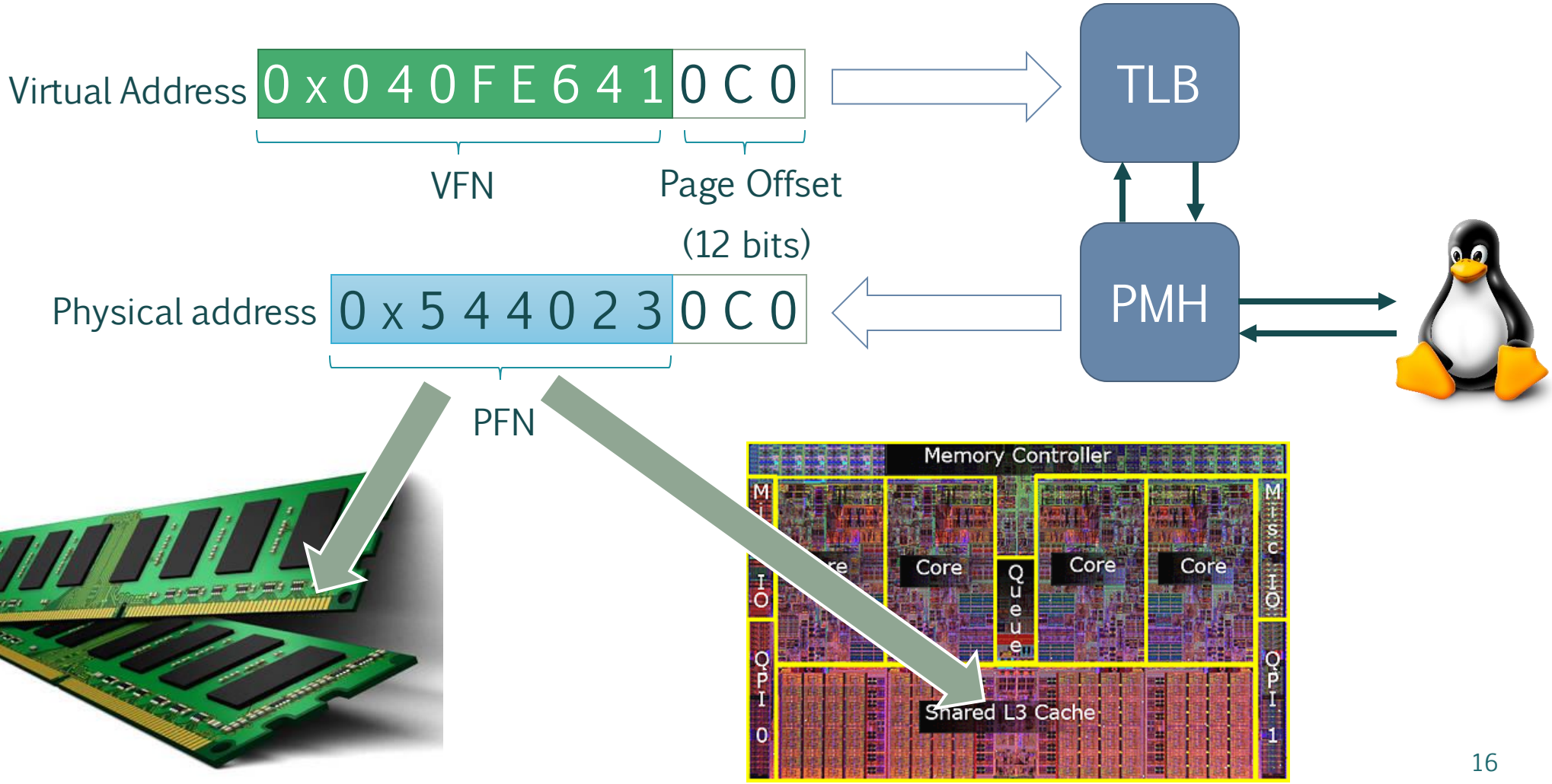
Virtual & Physical Addresses
What are store and load addresses?

Virtual & Physical Addresses
What are store and load addresses?

Virtual & Physical Addresses
What are store and load addresses?

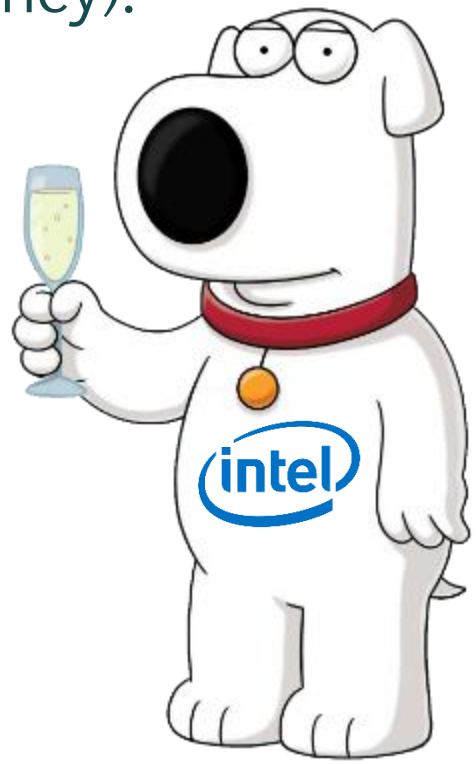


Virtual & Physical Addresses
What are store and load addresses?



Design Challenges?

- Loads are executed out-of-order and speculatively to avoid performance loss.
- Load may be dependent on preceding stores (dependency).
- Dependency check is difficult:
  - Virtual addresses may be aliased.
  - Physical addresses are not available immediately.
  - Stores may stay in-flight for a while.
    - We can't wait for them to succeed.
    - Can we forward the data from the store to the load?



SPOILER

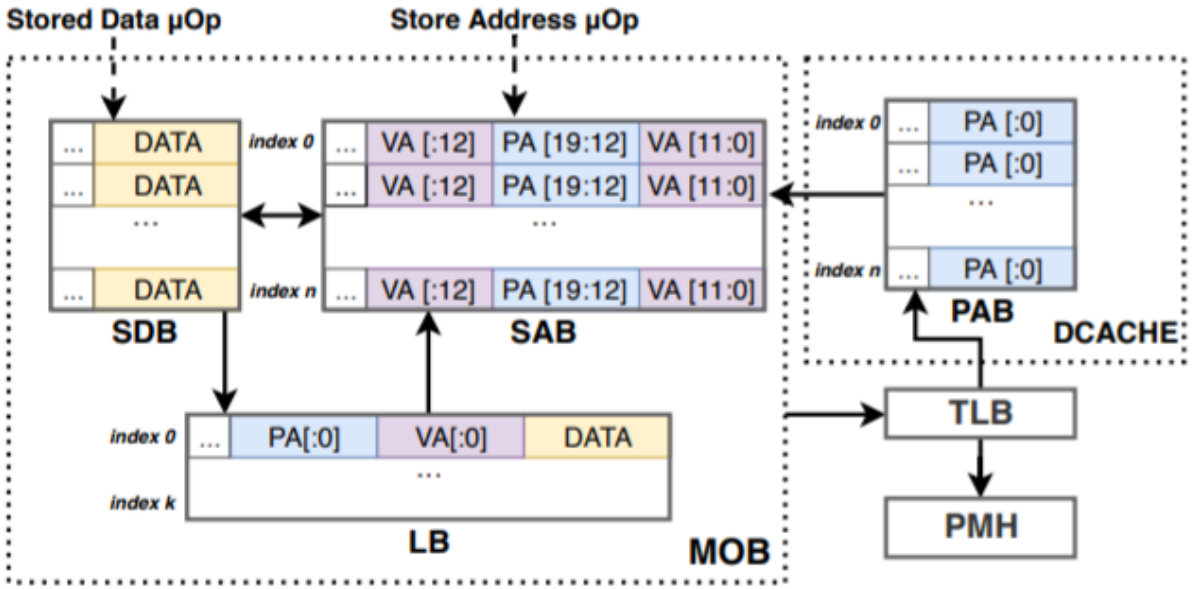
US 7,603,527 B2 RESOLVING FALSE DEPENDENCIES OF SPECULATIVE LOAD INSTRUCTIONS

"an operation X may determine whether the lower portion of the virtual address of a speculative load instruction matches the lower portion of virtual addresses of older store operations" LoosnetCheck

"an operation Y may determine whether the upper portion of the virtual address of the speculative load matches the upper portion of virtual addresses of older store" "If there is a hit at operation Y then the load may be blocked"

"in an embodiment, the load instruction may have its input data forwarded from the store operation from which the load instruction depends at operation" StoreForwarding

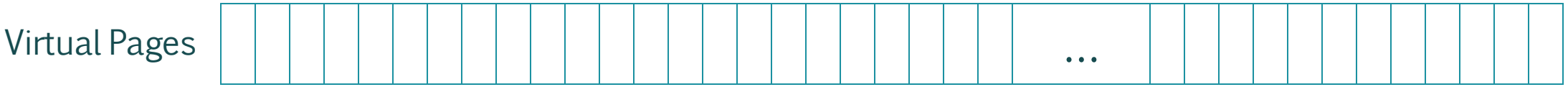
"If there is a hit at operation X and a miss at operation Y, ... the physical addresses of the load and the store may be compared at an operation Z" "In one embodiment, if there is a hit at operation X and ***the physical address of the load or the store operation is not valid, the physical address check at operation Z may be considered as a hit"*** *"In some embodiments,* ***the physical address check at operation Z may use a partial physical address****, e.g., base on data stored in the SAB. This makes the checking at operation Z conservative. Accordingly, in some embodiments,* ***a match may occur on a partial address and block****..."* FinenetCheck



