How Sharp is SHARP?

WOOT’19@USENIX-SECURITY

Dixit Kumar (Indian Institute of Technology Kanpur),
Chavhan Sujeet Yashavant (Indian Institute of Technology Kanpur),
Biswaabandan Panda (Indian Institute of Technology Kanpur), and
Vishal Gupta (Manipal University Jaipur and Indian Institute of Technology Kanpur)
SHARP [Yan et al., ISCA ‘17 ]
SHARP [Yan et al., ISCA ‘17 ]

Secure hierarchy-aware cache replacement policy
SHARP [Yan et al., ISCA ‘17 ]

Secure hierarchy-aware cache replacement policy

Mitigation for side-channel attacks
Side Channel Attacks

Attacker

Victim
Eviction based Cache Attacks: (Prime+Probe)
Eviction based Cache Attacks: (Prime+Probe)

Step 0: Attacker *fills* the entire shared cache (set)
Eviction based Cache Attacks: (Prime+Probe)

Step 0: Attacker *fills* the entire shared cache (set)

Step 1: Victim *evicts* cache blocks while running

Voila
Eviction based Cache Attacks: (Prime+Probe)

Step 0: Attacker *fills* the entire shared cache (set)

Step 1: Victim *evicts* cache blocks while running

Step 2: Attacker *probes* the cache set

Voila
Eviction based Cache Attacks: (Prime+Probe)

Step 0: Attacker fills the entire shared cache (set)

Step 1: Victim evicts cache blocks while running

Step 2: Attacker probes the cache set

If misses then victim has accessed the set
Various Mitigations
Various Mitigations

Cache Layout [HPCA ‘16]
Various Mitigations

Cache Layout [HPCA ‘16]

Fuzzing the timer [ISCA ‘12]
Various Mitigations

- Cache Layout [HPCA ‘16]
- Fuzzing the timer [ISCA ‘12]
- Cache Addressing [MICRO ‘18]
Various Mitigations

- Cache Layout [HPCA ‘16]
- Fuzzing the timer [ISCA ‘12]
- Cache Addressing [MICRO ‘18]
- Cache replacement policy [ISCA ‘17]
SHARP [Yan et al., ISCA ‘17 ]

Secure hierarchy-aware cache replacement policy

Mitigation for side-channel attacks
SHARP [Yan et al., ISCA ‘17 ]

Secure hierarchy-aware cache replacement policy

Mitigation for side-channel attacks

Prevents cross-core back invalidation
Cross-core Back-Invalidation - I

L1/L2

LLC
Cross-core Back-Invalidation - II

L1/L2

LLC

Cross-core back-invalidation

Miss
Cross-core Back-Invalidation - III

Attacker knows whether victim has accessed it or not

L1/L2

LLC
How SHARP Works?
How SHARP Works?

LLC Miss
How SHARP Works?

LLC Miss

STAGE - 1, LLC only block
How SHARP Works?

Private Caches

C0
A B

C1

C2
Y

C3

Shared Cache

A B Y N

Stage-1
How SHARP Works?

Private Caches

C0

A
B

C1

C2

Y

C3

Shared Cache

A
B
Y
N

Core-2 demands block X

Stage-1
How SHARP Works?

Stage-1

Private Caches

C0
A B

C1

C2
Y

C3

Shared Cache

A B Y N

Core-2 demands block X
How SHARP Works?

Private Caches:
- C0: A, B
- C1: Empty
- C2: Y
- C3: Empty

Shared Cache:
- A, B, Y, N

Core-2 demands block X

Stage-1
How SHARP Works?

Private Caches:
- C0: A, B
- C1: 
- C2: Y
- C3: 

Shared Cache:
- A, B, Y, N

Core-2 demands block X

Stage-1: Not LLC Only
How SHARP Works?

Private Caches

<table>
<thead>
<tr>
<th>C0</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Shared Cache

<table>
<thead>
<tr>
<th></th>
<th></th>
<th>Y</th>
<th>N</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>Y</td>
<td>N</td>
</tr>
</tbody>
</table>

Stage-1

Core-2 demands block X
How SHARP Works?

