Security Needs a New Hardware-Software Contract

Gernot Heiser
gernot.heiser@data61.csiro.au | gernot@unsw.edu.au | @GernotHeiser

https://trustworthy.systems

The New Year Shock

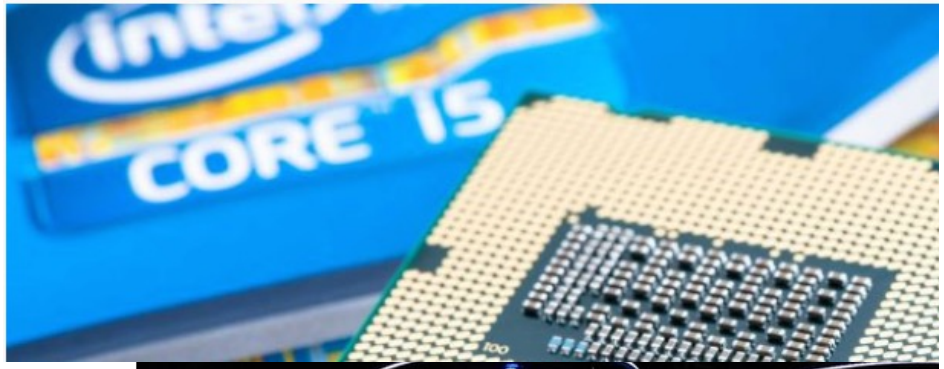
Vulnerabilities in modern Intel processors compromise the security of most compute

Share on Facebook | Share on Twitter

ABC Science By technology reporter Ariel Bogle

Updated 4 January 2018 at 3:36 pm

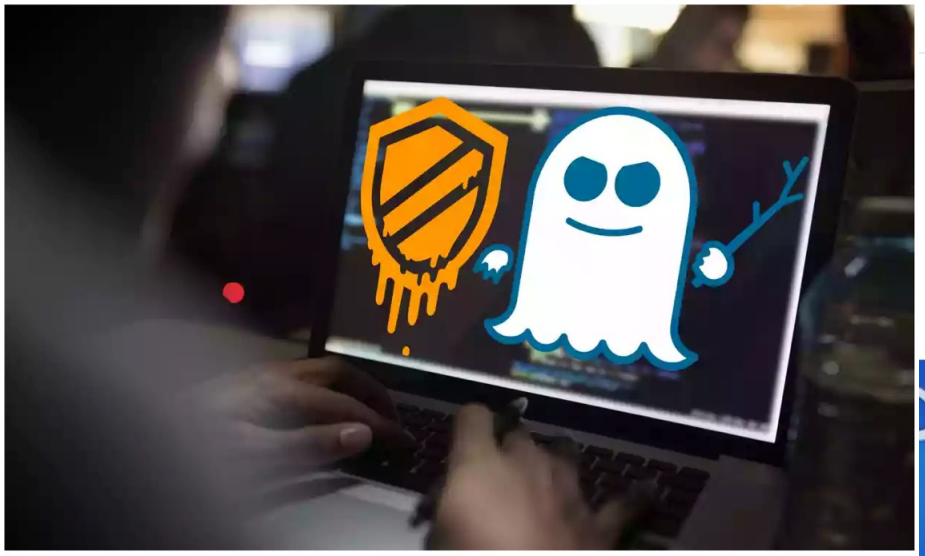
First posted 4 January 2018 at 12:45 pm



Data and computer security

Spectre and Meltdown processor security flaws – explained

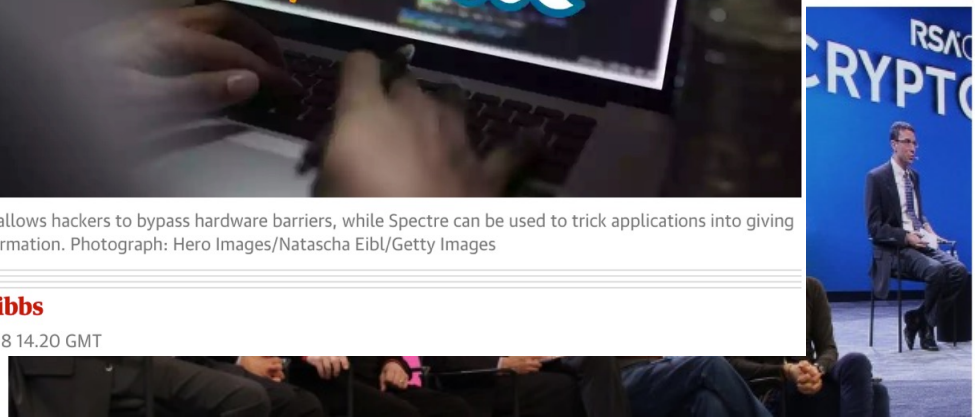
What are Meltdown and Spectre? Do they only affect Intel chips? Will the fixes slow my computer ... and what even is a processor?



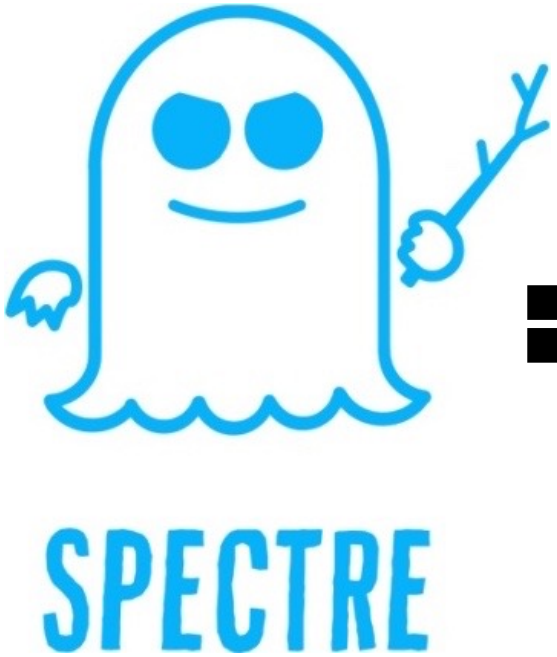
▲ Meltdown allows hackers to bypass hardware barriers, while Spectre can be used to trick applications into giving up secret information. Photograph: Hero Images/Natascha Eibl/Getty Images

Samuel Gibbs

Thu 4 Jan 2018 14.20 GMT



Threats

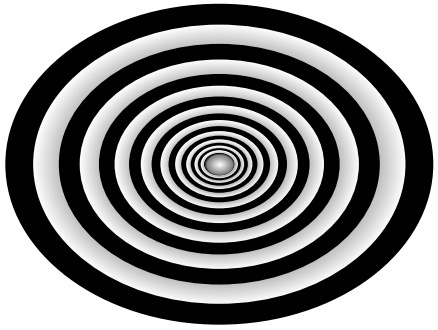


=

Speculation

An “unknown unknown” until recently

+

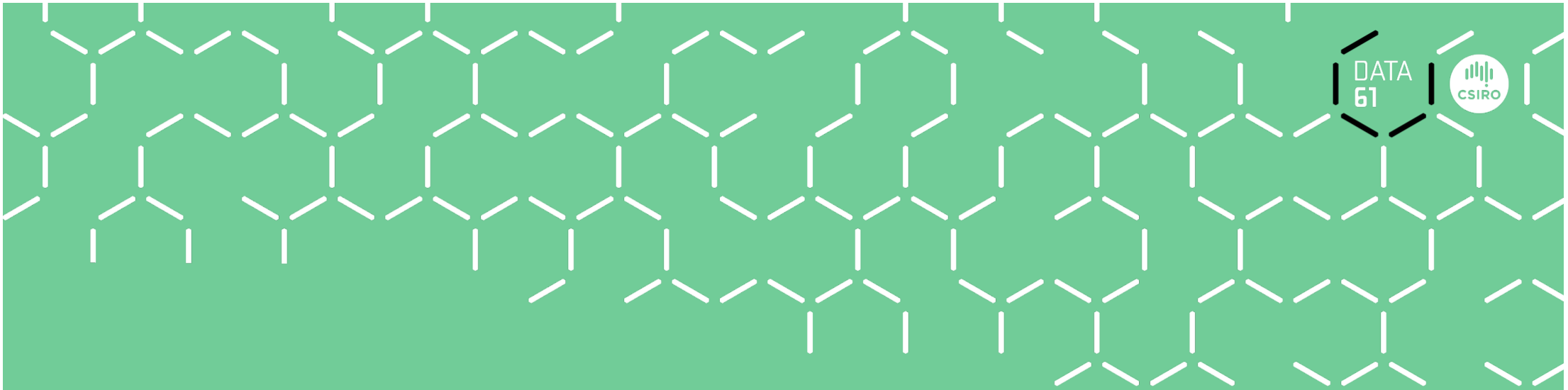


Timing Channel

A “known unknown” for decades

Overview

- What are timing channels?
- *Time protection:* OS must close microarchitectural channels
- How helpful is present hardware?
- What are the requirements on hardware for closing timing channels?
- Defining the new hardware-software contract – aISA



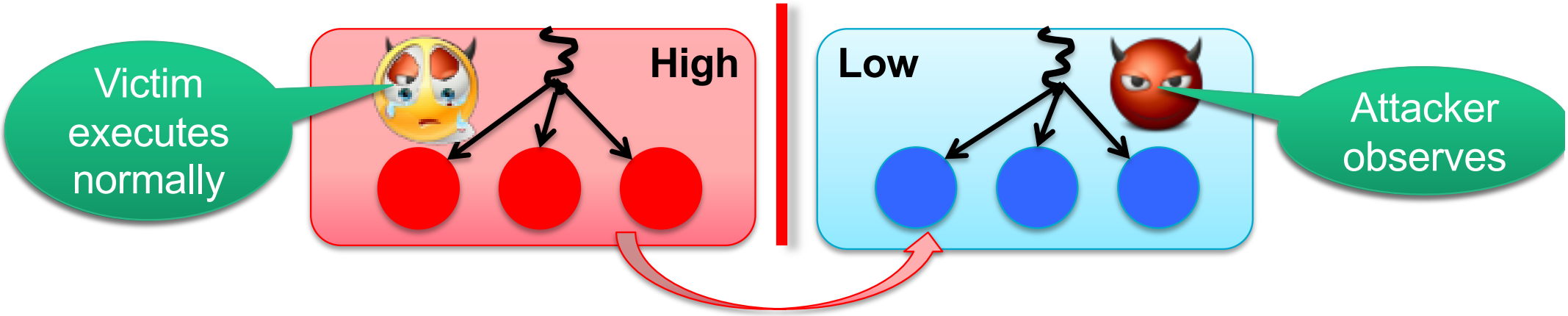
What Are Timing Channels?

