The Story of RowHammer

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu
19 January 2021
SEHAS Keynote @ HiPEAC
How Reliable/Secure/Safe is This Bridge?

Source: http://www.technologystudent.com/struct1/tacom1.png
Collapse of the “Galloping Gertie”

Source: AP
http://www.wsdot.wa.gov/tnbhistory/connections/connections3.htm
How Secure Are These People?

Security is about preventing unforeseen consequences
The Story of RowHammer

- One can predictably induce bit flips in commodity DRAM chips
  - >80% of the tested DRAM chips are vulnerable

- First example of how a simple hardware failure mechanism can create a widespread system security vulnerability
An “Early” Position Paper [IMW’13]

- Onur Mutlu, "Memory Scaling: A Systems Architecture Perspective"
  Proceedings of the 5th International Memory Workshop (IMW), Monterey, CA, May 2013. Slides (pptx) (pdf) EETimes Reprint

Memory Scaling: A Systems Architecture Perspective

Onur Mutlu
Carnegie Mellon University
onur@cmu.edu
http://users.ece.cmu.edu/~omutlu/

The DRAM Scaling Problem

- DRAM stores charge in a capacitor (charge-based memory)
  - Capacitor must be large enough for reliable sensing
  - Access transistor should be large enough for low leakage and high retention time
  - Scaling beyond 40-35nm (2013) is challenging [ITRS, 2009]

- DRAM capacity, cost, and energy/power hard to scale
As Memory Scales, It Becomes Unreliable

- Data from all of Facebook’s servers worldwide
- Meza+, “Revisiting Memory Errors in Large-Scale Production Data Centers,” DSN’15.

*Intuition: quadratic increase in capacity*
Large-Scale Failure Analysis of DRAM Chips

- Analysis and modeling of memory errors found in all of Facebook’s server fleet

- Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu, "Revisiting Memory Errors in Large-Scale Production Data Centers: Analysis and Modeling of New Trends from the Field" Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Rio de Janeiro, Brazil, June 2015. [Slides (pptx) (pdf)] [DRAM Error Model]

Revisiting Memory Errors in Large-Scale Production Data Centers: Analysis and Modeling of New Trends from the Field

Justin Meza  Qiang Wu*  Sanjeev Kumar*  Onur Mutlu
Carnegie Mellon University  *Facebook, Inc.
Infrastructures to Understand Such Issues

An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms (Liu et al., ISCA 2013)

The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study (Khan et al., SIGMETRICS 2014)

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors (Kim et al., ISCA 2014)

Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case (Lee et al., HPCA 2015)

AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems (Qureshi et al., DSN 2015)
Infrastructures to Understand Such Issues

SoftMC: Open Source DRAM Infrastructure


- Flexible
- Easy to Use (C++ API)
- Open-source
  
github.com/CMU-SAFARI/SoftMC
SoftMC: Open Source DRAM Infrastructure

- https://github.com/CMU-SAFARI/SoftMC

SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies

Hasan Hassan\textsuperscript{1,2,3} Nandita Vijaykumar\textsuperscript{3} Samira Khan\textsuperscript{4,3} Saugata Ghose\textsuperscript{3} Kevin Chang\textsuperscript{3} Gennady Pekhimenko\textsuperscript{5,3} Donghyuk Lee\textsuperscript{6,3} Oguz Ergin\textsuperscript{2} Onur Mutlu\textsuperscript{1,3}

\textsuperscript{1}ETH Zürich \textsuperscript{2}TOBB University of Economics & Technology \textsuperscript{3}Carnegie Mellon University
\textsuperscript{4}University of Virginia \textsuperscript{5}Microsoft Research \textsuperscript{6}NVIDIA Research
Retention Time Profile of DRAM looks like this:

- Location dependent
- Stored value pattern dependent
- Time dependent

64-128ms

>256ms

128-256ms
RAIDR: Heterogeneous Refresh [ISCA’12]

- Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu, "RAIDR: Retention-Aware Intelligent Intelligent DRAM Refresh"
An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms

Proceedings of the 40th International Symposium on Computer Architecture (ISCA), Tel-Aviv, Israel, June 2013. Slides (ppt) Slides (pdf)
Mitigation of Retention Issues [SIGMETRICS’14]

- Samira Khan, Donghyuk Lee, Yoongu Kim, Alaa Alameldeen, Chris Wilkerson, and Onur Mutlu,
"The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study"

The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study

Samira Khan†*  
samirakhan@cmu.edu

Donghyuk Lee†  
donghyuk1@cmu.edu

Yoongu Kim†  
yoongukim@cmu.edu

Alaa R. Alameldeen*  
alaa.r.alameldeen@intel.com

Chris Wilkerson*  
chris.wilkerson@intel.com

Onur Mutlu†  
onur@cmu.edu

†Carnegie Mellon University  *Intel Labs
Mitigation of Retention Issues [DSN’15]

- Moinuddin Qureshi, Dae Hyun Kim, Samira Khan, Prashant Nair, and Onur Mutlu,
"AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems"
Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Rio de Janeiro, Brazil, June 2015.
[Slides (pptx) (pdf)]

AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems

Moinuddin K. Qureshi† Dae-Hyun Kim† Samira Khan‡ Prashant J. Nair† Onur Mutlu‡
†Georgia Institute of Technology
‡Carnegie Mellon University
{moin, dhkim, pnair6}@ece.gatech.edu
{samirakhan, onur}@cmu.edu
Mitigation of Retention Issues [DSN’16]

- Samira Khan, Donghyuk Lee, and Onur Mutlu,
"PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM"
[Slides (pptx) (pdf)]

PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM

Samira Khan* Donghyuk Lee†‡ Onur Mutlu*†
*University of Virginia †Carnegie Mellon University ‡Nvidia *ETH Zürich
Mitigation of Retention Issues [MICRO’17]

- Samira Khan, Chris Wilkerson, Zhe Wang, Alaa R. Alameldeen, Donghyuk Lee, and Onur Mutlu,

"Detecting and Mitigating Data-Dependent DRAM Failures by Exploiting Current Memory Content"

Proceedings of the 50th International Symposium on Microarchitecture (MICRO), Boston, MA, USA, October 2017.

[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Poster (pptx) (pdf)]
Mitigation of Retention Issues [ISCA'17]

- Minesh Patel, Jeremie S. Kim, and Onur Mutlu,
  "The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions"

[Slides (pptx) (pdf)]
[Lightning Session Slides (pptx) (pdf)]

- First experimental analysis of (mobile) LPDDR4 chips
- Analyzes the complex tradeoff space of retention time profiling
- Idea: enable fast and robust profiling at higher refresh intervals & temperatures

The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions

Minesh Patel$‡‡ Jeremie S. Kim‡§ Onur Mutlu$‡‡
$ETH Zürich ‡Carnegie Mellon University
Mitigation of Retention Issues [DSN’19]


Understanding and Modeling On-Die Error Correction in Modern DRAM: An Experimental Study Using Real Devices

Minesh Patel† Jeremie S. Kim‡‡ Hasan Hassan† Onur Mutlu‡‡
†ETH Zürich  ‡‡Carnegie Mellon University
Mitigation of Retention Issues [MICRO’20]

  - [Slides (pptx) (pdf)]
  - [Lightning Talk Slides (pptx) (pdf)]
  - [Talk Video (15 minutes)]
  - [Lightning Talk Video (1.5 minutes)]

Best paper award.

