Golmaal: Thanks to the Secure TimeCache for a Faster DRAM Covert Channel

Ajaykumar kushwaha, Ajay jain, Mahendra patel and Biswabandan Panda

Dept. of Computer Science and Engineering, Indian Institute of Technology Bombay
I can perform Flush+Reload attack.

You mean, you want to send first accesses of the reloaded cache lines to DRAM.

Won’t that increase bandwidth of Covert channel in DRAM?

I will Implement TimeCache to mitigate it.

Yes !!! Big Brains 😊
Background

- DRAM Covert channel
- TimeCache
DRAM Row-Buffer conflict covert channel*

- Sender and receiver agree on a bank (can be hardcoded)
- Both sender and receiver on host select a different row inside this bank
- Receiver measures access time for its selected row
- Sender can transmit 0 by doing nothing and 1 by causing row buffer conflict by accessing its own row.
- If measured timing was “fast” by the receiver then sender transmitted 0 otherwise 1.

Bank 1

Row 1

Row 2 (S)

Row 3

Row 4 (R)

Row 5

... 

Row N

Row Buffer

Sender and Receiver decide on One Bank
Row Buffer

Row 4 (R)

Row 3

Row 2 (S)

Row 1

Bank 1

Activates

Copy

Receiver measures access time to its address
Bank 1

- Row 1
- Row 2 (S)
- Row 3
- Row 4 (R)
- Row 5
- Row N
- Row Buffer

Return HIGH

INITIALIZATION
Repeated access always have low access time.
Sender Accesses its Address

Row Buffer

Row 2 (S)

Row 3

Row 4 (R)

Row 5

Row N
On next access of receiver, there is a row miss.
Receiver has high access time
Background

- DRAM Covert channel
- TimeCache
**TimeCache**

- For flush+reload attack, **first access** of attacker for **time measurement** is important.

- Resolution: **Delay first access**
  - To all cache lines → For each process, first access is **always Miss**.
  - Create a miss, even if it is a hit
  - **S-bit** denotes if it is the first access to the cache line by the process.

---

Example
Example

```
00010000
```

Victim

```
00010000
```

Spy

flush
Example

S-bit update

Victim

Spy

Wait
Example
Example
Example

S-bit update

00010000

Victim

Access

00000000

Spy

Wait

Memory
Example
Example
Example

Victim

Spy

Memory

Discard

Access

S-bit update

00010000

00010000
Golmaal Covert Channel

- Assumptions and Protocol
- Timing Diagram
- Transmission Channel
- Evaluation
Golmaal Covert Channel Assumptions

Theoretical Assumptions :
- Shared Memory Agreement.
- Senders and multiple receiver processes are allocated different rows of one DRAM bank of a given rank and channel.

Numerical Assumptions :
- Clflush Cycles = 300
- DRAM Access Cycles = 375(Row Buffer Hit) and 220(Row Buffer Miss)*
- Synchronization Cycle = 400 cycles
- Processor Speed = 4Ghz

** “DRAM latency in IceLake,” https://www.7-cpu.com/cpu/IceLake.html, 2021
Golmaal Covert Channel

- Assumptions and Protocol
- **Timing Diagram**
- Transmission Channel
- Evaluation
Timing diagram **Without** time-cache

300 Cycles

cflush A1
Timing diagram **Without** time-cache

- clflush A1
- Access A1
- Receiver 1

- 300 Cycles
- 375 Cycles
Timing diagram **Without** time-cache

300 Cycles → 375 Cycles → 300 Cycles

- clflush A1
- Access A1
- Receiver 1
- clflush A1
Timing diagram **Without** time-cache

- 300 Cycles
- 375 Cycles
- 300 Cycles
- 220 Cycles

- Access A1
- Receiver 1
- Access A1
- Receiver 2

- clflush A1
- clflush A1
Timing diagram *With* time-cache

![Diagram showing 300 cycles and clflush A1]
Timing diagram *With* time-cache
Timing diagram **With** time-cache

- **300 Cycles**
  - `clflush A1`
  - `Receiver 1`

- **375 Cycles**
  - `Access A1`

- **220 Cycles**
  - `Access A1`
  - `Receiver 2`
Timing diagram With time-cache

- 300 Cycles
- 375 Cycles
- 220 Cycles
- 220 Cycles

| clflush A1 | Access A1 | Receiver 1 | Access A1 | Receiver 2 | Access A1 | Receiver 3 |
Comparison of timing diagram Without and With time-cache

Without

300 Cycles
clflush A1

375 Cycles
Access A1
Receiver 1

300 Cycles
Access A1
clflush A1

With

220 Cycles
Access A1
Receiver 2

220 Cycles
Access A1
Receiver 3

220 Cycles
Access A1
Receiver 4
Golmaal Covert Channel

- Assumptions and Protocol
- Timing Diagram
- Transmission Channel
- Evaluation
TRANSMISSION

Data: [INIT]01001

<table>
<thead>
<tr>
<th>Sender</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Receiver</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td></td>
</tr>
<tr>
<td>R1</td>
<td>0</td>
</tr>
<tr>
<td>R2</td>
<td>0</td>
</tr>
<tr>
<td>R3</td>
<td>0</td>
</tr>
<tr>
<td>R4</td>
<td>0</td>
</tr>
</tbody>
</table>

Time Cache:
- w
- x
- y
- z

DRAM:
- A{w}
- B{y}
- GARBAGE

Access addr y
S bit 0 Cache miss
Row conflict
TRANSMISSION

Data: [INIT]01001

<table>
<thead>
<tr>
<th>Sender</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Receiver</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>1</td>
</tr>
<tr>
<td>R2</td>
<td>0</td>
</tr>
<tr>
<td>R3</td>
<td>0</td>
</tr>
<tr>
<td>R4</td>
<td>0</td>
</tr>
</tbody>
</table>

Sender sends w to Receiver.