Timing Channels

Information leakage through timing of events

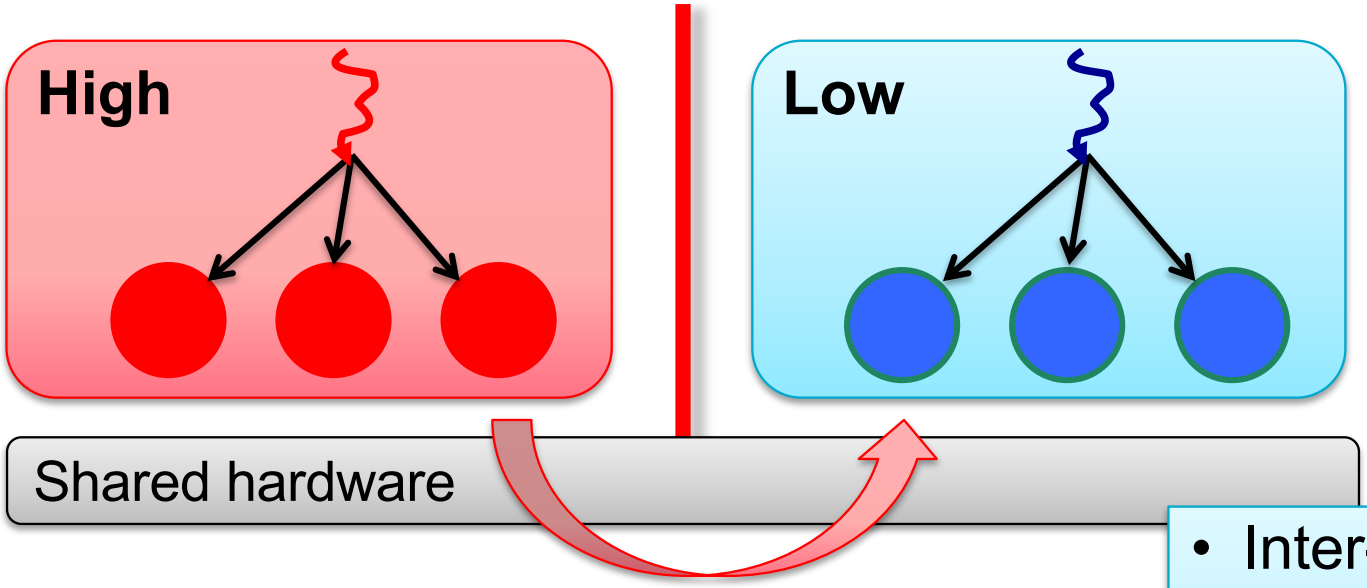
- Typically by observing response latencies or own execution speed

Covert channel: Information flow that bypasses the security policy



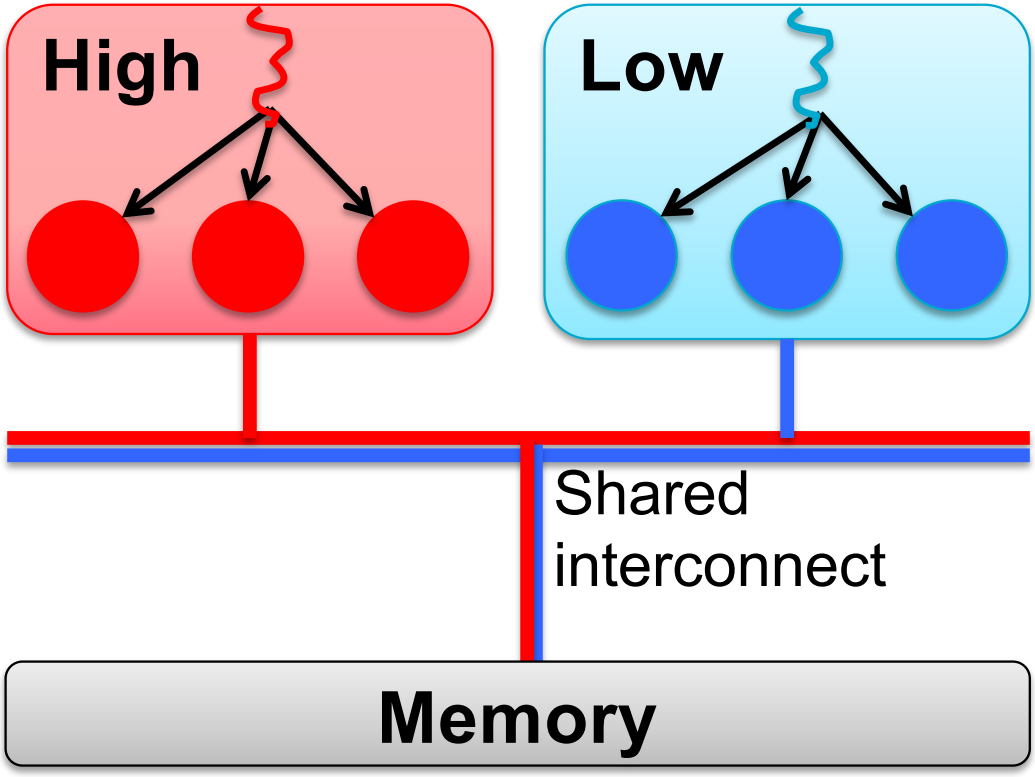
Side channel: Covert channel exploitable without insider help

Cause: Temporal Interference



- Inter-process interference
- Competing access to micro-architectural features
  - not exposed by the ISA
  - hidden by the HW-SW contract!

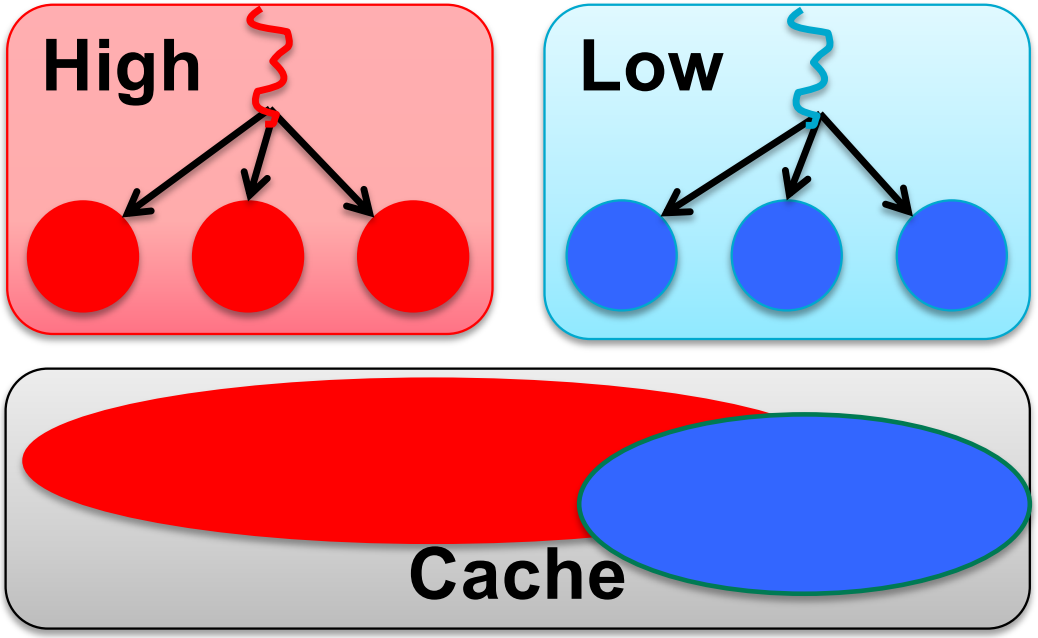
Sharing 1: Stateless Interconnect



H/W is *bandwidth-limited*

- Interference during concurrent access
- Generally reveals no data or addresses
- Must encode info into access patterns
- Only usable as covert channel, not side channel