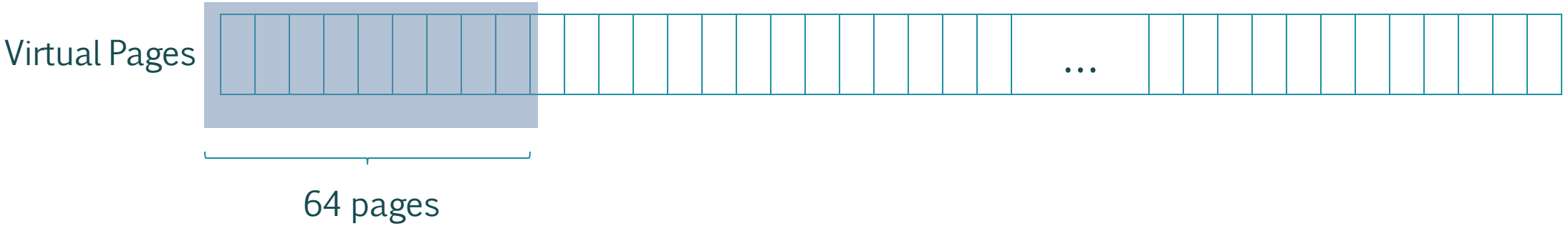
SPOILER Attack

Dependency Resolution

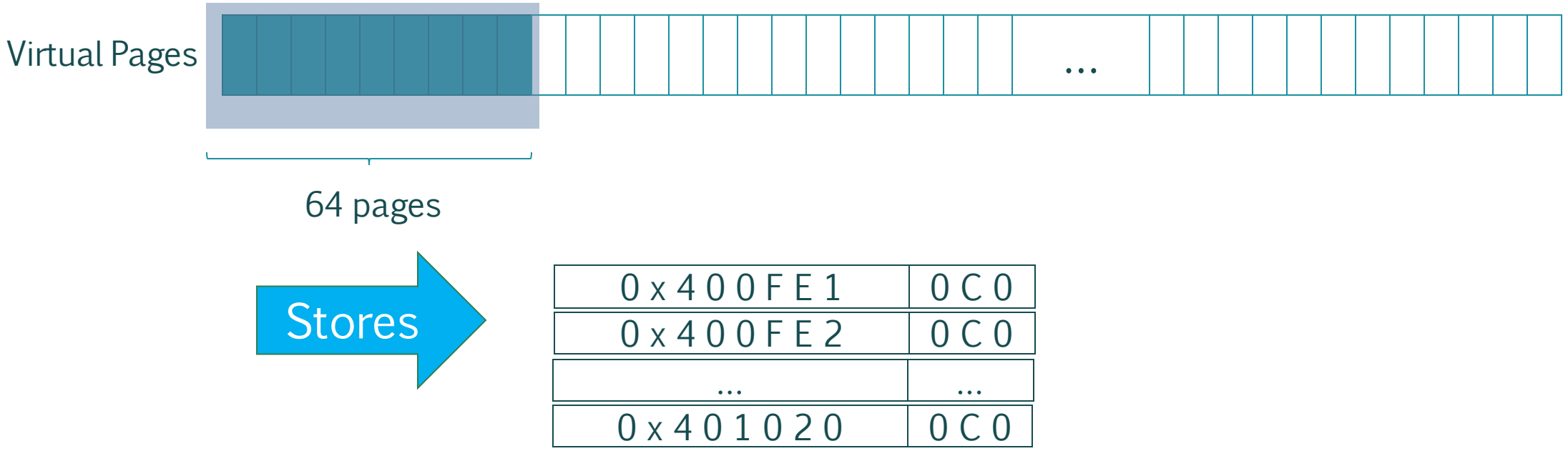
SPOILER Attack



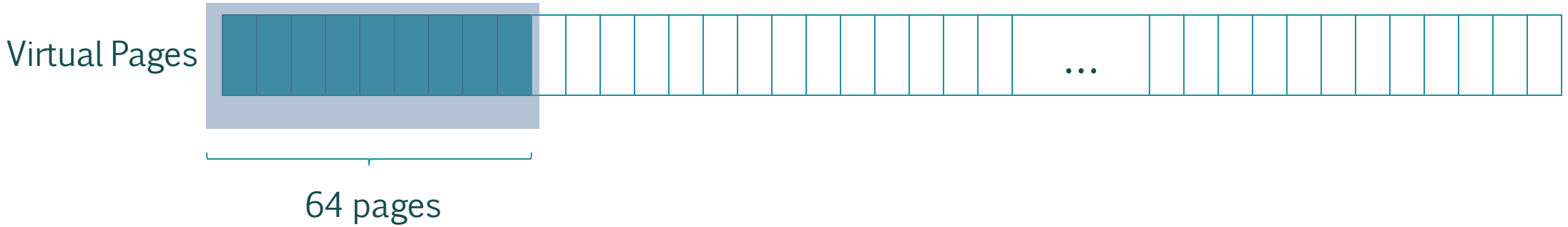
SPOILER Attack



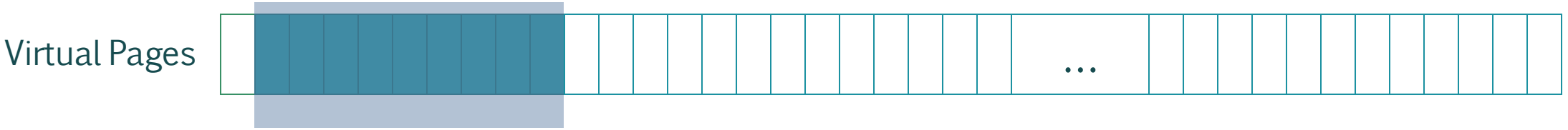
SPOILER Attack



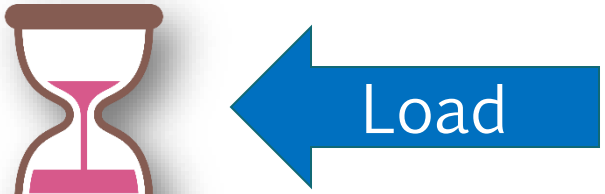
SPOILER Attack



SPOILER Attack

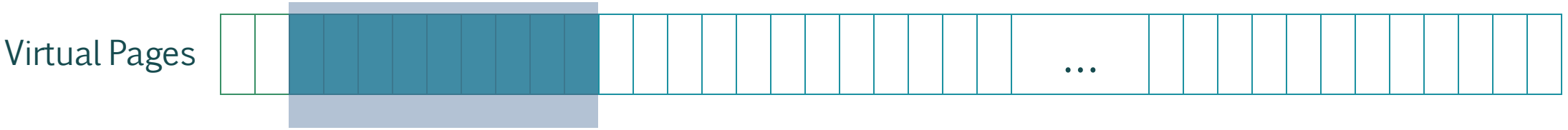


0x400FE2	0C0
0x400FE3	0C0
...	...
0x401021	0C0

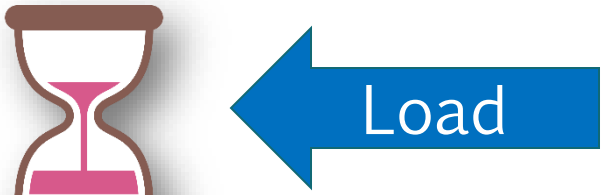


0x4F1234	0C0

SPOILER Attack

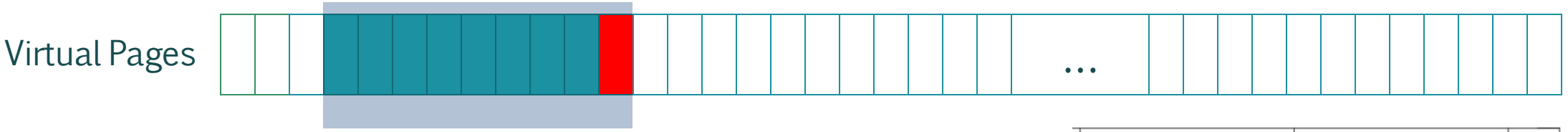


0x400FE3	0C0
0x400FE4	0C0
...	...
0x401022	0C0



0x4F1234	0C0

SPOILER Attack



Virtual Addresses

Stores

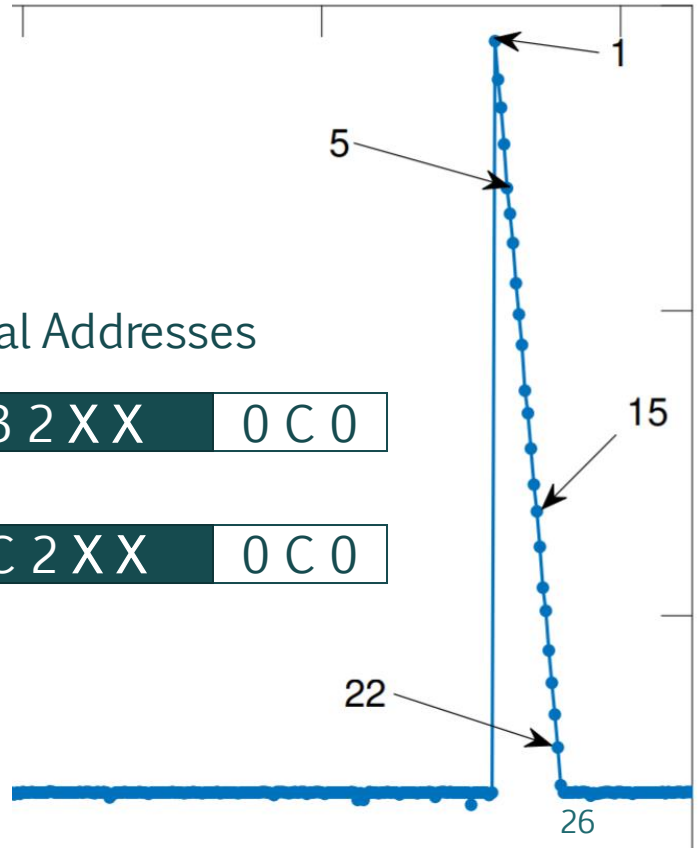
0x400FE4	0C0
0x400FE5	0C0
...	...
0x401023	0C0

