



Cross-VM and Cross-Processor Covert Channels Exploiting Processor Idle Power Management

Paizhuo Chen, Lei Li, **Zhice Yang**

ShanghaiTech University



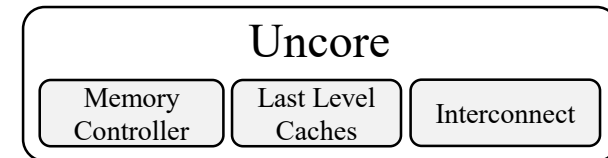
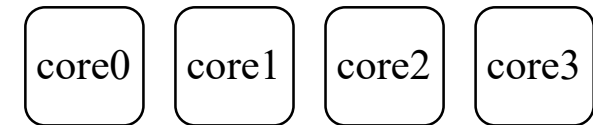
上海科技大学
ShanghaiTech University

Overview

- A new type of covert/side channel
 - Rooted in **processor idle power management**
 - Not based on cache or memory
 - Cross-core and cross-processor
 - Cross-VM
 - in-house testbed
 - public cloud

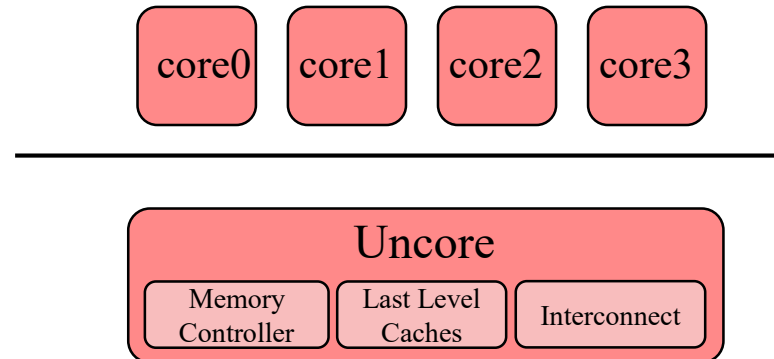
Processor Power Management

- **Active** power management
 - Dynamic Voltage and Frequency Scaling (DVFS)
 - e.g., Intel SpeedStep
 - Well studied
 - heavy load -> high frequency/power
 - power budget limit -> covert/side channels
- **Idle** power management
 - Turn off/down hardware components when there are no tasks
 - Must take into account hardware dependencies
 - uncore is shared by cores



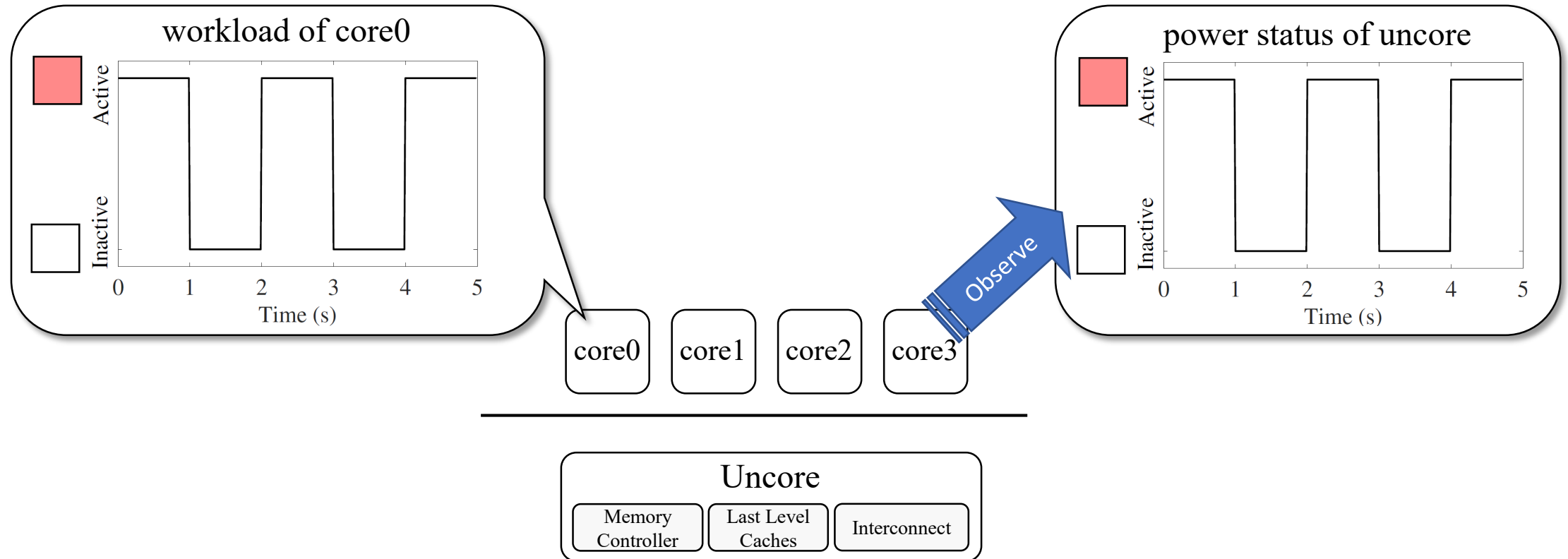
Idle Power Management Dependency

- When any core is active, the uncore stays active
 - To support shared functions
- The uncore is turned off when all cores are idle
 - To save power