Sharing 2: Stateful Hardware



HW is *capacity-limited*

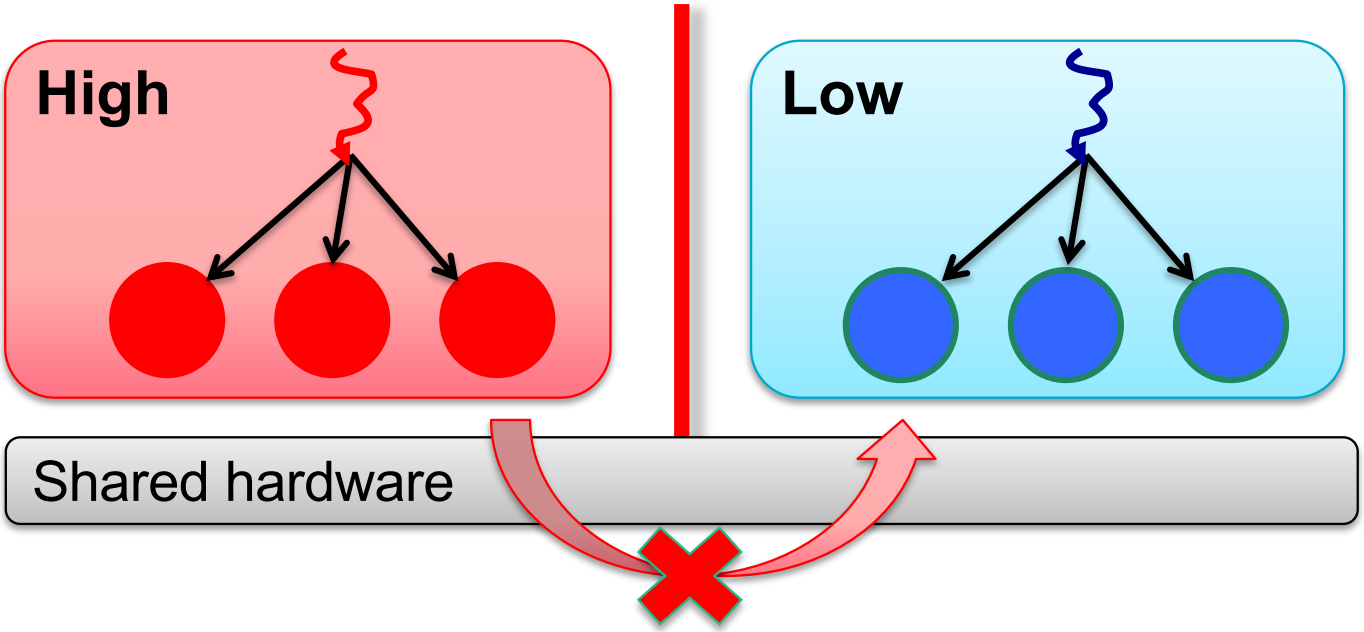
- Interference during
  - concurrent access
  - time-shared access
- Collisions reveal data or addresses
- *Usable as side channel*

Any state-holding microarchitectural feature:

- cache, branch predictor, pre-fetcher state machine

Time Protection

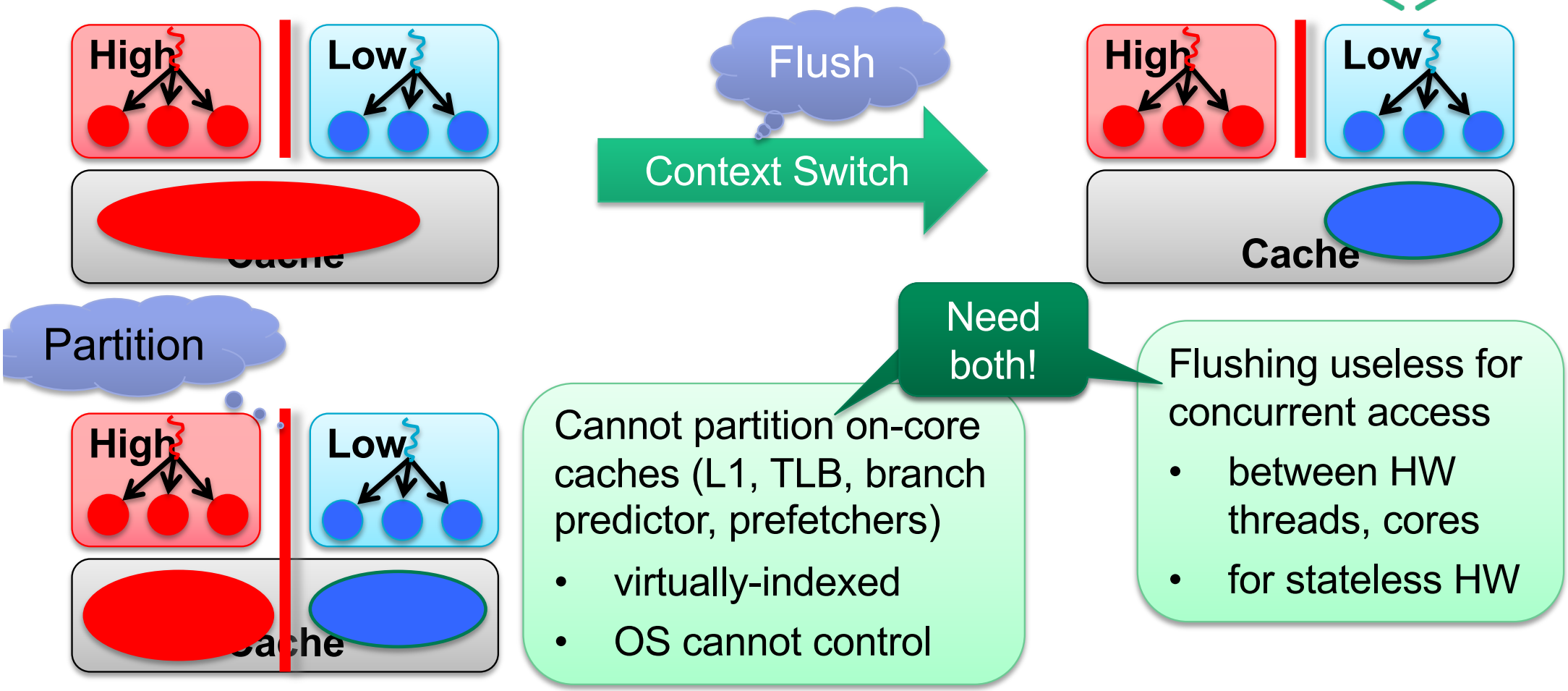
OS Must Enforce *Time Protection*



Preventing interference is core duty of the OS!

- *Memory protection* is well established
- *Time protection* is completely absent

Time Protection: No Sharing of State



Requirements For Time Protection

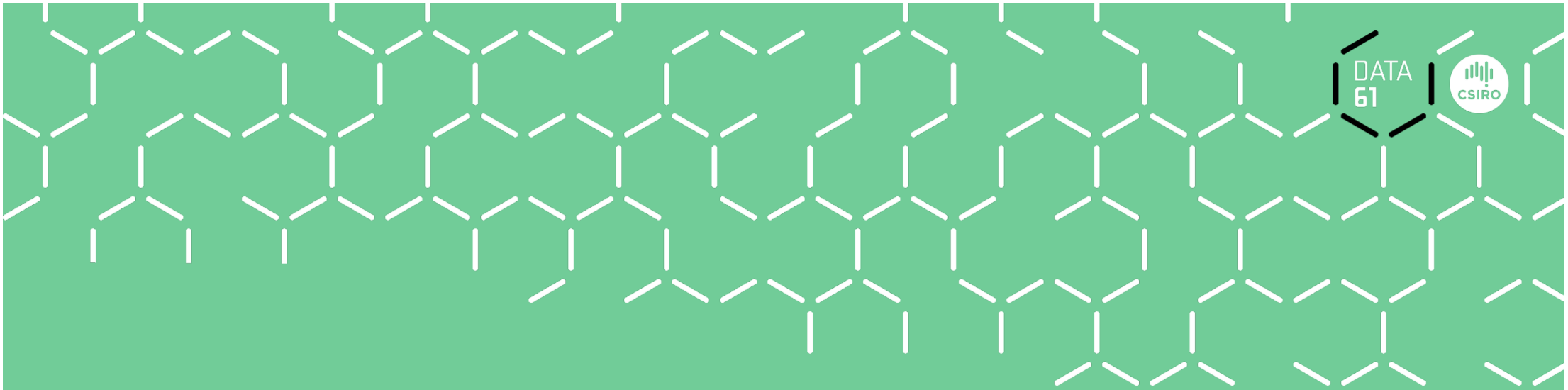
Off-core state & stateless HW

On-core state

Timing channels can be closed *iff* the OS can

- partition or
- reset

all shared hardware



Implementing Time Protection: Stateful Hardware

Flush on Domain Switch

Must remove any history dependence

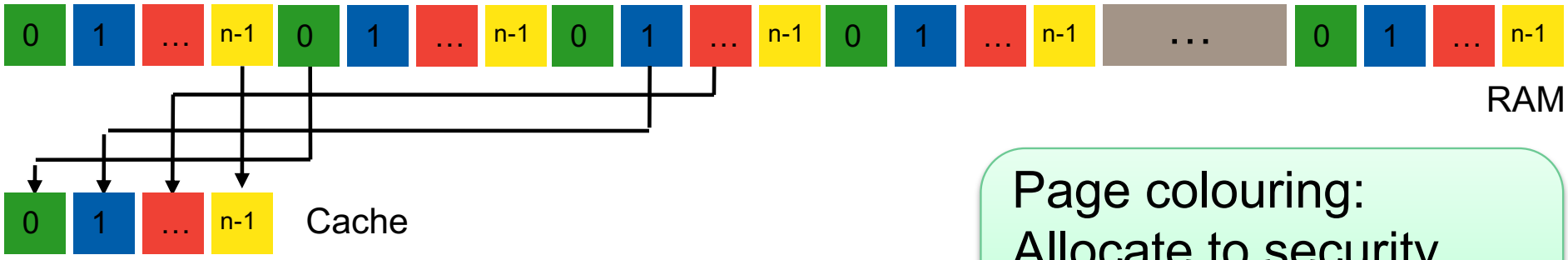
Latency depends on prior execution!

