Security Needs a New Hardware-Software Contract

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

https://trustworthy.systems

CSIRO

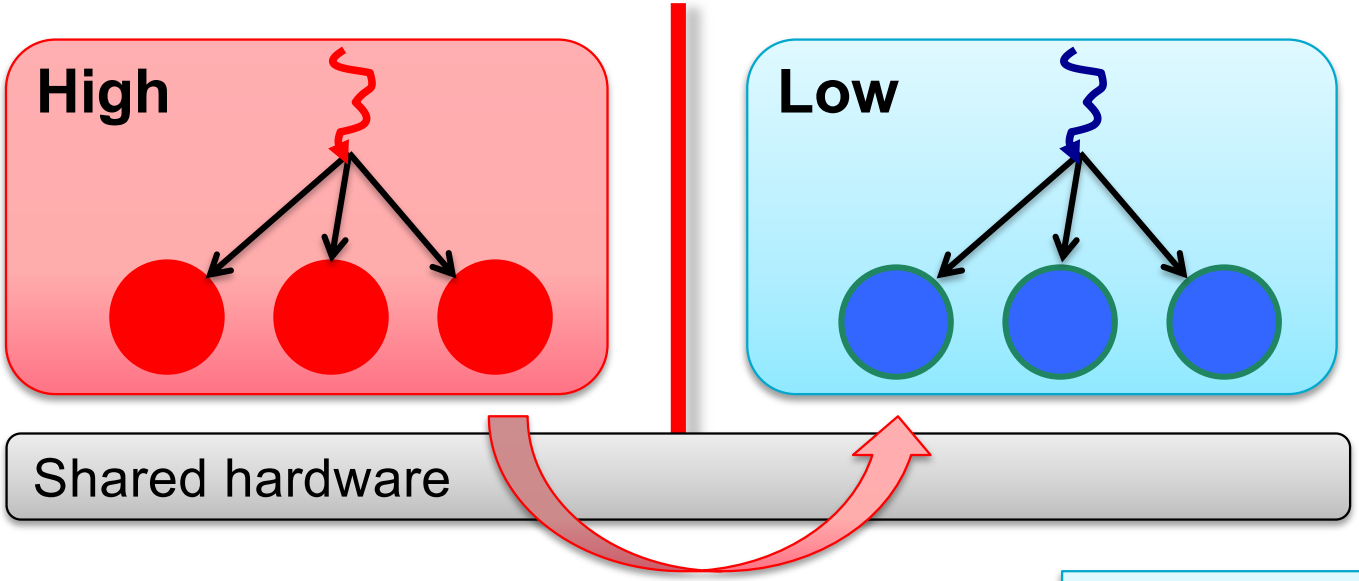
My takeaway from this morning

1. 2018-01-03: Spectre & Meltdown happen
2. Houston, we have a problem!
3. Let's refine contract so programmers can write non-leaking programs