Private Caches

C0
A
B

C1


C2
Y

C3


Shared Cache

A
B
Y
N

Stage-1

LLC Only block found

Core-2 demands block X
How SHARP Works?

Private Caches

C0: A B
C1: 
C2: Y
C3: 

Shared Cache

A B Y N

Stage-1

Evicts it

Core-2 demands block X

31
How SHARP Works?

Private Caches

C0: A B

C1: (empty)

C2: X Y

C3: (empty)

Shared Cache

A B Y X

Stage-1
How SHARP Works?

Private Caches

C0
A B

C1

C2
X Y

C3

Shared Cache

A B Y X

Stage-1

Core-2 demands block Z
How SHARP Works?

Private Caches

C0: A, B
C1: 
C2: X, Y
C3: 

Shared Cache

A, B, Y, X

Stage-1

Core-2 demands block Z
How SHARP Works?

Private Caches

C0
A
B

C1

C2
X
Y

C3

Shared Cache

A
B
Y
X

Stage-1
No LLC only block found
(Stage-1 failed)

Core-2 demands block Z

Stage-1
35
How SHARP Works?

LLC Miss

STAGE - 1, LLC only block
How SHARP Works?

- LLC Miss
  - STAGE - 1, LLC only block
    - STAGE - 2, Same core block (intra-core eviction)
How SHARP Works?

Private Caches

C0: A  B
C1:  
C2: X  Y
C3:  

Shared Cache

A  B  Y  X

Stage-2

Core-2 demands block Z
How SHARP Works?

Private Caches

C0

A
B

C1

C2

X
Y

C3

Shared Cache

A
B
Y
X

Core-2 demands block Z

Stage-2

Not Intra core block

Z
How SHARP Works?

Private Caches

C0
A
B

C1

C2
X
Y

C3

Shared Cache

A
B
Y
X

Intra core block found

Stage-2

Core-2 demands block Z
How SHARP Works?

Private Caches

C0

A

B

Stage-2

Shared Cache

A

B

Y

X

Evicts it

Core-2 demands block Z

C1

C2

X

Y

C3

Z

Core-2

demands

block Z

Stage-2
How SHARP Works?

Private Caches

C0
- A
- B

C1
- Empty

C2
- Z
- Y

C3
- Empty

Shared Cache

- A
- B
- Y
- Z

Stage-2
How SHARP Works?

Private Caches

<table>
<thead>
<tr>
<th>C0</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>B</td>
<td>Z</td>
<td>Y</td>
</tr>
</tbody>
</table>

Shared Cache

|      | A | B | Y | Z |

Core-3 demands block P

Stage-2
How SHARP Works?

Private Caches

- C0: A, B
- C1: A

Shared Cache

- A, B, Y, Z

Core-3 demands block P

Stage-2
How SHARP Works?

Private Caches:
- C0: A, B
- C1: Empty
- C2: Z, Y
- C3: Empty

Shared Cache:
- A, B, Y, Z

Core-3 demands block P

Stage-2
How SHARP Works?

Private Caches
- C0: A B
- C1: blank
- C2: Z Y
- C3: blank

Shared Cache
- A B Y Z

Core-3 demands block P

Stage-2
- No LLC only block found (Stage-1 failed)
How SHARP Works?

Core-3 demands block P

Private Caches

C0

C1

C2

C3

Shared Cache

A B Y Z

No Intra core block found (Stage-2 failed)

Stage-2

No Intra core block found (Stage-2 failed)
How SHARP Works?

LLC Miss

STAGE - 1, LLC only block

STAGE - 2, Same core block (intra-core eviction)
How SHARP Works?

LLC Miss

STAGE - 1, LLC only block

STAGE - 2, Same core block (intra-core eviction)

STAGE - 3, Random block (inter-core eviction)
How SHARP Works?

Private Caches

Core-3 demands block P

Shared Cache

Stage-3 Evicts random block
How SHARP Works?

Private Caches
- C0
  - A
  - B

Core-3 demands block P

Shared Cache
- A
- B
- Y
- Z

Causes cross-core back invalidation

Stage-3

Core-3
How SHARP Works?