Bit-Exact ECC Recovery (BEER):
Determining DRAM On-Die ECC Functions by Exploiting DRAM Data Retention Characteristics

Minesh Patel† Jeremie S. Kim†† Taha Shahroodi† Hasan Hassan† Onur Mutlu††
†ETH Zürich ‡Carnegie Mellon University
A Curious Phenomenon
A Curious Discovery [Kim et al., ISCA 2014]

One can predictably induce errors in most DRAM memory chips
DRAM RowHammer

A simple hardware failure mechanism can create a widespread system security vulnerability.
Modern DRAM is Prone to Disturbance Errors

Repeatedly reading a row enough times (before memory gets refreshed) induces disturbance errors in adjacent rows in most real DRAM chips you can buy today.

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors, (Kim et al., ISCA 2014)
Most DRAM Modules Are Vulnerable

A company: 86% (37/43) errors
B company: 83% (45/54) errors
C company: 88% (28/32) errors

Up to $1.0 \times 10^7$ errors
Up to $2.7 \times 10^6$ errors
Up to $3.3 \times 10^5$ errors

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors, (Kim et al., ISCA 2014)
Recent DRAM Is More Vulnerable

- A Modules
- B Modules
- C Modules

Errors per $10^9$ Cells

Module Vintage

- 2008
- 2009
- 2010
- 2011
- 2012
- 2013
- 2014

Log scale for Y-axis
Recent DRAM Is More Vulnerable

Errors per $10^9$ Cells

Module Vintage
Recent DRAM Is More Vulnerable

All modules from 2012–2013 are vulnerable
Why Is This Happening?

- DRAM cells are too close to each other!
  - They are not electrically isolated from each other

- Access to one cell affects the value in nearby cells
  - due to electrical interference between
    - the cells
    - wires used for accessing the cells
  - Also called cell-to-cell coupling/interference

- Example: When we activate (apply high voltage) to a row, an adjacent row gets slightly activated as well
  - Vulnerable cells in that slightly-activated row lose a little bit of charge
  - If row hammer happens enough times, charge in such cells gets drained
Higher-Level Implications

- This simple circuit level failure mechanism has enormous implications on upper layers of the transformation hierarchy.
A Simple Program Can Induce Many Errors

```asm
loop:
    mov (X), %eax
    mov (Y), %ebx
    clflush (X)
    clflush (Y)
    mfence
    jmp loop
```

Download from: https://github.com/CMU-SAFARI/rowhammer
A Simple Program Can Induce Many Errors

1. Avoid *cache hits*
   – Flush \( X \) from cache

2. Avoid *row hits* to \( X \)
   – Read \( Y \) in another row

Download from: [https://github.com/CMU-SAFARI/rowhammer](https://github.com/CMU-SAFARI/rowhammer)
A Simple Program Can Induce Many Errors

```plaintext
loop:
mov (X), %eax
mov (Y), %ebx
clflush (X)
clflush (Y)
mfence
jmp loop
```

Download from: https://github.com/CMU-SAFARI/rowhammer
A Simple Program Can Induce Many Errors

```
loop:
  mov (X), %eax
  mov (Y), %ebx
  clflush (X)
  clflush (Y)
  mfence
  jmp loop
```

Download from: [https://github.com/CMU-SAFARI/rowhammer](https://github.com/CMU-SAFARI/rowhammer)
A Simple Program Can Induce Many Errors

```plaintext
loop:
mov (X), %eax
mov (Y), %ebx
clflush (X)
clflush (Y)
mfence
jmp loop
```

Download from: https://github.com/CMU-SAFARI/rowhammer
## Observed Errors in Real Systems

<table>
<thead>
<tr>
<th>CPU Architecture</th>
<th>Errors</th>
<th>Access-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Haswell (2013)</td>
<td>22.9K</td>
<td>12.3M/sec</td>
</tr>
<tr>
<td>Intel Ivy Bridge (2012)</td>
<td>20.7K</td>
<td>11.7M/sec</td>
</tr>
<tr>
<td>Intel Sandy Bridge (2011)</td>
<td>16.1K</td>
<td>11.6M/sec</td>
</tr>
<tr>
<td>AMD Piledriver (2012)</td>
<td>59</td>
<td>6.1M/sec</td>
</tr>
</tbody>
</table>

A real reliability & security issue

One Can Take Over an Otherwise-Secure System

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors (Kim et al., ISCA 2014)

Exploiting the DRAM rowhammer bug to gain kernel privileges (Seaborn, 2015)
“Rowhammer” is a problem with some recent DRAM devices in which repeatedly accessing a row of memory can cause bit flips in adjacent rows (Kim et al., ISCA 2014).

- Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors (Kim et al., ISCA 2014)

We tested a selection of laptops and found that a subset of them exhibited the problem.

We built two working privilege escalation exploits that use this effect.

- Exploiting the DRAM rowhammer bug to gain kernel privileges (Seaborn+, 2015)

One exploit uses rowhammer-induced bit flips to gain kernel privileges on x86-64 Linux when run as an unprivileged userland process.

- When run on a machine vulnerable to the rowhammer problem, the process was able to induce bit flips in page table entries (PTEs).
- It was able to use this to gain write access to its own page table, and hence gain read-write access to all of physical memory.

Exploiting the DRAM rowhammer bug to gain kernel privileges (Seaborn & Dullien, 2015)
Security Implications

Rowhammer
It’s like breaking into an apartment by repeatedly slamming a neighbor’s door until the vibrations open the door you were after.
Selected Readings on RowHammer (I)

- **Our first detailed study: Rowhammer analysis and solutions** (June 2014)

- **Our Source Code to Induce Errors in Modern DRAM Chips** (June 2014)
  - [https://github.com/CMU-SAFARI/rowhammer](https://github.com/CMU-SAFARI/rowhammer)

- **Google Project Zero’s Attack to Take Over a System** (March 2015)
  - Exploiting the DRAM rowhammer bug to gain kernel privileges (Seaborn+, 2015)
  - [https://github.com/google/rowhammer-test](https://github.com/google/rowhammer-test)
  - **Double-sided Rowhammer**
Selected Readings on RowHammer (II)

- Remote RowHammer Attacks via JavaScript (July 2015)
  - https://github.com/IAIK/rowhammerjs
  - Gruss et al., DIMVA 2016.
  - **CLFLUSH-free Rowhammer**
    - “A fully automated attack that requires nothing but a website with JavaScript to trigger faults on remote hardware.”
    - “We can gain unrestricted access to systems of website visitors.”