- **Unreasonable burden on programmer**
- **Unreasonably fine-grained**

We Need Time Protection

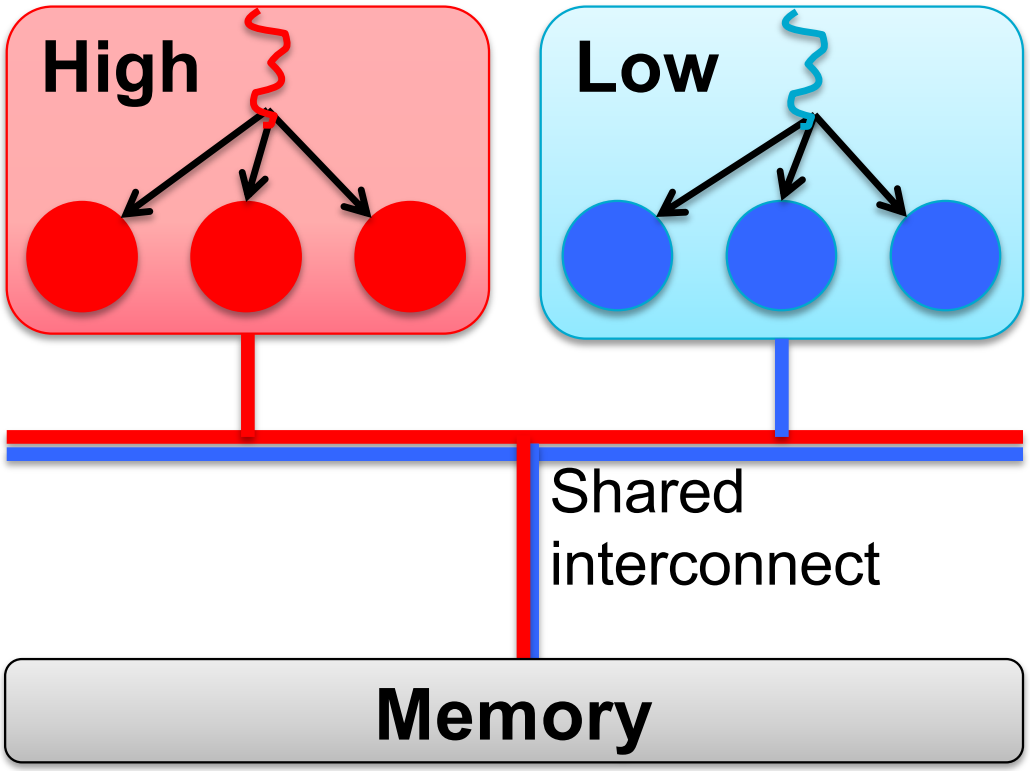
Cause: Temporal Interference



Affect execution speed

- Inter-process interference
- Competing access to micro-architectural features
- Hidden by the HW-SW contract!