Load

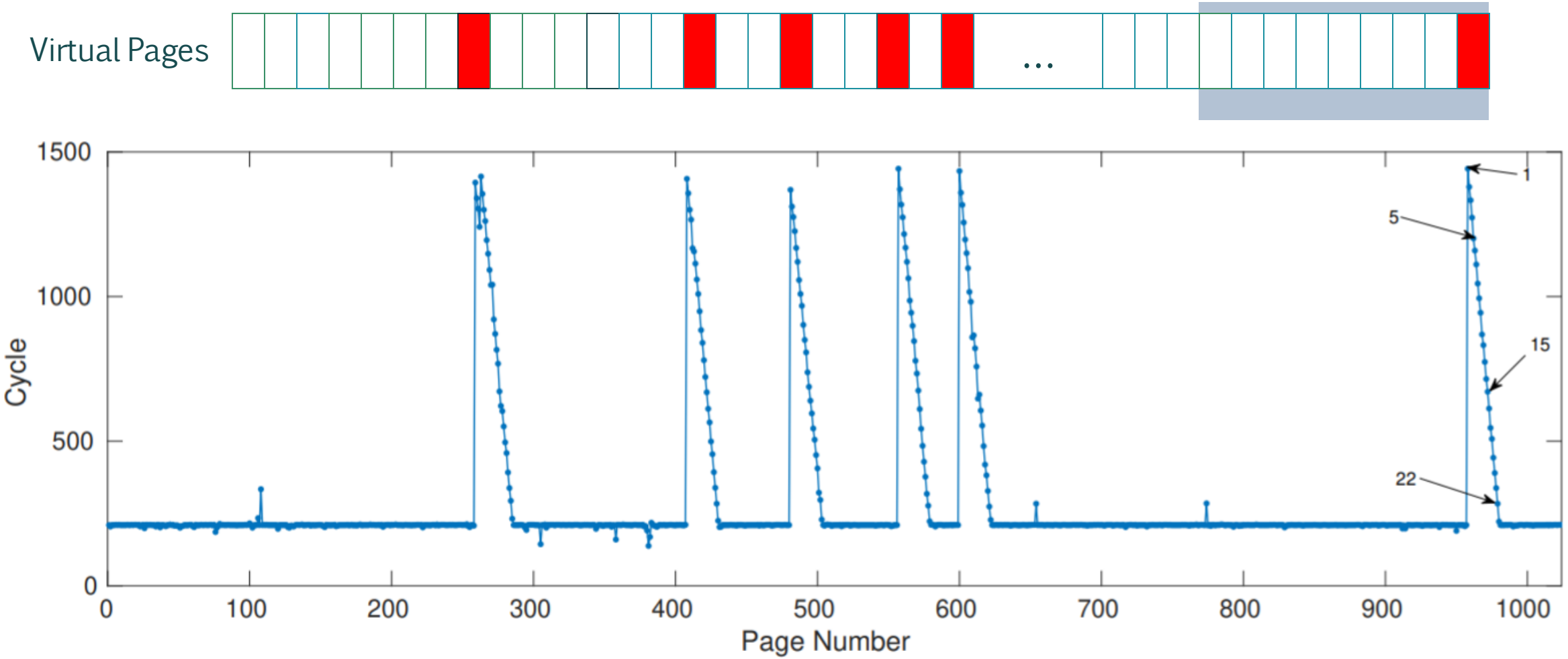
0x4F1234	0C0

Physical Addresses

0x65F32XX	0C0
0x32AC2XX	0C0



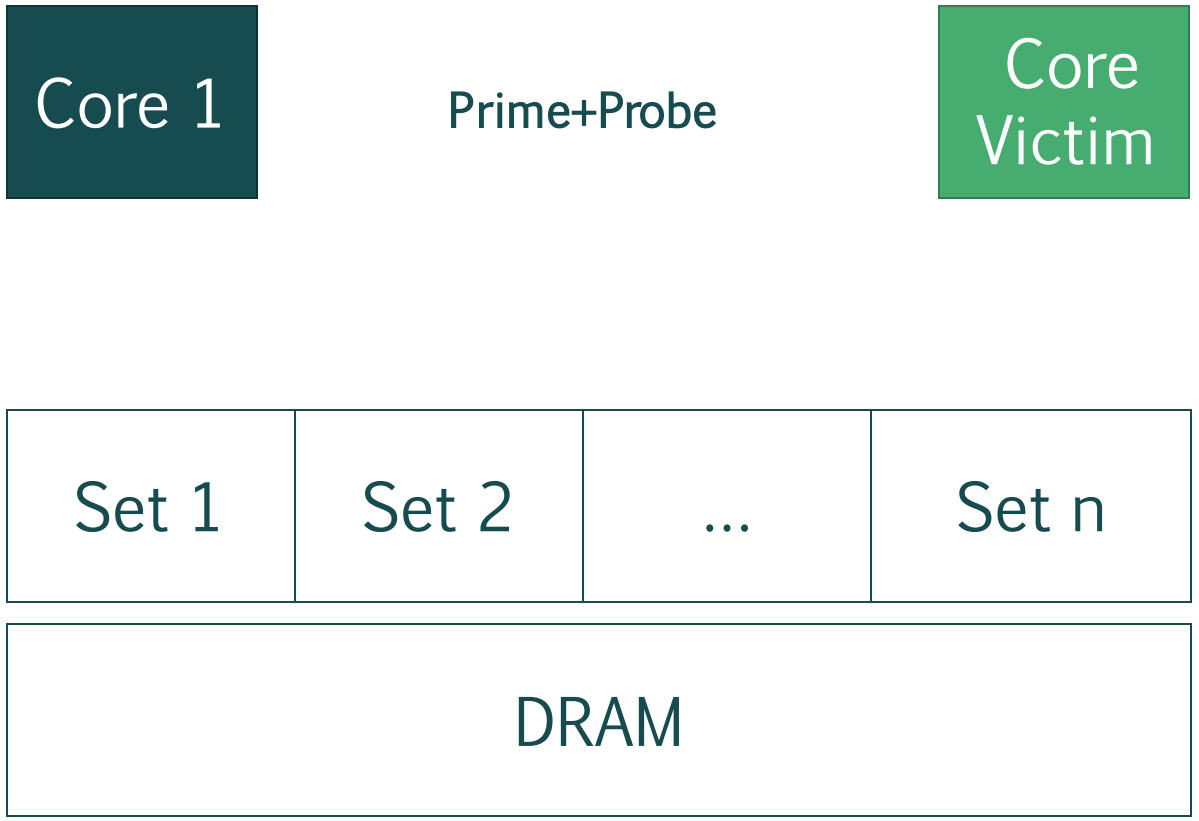
SPOILER Attack



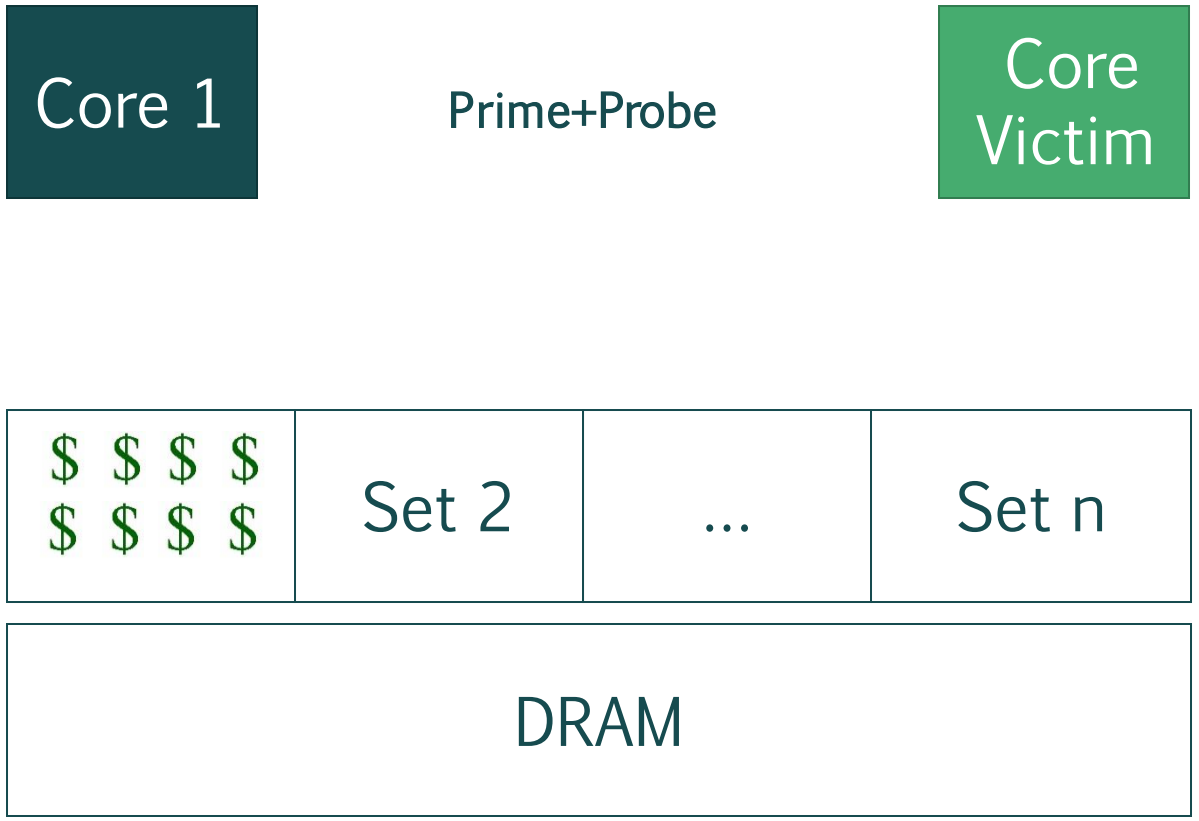
SPOILER Boosts Cache Attacks

SPOILER Boosts Cache Attacks

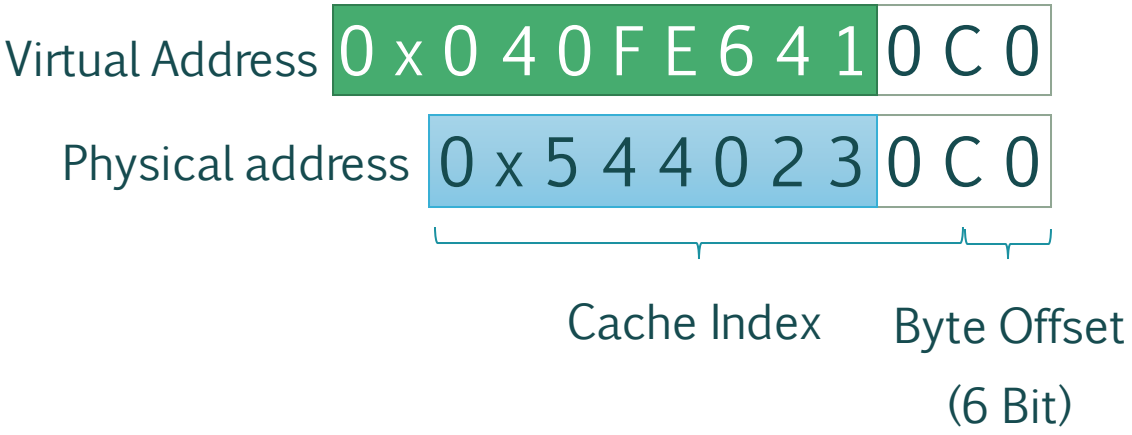
SPOILER Boosts Cache Attacks

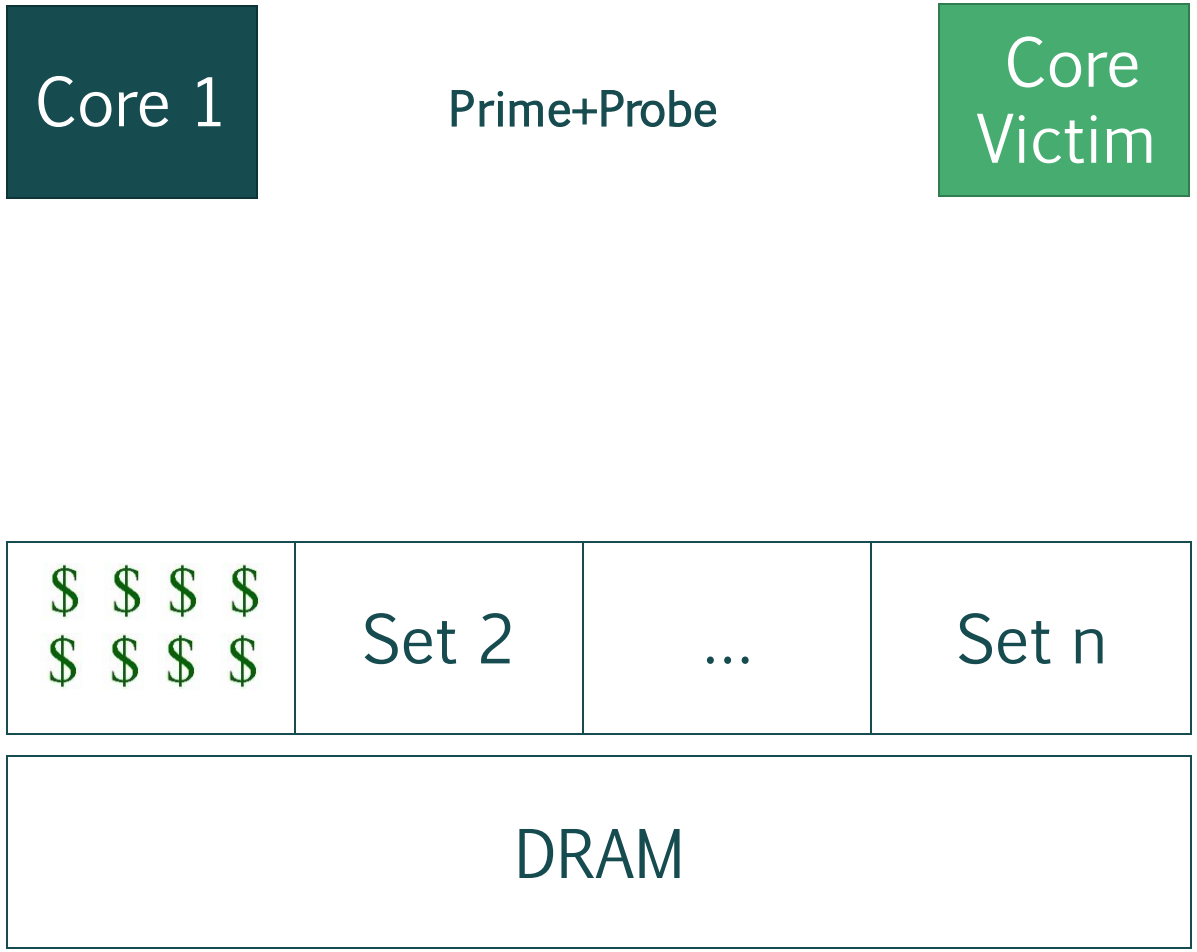


SPOILER Boosts Cache Attacks

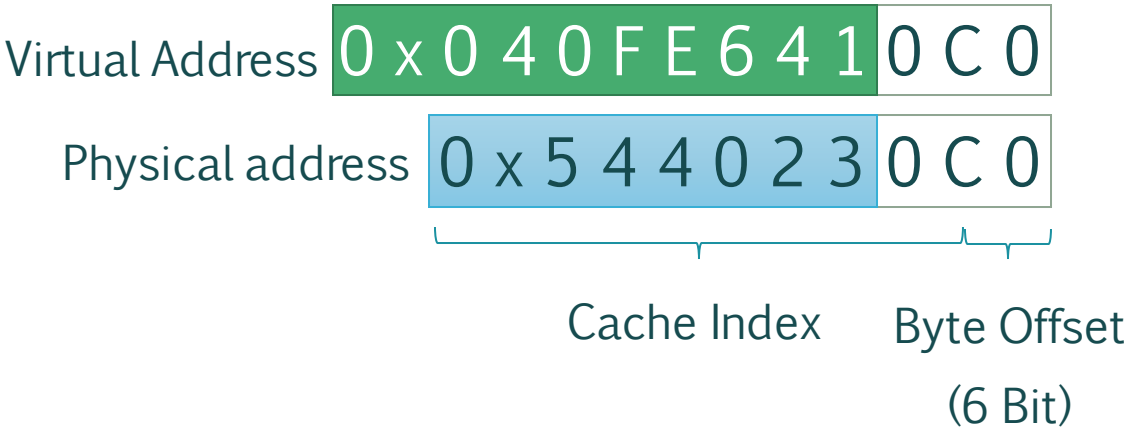


SPOILER Boosts Cache Attacks





SPOILER Boosts Cache Attacks

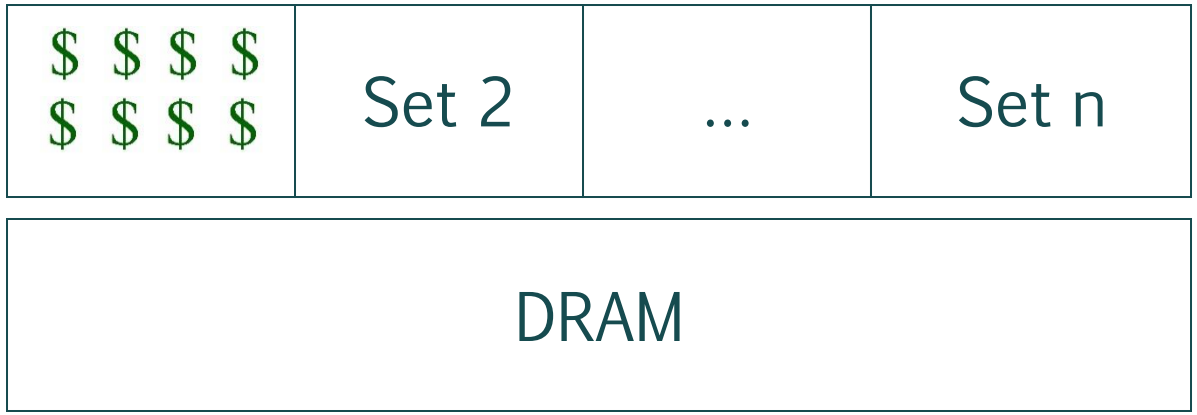


Skylake Client

L1: 64 Sets, 6 bit Index

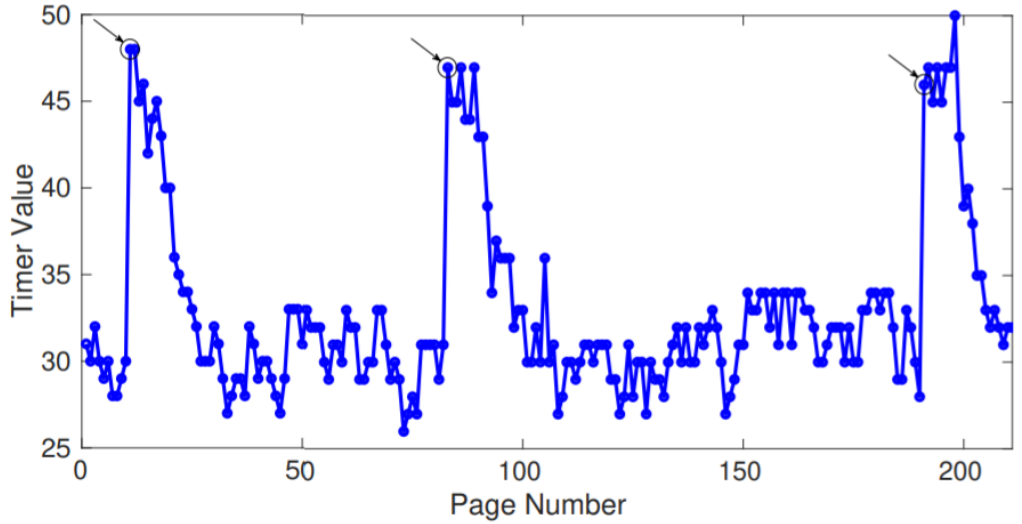
L2: 1024 Sets, 10 bit Index