- ANVIL: Software-Based Protection Against Next-Generation Rowhammer Attacks (March 2016)
  - http://dl.acm.org/citation.cfm?doid=2872362.2872390
  - Aweke et al., ASPLOS 2016
  - **CLFLUSH-free Rowhammer**
  - Software based monitoring for rowhammer detection
Selected Readings on RowHammer (III)

- Dedup Est Machina: Memory Deduplication as an Advanced Exploitation Vector (May 2016)
  - Exploits Rowhammer and Memory Deduplication to overtake a browser
  - “We report on the first reliable remote exploit for the Rowhammer vulnerability running entirely in Microsoft Edge.”
  - “[an attacker] ... can reliably “own” a system with all defenses up, even if the software is entirely free of bugs.”

- CAn’t Touch This: Software-only Mitigation against Rowhammer Attacks targeting Kernel Memory (August 2017)
  - Partitions physical memory into security domains, user vs. kernel; limits rowhammer-induced bit flips to the user domain.
Selected Readings on RowHammer (IV)

- A New Approach for Rowhammer Attacks (May 2016)
  - Qiao et al., HOST 2016
  - **CLFLUSH-free RowHammer**
  - “Libc functions memset and memcpy are found capable of rowhammer.”
  - Triggers RowHammer with malicious inputs but benign code

- One Bit Flips, One Cloud Flops: Cross-VM Row Hammer Attacks and Privilege Escalation (August 2016)
  - “Technique that allows a malicious guest VM to have read and write accesses to arbitrary physical pages on a shared machine.”
  - Graph-based algorithm to reverse engineer mapping of physical addresses in DRAM
Selected Readings on RowHammer (V)

- **Curious Case of RowHammer: Flipping Secret Exponent Bits using Timing Analysis** (August 2016)
  - Bhattacharya et al., CHES 2016
  - Combines timing analysis to perform rowhammer on cryptographic keys stored in memory

- **DRAMA: Exploiting DRAM Addressing for Cross-CPU Attacks** (August 2016)
  - Pessl et al., USENIX Security 2016
  - Shows RowHammer failures on DDR4 devices despite TRR solution
  - Reverse engineers address mapping functions to improve existing RowHammer attacks
Selected Readings on RowHammer (VI)

- **Flip Feng Shui: Hammering a Needle in the Software Stack** (August 2016)
  - Razavi et al., USENIX Security 2016.
  - Combines memory deduplication and RowHammer
  - “A malicious VM can gain unauthorized access to a co-hosted VM running OpenSSH.”
  - Breaks OpenSSH public key authentication

- **Drammer: Deterministic Rowhammer Attacks on Mobile Platforms** (October 2016)
  - [http://dl.acm.org/citation.cfm?id=2976749.2978406](http://dl.acm.org/citation.cfm?id=2976749.2978406)
  - Van Der Veen et al., ACM CCS 2016
  - Can take over an ARM-based Android system deterministically
  - Exploits predictable physical memory allocator behavior
    - Can deterministically place security-sensitive data (e.g., page table) in an attacker-chosen, vulnerable location in memory
Selected Readings on RowHammer (VII)

- **When Good Protections go Bad: Exploiting anti-DoS Measures to Accelerate Rowhammer Attacks** (May 2017)
  - Aga et al., HOST 2017
  - “A virtual-memory based cache-flush free attack that is sufficiently fast to **rowhammer with double rate refresh**.”
  - Enabled by Cache Allocation Technology

- **SGX-Bomb: Locking Down the Processor via Rowhammer Attack** (October 2017)
  - [https://dl.acm.org/citation.cfm?id=3152709](https://dl.acm.org/citation.cfm?id=3152709)
  - Jang et al., SysTEX 2017
  - “Launches the Rowhammer attack against enclave memory to trigger the processor lockdown.”
  - **Running unknown enclave programs on the cloud can shut down servers shared with other clients.**
Selected Readings on RowHammer (VIII)

- **Another Flip in the Wall of Rowhammer Defenses** (May 2018)
  - Gruss et al., IEEE S&P 2018
  - A new type of Rowhammer attack which only hammers one single address, which can be done without knowledge of physical addresses and DRAM mappings
  - Defeats static analysis and performance counter analysis defenses by running inside an SGX enclave

- **GuardION: Practical Mitigation of DMA-Based Rowhammer Attacks on ARM** (June 2018)
  - [https://link.springer.com/chapter/10.1007/978-3-319-93411-2_5](https://link.springer.com/chapter/10.1007/978-3-319-93411-2_5)
  - Van Der Veen et al., DIMVA 2018
  - Presents RAMPAGE, a DMA-based RowHammer attack against the latest Android OS
Grand Pwning Unit: Accelerating Microarchitectural Attacks with the GPU (May 2018)
- The first end-to-end remote Rowhammer exploit on mobile platforms that use our GPU-based primitives in orchestration to **compromise browsers on mobile devices in under two minutes**.

Throwhammer: Rowhammer Attacks over the Network and Defenses (July 2018)
- Tatar et al., USENIX ATC 2018.
- “[We] show that an attacker can trigger and exploit Rowhammer bit flips directly from a remote machine by only sending network packets.”
Selected Readings on RowHammer (X)

- **Nethammer: Inducing Rowhammer Faults through Network Requests** (July 2018)
  - Lipp et al., arxiv.org 2018.
  - “Nethammer is the first truly remote Rowhammer attack, without a single attacker-controlled line of code on the targeted system.”

- **ZebRAM: Comprehensive and Compatible Software Protection Against Rowhammer Attacks** (October 2018)
  - Konoth et al., OSDI 2018
  - A new pure-software protection mechanism against RowHammer.
Selected Readings on RowHammer (XI.A)

- PassMark Software, memtest86, since 2014
  - [https://www.memtest86.com/troubleshooting.htm#hammer](https://www.memtest86.com/troubleshooting.htm#hammer)

Why am I only getting errors during Test 13 Hammer Test?

The Hammer Test is designed to detect RAM modules that are susceptible to disturbance errors caused by charge leakage. This phenomenon is characterized in the research paper *Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors* by Yoongu Kim et al. According to the research, a significant number of RAM modules manufactured 2010 or newer are affected by this defect. In simple terms, susceptible RAM modules can be subjected to disturbance errors when repeatedly accessing addresses in the same memory bank but different rows in a short period of time. Errors occur when the repeated access causes charge loss in a memory cell, before the cell contents can be refreshed at the next DRAM refresh interval.

Starting from MemTest86 v6.2, the user may see a warning indicating that the RAM may be vulnerable to high frequency row hammer bit flips. This warning appears when errors are detected during the first pass (maximum hammer rate) but no errors are detected during the second pass (lower hammer rate). See MemTest86 Test Algorithms for a description of the two passes that are performed during the Hammer Test (Test 13). When performing the second pass, address pairs are hammered only at the rate deemed as the maximum allowable by memory vendors (200K accesses per 64ms). Once this rate is exceeded, the integrity of memory contents may no longer be guaranteed. If errors are detected in both passes, errors are reported as normal.