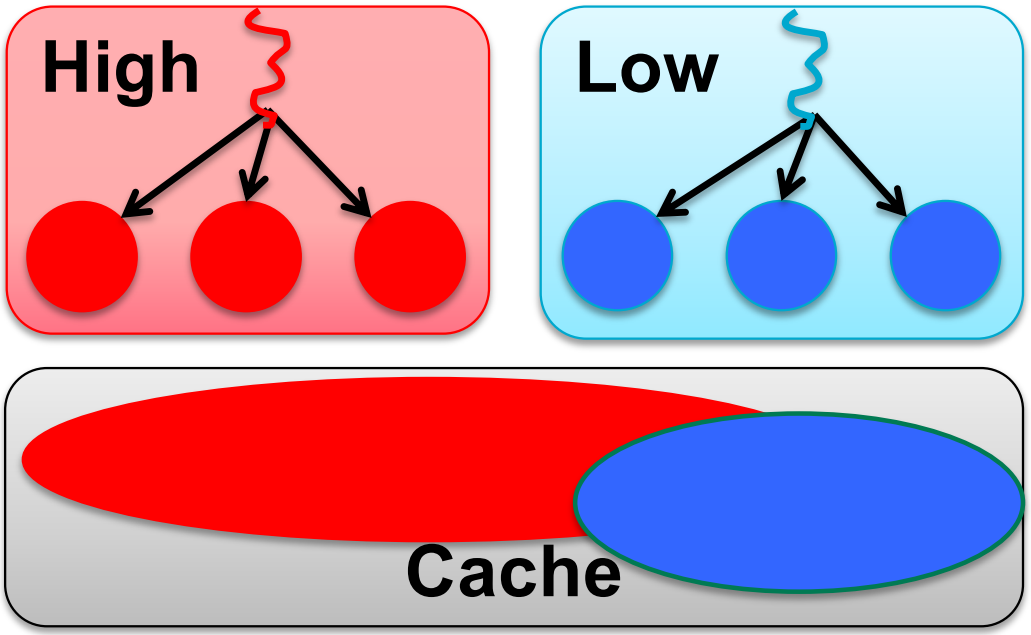
Sharing 1: Stateless Interconnect



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



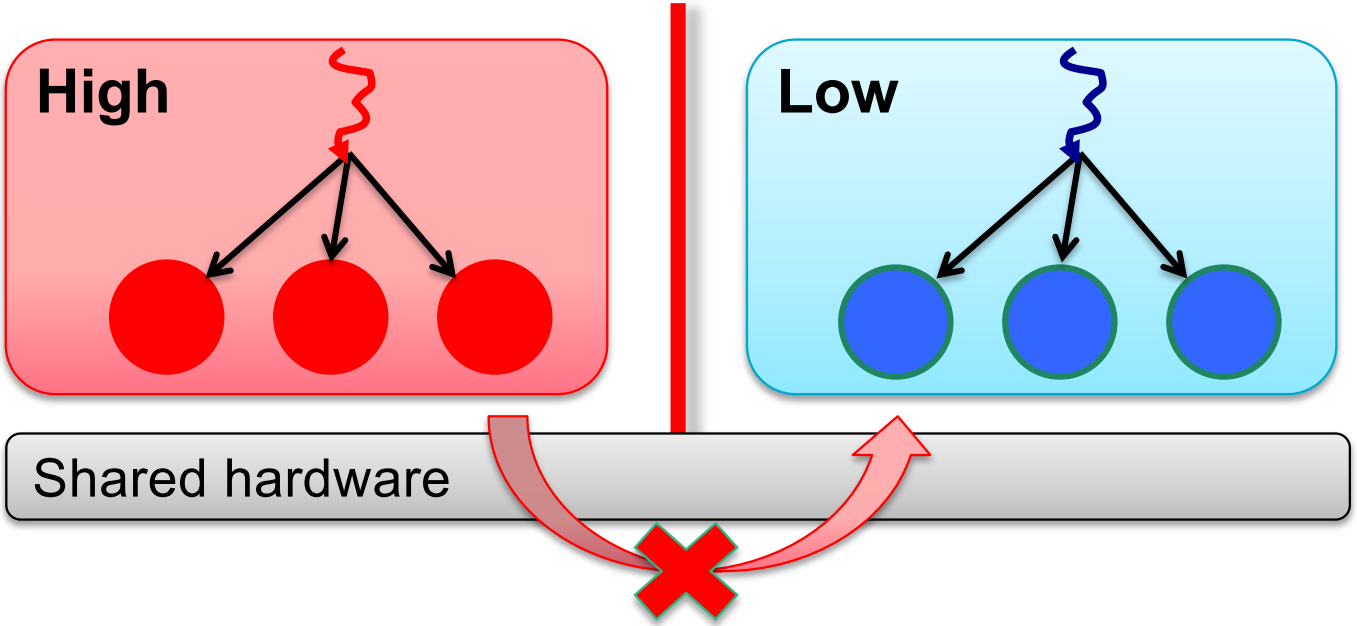
HW is *capacity-limited*

- 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

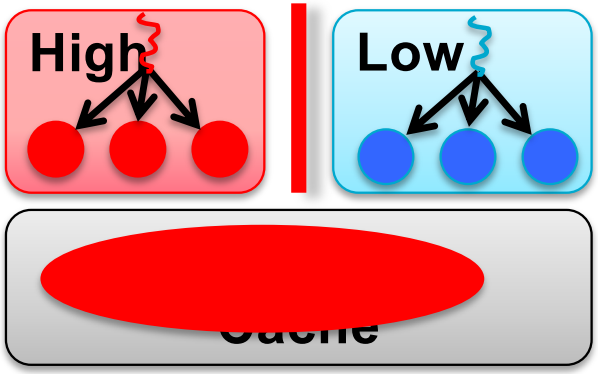
Time Protection



Preventing interference is core duty of the OS!

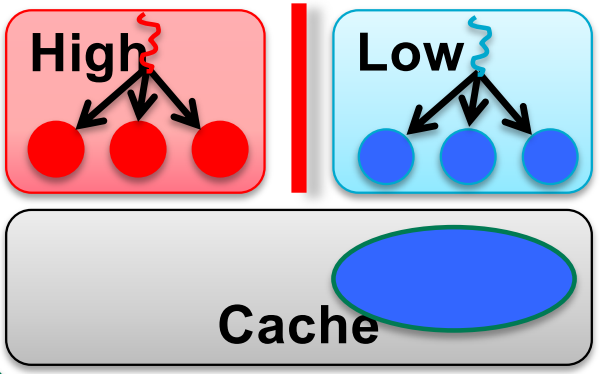
- *Memory protection* is well established
- We also need *time protection*

Time Protection: Control Sharing

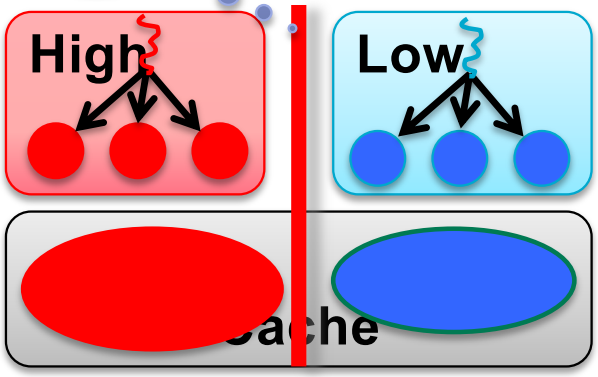


Flush

Context Switch



Partition



Need both!

Cannot partition on-core caches (L1, TLB, branch predictor, prefetchers)