1. T_0 = current_time()
2. Switch user context
3. Flush on-core state
4. Touch all code/data needed for return
5. while (T_0+WCET < current_time()) ;
6. Reprogram timer
7. return

Ensure deterministic execution

Time padding to Remove dependency

Partition Caches: Page Colouring



Page colouring:
Allocate to security partitions only memory of disjoint colours

Exploit associative cache lookup:

- Particular address maps to specific cache subset, called *cache colour*
- # colours = cache size / (page size * associativity)

Testbed: seL4 Microkernel

seL4: The world's only operating-system kernel with provable security enforcement (incl. memory protection)

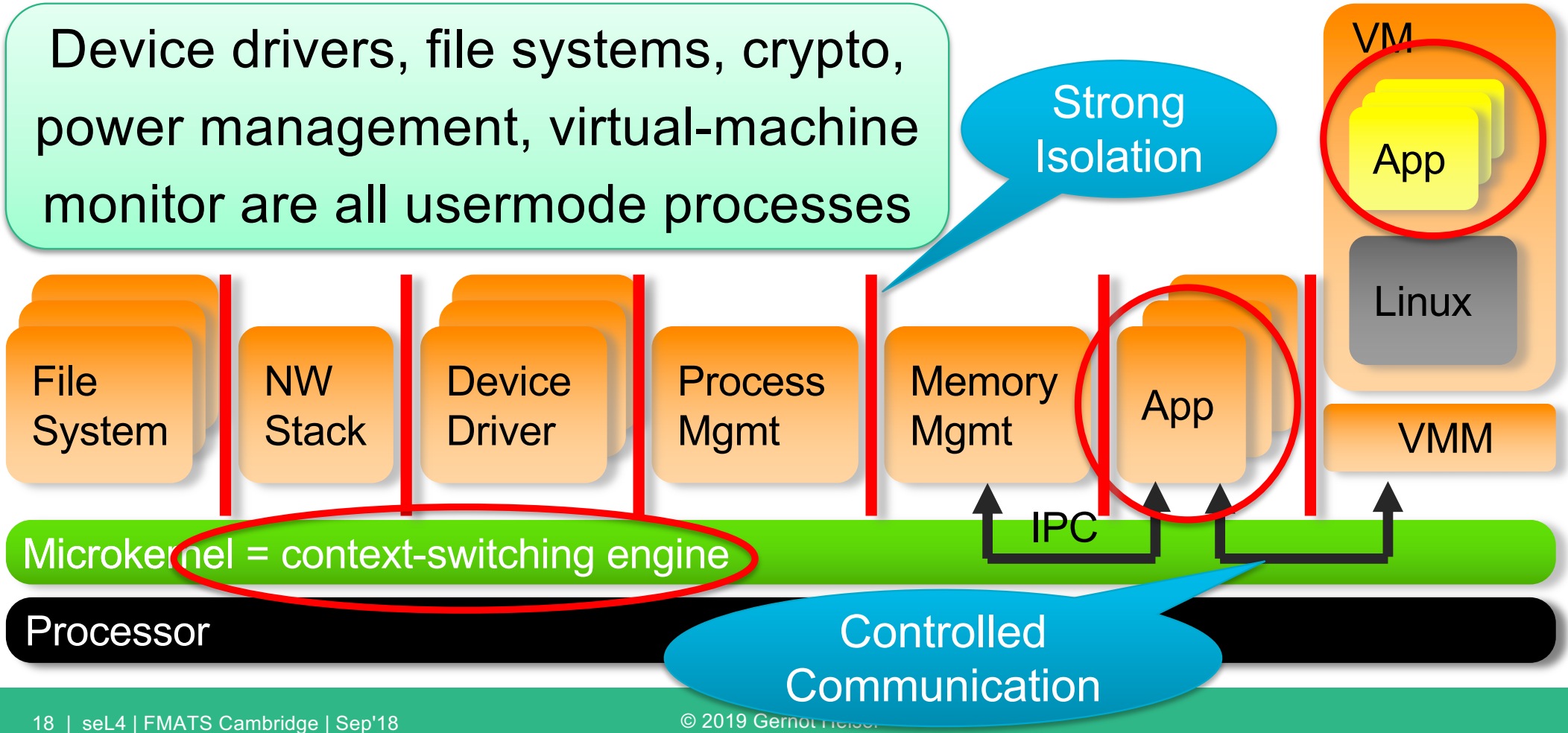
Open Source

seL4: The world's only protected-mode OS with complete, sound timeliness analysis

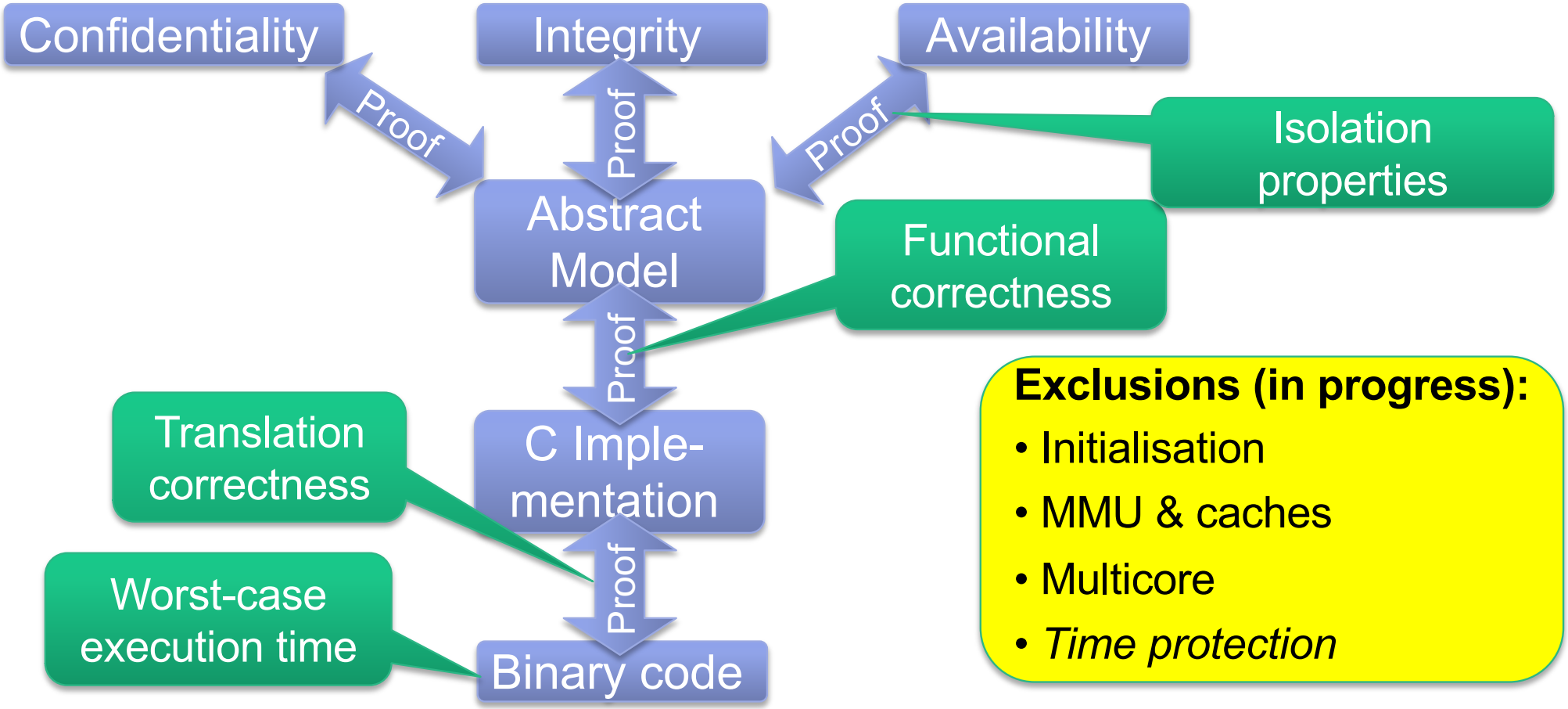
seL4: The world's fastest microkernel

A Microkernel is not an OS

Device drivers, file systems, crypto, power management, virtual-machine monitor are all usermode processes



Security Proof Chain



Exclusions (in progress):