LLC: 2048 Sets, 11 bit Index, 1-2 bit slices



SPOILER – Javascript Eviction Sets

- 1 MB Aliasing Leakage



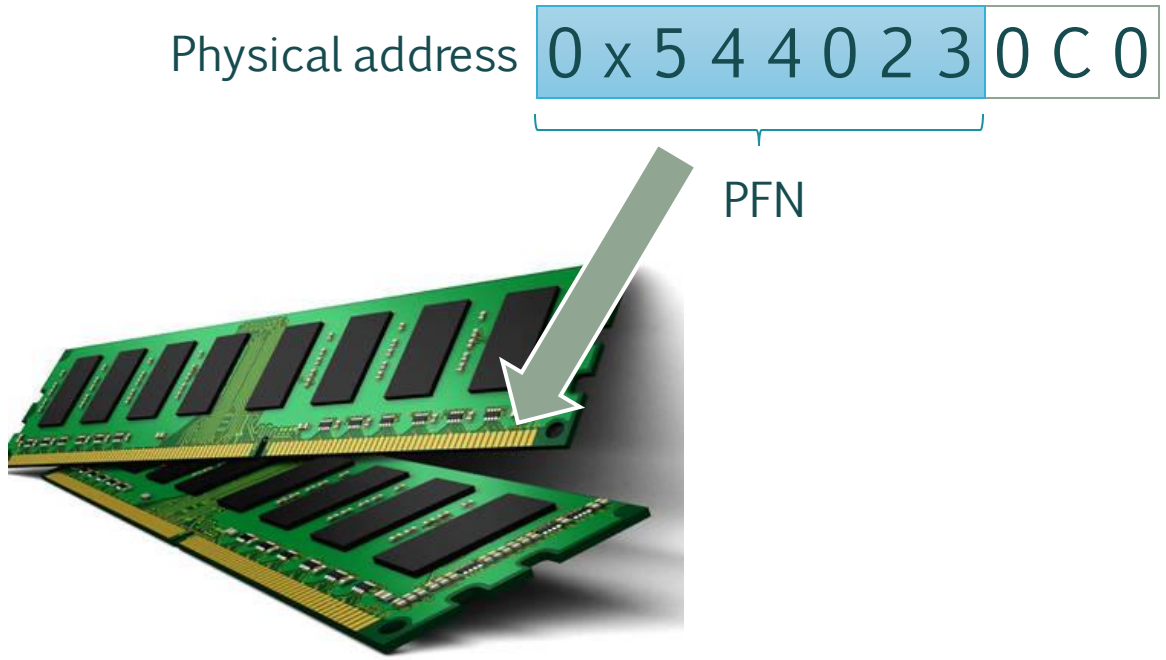
- Eviction Set Finding Comparison

Algorithm	R	t_{total}	t_{AAS}	t_{ESS}	Success
Classic [42]	3	46s	-	100%	80%
Improved [14]	3	35s	-	100%	80%
AA (ours)	10	10s	54%	46%	67%
AA (ours)	20	12s	75%	25%	100%

SPOILER Boosts Rowhammer

SPOILER Boosts Rowhammer

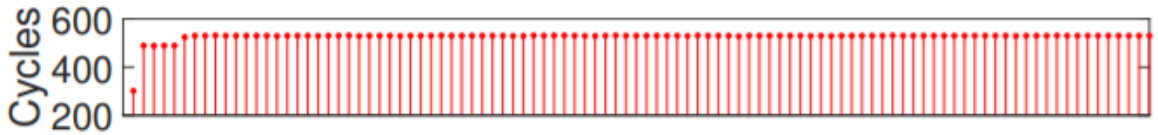
- Physical addresses are used for mapping DRAM banks
  - More Banks, More Physical Address Bits
- Single-Sided Rowhammer:
  - Requirement: Bank Co-location
- Double-Sided Rowhammer:
  - Contiguous Memory Pages



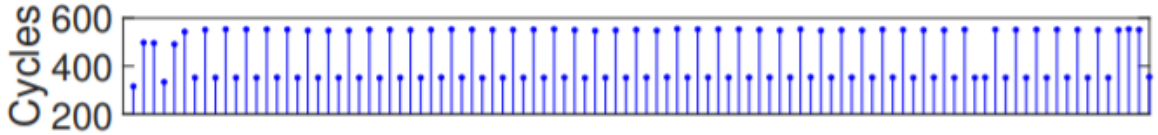