- virtually-indexed
- OS cannot control

Flushing useless for concurrent access

- between HW threads, cores
- for stateless HW

Requirements For Time Protection

Off-core state & stateless HW

On-core state

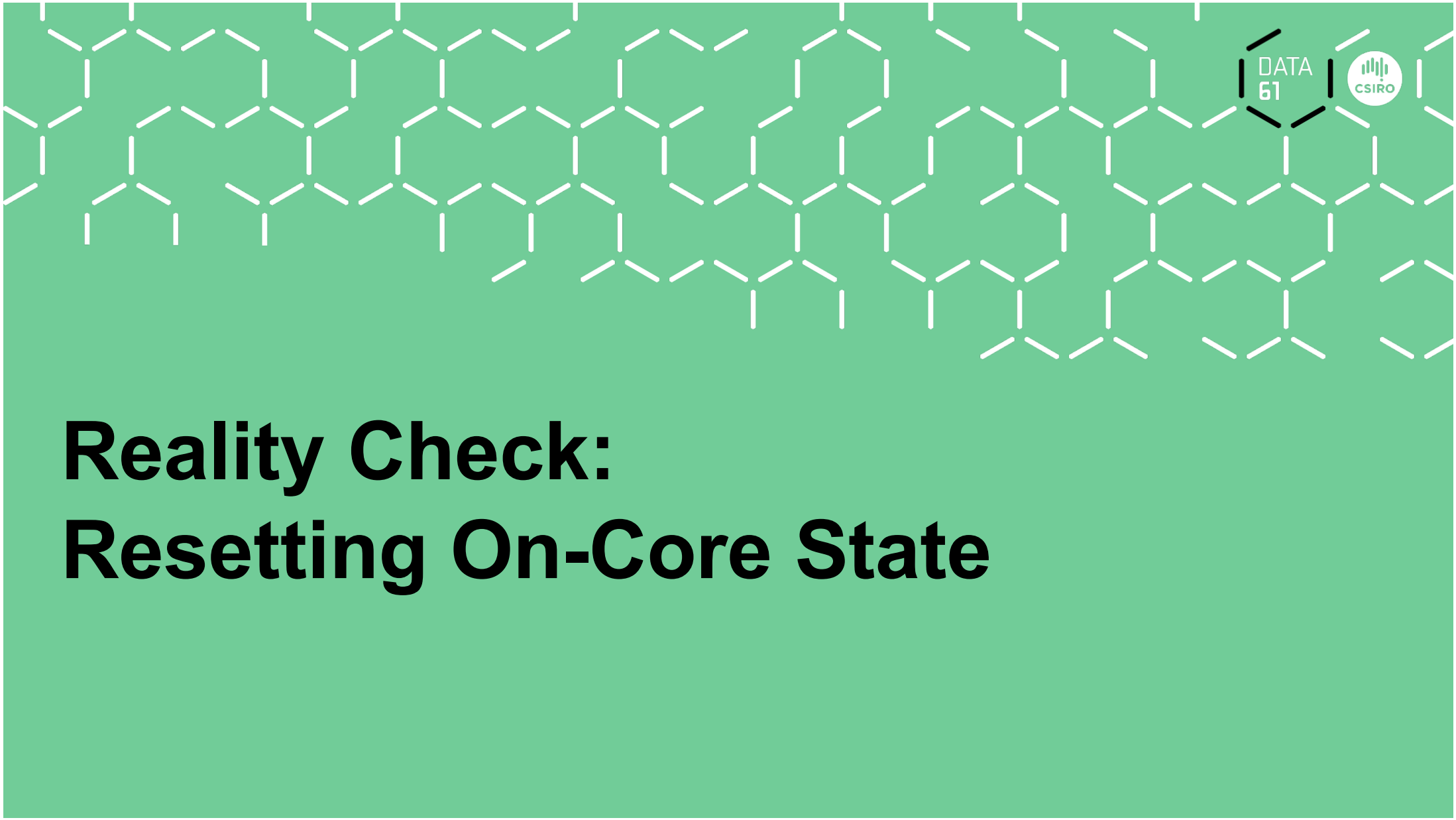
Timing channels can be closed *iff* the OS can

- partition or
- reset

all shared hardware

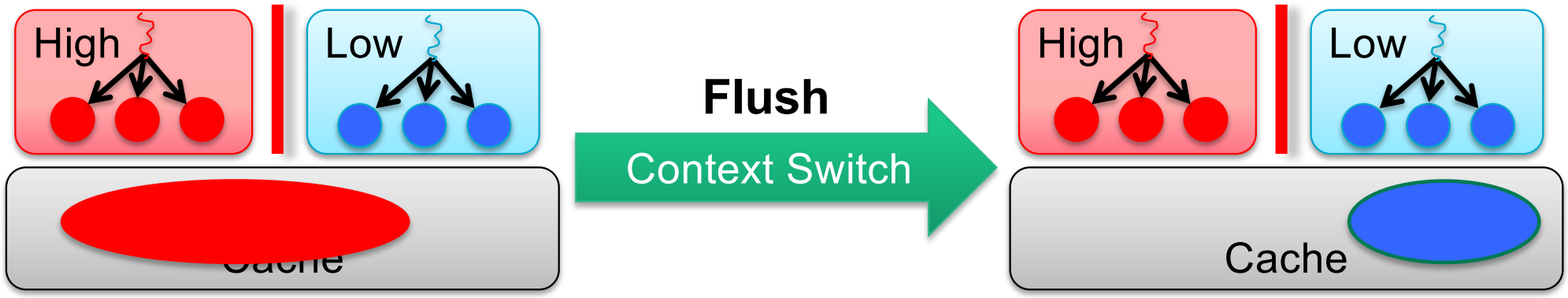
Problem:

- **Cannot partition interconnect bandwidth**
- **Cannot prevent inter-core covert channels!**