Private Caches

C0

A

C1

C2

Z Y

C3

P

Shared Cache

A P Y Z

52
How SHARP Works?

1. LLC Miss
2. STAGE - 1, LLC only block
3. STAGE - 2, Same core block (intra-core eviction)
4. STAGE - 3, Random block (inter-core eviction)
How SHARP Works?

1. LLC Miss

2. STAGE - 1, LLC only block

3. STAGE - 2, Same core block (intra-core eviction)

4. STAGE - 3, Random block (inter-core eviction)

5. Increments alarm-counter
SHARP Alarm Counter

Counter per core
SHARP Alarm Counter

Counter per core

Increments on inter-core eviction
SHARP Alarm Counter

- Counter *per core*
- Increments on inter-core *eviction*

For *1 billion cycles*, the threshold value is *2000*
SHARP Alarm Counter

Counter per core

Increments on inter-core eviction

For 1 billion cycles, the threshold value is 2000

On exceeding threshold, SHARP triggers OS interrupt
Questions That We Ask?
Questions That We Ask?

Does SHARP mitigate all attacks?
Prime+Reprime+Probe

Private Caches

Shared Cache

C0  C1  C2  C3
Prime+Reprime+Probe

Private Caches

Shared Cache

C0 C1 C2 C3

Critical Set
Prime+Reprime+Probe

Multi-threaded Attacker

Critical Set

Private Caches

Shared Cache
Prime+Reprime+Probe

Attacker Primes

Private Caches

C0

C1

C2

C3

Shared Cache

Critical Set

64
Prime+Reprime+Probe

Private Caches

Shared Cache

Attacker Primes

Critical Set
Prime+Reprime+Probe

Private Caches

Shared Cache

Attacker Primes

Critical Set
Prime+Reprime+Probe

Attacker Primes

Shared Cache

Critical Set

Private Caches
Prime+Reprime+Probe

Private Caches

Shared Cache

Critical Set
Prime+Reprime+Probe

Private Caches

Shared Cache

Attacker Reprimes

Critical Set
Prime+Reprime+Probe

Private Caches

Shared Cache

Attacker Reprimes

Critical Set
Prime+Reprime+Probe

Private Caches

Shared Cache

Attacker Reprimes

Critical Set
Prime+Reprime+Probe

Private Caches
- C0: E
- C1: F
- C2: G
- C3: H

Shared Cache
- A
- B
- C
- D

Attacker Reprimes

Critical Set
- E
- F
- G
- H

72
Prime+Reprime+Probe

Private Caches

Shared Cache

Critical Set
Prime+Reprime+Probe

Private Caches

Shared Cache

Victim Comes

Critical Set
Prime+Reprime+Probe

Private Caches

Shared Cache

Victim Accesses

Critical Set
Prime+Reprime+Probe

Private Caches

Shared Cache

Victim Accesses

Critical Set

Y F G H
X Y C D
E F G H
Prime+Reprime+Probe

Private Caches

Shared Cache

Attacker Comes

Critical Set
Prime+Reprime+Probe

- **Private Caches**
  - C0
  - C1
  - C2
  - C3

- **Shared Cache**
  - X
  - Y
  - C
  - D
  - E
  - F
  - G
  - H

**Attackers**
- Probe

**Critical Set**
Prime+Reprime+Probe

Private Caches
- C0
- C1
- C2
- C3

Shared Cache
- ?
- Y
- C
- D
- E
- F
- G
- H

Attacker Probes

Critical Set
Prime+Reprime+Probe

Private Caches
- C0
- C1: F
- C2: G
- C3: H

Shared Cache
- ?
- ?
- C
- D
- E
- F
- G
- H

Critical Set

Attacker Probes
Questions That We Ask?

Does SHARP *mitigate all attacks*?

No 😞
Questions That We Ask?