The errors detected during Test 13, albeit exposed only in extreme memory access cases, are most certainly real errors. During typical home PC usage (e.g. web browsing, word processing, etc.), it is less likely that the memory usage pattern will fall into the extreme case that make it vulnerable to disturbance errors. It may be of greater concern if you were running highly sensitive equipment such as medical equipment, aircraft control systems, or bank database servers. It is impossible to predict with any accuracy if these errors will occur in real life applications. One would need to do a major scientific study of 1000 of computers and their usage patterns, then do a forensic analysis of each application to study how it makes use of the RAM while it executes. To date, we have only seen 1-bit errors as a result of running the Hammer Test.
Selected Readings on RowHammer (XI.B)

- PassMark Software, memtest86, since 2014
  - https://www.memtest86.com/troubleshooting.htm#hammer

Detection and mitigation of row hammer errors

The ability of MemTest86 to detect and report on row hammer errors depends on several factors and what mitigations are in place. To generate errors adjacent memory rows must be repeatedly accessed. But hardware features such as multiple channels, interleaving, scrambling, Channel Hashing, NUMA & XOR schemes make it nearly impossible (for an arbitrary CPU & RAM stick) to know which memory addresses correspond to which rows in the RAM. Various mitigations might also be in place. Different BIOS firmware might set the refresh interval to different values (tREFI). The shorter the interval the more resistant the RAM will be to errors. But shorter intervals result in higher power consumption and increased processing overhead. Some CPUs also support pseudo target row refresh (pTRR) that can be used in combination with pTRR-compliant RAM. This field allows the RAM stick to indicate the MAC (Maximum Active Count) level which is the RAM can support. A typical value might be 200,000 row activations. Some CPUs also support the Joint Electron Design Engineering Council (JEDEC) Targeted Row Refresh (TRR) algorithm. The TRR is an improved version of the previously implemented pTRR algorithm and does not inflict any performance drop or additional power usage. As a result the row hammer test implemented in MemTest86 maybe not be the worst case possible and vulnerabilities in the underlying RAM might be undetectable due to the mitigations in place in the BIOS and CPU.
Security Implications (ISCA 2014)

• **Breach of memory protection**
  – OS page (4KB) fits inside DRAM row (8KB)
  – Adjacent DRAM row \(\rightarrow\) Different OS page

• **Vulnerability:** *disturbance attack*
  – By accessing its own page, a program could corrupt pages belonging to another program

• **We constructed a proof-of-concept**
  – Using only user-level instructions
More Security Implications (I)

“We can gain unrestricted access to systems of website visitors.”

Not there yet, but ...

ROOT privileges for web apps!

Source: https://lab.dsst.io/32c3-slides/7197.html

Rowhammer.js: A Remote Software-Induced Fault Attack in JavaScript (DIMVA’16)
More Security Implications (II)

“Can gain control of a smart phone deterministically”
More Security Implications (III)

- Using an integrated GPU in a mobile system to remotely escalate privilege via the WebGL interface. IEEE S&P 2018

"GRAND PWNING UNIT" —

Drive-by Rowhammer attack uses GPU to compromise an Android phone

JavaScript based GLitch pwns browsers by flipping bits inside memory chips.

DAN GOODIN - 5/3/2018, 12:00 PM

Grand Pwning Unit: Accelerating Microarchitectural Attacks with the GPU

Pietro Frigo
Vrije Universiteit Amsterdam
p.frigo@vu.nl

Cristiano Giuffrida
Vrije Universiteit Amsterdam
giuffrida@cs.vu.nl

Herbert Bos
Vrije Universiteit Amsterdam
herbertb@cs.vu.nl

Kaveh Razavi
Vrije Universiteit Amsterdam
kaveh@cs.vu.nl
Packets over a LAN are all it takes to trigger serious Rowhammer bit flips

The bar for exploiting potentially serious DDR weakness keeps getting lower.

THROWHAMMER —

Throwhammer: Rowhammer Attacks over the Network and Defenses

Andrei Tatar
VU Amsterdam

Radhesh Krishnan
VU Amsterdam

Elias Athanasopoulos
University of Cyprus

Cristiano Giuffrida
VU Amsterdam

Herbert Bos
VU Amsterdam

Kaveh Razavi
VU Amsterdam
More Security Implications (V)

- Rowhammer over RDMA (II)

Nethammer—Exploiting DRAM Rowhammer Bug Through Network Requests

Nethammer:
Inducing Rowhammer Faults through Network Requests

Moritz Lipp
Graz University of Technology

Misiker Tadesse Aga
University of Michigan

Daniel Gruss
Graz University of Technology

Clémentine Maurice
Univ Rennes, CNRS, IRISA

Michael Schwarz
Graz University of Technology

Lukas Raab
Graz University of Technology

Lukas Lamster
Graz University of Technology
More Security Implications (VI)

IEEE S&P 2020

RAMBleed

RAMBleed: Reading Bits in Memory Without Accessing Them

Andrew Kwong  
University of Michigan  
ankwong@umich.edu

Daniel Genkin  
University of Michigan  
genkin@umich.edu

Daniel Gruss  
Graz University of Technology  
daniel.gruss@iaik.tugraz.at

Yuval Yarom  
University of Adelaide and Data61  
yval@cs.adelaide.edu.au
More Security Implications (VII)

- USENIX Security 2019

Terminal Brain Damage: Exposing the Graceless Degradation in Deep Neural Networks Under Hardware Fault Attacks

Sanghyun Hong, Pietro Frigo†, Yiğitcan Kaya, Cristiano Giuffrida†, Tudor Dumitraș

University of Maryland, College Park
†Vrije Universiteit Amsterdam

A Single Bit-flip Can Cause Terminal Brain Damage to DNNs
One specific bit-flip in a DNN’s representation leads to accuracy drop over 90%

Our research found that a specific bit-flip in a DNN’s bitwise representation can cause the accuracy loss up to 90%, and the DNN has 40-50% parameters, on average, that can lead to the accuracy drop over 10% when individually subjected to such single bitwise corruptions...

Read More
More Security Implications (VIII)

USENIX Security 2020

DeepHammer: Depleting the Intelligence of Deep Neural Networks through Targeted Chain of Bit Flips

Fan Yao  
University of Central Florida  
fan.yao@ucf.edu

Adnan Siraj Rakin  
Arizona State University  
asrakin@asu.edu

Deliang Fan  

dfan@asu.edu

Degrade the **inference accuracy** to the level of **Random Guess**

Example: ResNet-20 for CIFAR-10, 10 output classes

Before attack, **Accuracy: 90.2%**  
After attack, **Accuracy: ~10% (1/10)**
More Security Implications (IX)

- Rowhammer on MLC NAND Flash (based on [Cai+, HPCA 2017])

Security

Rowhammer RAM attack adapted to hit flash storage

Project Zero's two-year-old dog learns a new trick

By Richard Chirgwin 17 Aug 2017 at 04:27

From random block corruption to privilege escalation:
A filesystem attack vector for rowhammer-like attacks

Anil Kurmus       Nikolas Ioannou       Matthias Neugschwandtner     Nikolaos Papandreou
Thomas Parnell
IBM Research – Zurich
More Security Implications?
Understanding RowHammer
Root Causes of Disturbance Errors

• **Cause 1: Electromagnetic coupling**
  – Toggling the wordline voltage briefly increases the voltage of adjacent wordlines
  – Slightly opens adjacent rows \( \rightarrow \) Charge leakage