Reality Check:
Resetting On-Core State

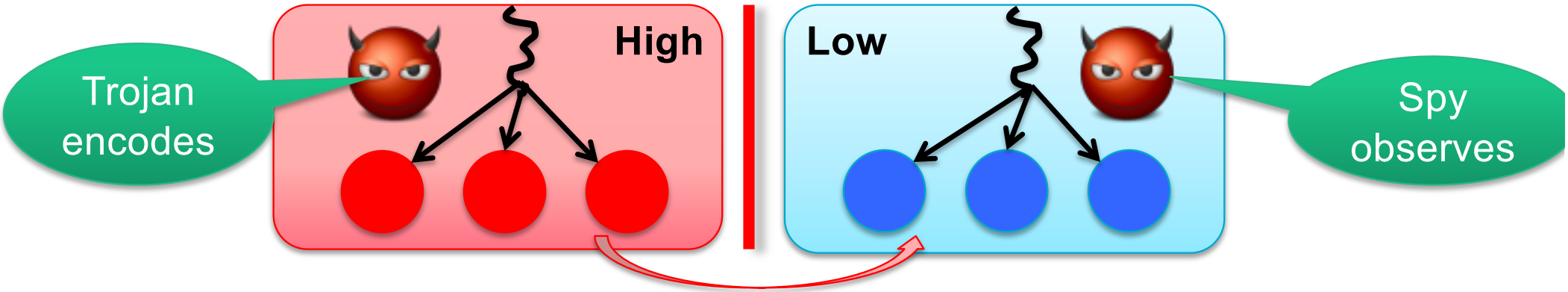
Evaluating Intra-Core Channels



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

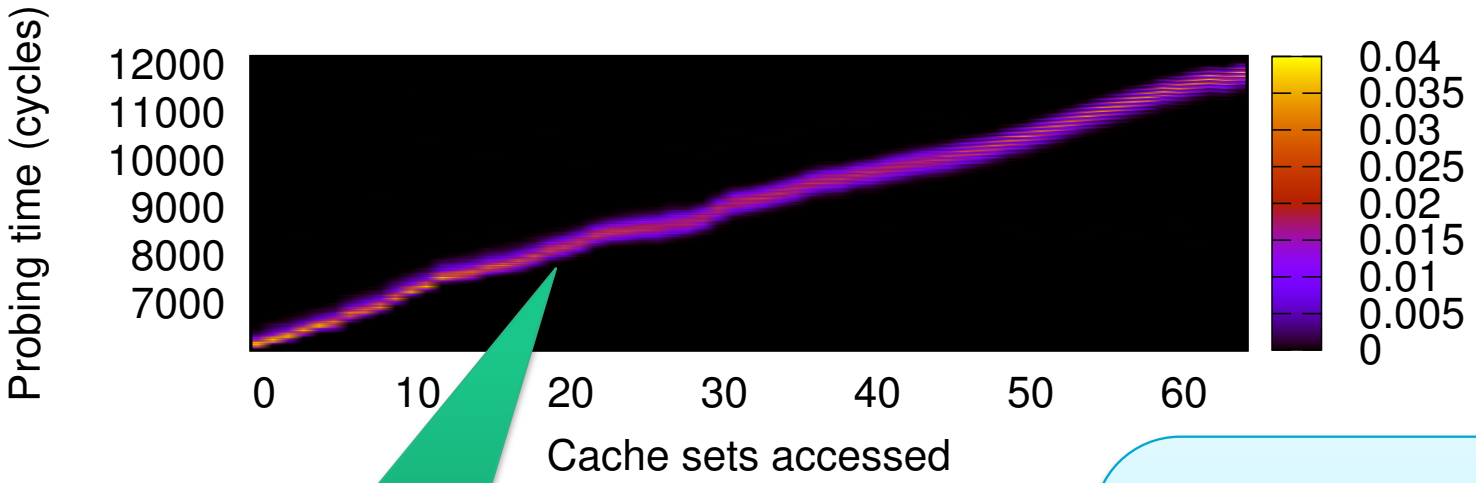


1. Fill cache with own data
2. Touch *n* cache lines
3. Traverse cache, measure execution time

Input Signal

Output Signal

Methodology: Channel Matrix



Raw I-cache channel
Intel Sandy Bridge

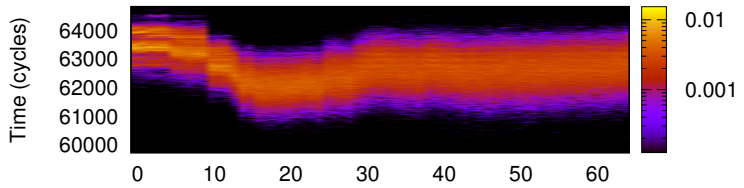
Horizontal variation indicates channel

Channel Matrix:

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

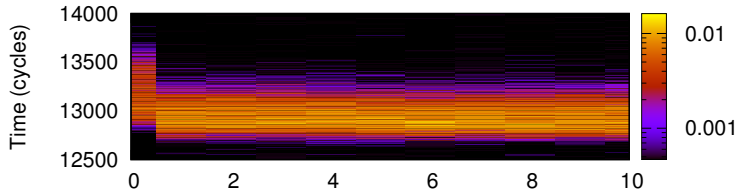
I-Cache Channel With Full State Flush

CHANNEL!