Idle Power Management Dependency

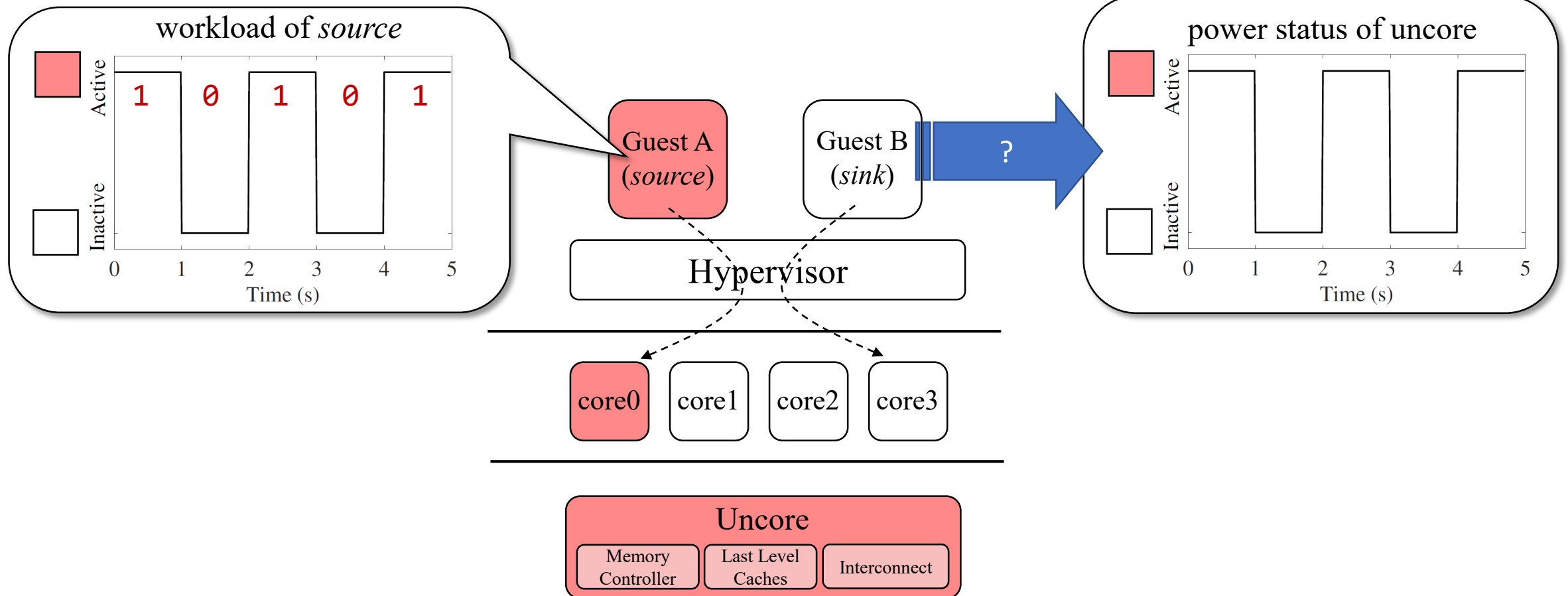
- Lead to covert/side channels
 - Use uncore power status to probe the activity of cores