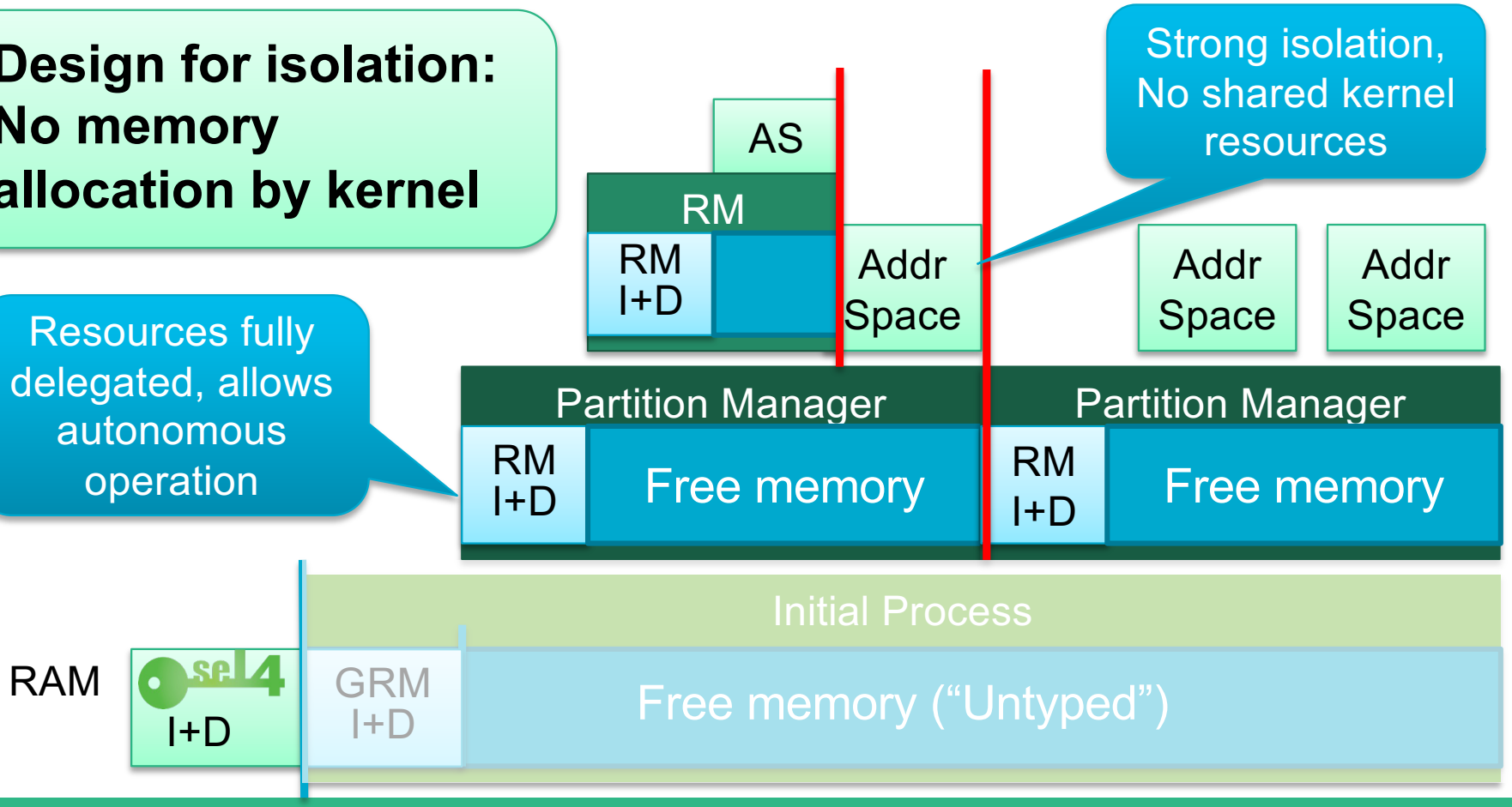
- Initialisation
- MMU & caches
- Multicore
- *Time protection*

Memory Management Model

Design for isolation: No memory allocation by kernel

Strong isolation, No shared kernel resources

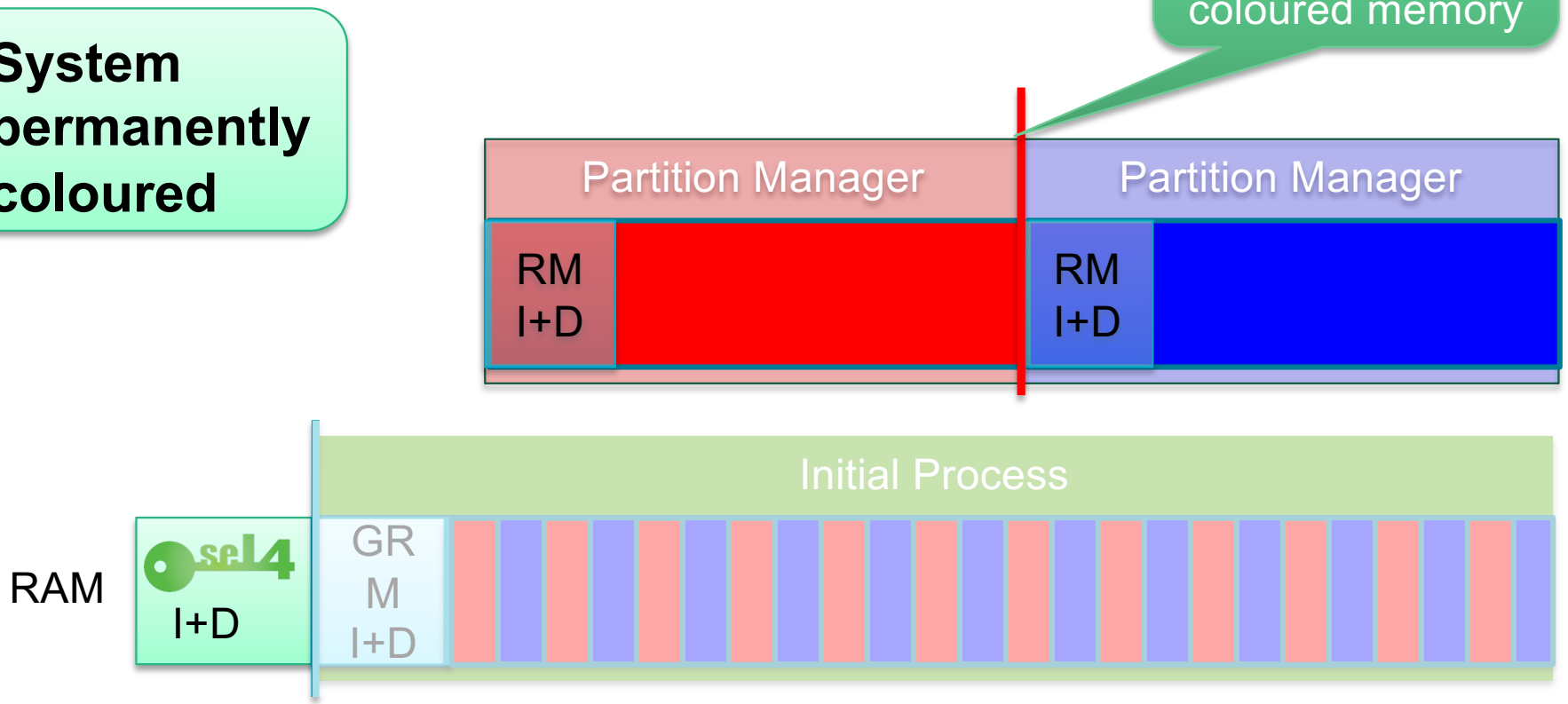
Resources fully delegated, allows autonomous operation



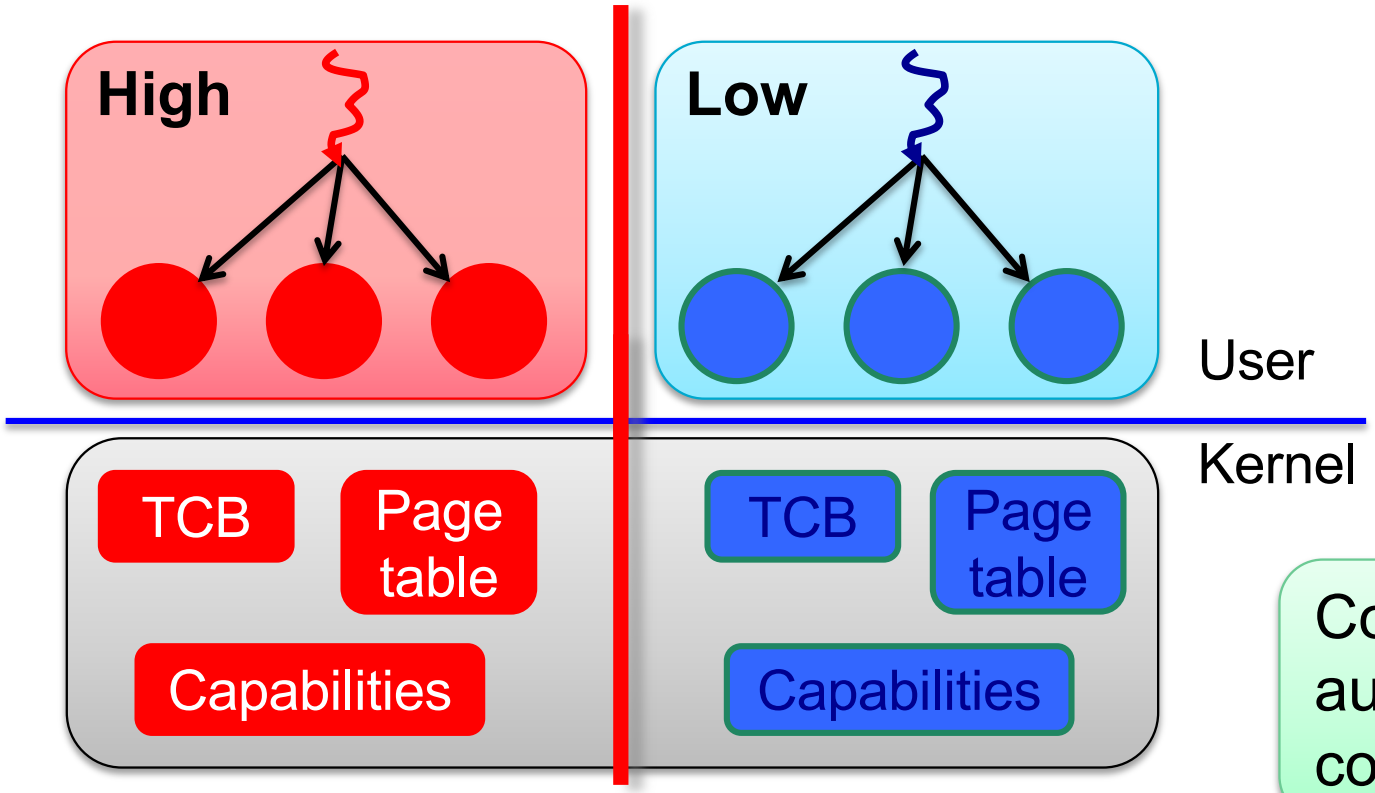
Colouring User Memory Is Easy

System permanently coloured

Partitions restricted to coloured memory



Isolation Goes Deep



seL4 has no heap

- All kernel memory supplied by user-level managers

Kernel code?