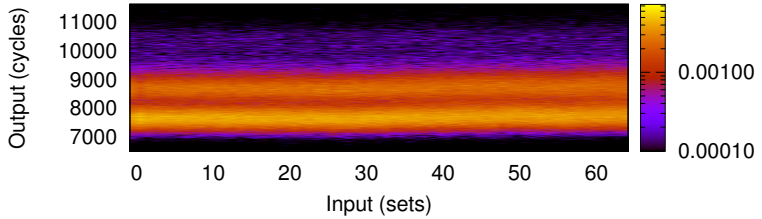
Intel Sandy Bridge

CHANNEL!



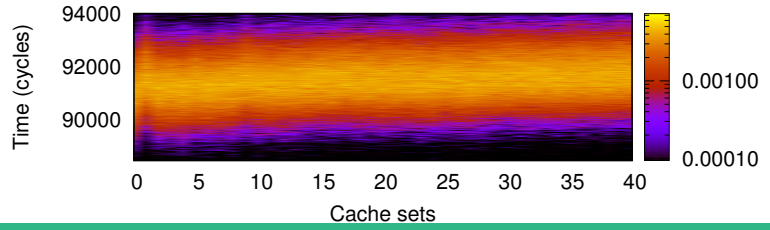
Intel Haswell

No evidence of channel



Intel Skylake

SMALL CHANNEL!