Cross VM Attacks

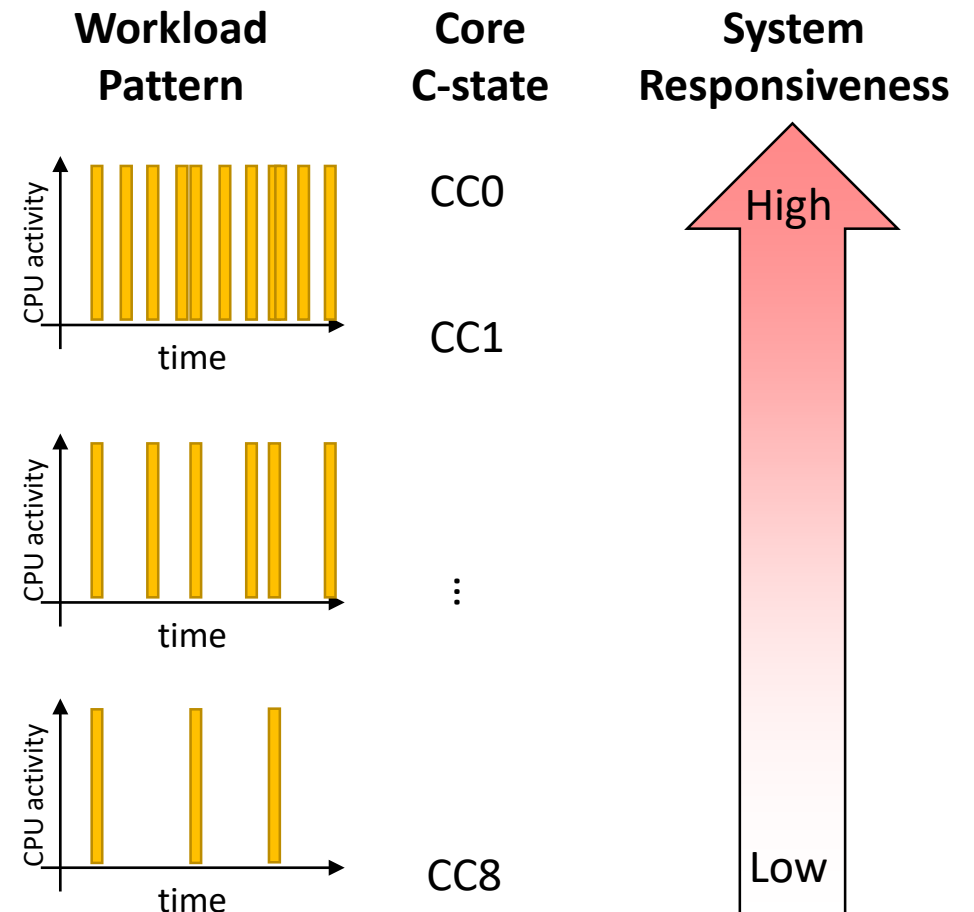
Covert Channel

- How to probe the uncore power status within a VM guest ?



Idle Power Management - Core

- Core low-power idle state
 - Core C-state (CC)
 - CC0 is the active state
 - subject to active power management
 - CC1, CC2, ..., CCn are the “sleeping” states
 - the core does not execute instructions
 - Numerically **larger (deeper) CCs** imply
 - more hardware components are turned off
 - more power saving
 - **take more time to switch to the active state**
- Managed by OS
 - Based on workload pattern statistics, put cores into appropriate CCs
 - to balance latency performance and power saving
 - light load -> deep CC
 - heavy load -> light CC



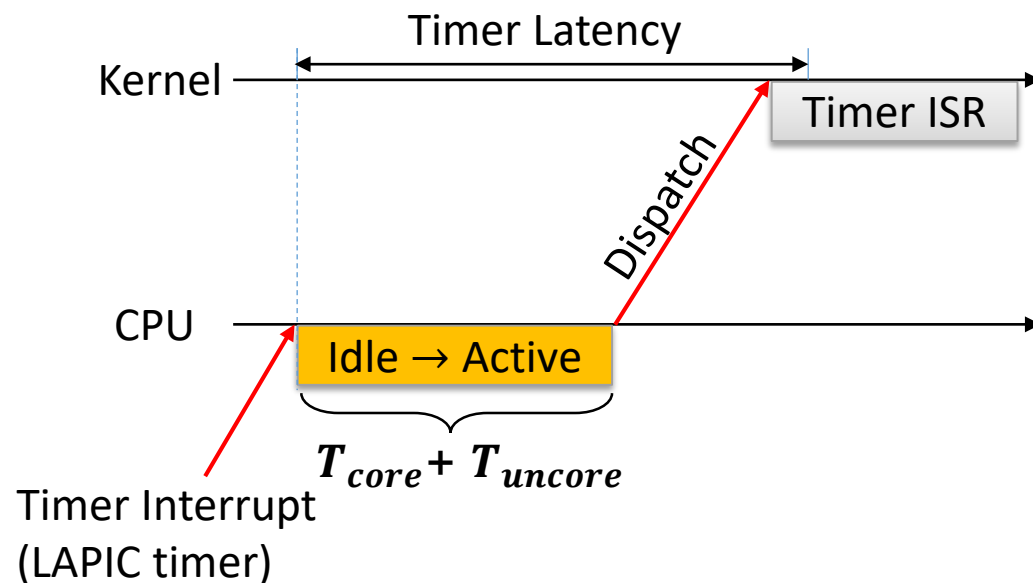
Idle Power Management - Unore

- Uncore idle power management mechanisms
 - Package C-state (PC)
 - similar to core C-state
 - control on/off of MC, LLC, etc.
 - Uncore Frequency Scaling
 - control clock frequency of uncore components
 - Other mechanisms
- Managed by processor firmware
 - Blackbox