Colouring user data automatically colours kernel data

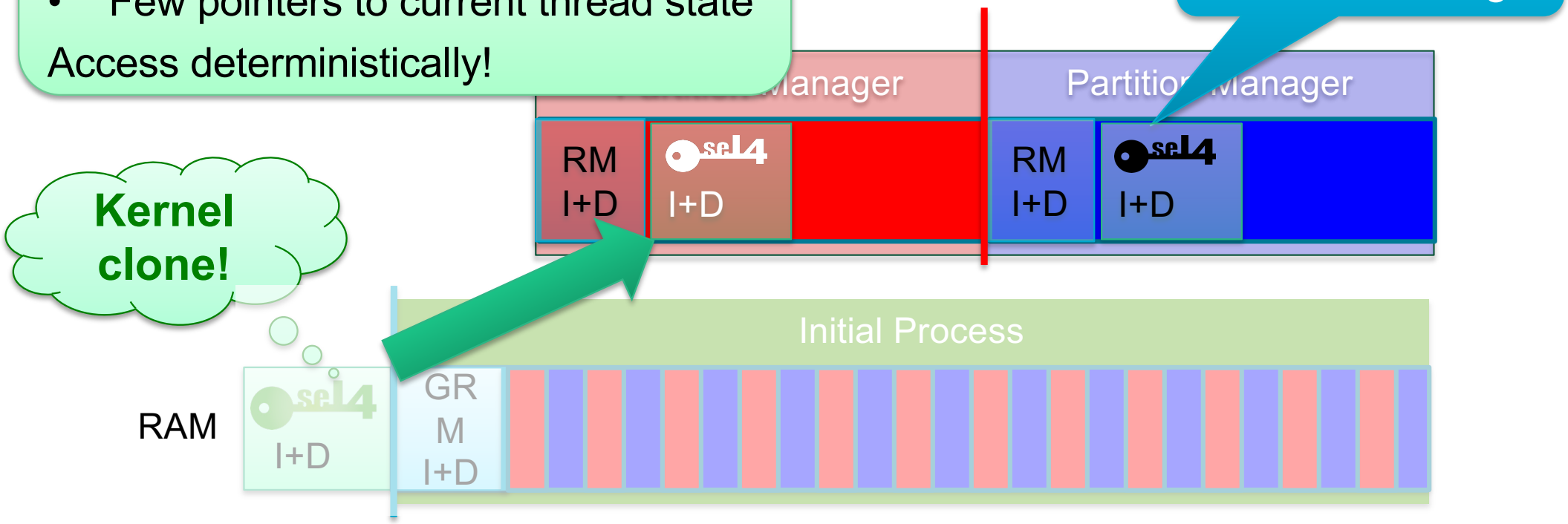
Colouring the Kernel

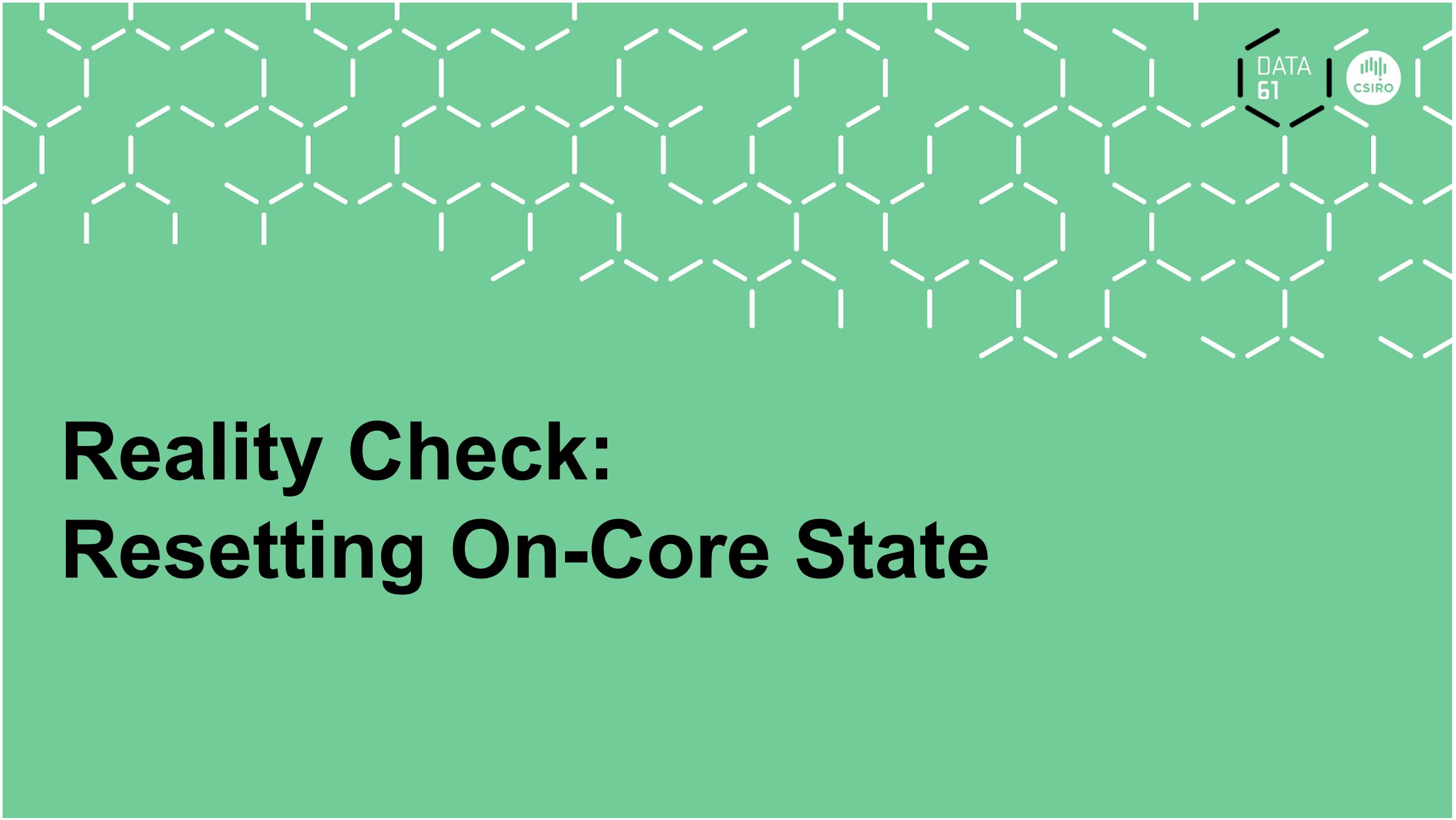
Remaining shared kernel data:

- Scheduler queue array & bitmap
- Few pointers to current thread state

Access deterministically!