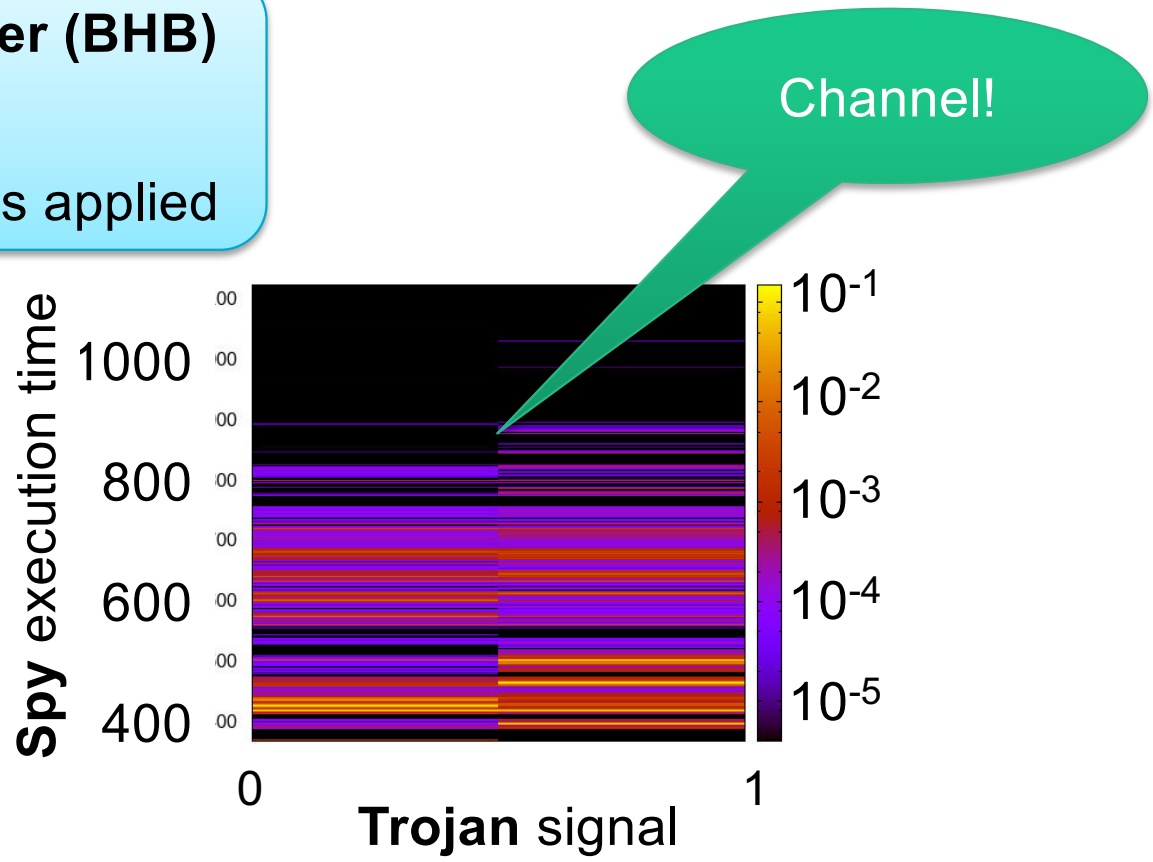


HiSilicon A53

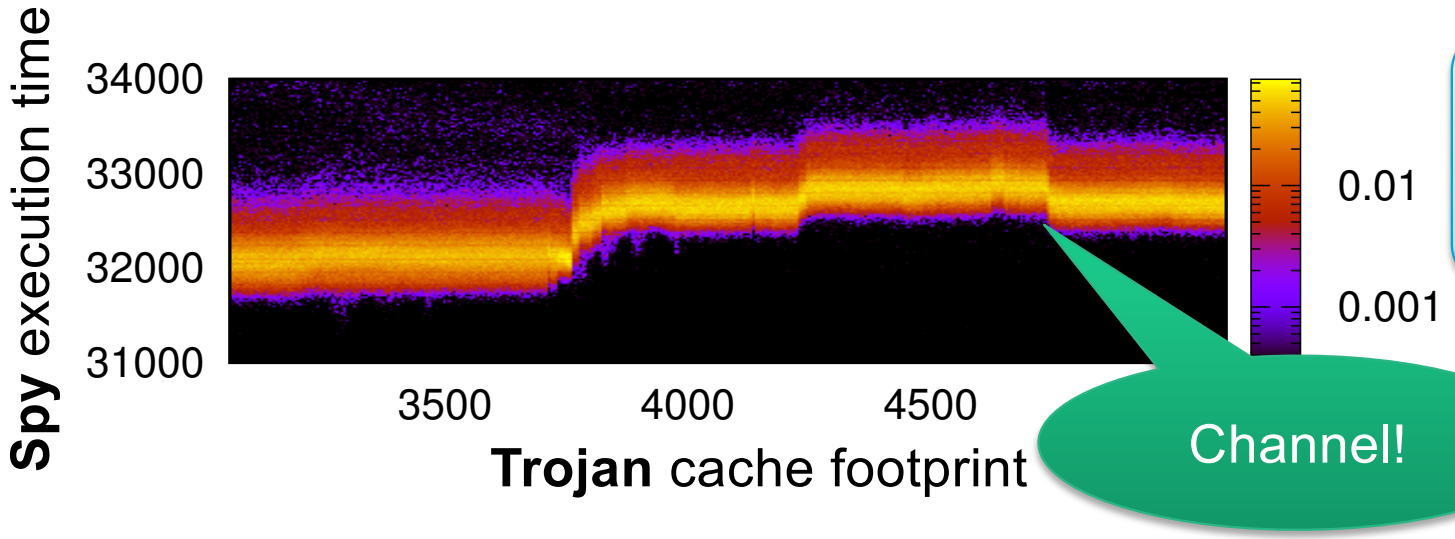
HiSilicon A53 Branch History Buffer

Branch history buffer (BHB)