Deeper Idle Status -> Larger Latency

Probing Uncore Power Status in VM

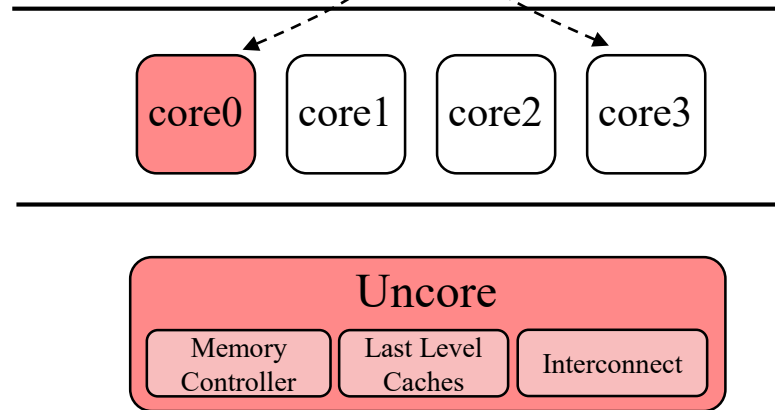
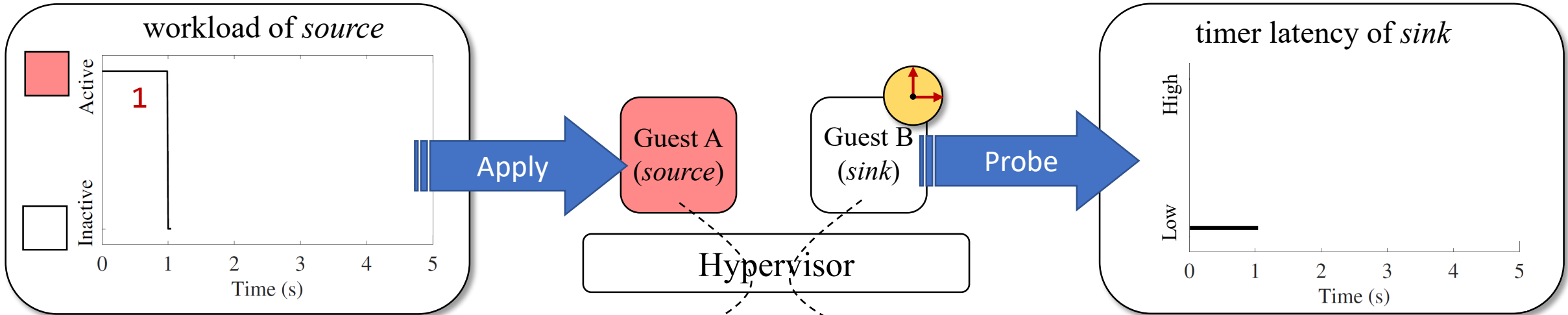
- Approach: timer latency
 - Schedule a periodic timer
 - Timer interrupts will be delayed if the CPU is idle when interrupts arrive
 - deep idle status -> large timer latency
 - Both core (T_{core}) and uncore (T_{uncore}) contribute to the timer latency



Core C-state	T_{core}	T_{uncore}
CC0	12	1
CC1	20	11
CC1E	20	40
CC3	50	75
CC6	62	102
CC7s	62	146
CC8	62	304

Decoupled Timer Latency (μs)