• **Cause 2: Conductive bridges**

• **Cause 3: Hot-carrier injection**

*Confirmed by at least one manufacturer*
An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms (Liu et al., ISCA 2013)

The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study (Khan et al., SIGMETRICS 2014)

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors (Kim et al., ISCA 2014)

Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case (Lee et al., HPCA 2015)

AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems (Qureshi et al., DSN 2015)
Where RowHammer Was Discovered

## Tested DRAM Modules

(129 total)

<table>
<thead>
<tr>
<th>Manufacturer</th>
<th>Module</th>
<th>Date* (yy-mm)</th>
<th>Timing†</th>
<th>Organization</th>
<th>Chip</th>
<th>Victims-per-Module</th>
<th>$R_{th}$ (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>A</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>A1</td>
<td>10-08</td>
<td>1066</td>
<td>50.625</td>
<td>0.5</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>A2</td>
<td>10-20</td>
<td>1066</td>
<td>50.625</td>
<td>0.5</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>A3.5</td>
<td>10-20</td>
<td>1066</td>
<td>50.625</td>
<td>0.5</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>A4.7</td>
<td>11-24</td>
<td>1066</td>
<td>49.125</td>
<td>1</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>A5.12</td>
<td>11-12</td>
<td>1066</td>
<td>49.125</td>
<td>1</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>A6.14</td>
<td>11-50</td>
<td>1066</td>
<td>49.125</td>
<td>1</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>A12.66</td>
<td>12-22</td>
<td>1600</td>
<td>49.125</td>
<td>2</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>A16.16</td>
<td>12-26</td>
<td>1600</td>
<td>49.125</td>
<td>2</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>Total of 43 Modules</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>B</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>B1</td>
<td>08-49</td>
<td>1066</td>
<td>50.625</td>
<td>1</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>B2</td>
<td>09-49</td>
<td>1066</td>
<td>50.625</td>
<td>1</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>B3</td>
<td>10-19</td>
<td>1066</td>
<td>50.625</td>
<td>1</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>B4</td>
<td>11-31</td>
<td>1333</td>
<td>49.125</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>B5</td>
<td>11-16</td>
<td>1066</td>
<td>50.625</td>
<td>1</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>B6</td>
<td>11-19</td>
<td>1066</td>
<td>50.625</td>
<td>1</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>B7</td>
<td>12-01</td>
<td>1866</td>
<td>47.125</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Total of 54 Modules</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>C1</td>
<td>10-18</td>
<td>1333</td>
<td>49.125</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>C2</td>
<td>10-20</td>
<td>1066</td>
<td>50.625</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>C3.22</td>
<td>10-22</td>
<td>1066</td>
<td>50.625</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>C4.25</td>
<td>11-26</td>
<td>1333</td>
<td>49.125</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>C5</td>
<td>10-43</td>
<td>1333</td>
<td>49.125</td>
<td>1</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>C6</td>
<td>11-51</td>
<td>1333</td>
<td>49.125</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>C7</td>
<td>11-12</td>
<td>1333</td>
<td>49.125</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>C8</td>
<td>11-19</td>
<td>1333</td>
<td>49.125</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>C9</td>
<td>11-31</td>
<td>1333</td>
<td>49.125</td>
<td>2</td>
<td>8</td>
<td>8</td>
</tr>
<tr>
<td>Total of 32 Modules</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* We report the manufacture date marked on the chip packages, which is more accurate than other dates that can be gleaned from a module.
† We report timing constraints stored in the module’s on-board ROM [33], which is read by the system BIOS to calibrate the memory controller.
‡ The maximum DRAM chip size supported by our testing platform is 2Gb.
¶ We report DRAM die versions marked on the chip packages, which typically progress in the following manner: $A \rightarrow A \rightarrow B \rightarrow C \rightarrow \cdots$.

Table 3. Sample population of 129 DDR3 DRAM modules, categorized by manufacturer and sorted by manufacture date.
RowHammer Characterization Results

1. Most Modules Are at Risk
2. Errors vs. Vintage
3. Error = Charge Loss
4. Adjacency: Aggressor & Victim
5. Sensitivity Studies
6. Other Results in Paper
7. Solution Space

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors, (Kim et al., ISCA 2014)
4. Adjacency: Aggressor & Victim

Note: For three modules with the most errors (only first bank)

Most aggressors & victims are adjacent
1 Access Interval (Aggressor)

Note: For three modules with the most errors (only first bank)

Less frequent accesses → Fewer errors
2 Refresh Interval

Note: Using three modules with the most errors (only first bank)

More frequent refreshes ➔ Fewer errors
Data Pattern

Solid

111111
111111
111111
111111

~Solid

000000
000000
000000
000000

RowStripe

111111
000000
111111
000000

10x Errors

111111
000000
111111
000000
111111

Errors affected by data stored in other cells
6. Other Results (in Paper)

- Victim Cells $\neq$ Weak Cells (i.e., leaky cells)
  - Almost no overlap between them

- Errors not strongly affected by temperature
  - Default temperature: 50°C
  - At 30°C and 70°C, number of errors changes <15%

- Errors are repeatable
  - Across ten iterations of testing, >70% of victim cells had errors in every iteration
6. Other Results (in Paper) cont’d

• As many as 4 errors per cache-line
  – Simple ECC (e.g., SECDED) cannot prevent all errors

• Number of cells & rows affected by aggressor
  – Victims cells per aggressor: \( \leq 110 \)
  – Victims rows per aggressor: \( \leq 9 \)

• Cells affected by two aggressors on either side
  – Very small fraction of victim cells \((<100)\) have an error when either one of the aggressors is toggled
First RowHammer Analysis

- Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu,

"Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors"
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Source Code and Data]
Two Types of RowHammer Solutions

- **Immediate**
  - To protect the vulnerable DRAM chips in the field
  - Limited possibilities

- **Longer-term**
  - To protect future DRAM chips
  - Wider range of protection mechanisms

- Our ISCA 2014 paper proposes both types of solutions
  - Seven solutions in total
  - PARA proposed as best solution → already employed in the field
### Some Potential Solutions (ISCA 2014)

<table>
<thead>
<tr>
<th>Solution</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Make better DRAM chips</td>
<td></td>
</tr>
<tr>
<td>Refresh frequently</td>
<td>Power, Performance</td>
</tr>
<tr>
<td>Sophisticated ECC</td>
<td>Cost, Power</td>
</tr>
<tr>
<td>Access counters</td>
<td>Cost, Power, Complexity</td>
</tr>
</tbody>
</table>
Apple’s Patch for RowHammer


Available for: OS X Mountain Lion v10.8.5, OS X Mavericks v10.9.5

Impact: A malicious application may induce memory corruption to escalate privileges

Description: A disturbance error, also known as Rowhammer, exists with some DDR3 RAM that could have led to memory corruption. **This issue was mitigated by increasing memory refresh rates.**

CVE-ID

CVE-2015-3693: Mark Seaborn and Thomas Dullien of Google, working from original research by Yoongu Kim et al (2014)

HP, Lenovo, and many other vendors released similar patches
Our Best Solution to RowHammer

• **PARA:** *Probabilistic Adjacent Row Activation*

• **Key Idea**
  - After closing a row, we activate (i.e., refresh) one of its neighbors with a low probability: $p = 0.005$