SPOILER Boosts Rowhammer

- Reverse Engineering DRAM Banks using DRAMA Tool
- Rowbuffer Conflict

System Model	DRAM Configuration	# of Bits
Dell XPS-L702x (Sandy Bridge)	1 x (4GB 2Rx8)	21
	2 x (4GB 2Rx8)	22
Dell Inspiron-580 (Nehalem)	1 x (2GB 2Rx8) **(b)**	21
	2 x (2GB 2Rx8) **(c)**	22
	4 x (2GB 2Rx8) **(d)**	23
Dell Optiplex-7010 (Ivy Bridge)	1 x (2GB 1Rx8) **(a)**	19
	2 x (2GB 1Rx8)	20
	1 x (4GB 2Rx8) **(e)**	21
	2 x (4GB 2Rx8)	22



(a) 19 bits used by memory controller, no unknown bits

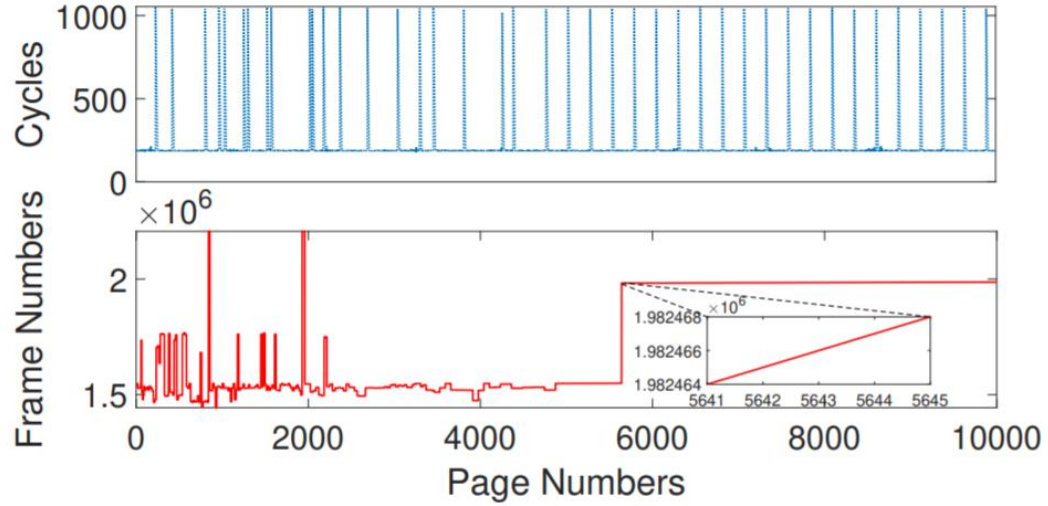


(b) 21 bits used by memory controller, 1 unknown bit

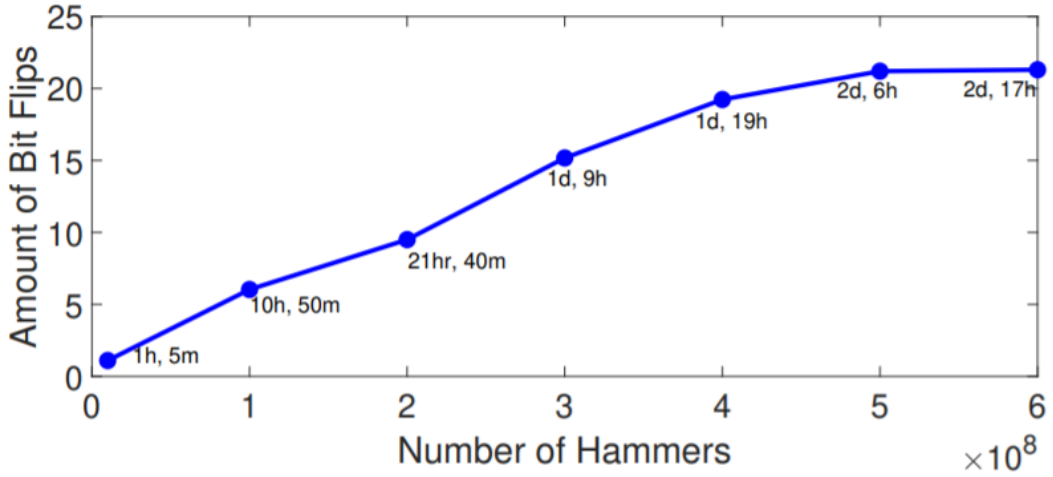
(c) 22 bits used by memory controller, 2 unknown bits