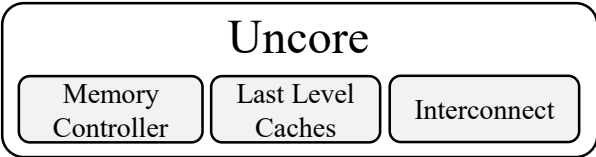
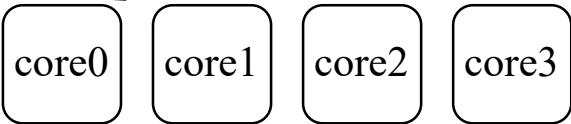
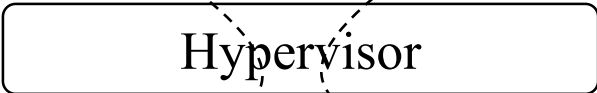
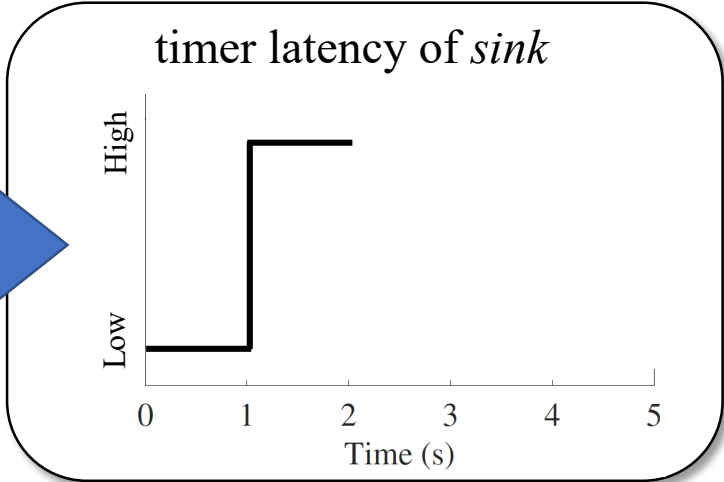
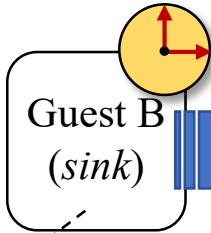
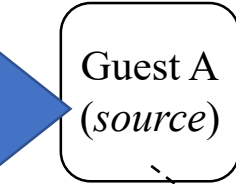
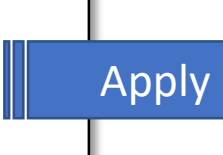
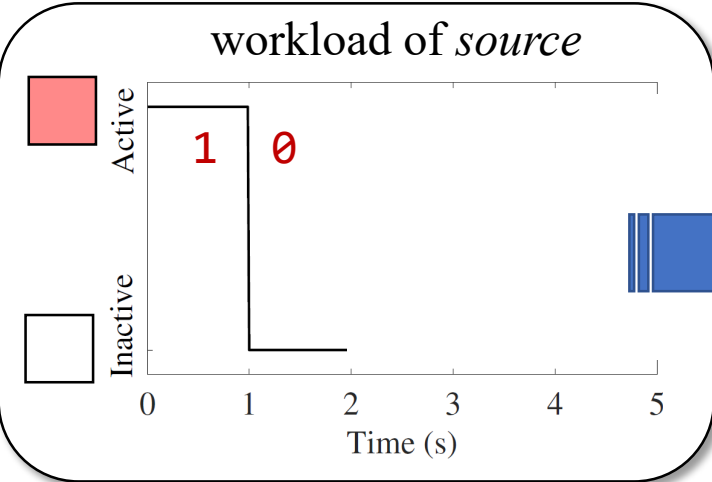
Example



Core C-state	T_{core}	T_{uncore}
CC0	12	1
CC1	20	11
CC1E	20	40
CC3	50	75
CC6	62	102
CC7s	62	146
CC8	62	304

Decoupled Timer Latency (μ s)

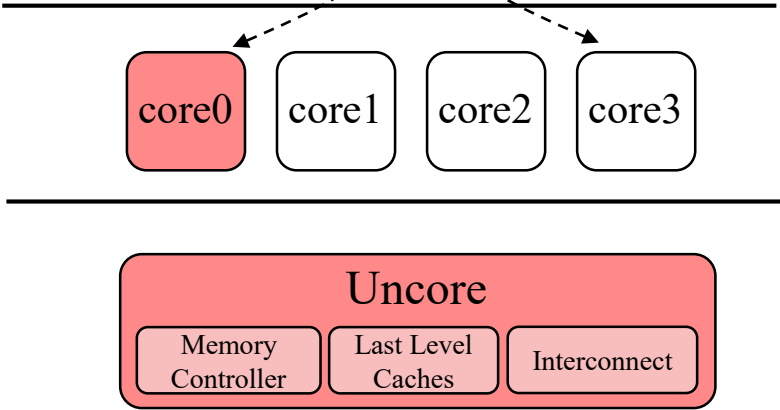
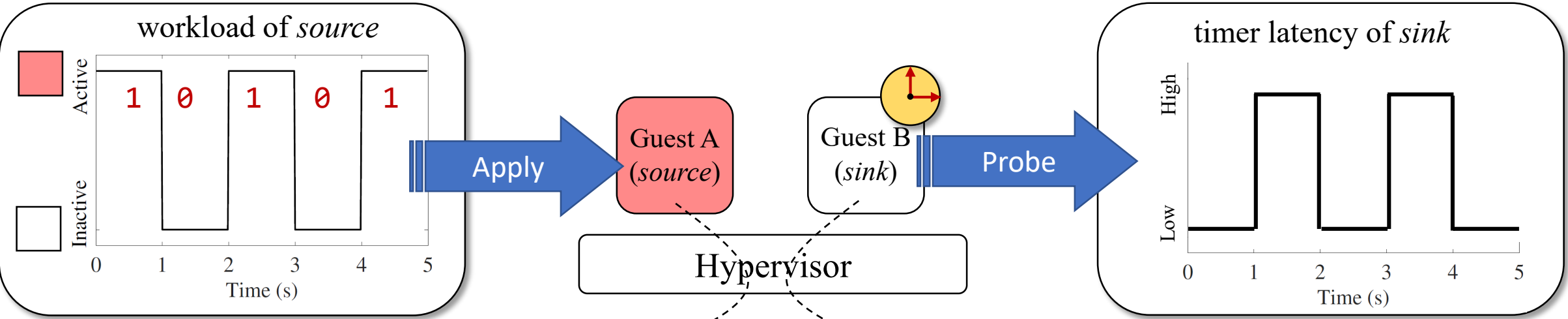
Example