• **Reliability Guarantee**
  - When $p=0.005$, errors in one year: $9.4 \times 10^{-14}$
  - By adjusting the value of $p$, we can vary the strength of protection against errors
Advantages of PARA

• **PARA refreshes rows infrequently**
  – Low power
  – Low performance-overhead
    • Average slowdown: 0.20% (for 29 benchmarks)
    • Maximum slowdown: 0.75%

• **PARA is stateless**
  – Low cost
  – Low complexity

• **PARA is an effective and low-overhead solution to prevent disturbance errors**
Requirements for PARA

• If implemented in **DRAM chip** (done today)
  – Enough slack in timing and refresh parameters
  – Plenty of slack today:
    • Chang et al., “Understanding Latency Variation in Modern DRAM Chips,” SIGMETRICS 2016.
    • Lee et al., “Design-Induced Latency Variation in Modern DRAM Chips,” SIGMETRICS 2017.
    • Chang et al., “Understanding Reduced-Voltage Operation in Modern DRAM Devices,” SIGMETRICS 2017.

• If implemented in **memory controller**
  – Better coordination between memory controller and DRAM
  – Memory controller should know which rows are physically adjacent
Probabilistic Activation in Real Life (I)

https://twitter.com/isislovecruft/status/1021939922754723841
Seven RowHammer Solutions Proposed

- Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu,

"Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors"
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Source Code and Data]

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

Yoongu Kim¹ Ross Daly* Jeremie Kim¹ Chris Fallin* Ji Hye Lee¹ Donghyuk Lee¹ Chris Wilkerson² Konrad Lai Onur Mutlu¹
¹Carnegie Mellon University ²Intel Labs
A Takeaway

Main Memory Needs
Intelligent Controllers for Security
Aside: Intelligent Controller for NAND Flash

Error Characterization, Mitigation, and Recovery in Flash-Memory-Based Solid-State Drives

This paper reviews the most recent advances in solid-state drive (SSD) error characterization, mitigation, and data recovery techniques to improve both SSD’s reliability and lifetime.

By Yu Cai, Saugata Ghose, Erich F. Haratsch, Yixin Luo, and Onur Mutlu

https://arxiv.org/pdf/1706.08642
Detailed Lectures on RowHammer

- Computer Architecture, Fall 2020, Lecture 4b
  - RowHammer (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=K Dy632z23UE&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=8](https://www.youtube.com/watch?v=K Dy632z23UE&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=8)

- Computer Architecture, Fall 2020, Lecture 5c
  - Secure and Reliable Memory (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=HvswnsfG3oQ&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=11](https://www.youtube.com/watch?v=HvswnsfG3oQ&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=11)

[https://www.youtube.com/onurmutlullectures](https://www.youtube.com/onurmutlullectures)
First RowHammer Analysis

- Yoongu Kim, Ross Daly, Jeremie Kim, Chris Fallin, Ji Hye Lee, Donghyuk Lee, Chris Wilkerson, Konrad Lai, and Onur Mutlu, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors"
  [Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Source Code and Data]
Onur Mutlu,
"The RowHammer Problem and Other Issues We May Face as Memory Becomes Denser"
[Slides (pptx) (pdf)]

The RowHammer Problem
and Other Issues We May Face as Memory Becomes Denser

Onur Mutlu
ETH Zürich
onur.mutlu@inf.ethz.ch
https://people.inf.ethz.ch/omutlu
A More Recent RowHammer Retrospective

- [Preliminary arXiv version](#)
- [Slides from COSADE 2019 (pptx)](#)
- [Slides from VLSI-SOC 2020 (pptx) (pdf)](#)
- [Talk Video (30 minutes)](#)

**RowHammer: A Retrospective**

Onur Mutlu§‡
§ETH Zürich

Jeremie S. Kim‡§
‡Carnegie Mellon University
RowHammer in 2020
RowHammer in 2020 (I)

- Jeremie S. Kim, Minesh Patel, A. Giray Yaglikci, Hasan Hassan, Roknoddin Azizi, Lois Orosa, and Onur Mutlu,

"Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques"

[Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (20 minutes)]
[Lightning Talk Video (3 minutes)]
TRRespass: Exploiting the Many Sides of Target Row Refresh

Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi,

"TRRespass: Exploiting the Many Sides of Target Row Refresh"


[Slides (pptx) (pdf)]
[Lecture Slides (pptx) (pdf)]
[Talk Video (17 minutes)]
[Lecture Video (59 minutes)]
[Source Code]
[Web Article]

Best paper award.
Pwnie Award 2020 for Most Innovative Research. Pwnie Awards 2020
RowHammer in 2020 (III)

- Lucian Cojocar, Jeremie Kim, Minesh Patel, Lillian Tsai, Stefan Saroiu, Alec Wolman, and Onur Mutlu,

"Are We Susceptible to Rowhammer? An End-to-End Methodology for Cloud Providers"

[Slides (pptx) (pdf)]
[Talk Video (17 minutes)]

Are We Susceptible to Rowhammer?
An End-to-End Methodology for Cloud Providers

Lucian Cojocar, Jeremie Kim§†, Minesh Patel§, Lillian Tsai‡, Stefan Saroiu, Alec Wolman, and Onur Mutlu§†
Microsoft Research, §ETH Zürich, †CMU, ‡MIT
TRRespass
RowHammer in 2020 (II)

- Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi,

"TRRespass: Exploiting the Many Sides of Target Row Refresh"


[Slides (pptx) (pdf)]
[Lecture Slides (pptx) (pdf)]
[Talk Video (17 minutes)]
[Lecture Video (59 minutes)]
[Source Code]
[Web Article]

*Best paper award.*

*Pwnie Award 2020 for Most Innovative Research. Pwnie Awards 2020*

TRRespass: Exploiting the Many Sides of Target Row Refresh

Pietro Frigo*† Emanuele Vannacci*† Hasan Hassan§ Victor van der Veen¶
Onur Mutlu§ Cristiano Giuffrida* Herbert Bos* Kaveh Razavi*

*Vrije Universiteit Amsterdam
§ETH Zürich
¶Qualcomm Technologies Inc.*
TRRespass

- First work to show that TRR-protected DRAM chips are vulnerable to RowHammer in the field
  - Mitigations advertised as secure are not secure

- Introduces the Many-sided RowHammer attack
  - Idea: Hammer many rows to bypass TRR mitigations (e.g., by overflowing proprietary TRR tables that detect aggressor rows)

- (Partially) reverse-engineers the TRR and pTRR mitigation mechanisms implemented in DRAM chips and memory controllers

- Provides an automatic tool that can effectively create many-sided RowHammer attacks in DDR4 and LPDDR4(X) chips
Example Many-Sided Hammering Patterns

(a) Assisted double-sided  (b) 4-sided

Fig. 12: Hammering patterns discovered by TRRespass. Aggressor rows are in red (■) and victim rows are in blue (■).
Fig. 10: Bit flips vs. number of aggressor rows. Module $C_{12}$: Number of bit flips in bank 0 as we vary the number of aggressor rows. Using SoftMC, we refresh DRAM with standard tREFI and run the tests until each aggressor rows is hammered 500K times.