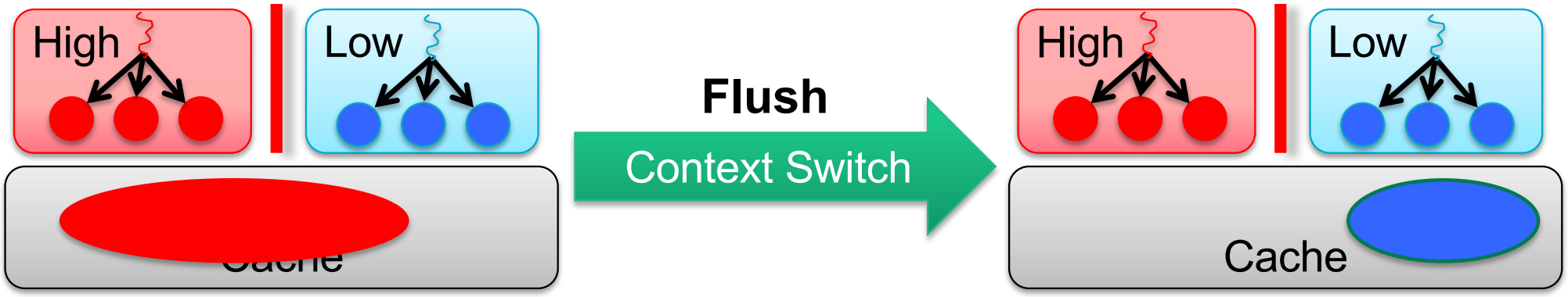
Each partition has own kernel image





Reality Check: Resetting On-Core State

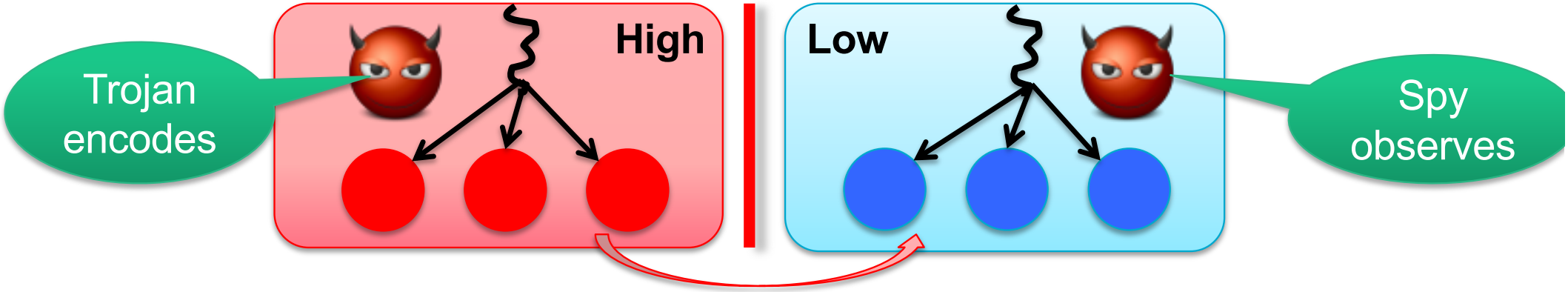
Evaluating Intra-Core Channels



Mitigation on Intel and Arm processors:

- Disable data prefetcher (just to be sure)
- On context switch, perform all architected flush operations:
  - Intel: wbinvd + invpcid (no targeted L1-cache flush supported!)
  - Arm: DCCISW + ICIALLU + TLBIALL + BPIALL

Methodology: Prime & Probe



1. Fill cache with own data

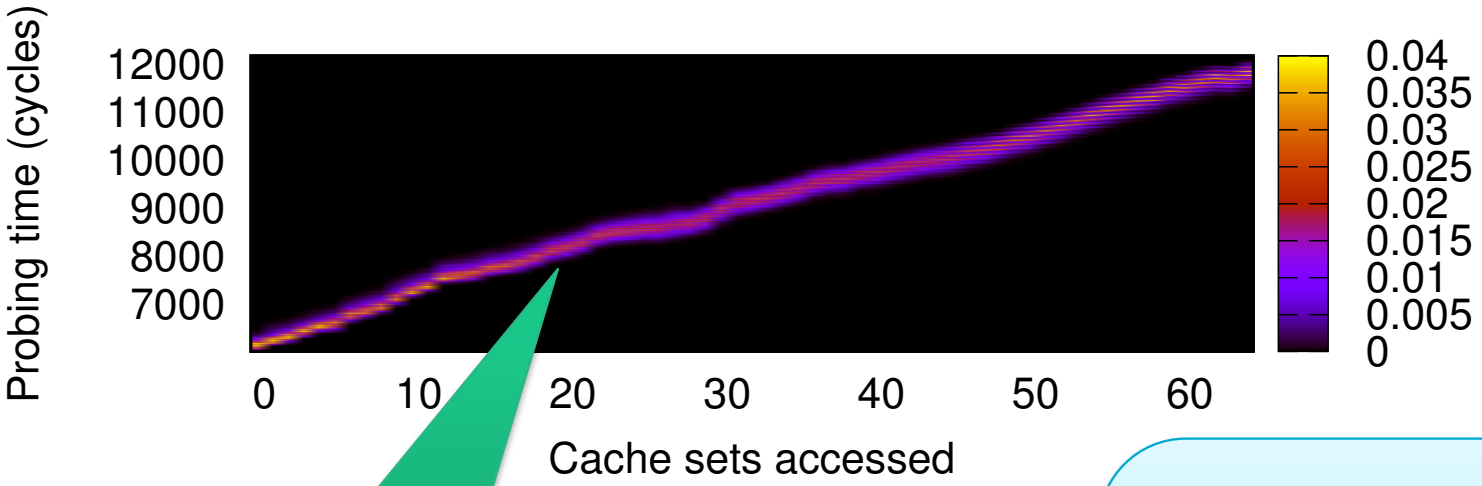
2. Touch *n* cache lines

Input Signal

3. Traverse cache, measure execution time

Output Signal

Methodology: Channel Matrix



Raw I-cache channel
Intel Sandy Bridge

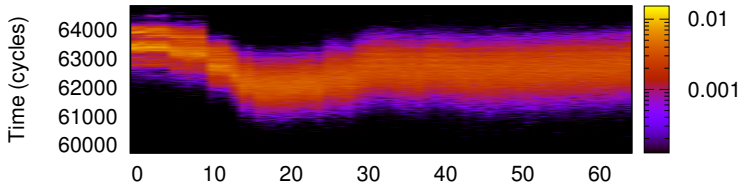
Horizontal variation indicates channel

Channel Matrix:

- Conditional probability of observing time, *t*, given input, *n*.
- Represented as heat map:
  - bright = high probability
  - dark = low probability

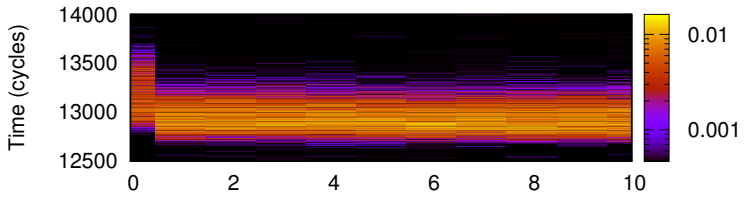
I-Cache Channel With Full State Flush

CHANNEL!



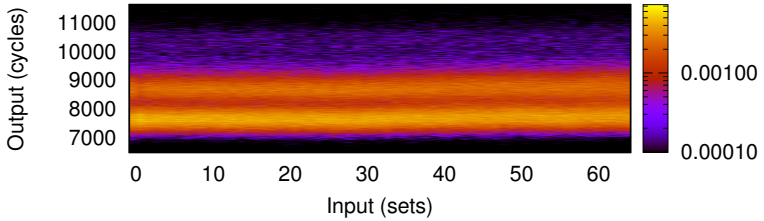
Intel Sandy Bridge

CHANNEL!



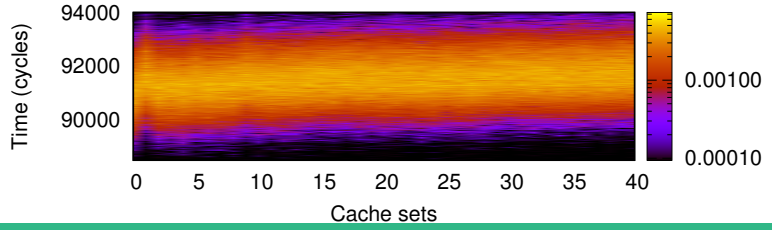
Intel Haswell

No evidence of channel



Intel Skylake

SMALL CHANNEL!