Core C-state	T_{core}	T_{uncore}
CC0	12	1
CC1	20	11
CC1E	20	40
CC3	50	75
CC6	62	102
CC7s	62	146
CC8	62	304

Decoupled Timer Latency (μ s)

Example



Core C-state	T_{core}	T_{uncore}
CC0	12	1
CC1	20	11
CC1E	20	40
CC3	50	75
CC6	62	102
CC7s	62	146
CC8	62	304

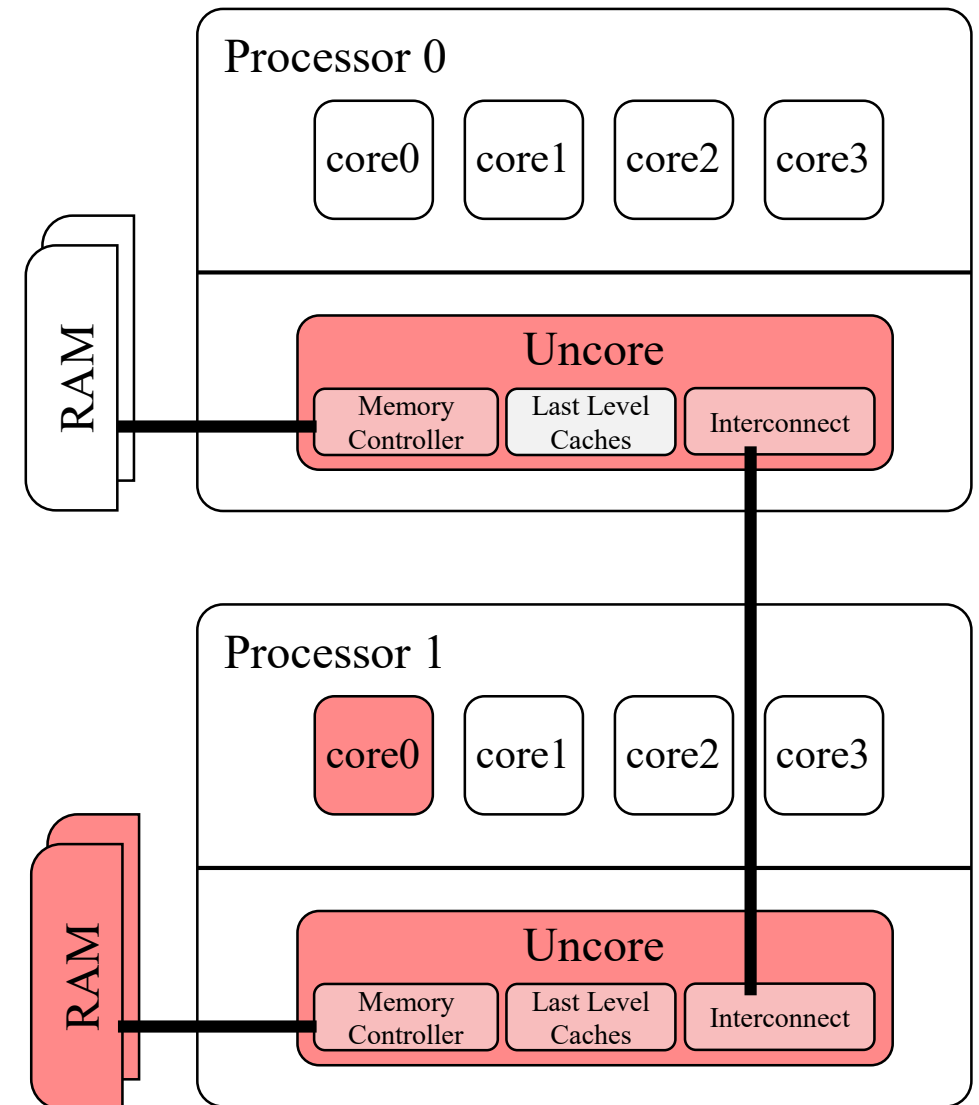
Decoupled Timer Latency (μ s)

Implementation

- KVM hypervisor
 - Use default configurations
 - *sink* and *source* are hosted by different VM guests
- Cross-core
 - *sink* and *source* are assigned to different cores
 - i5 6500
 - 2.3 kbps
 - Xeon E5 2630v4
 - 200 bps
- Cross-processor
 - *sink* and *source* are assigned to different sockets
 - 2 × Xeon E5 2630v4
 - 120 bps

Affected by processor architectures, power management preferences, OS versions, etc.