SPOILER Boosts Rowhammer

- Detecting Contiguous Memory

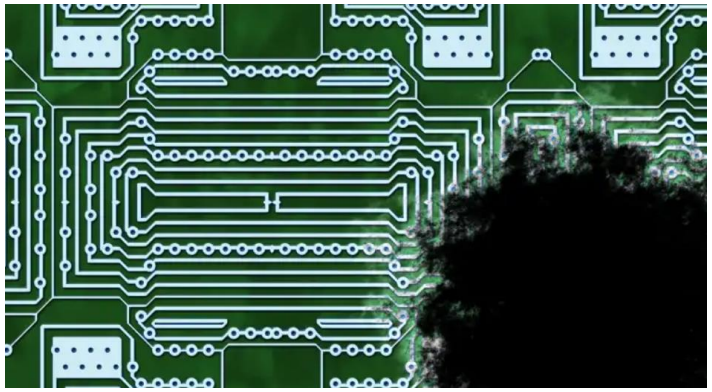


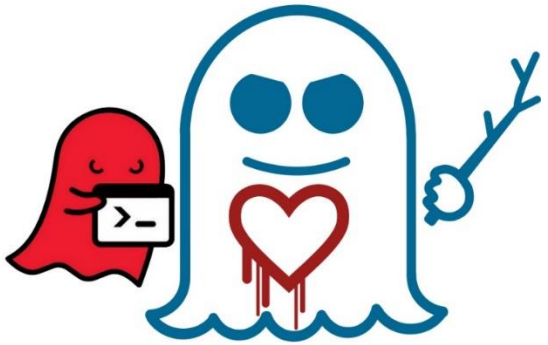
- Rowhammer Bitflips



CVE-2019-0162

- 12/01/2018: We informed our findings to iPSIRT.
- 12/03/2018: iPSIRT acknowledged the receipt.
- 03/01/2018: We published the paper.
- 04/09/2019: iPSIRT released public advisory (INTELSA-00238) (CVE-2019-0162).
- And we got some free logos, *Thanks to Media* !!!



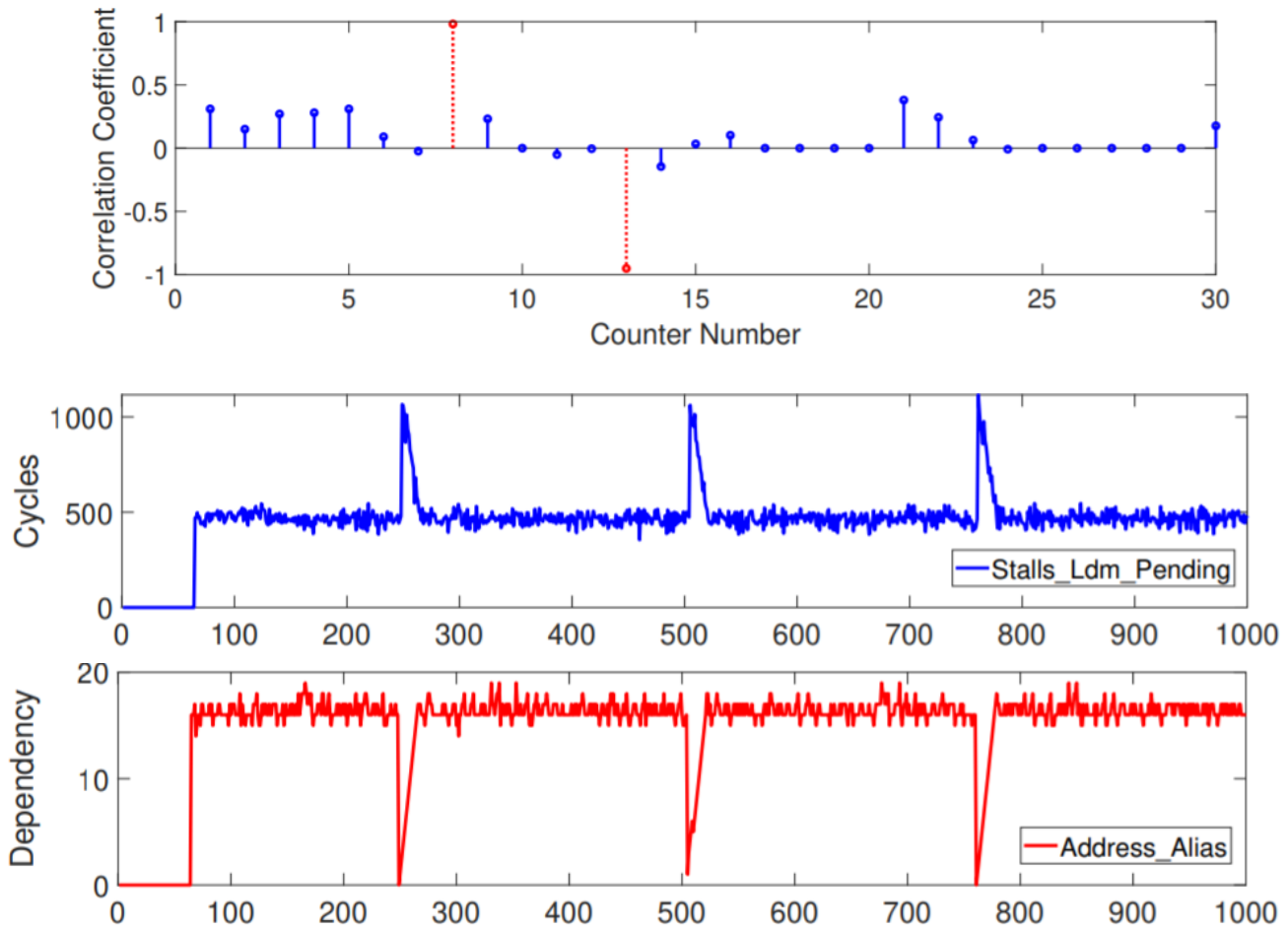


Questions?!

@danielmgmi

https://github.com/UzL-ITS/Spoiler

SPOILER Attack – HPC Analysis



SPOILER: 1 MB Aliasing

- Significant delay on Load when it matches with 20 bits of a store address (1 MB aliasing)
- The delay is highest when the store appears later in the store buffer.
- The number of steps has a correlation with the store buffer size.
- HPC Analysis:
  - STALLS_LDM_PENDING: Direct correlation, confirms that the delay is due to the Load
  - Ld_Blocks_Partial:Address_Alias: Negative Correlation, confirms that the delay is not due to *Loosenet* check