Does SHARP mitigate all attacks?  
No 😞

Does SHARP facilitate few more attacks?
Denial of Service Attack

Private Caches

Shared Cache

Critical Set
Denial of Service Attack

Multi-threaded Attacker

Private Caches

Shared Cache

Critical Set
Denial of Service Attack

Attacker accesses **block A**
Denial of Service Attack

Private Caches
- C0: A
- C1: B
- C2: 
- C3: 

Shared Cache
- A: A
- B: B

Critical Set

Attacker accesses block B
Denial of Service Attack

Private Caches

Shared Cache

Attacker accesses block C

Critical Set
Denial of Service Attack

Private Caches

Shared Cache

Attacker accesses block D
Denial of Service Attack

Private Caches

Shared Cache

Attacker occupies entire set
Denial of Service Attack

Private Caches

Shared Cache

Victim comes

Critical Set
Denial of Service Attack

Private Caches

Shared Cache

Victim accesses block X
Denial of Service Attack

Private Caches

Shared Cache

Victim accesses block Y

Critical Set
Denial of Service Attack

Private Caches

Z   B

C0  C1  C2  C3

Shared Cache

Z   B   C   D

Critical Set

Victim accesses block Z
Denial of Service Attack

Victim strives for other ways

Critical Set
Questions That We Ask?

Does SHARP mitigate all attacks?  
No 😞

Does SHARP facilitate few more attacks?  
Yes 😞
Questions That We Ask?

Does SHARP **mitigate** all attacks?  
No 😞

Does SHARP **facilitate** few more attacks?  
Yes 😞

Does threshold **affect** benign applications?
Simulation
Simulation

**ChampSim**, a trace driven simulator
Simulation

ChampSim, a trace driven simulator

Simulated SHARP on a 16-core system with three levels of caches and huge pages
Simulation

ChampSim, a trace driven simulator

Simulated SHARP on a 16-core system with three levels of caches and huge pages

Used different combinations of LLC thrashing and LLC fitting applications
Example, 16:0 denotes 16 thrashing and zero fitting
# LLC Thrashing Benchmarks [SPEC CPU 2017]

<table>
<thead>
<tr>
<th>Mix No</th>
<th>Thrashing Benchmarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>605.mcf-484B</td>
</tr>
<tr>
<td>2</td>
<td>605.mcf-665B</td>
</tr>
<tr>
<td>3</td>
<td>605.mcf-994B</td>
</tr>
<tr>
<td>4</td>
<td>607.cactubssn-2421B</td>
</tr>
<tr>
<td>5</td>
<td>620.omnetpp-141B</td>
</tr>
<tr>
<td>6</td>
<td>620.omnetpp-874B</td>
</tr>
<tr>
<td>7</td>
<td>621.wrf-6673B</td>
</tr>
<tr>
<td>8</td>
<td>623.xalancbmk-10B</td>
</tr>
<tr>
<td>9</td>
<td>649.fotonik-10881B</td>
</tr>
<tr>
<td>10</td>
<td>654.roms-523B</td>
</tr>
</tbody>
</table>
Interbackhit Rate

![Interbackhit Rate Chart]

- **Baseline**
- **Sharp**
Interbackhit Rate

![Interbackhit Rate Chart](chart.png)

- **Baseline**
- **Sharp**

Mix Number

Interbackhit rate (%)
Interbackhit Counter

Counter per billion cycles

Mix Number

1  2  3  4  5  6  7  8  9  10
16 22  5  79  83 27  5  98 10 10
45 26  6 155 154 67 10 222 33
26  5  16 264 222 16  5  10  10
Interbackhit Counter

![Bar chart showing interbackhit counter for different mix numbers. The chart compares average (green) and maximum (red) values.]
Interbackhit Counter

132x times of SHARP threshold
Questions That We Ask?

Does SHARP **mitigate all attacks?**  
No 😞

Does SHARP **facilitate few more attacks?**  
Yes 😞

Does threshold **affect benign applications?**  
Yes 😞
Questions That We Ask?