To support remote memory access



Implementation

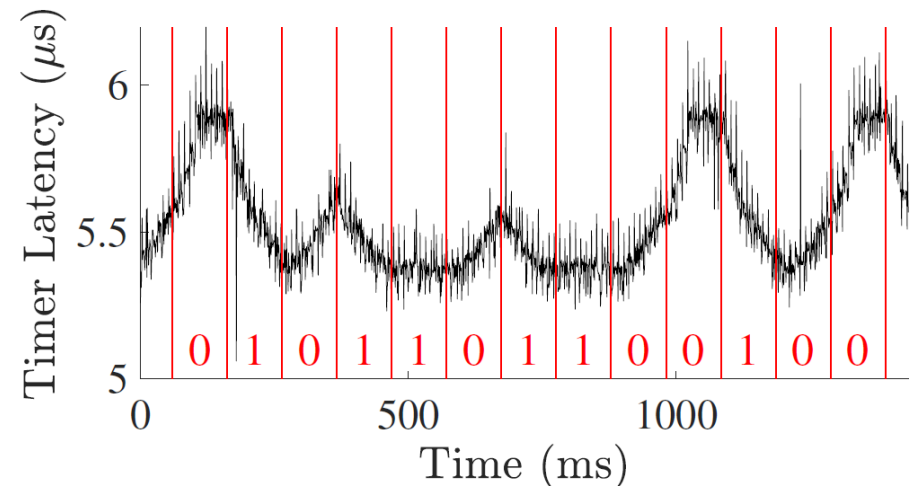
- KVM hypervisor
 - Use default configurations
 - *sink* and *source* are hosted by different VM guests
- Cross-core
 - *sink* and *source* are assigned to different cores
 - i5 6500
 - 2.3 kbps
 - Xeon E5 2630v4
 - 200 bps
- Cross-processor
 - *sink* and *source* are assigned to different sockets
 - 2 × Xeon E5 2630v4
 - 120 bps

Affected by processor architectures, power management preferences, OS versions, etc.

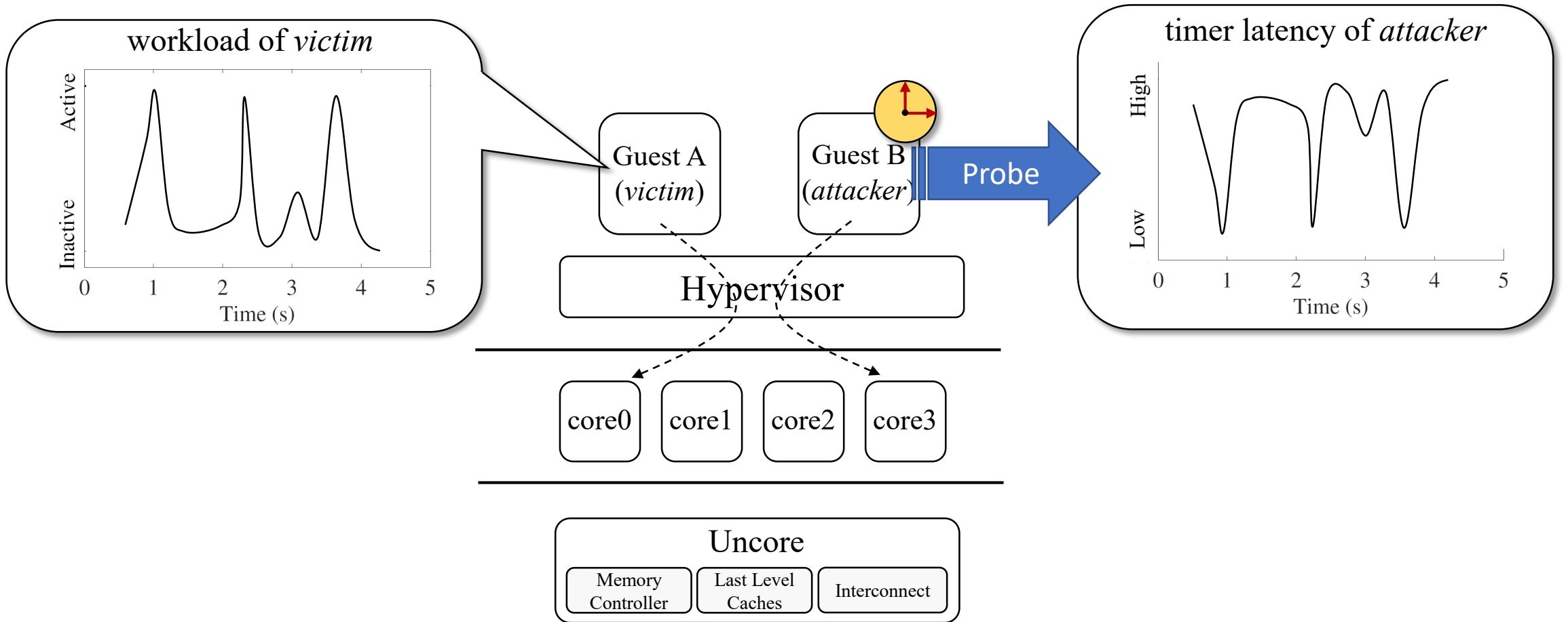
To support remote memory access

- Dedicated host in public cloud
 - Amazon EC2 c5
 - 2× Xeon 8124M
 - Azure Dsv3_Type2
 - Xeon 8171M
 - Performance
 - Several bps