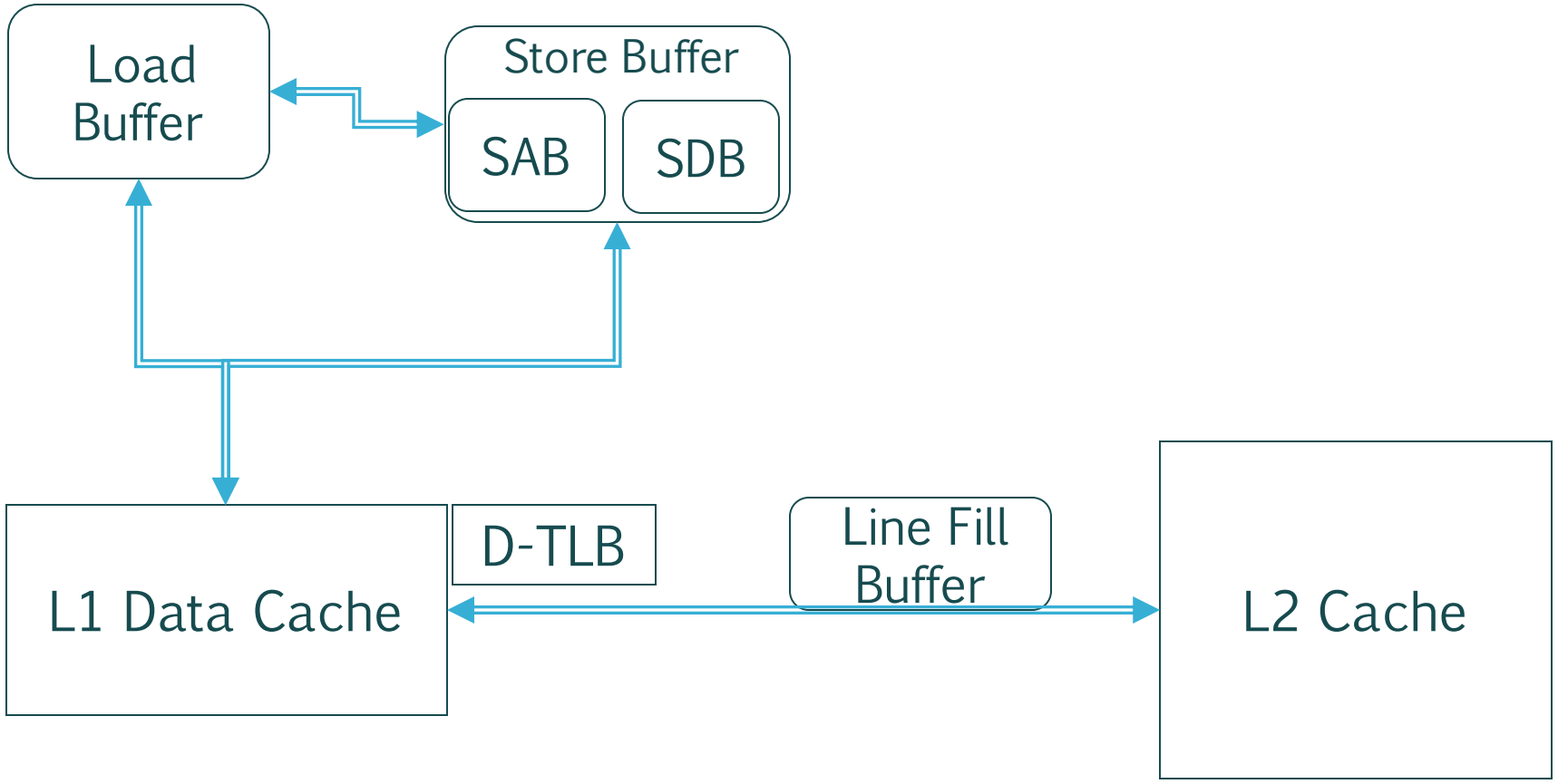
SPOILER Attack – Affected Machines

CPU Model	Architecture	Steps	SB Size
Intel Core i7-8650U	Kaby Lake R	22	56
Intel Core i7-7700	Kaby Lake	22	56
Intel Core i5-6440HQ	Skylake	22	56
Intel Xeon E5-2640v3	Haswell	17	42
Intel Xeon E5-2670v2	Ivy Bridge EP	14	36
Intel Core i7-3770	Ivy Bridge	12	36
Intel Core i7-2670QM	Sandy Bridge	12	36
Intel Core i5-2400	Sandy Bridge	12	36
Intel Core i5 650	Nehalem	11	32
Intel Core2Duo T9400	Core	N/A	20
Qualcomm Kryo 280	ARMv8-A	N/A	*
AMD A6-4455M	Bulldozer	N/A	*