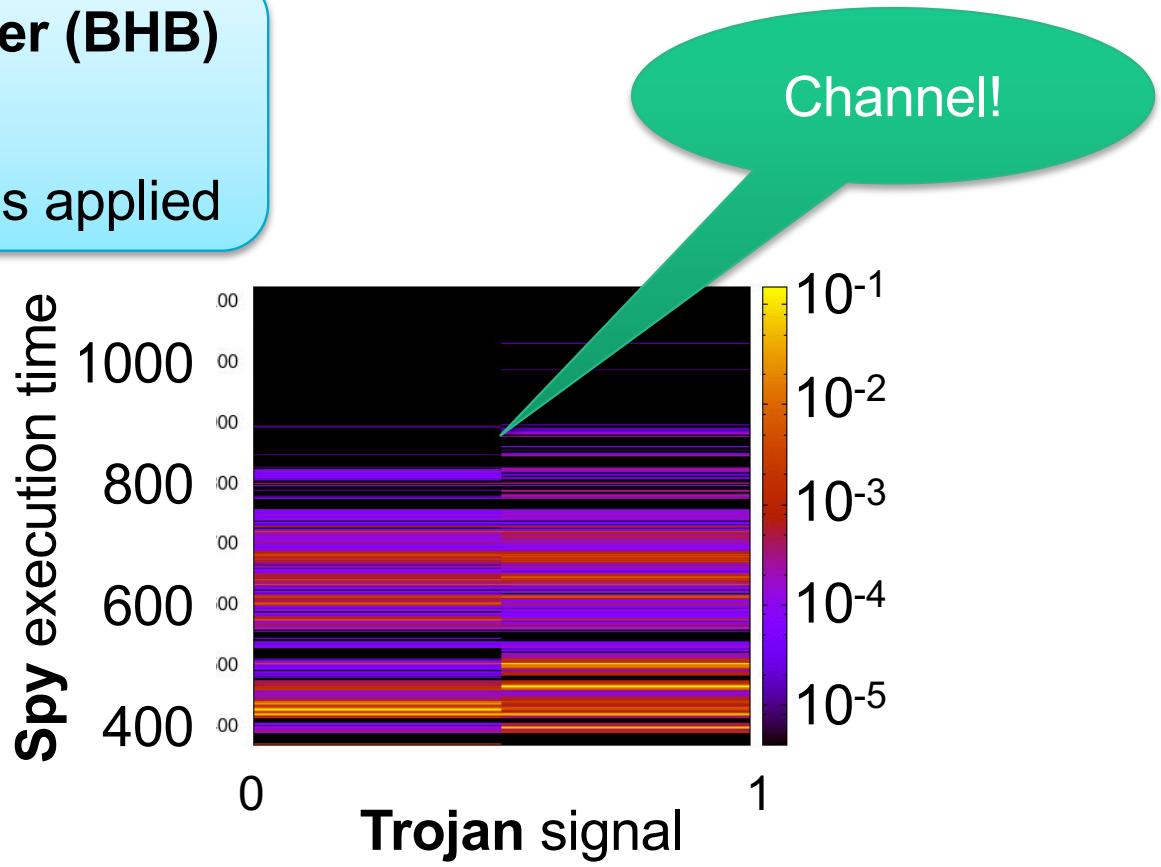


HiSilicon A53

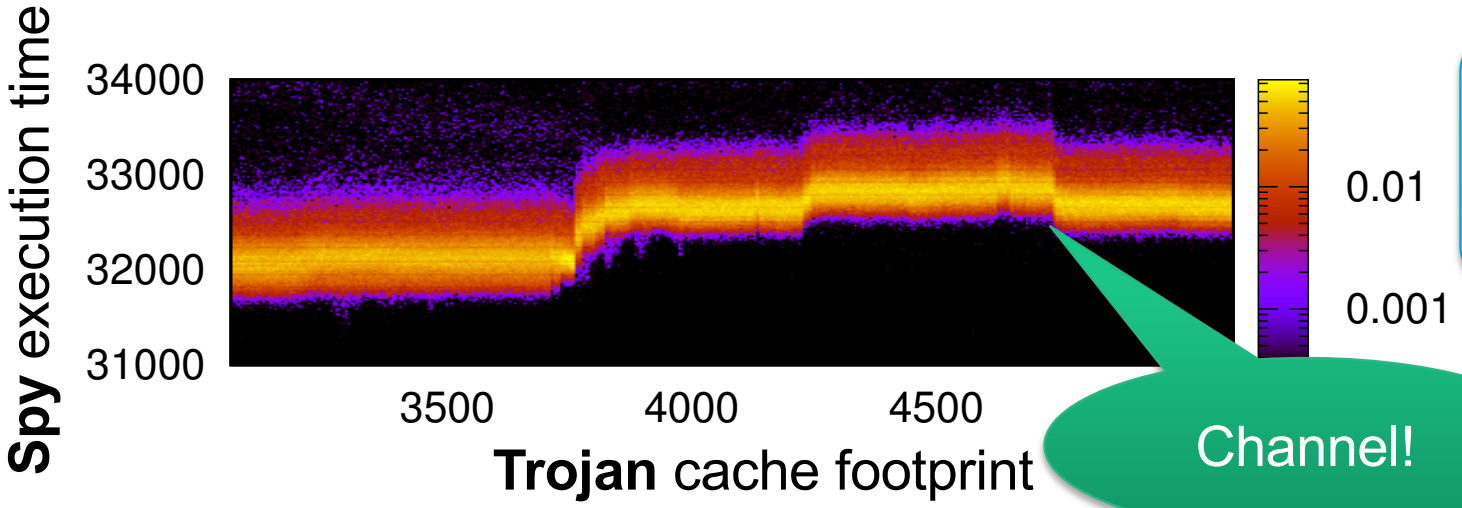
HiSilicon A53 Branch History Buffer

Branch history buffer (BHB)

- One-bit channel
- All reset operations applied



Example: Intel Haswell BTB



Branch target buffer

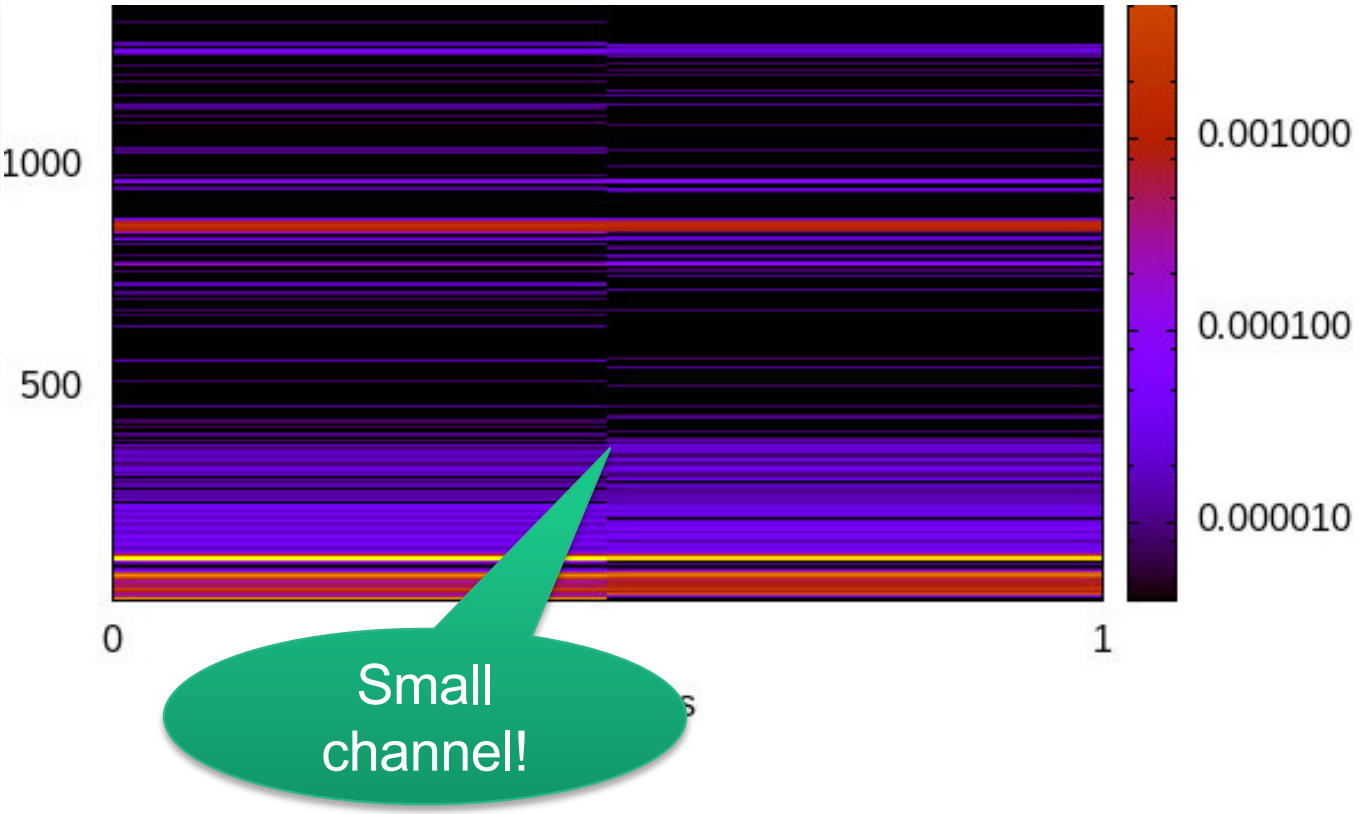
- All reset operations applied

Found residual channels in all recent Intel and ARM processors examined!

Intel Spectre Defences

Intel added *indirect branch control* (IBC) feature, which closes most channels, but…

Intel Skylake
Branch history buffer



https://ts.data61.csiro.au/projects/TS/timingchannels/arch-mitigation.pml

Requirements on Hardware

Hardware-Software Contract: ISA

- The ISA is a purely operational contract
  - sufficient to ensure *functional correctness*
  - abstracts away *time*
  - insufficient for ensuring either timing safety or security
- For security need an abstraction of microarchitectural state
  - essential for letting OS provide time protection

New HW/SW Contract: aISA

Augmented ISA supporting time protection

For all shared microarchitectural resources:

1. Resource must be partitionable or resetable
2. Concurrently shared resource must be partitioned
3. Resource accessed solely by virtual address must be reset and not concurrently accessed
   - Implies cannot share HW threads across security domains!
4. Mechanisms must be sufficiently specified for OS to partition or reset
   - Must be constant time or of specified, bounded latency
5. OS must know if resettable state is derived from data, instructions, data addresses or instruction addresses

Cost of Reset

- **Flushing on-core state** is not a performance issue:
  - no cost when not used
  - direct flush cost should for dirty L1-D in the order of 1µs
  - direct flush cost for everything else in the order of 100 cycles
  - indirect cost is negligible, if used on security-partition switch
    - eg VM switch, 10–100 Hz rate
    - no hot data in cache after other partition's execution

- **Hardware support (eg targeted L1 flush) is essential!**

Summary

- Timing channels are a mainstream security threat
- They are based on competition for shared hardware
- Prevention through OS-enforced *time protection*
  - OS must prevent sharing by partitioning or flushing
- The shared hardware is hidden by the ISA, the present HW-SW contract
  - OS cannot systematically prevent timing channels based on ISA
- Need a new, security-oriented contract, the aISA
  - aISA must expose enough microarchitecture for OS to enforce time protection

Thank You

Gernot Heiser
gernot.heiser@data61.csiro.au | @GernotHeiser

https://trustworthy.systems