Fig. 11: Bit flips vs. number of aggressor rows. Module $A_{15}$: Number of bit flips in bank 0 as we vary the number of aggressor rows. Using SoftMC, we refresh DRAM with standard tREFI and run the tests until each aggressor rows is hammered 500K times.

Fig. 13: Bit flips vs. number of aggressor rows. Module $A_{10}$: Number of bit flips triggered with $N$-sided RowHammer for varying number of $N$ on Intel Core i7-7700K. Each aggressor row is one row away from the closest aggressor row (i.e., VAVAVA... configuration) and aggressor rows are hammered in a round-robin fashion.
# TRRespass Vulnerable DRAM Modules

## TABLE II: TRRespass results. We report the number of patterns found and bit flips detected for the 42 DRAM modules in our set.

<table>
<thead>
<tr>
<th>Module</th>
<th>Date (yy-ww)</th>
<th>Freq. (MHz)</th>
<th>Size (GB)</th>
<th>Organization</th>
<th>MAC</th>
<th>Found Patterns</th>
<th>Best Pattern</th>
<th>Corruptions</th>
<th>Double Refresh</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Ranks</td>
<td>Banks</td>
<td>Pins</td>
<td></td>
<td>Total</td>
<td>1 → 0</td>
</tr>
<tr>
<td>A_{0,1,2,3}</td>
<td>16-37</td>
<td>2132</td>
<td>4</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>A_{4}</td>
<td>16-51</td>
<td>2132</td>
<td>4</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>4</td>
<td>9-sided</td>
</tr>
<tr>
<td>A_{5}</td>
<td>18-51</td>
<td>2400</td>
<td>4</td>
<td>1</td>
<td>8</td>
<td>×16</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>A_{6,7}</td>
<td>18-15</td>
<td>2666</td>
<td>4</td>
<td>1</td>
<td>8</td>
<td>×16</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>A_{8}</td>
<td>17-09</td>
<td>2400</td>
<td>8</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>33</td>
<td>19-sided</td>
</tr>
<tr>
<td>A_{9}</td>
<td>17-31</td>
<td>2400</td>
<td>8</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>33</td>
<td>19-sided</td>
</tr>
<tr>
<td>A_{10}</td>
<td>19-02</td>
<td>2400</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>488</td>
<td>10-sided</td>
</tr>
<tr>
<td>A_{11}</td>
<td>19-02</td>
<td>2400</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>523</td>
<td>10-sided</td>
</tr>
<tr>
<td>A_{12,13}</td>
<td>18-50</td>
<td>2666</td>
<td>8</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>A_{14}</td>
<td>19-08†</td>
<td>3200</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>120</td>
<td>14-sided</td>
</tr>
<tr>
<td>A_{15}†</td>
<td>17-08</td>
<td>2132</td>
<td>4</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>2</td>
<td>9-sided</td>
</tr>
<tr>
<td>B_{0}</td>
<td>18-11</td>
<td>2666</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>2</td>
<td>3-sided</td>
</tr>
<tr>
<td>B_{1}</td>
<td>18-11</td>
<td>2666</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>2</td>
<td>3-sided</td>
</tr>
<tr>
<td>B_{2}</td>
<td>18-49</td>
<td>3000</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>2</td>
<td>3-sided</td>
</tr>
<tr>
<td>B_{3}</td>
<td>19-08†</td>
<td>3000</td>
<td>8</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>B_{4,5}</td>
<td>19-08†</td>
<td>2666</td>
<td>8</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>B_{6,7}</td>
<td>19-08†</td>
<td>2400</td>
<td>4</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>B_{8,9}</td>
<td>19-08†</td>
<td>2400</td>
<td>8</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>B_{9,10,11}</td>
<td>16-13†</td>
<td>2132</td>
<td>8</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>C_{0,1}</td>
<td>18-46</td>
<td>2666</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>C_{2,3}</td>
<td>19-08†</td>
<td>2800</td>
<td>4</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>C_{4,5}</td>
<td>19-08†</td>
<td>3000</td>
<td>8</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>C_{6,7}</td>
<td>19-08†</td>
<td>3000</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>C_{8}</td>
<td>19-08†</td>
<td>3200</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>C_{9}</td>
<td>18-47</td>
<td>2666</td>
<td>16</td>
<td>2</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>C_{10,11}</td>
<td>19-04</td>
<td>2933</td>
<td>8</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UL</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>C_{12}†</td>
<td>15-01†</td>
<td>2132</td>
<td>4</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UT</td>
<td>25</td>
<td>10-sided</td>
</tr>
<tr>
<td>C_{13}†</td>
<td>18-49</td>
<td>2132</td>
<td>4</td>
<td>1</td>
<td>16</td>
<td>×8</td>
<td>UT</td>
<td>3</td>
<td>9-sided</td>
</tr>
</tbody>
</table>

† The module does not report manufacturing date. Therefore, we report purchase date as an approximation.
‡ Analyzed using the FPGA-based SoftMC.
§ The system runs with double refresh frequency in standard conditions. We configured the refresh interval to be 64 ms in the BIOS settings.

UL = Unlimited

UT = Untested
TRRespass Vulnerable Mobile Phones

**TABLE III: LPDDR4(X) results.** Mobile phones tested against TRRespass on ARMv8 sorted by production date. We found bit flip inducing RowHammer patterns on 5 out of 13 mobile phones.

<table>
<thead>
<tr>
<th>Mobile Phone</th>
<th>Year</th>
<th>SoC</th>
<th>Memory (GB)</th>
<th>Found Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google Pixel</td>
<td>2016</td>
<td>MSM8996</td>
<td>4(^\d)</td>
<td>✓</td>
</tr>
<tr>
<td>Google Pixel 2</td>
<td>2017</td>
<td>MSM8998</td>
<td>4</td>
<td>—</td>
</tr>
<tr>
<td>Samsung G960F/DS</td>
<td>2018</td>
<td>Exynos 9810</td>
<td>4</td>
<td>—</td>
</tr>
<tr>
<td>Huawei P20 DS</td>
<td>2018</td>
<td>Kirin 970</td>
<td>4</td>
<td>—</td>
</tr>
<tr>
<td>Sony XZ3</td>
<td>2018</td>
<td>SDM845</td>
<td>4</td>
<td>—</td>
</tr>
<tr>
<td>HTC U12+</td>
<td>2018</td>
<td>SDM845</td>
<td>6</td>
<td>—</td>
</tr>
<tr>
<td>LG G7 ThinQ</td>
<td>2018</td>
<td>SDM845</td>
<td>4(^\d)</td>
<td>✓</td>
</tr>
<tr>
<td>Google Pixel 3</td>
<td>2018</td>
<td>SDM845</td>
<td>4</td>
<td>✓</td>
</tr>
<tr>
<td>Google Pixel 4</td>
<td>2019</td>
<td>SM8150</td>
<td>6</td>
<td>—</td>
</tr>
<tr>
<td>OnePlus 7</td>
<td>2019</td>
<td>SM8150</td>
<td>8</td>
<td>✓</td>
</tr>
<tr>
<td>Samsung G970F/DS</td>
<td>2019</td>
<td>Exynos 9820</td>
<td>6</td>
<td>✓</td>
</tr>
<tr>
<td>Huawei P30 DS</td>
<td>2019</td>
<td>Kirin 980</td>
<td>6</td>
<td>—</td>
</tr>
<tr>
<td>Xiaomi Redmi Note 8 Pro</td>
<td>2019</td>
<td>Helio G90T</td>
<td>6</td>
<td>—</td>
</tr>
</tbody>
</table>

\(^\d\) LPDDR4 (not LPDDR4X)
# TRRespass Based RowHammer Attack

## TABLE IV: Time to exploit

Time to find the first exploitable template on two sample modules from each DRAM vendor.