Raw Timer Latency Trace of Amazon EC2

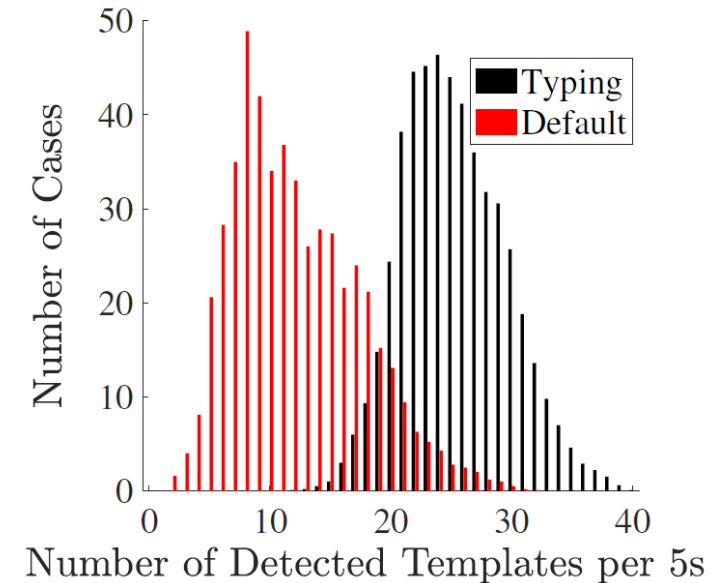
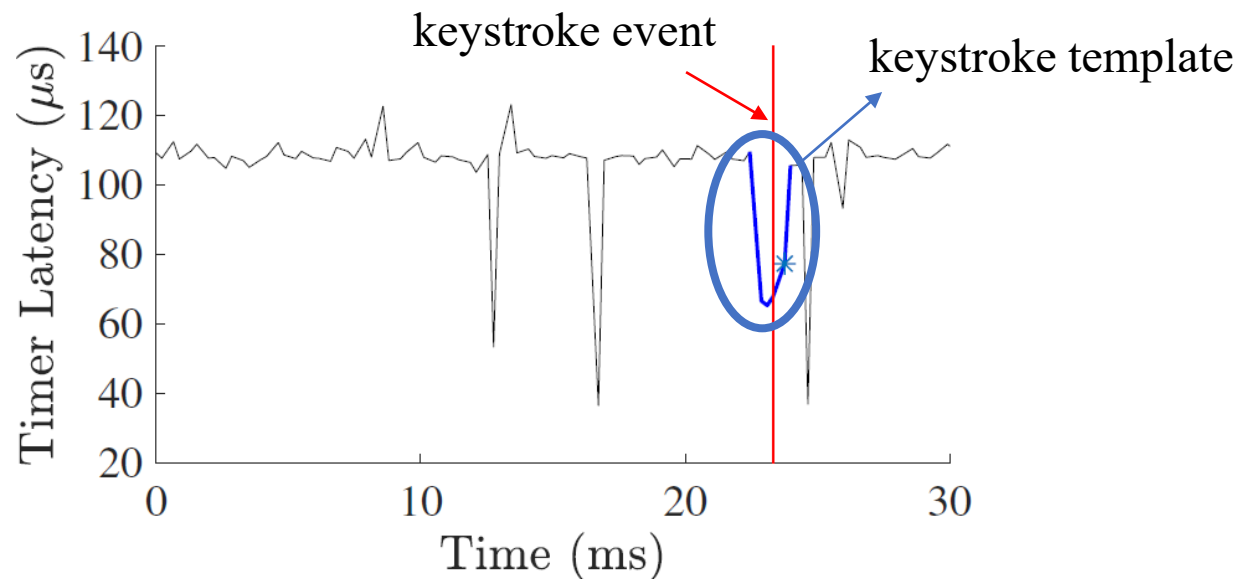


Side Channel



Example: Probing SSH Keystroke Activity

- SSH keystroke invokes CPU activities
 - To process network stack, decryption, stdin stream, etc.
- Core activity -> uncore activity -> latency variation
 - Unique latency footprint of SSH keystrokes is observed



Countermeasures

- Restrict timer resolution
- Adopt performance-oriented power plans
- Detect high-rate interrupts

Conclusion

- A new type of covert/side channel
 - Cross-core and cross-processor
 - Idle power management dependency
 - core's activity is reflected in the power status of shared hardware, e.g., uncore
 - Cross-VM
 - Timer/interrupt latency

Thank You !

Contact: Zhice Yang yangzhc@shanghaitech.edu.cn