- One-bit channel
- All reset operations applied



Example: Intel Haswell BTB



Branch target buffer

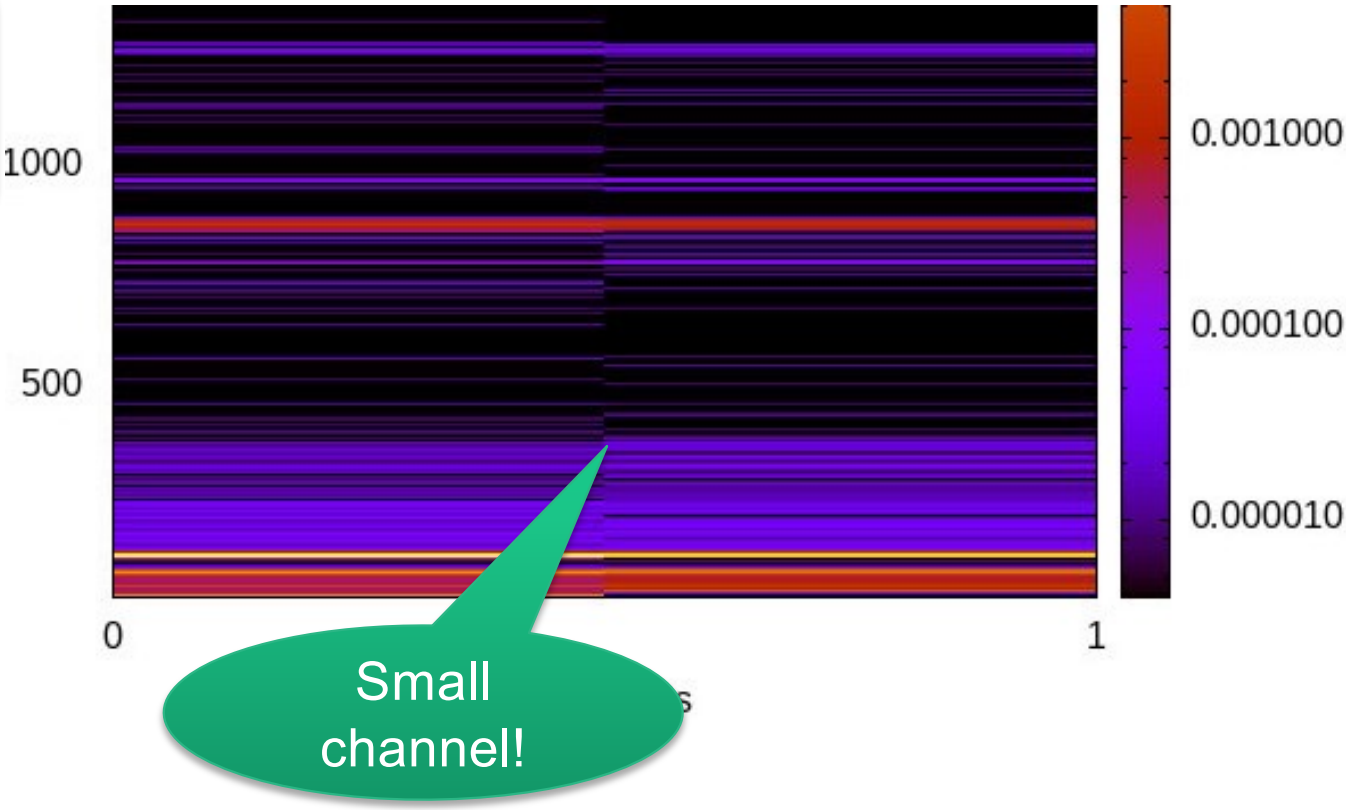
- All reset operations applied

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



Ge et al., APSys'18

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

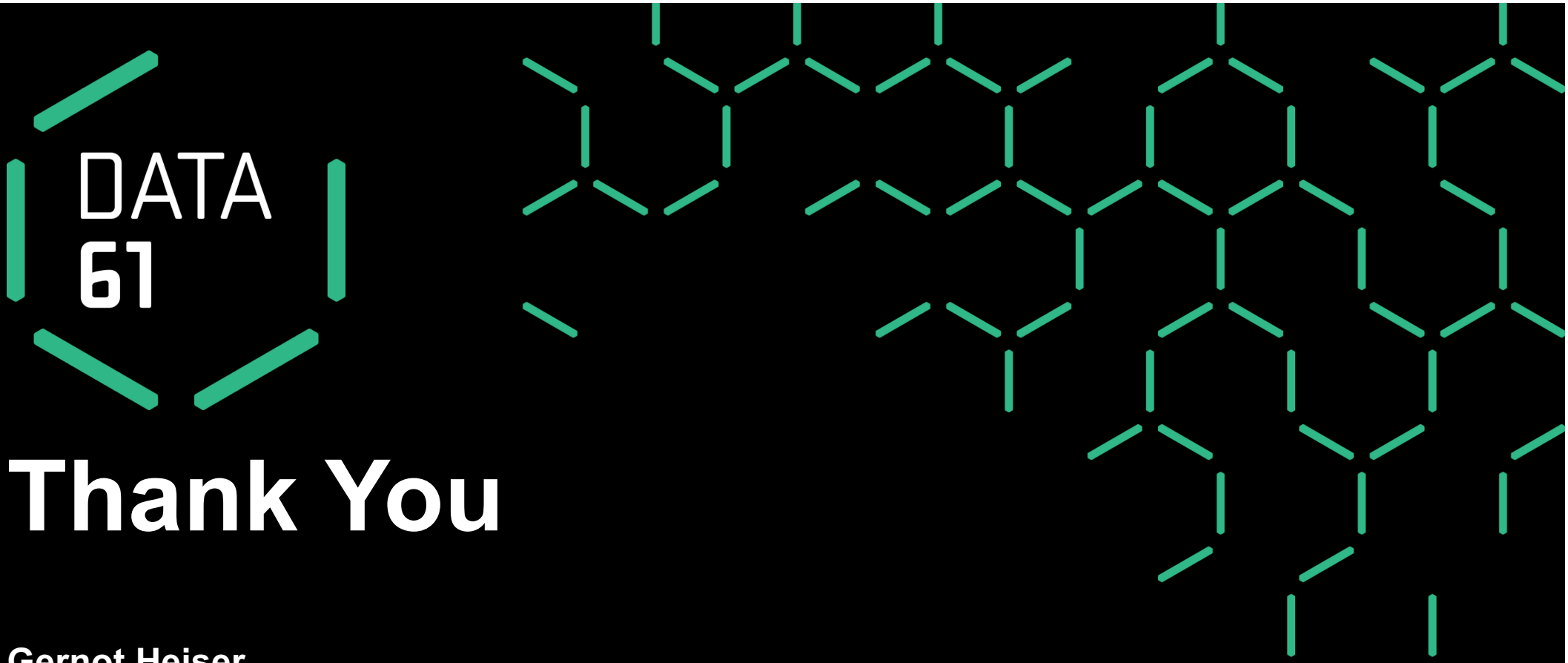
Augmented ISA supporting time protection

For all shared microarchitectural resources:

1. Resource must be partitionable or resetable
2. Concurrently shared resources must be partitionable
3. Resource accessed solely by virtual address must be resettable and not concurrently accessed
4. Mechanisms must be sufficiently specified for OS to use
   - Must be constant time or of specified, bounded latency
5. *Desirable:* OS must know if resettable state is derived from data, instructions, data addresses or instruction addresses

My Message

Treat the OS as your friend, not a nuisance!



Thank You

Gernot Heiser

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

https://trustworthy.systems