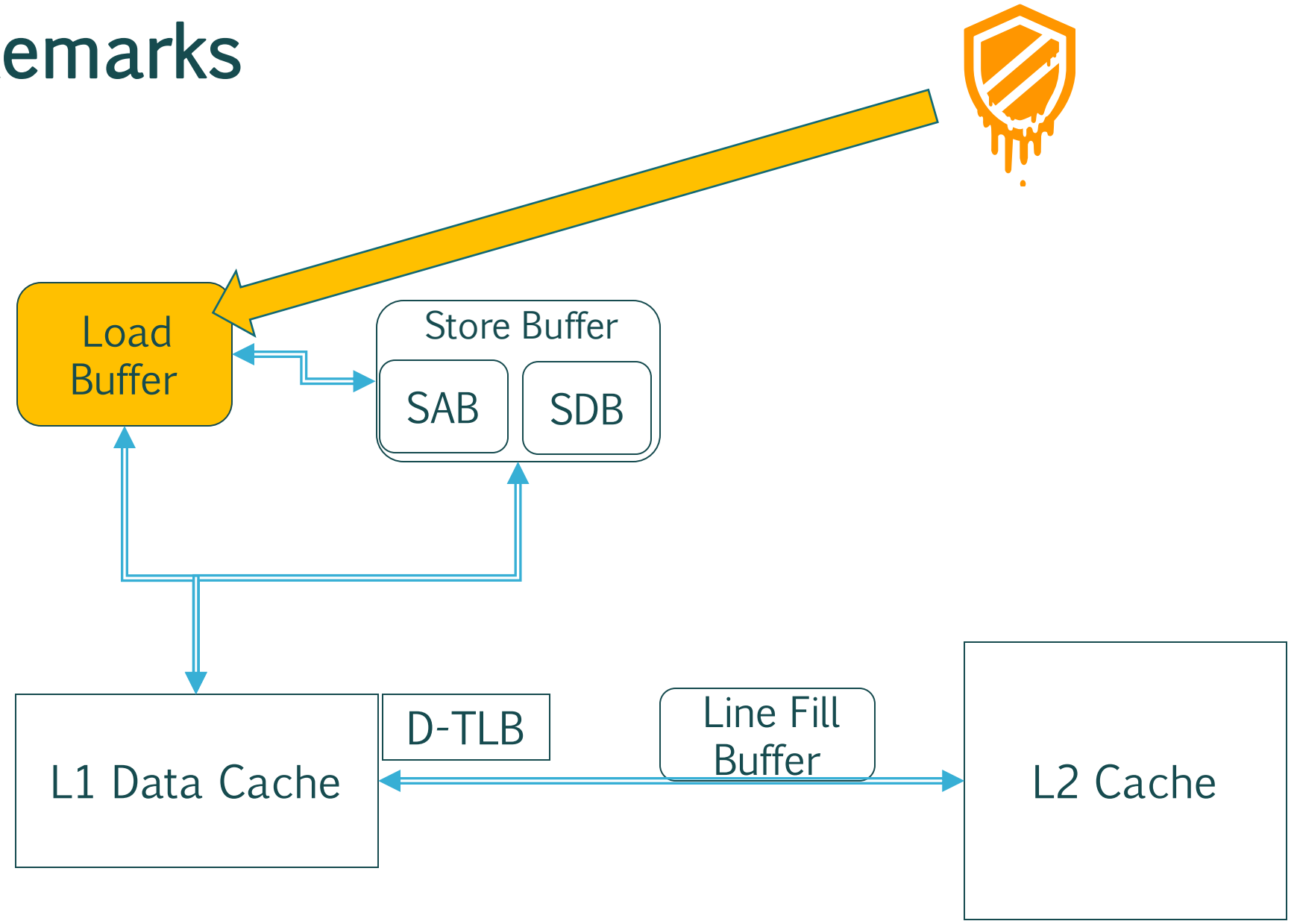
SPOILER Boosts Cache Attacks

SPOILER Boosts Cache Attacks

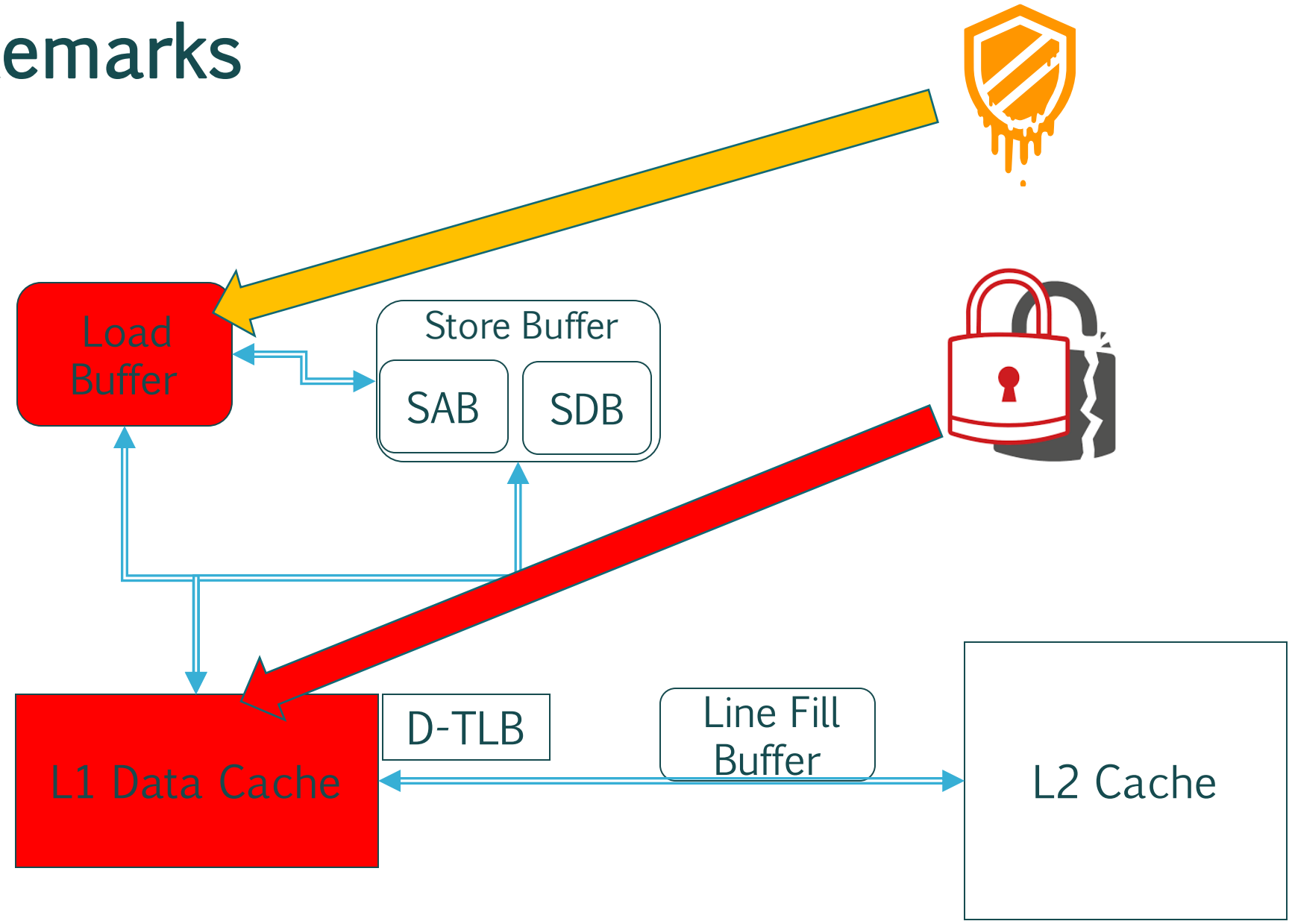
Remarks


Load Buffer
Store Buffer
SAB
SDB
L1 Data Cache
D-TLB
Line Fill Buffer
L2 Cache

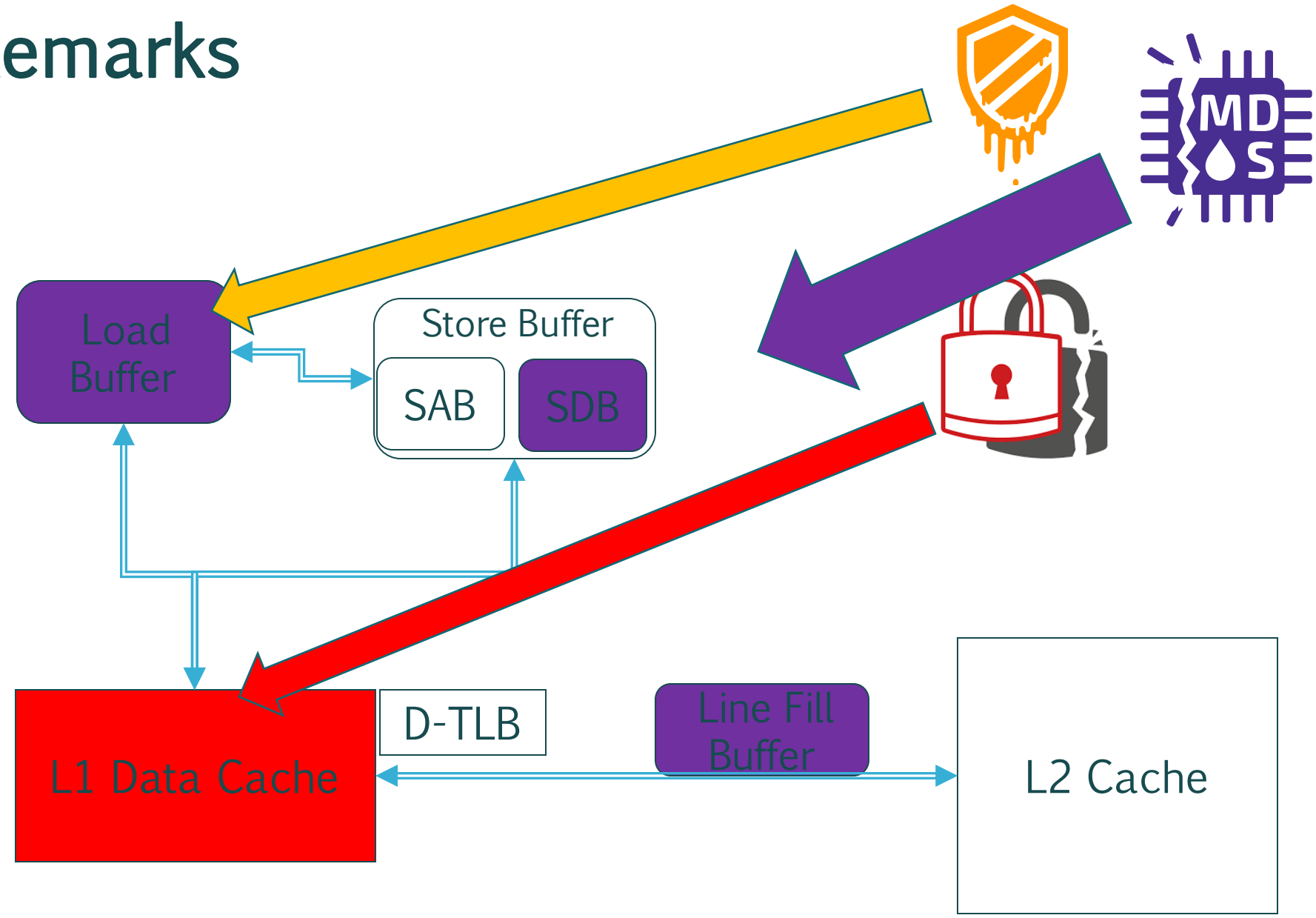
Remarks


Load Buffer
Store Buffer
SAB
SDB
L1 Data Cache
D-TLB
Line Fill Buffer
L2 Cache

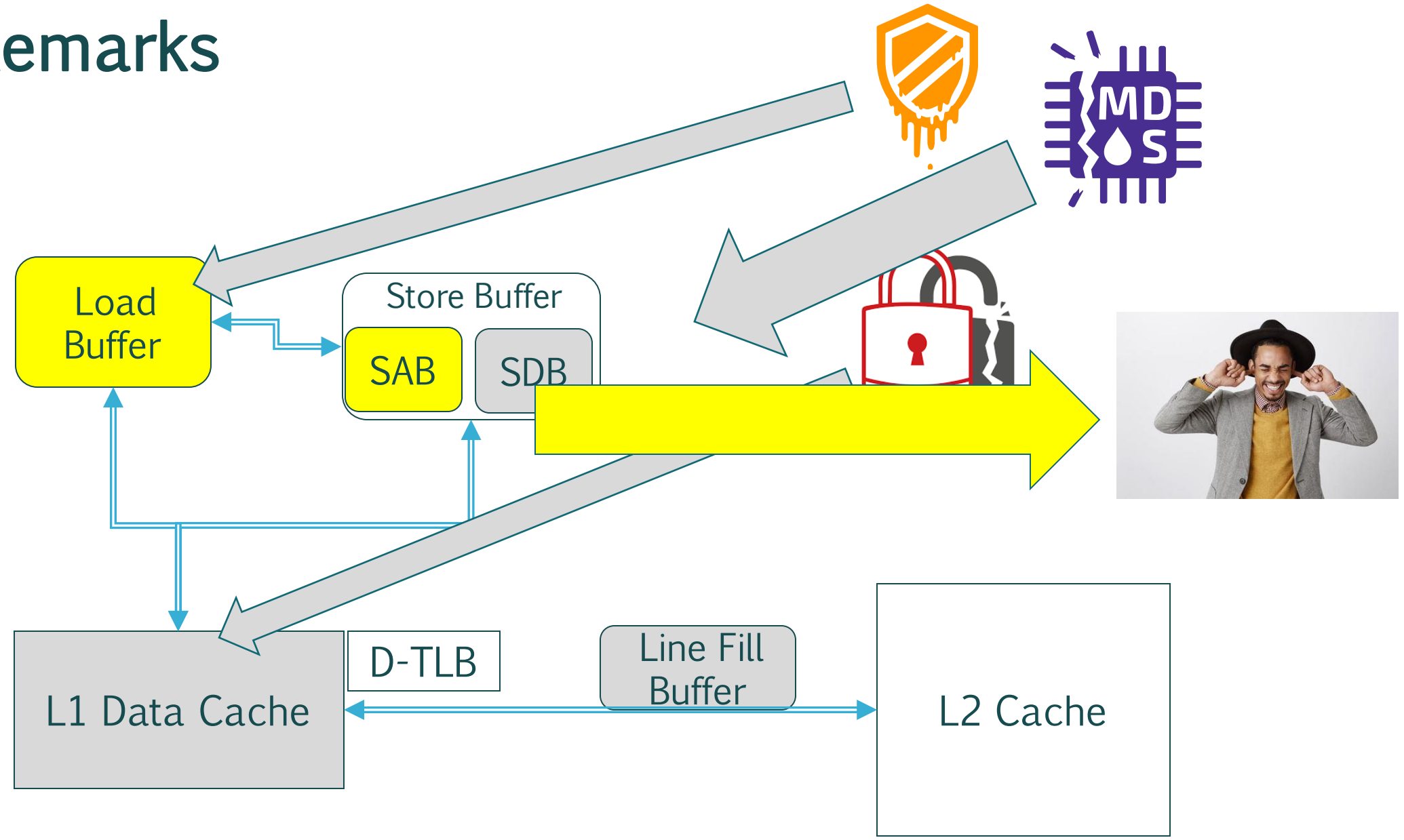
Remarks


Load Buffer
Store Buffer
SAB
SDB
L1 Data Cache
D-TLB
Line Fill Buffer
L2 Cache

Remarks


Load Buffer
Store Buffer
SAB
SDB
L1 Data Cache
D-TLB
Line Fill Buffer
L2 Cache

Remarks


Load Buffer
Store Buffer
SAB
SDB
L1 Data Cache
D-TLB
Line Fill Buffer
L2 Cache