Receiver checks S-bit for each entry in DRAM:
- A{w} - S-bit 0
- B{y} - S-bit 0
- B{y} - S-bit 0

Receiver changes S-bit for R1 to 1.

Get row B{y} from DRAM.
TRANSMISSION

Data: [INIT]01001

Sender

<table>
<thead>
<tr>
<th>Sender</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
</tr>
</tbody>
</table>

Receiver

<table>
<thead>
<tr>
<th>Receiver</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
</tr>
</tbody>
</table>

Time Cache

- w
- x
- y
- z

DRAM

- A{w}
- B{y}
- B{y}

- Access addr y
- S bit 0
- Cache miss

Row hit

- Low access time
  ≈ ‘0’ bit
TRANSMISSION

Data: [INIT]01001

Sender

<table>
<thead>
<tr>
<th>Sender</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
</tr>
</tbody>
</table>

Access addr w

S bit 0
Cache miss

Receiver

<table>
<thead>
<tr>
<th>Receiver</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>0</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
</tr>
</tbody>
</table>

R1
R2
R3
R4

Row Conflict

DRAM

A{w}

B{y}

Time Cache

w

x

y

z

S-bit

y

1

1

0

0
Data: [INIT]01001

Sender

<table>
<thead>
<tr>
<th>Sender</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td></td>
</tr>
<tr>
<td>S1</td>
<td>1</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
</tr>
</tbody>
</table>

Receiver

<table>
<thead>
<tr>
<th>Receiver</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>1</td>
</tr>
<tr>
<td>R2</td>
<td>1</td>
</tr>
<tr>
<td>R3</td>
<td>0</td>
</tr>
<tr>
<td>R4</td>
<td>0</td>
</tr>
</tbody>
</table>

Time Cache

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td></td>
</tr>
<tr>
<td>x</td>
<td></td>
</tr>
<tr>
<td>y</td>
<td></td>
</tr>
<tr>
<td>z</td>
<td></td>
</tr>
</tbody>
</table>

DRAM

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A{w}</td>
<td></td>
</tr>
<tr>
<td>B{y}</td>
<td></td>
</tr>
<tr>
<td>A{w}</td>
<td></td>
</tr>
</tbody>
</table>

Change sbit to 1

Get row
TRANSMISSION

Data: [INIT]01001

Sender

<table>
<thead>
<tr>
<th>Sender</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td>1</td>
</tr>
<tr>
<td>S1</td>
<td>1</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
</tr>
<tr>
<td>S3</td>
<td>0</td>
</tr>
<tr>
<td>S4</td>
<td>0</td>
</tr>
</tbody>
</table>

Receiver

<table>
<thead>
<tr>
<th>Receiver</th>
<th>S-bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>y</td>
<td>1</td>
</tr>
<tr>
<td>R1</td>
<td>1</td>
</tr>
<tr>
<td>R2</td>
<td>1</td>
</tr>
<tr>
<td>R3</td>
<td>0</td>
</tr>
<tr>
<td>R4</td>
<td>0</td>
</tr>
</tbody>
</table>

Time Cache

- w
- x
- y
- z

Access

addr y

S bit 0

Cache miss

DRAM

- A{w}
- B{y}
- A{w}

Row conflict

≈ High access time
‘1’ bit
Golmaal Covert Channel

- Assumptions and Protocol
- Timing Diagram
- Transmission Channel
- Evaluation
Simulation infrastructure

Simulators:
- Extensively modified ChampSim* microarchitectural simulator
- Ramulator*, a fast and cycle-accurate DRAM simulator

Simulator Parameters and Configurations:
- DDR4 DRAM controllers with a data rate of 3200 MT/sec.
- To synchronize between a pair of sender and receiver processes, we use the wall clocktime.
- To further improve synchronization we use the `nanosleep` system call for 100ns after every DRAM access by a sender and a receiver.

Number of Sender-Receiver

- Data:
  - 000000....
  - Random...
  - 101010.....
  - 111111......

“Bandwidth increases with increase in number of sender and receiver”
Error Percentage

- Eight sender and receiver pair communicating a payload of “11111111..1”.
- Error percentage is maximum at 4.28 Mbps with 12.8% of error.

“As the raw-bandwidth increase so does the probability of error”
Evaluation

- Eight sender and receiver pair communicating a payload of “11111111…1”.
- The maximum True capacity we achieved is 3.72 Mbps while transferring 1111....
Thank You
Any Questions ??