Does SHARP mitigate all attacks?  
No 😞

Does SHARP facilitate few more attacks?  
Yes 😞

Does threshold affect benign applications?  
Yes 😞

What does OS do when it receives an interrupt?
Speculating Possible OS Mitigations
Speculating Possible OS Mitigations

To deschedule
Speculating Possible OS Mitigations

To deschedule

To migrate to another socket
Migration to Another Socket

Inter-socket thread migration

Socket 0

Node 0

Socket 1
Speculating Possible OS Mitigations

- To deschedule
- To migrate to another socket
Speculating Possible OS Mitigations

- To deschedule
- To migrate to another socket

Causes performance overhead
Delay Cost: IPC Slowdown

<table>
<thead>
<tr>
<th>Case</th>
<th>Mitigation Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1 Million cycle</td>
</tr>
<tr>
<td>II</td>
<td>16 Million cycle</td>
</tr>
<tr>
<td>III</td>
<td>256 Million cycle</td>
</tr>
</tbody>
</table>
Delay Cost: IPC Slowdown

### Case I
Mitigation Delay: 1 Million cycle

### Case II
Mitigation Delay: 16 Million cycle

### Case III
Mitigation Delay: 256 Million cycle

1.1x Slowdown
Delay Cost: IPC Slowdown

Case | Mitigation Delay
--- | ---
I | 1 Million cycle
II | 16 Million cycle
III | 256 Million cycle

1.1x Slowdown
2.5x Slowdown
Delay Cost: IPC Slowdown

<table>
<thead>
<tr>
<th>Case</th>
<th>Mitigation Delay</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>1 Million cycle</td>
</tr>
<tr>
<td>II</td>
<td>16 Million cycle</td>
</tr>
<tr>
<td>III</td>
<td>256 Million cycle</td>
</tr>
</tbody>
</table>

Case I
- IPC Slowdown
- 1.1x Slowdown

Case II
- IPC Slowdown
- 2.5x Slowdown

Case III
- IPC Slowdown
- 30x Slowdown
Speculating Possible OS Mitigations

To deschedule

To migrate to an another socket
Speculating Possible OS Mitigations

To deschedule

causes slowdown 😞

To migrate to an another socket

causes significant slowdown 😞
Speculating Possible OS Mitigations

- To deschedule
  - causes slowdown

- To migrate to another socket
  - causes significant slowdown

- To kill
Speculating Possible OS Mitigations

To deschedule

causes slowdown 😞

To migrate to another socket

causes significant slowdown 😞

To kill

16-0 Mix, 100% apps got killed 😞
Speculating Possible OS Mitigations

- To deschedule
  - causes slowdown 😞

- To migrate to another socket
  - causes significant slowdown 😞

- To kill
  - 16-0 Mix, 100% apps got killed 😞

Does mitigation strategy facilitate any new attack?
Threshold Aware Attack - I

Attacker runs

Attacker crosses the threshold

OS deschedules/migrates/kills attacker
Threshold Aware Attack - II

- Attacker runs
- Attacker reaches nearly the threshold
- Attacker sleeps
- OS schedules process X
- Process X crosses the threshold
- OS deschedules/migrates/kills process X
Speculating Possible OS Mitigations

To deschedule
causes slowdown 😞

To migrate to another socket
causes significant slowdown 😞

To kill
16-0 Mix, 100% apps got killed 😞

Does mitigation strategy facilitates any new attack? Yes 😞
Questions That We Ask?

Does SHARP mitigate all attacks?  No 😞

Does SHARP facilitate few more attacks? Yes 😞

Does threshold affect benign applications? Yes 😞

What does OS do when it receives an interrupt?
Questions That We Ask?

Does SHARP mitigate all attacks?  No 😞

Does SHARP facilitate few more attacks?  Yes 😞

Does threshold affect benign applications?  Yes 😞

What does OS do when it receives an interrupt?  

Is SHARP secure in terms of information leakage?  ☹️
Questions That We Ask?

Does SHARP mitigate all attacks?  
No 😞

Does SHARP facilitate few more attacks?  
Yes 😞

Does threshold affect benign applications?  
Yes 😞

What does OS do when it receives an interrupt?  

Is SHARP secure in terms of information leakage?  
Not really 😞
Conclusion

SHARP is not that sharp

Facilitates new attacks

Don’t mitigates all attacks

Role of OS is not defined

Performance overhead to benign applications
Thank You!

Semiconductor Research Corporation

INDIAN INSTITUTE OF TECHNOLOGY KANPUR