<table>
<thead>
<tr>
<th>Module</th>
<th>$\tau$ (ms)</th>
<th>PTE [81]</th>
<th>RSA-2048 [79]</th>
<th>sudo [27]</th>
</tr>
</thead>
<tbody>
<tr>
<td>$A_{14}$</td>
<td>188.7</td>
<td>4.9s</td>
<td>6m 27s</td>
<td>—</td>
</tr>
<tr>
<td>$A_4$</td>
<td>180.8</td>
<td>38.8s</td>
<td>39m 28s</td>
<td>—</td>
</tr>
<tr>
<td>$B_1$</td>
<td>360.7</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>$B_2$</td>
<td>331.2</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<td>$C_{12}$</td>
<td>300.0</td>
<td>2.3s</td>
<td>74.6s</td>
<td>54m 16s</td>
</tr>
<tr>
<td>$C_{13}$</td>
<td>180.9</td>
<td>3h 15m</td>
<td>—</td>
<td>—</td>
</tr>
</tbody>
</table>

$\tau$: Time to template a single row: time to fill the victim and aggressor rows + hammer time + time to scan the row.
TRRespass Key Results

- 13 out of 42 tested DDR4 DRAM modules are vulnerable
  - From all 3 major manufacturers
  - 3-, 9-, 10-, 14-, 19-sided hammer attacks needed

- 5 out of 13 mobile phones tested vulnerable
  - From 4 major manufacturers
  - With LPDDR4(X) DRAM chips

- These results are scratching the surface
  - TRRespass tool is not exhaustive
  - There is a lot of room for uncovering more vulnerable chips and phones
RowHammer is still an open problem

Security by obscurity is likely not a good solution
Detailed Lecture on TRRespass

- Computer Architecture, Fall 2020, Lecture 5a
  - RowHammer in 2020: TRRespass (ETH Zürich, Fall 2020)
  - https://www.youtube.com/watch?v=pwRw7QqK_qA&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=9

https://www.youtube.com/onurmutlulectures
RowHammer in 2020 (II)

- Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi,

"TRRespass: Exploiting the Many Sides of Target Row Refresh"


[Slides (pptx) (pdf)]
[Lecture Slides (pptx) (pdf)]
[Talk Video (17 minutes)]
[Lecture Video (59 minutes)]
[Source Code]
[Web Article]

Best paper award.
Pwnie Award 2020 for Most Innovative Research. Pwnie Awards 2020

TRRespass: Exploiting the Many Sides of Target Row Refresh

Pietro Frigo*† Emanuele Vannacci*† Hasan Hassan§ Victor van der Veen¶
Onur Mutlu§ Cristiano Giuffrida* Herbert Bos* Kaveh Razavi*

*Vrije Universiteit Amsterdam  §ETH Zürich
†University of California, Berkeley  ¶Qualcomm Technologies Inc.
Revisiting RowHammer
Revisiting RowHammer
An Experimental Analysis of Modern Devices and Mitigation Techniques

Jeremie S. Kim    Minesh Patel
A. Giray Yağlıkçı  Hasan Hassan
Roknoddin Azizi    Lois Orosa    Onur Mutlu

SAFARI
Key Conclusions

• We characterized **1580 DRAM** chips of different DRAM types, technology nodes, and manufacturers.

• We studied **five** state-of-the-art RowHammer mitigation mechanisms and an ideal refresh-based mechanism.

• We made **two key observations**

  1. **RowHammer is getting much worse.** It takes much fewer hammers to induce RowHammer bit flips in newer chips
     • e.g., **DDR3**: 69.2k to 22.4k, **DDR4**: 17.5k to 10k, **LPDDR4**: 16.8k to 4.8k

  2. **Existing mitigation mechanisms do not scale** to DRAM chips that are more vulnerable to RowHammer
     • e.g., 80% performance loss when the hammer count to induce the first bit flip is 128

• We **conclude** that it is **critical** to do more research on RowHammer and develop scalable mitigation mechanisms to prevent RowHammer in future systems.
DRAM Testing Infrastructures

Three separate testing infrastructures

1. **DDR3**: FPGA-based SoftMC [Hassan+, HPCA’17] (Xilinx ML605)

2. **DDR4**: FPGA-based SoftMC [Hassan+, HPCA’17] (Xilinx Virtex UltraScale 95)

3. **LPDDR4**: In-house testing hardware for LPDDR4 chips

All provide fine-grained control over DRAM commands, timing parameters and temperature
## DRAM Chips Tested

<table>
<thead>
<tr>
<th>DRAM type-node</th>
<th>Number of Chips (Modules) Tested</th>
<th>Mfr. A</th>
<th>Mfr. B</th>
<th>Mfr. C</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>DDR3-old</td>
<td>56 (10)</td>
<td>88 (11)</td>
<td>28 (7)</td>
<td></td>
<td>172 (28)</td>
</tr>
<tr>
<td>DDR3-new</td>
<td>80 (10)</td>
<td>52 (9)</td>
<td>104 (13)</td>
<td></td>
<td>236 (32)</td>
</tr>
<tr>
<td>DDR4-old</td>
<td>112 (16)</td>
<td>24 (3)</td>
<td>128 (18)</td>
<td></td>
<td>264 (37)</td>
</tr>
<tr>
<td>DDR4-new</td>
<td>264 (43)</td>
<td>16 (2)</td>
<td>108 (28)</td>
<td></td>
<td>388 (73)</td>
</tr>
<tr>
<td>LPDDR4-1x</td>
<td>12 (3)</td>
<td>180 (45)</td>
<td>N/A</td>
<td>192 (48)</td>
<td></td>
</tr>
<tr>
<td>LPDDR4-1y</td>
<td>184 (46)</td>
<td>N/A</td>
<td>144 (36)</td>
<td></td>
<td>328 (82)</td>
</tr>
</tbody>
</table>

**1580 total DRAM chips tested from 300 DRAM modules**

- **Three** major DRAM manufacturers {A, B, C}
- **Three** DRAM **types or standards** {DDR3, DDR4, LPDDR4}
  - LPDDR4 chips we test implement on-die ECC
- **Two** technology nodes per DRAM type {old/new, 1x/1y}
  - Categorized based on manufacturing date, datasheet publication date, purchase date, and characterization results

**Type-node**: configuration describing a chip’s type and technology node generation: DDR3-old/new, DDR4-old/new, LPDDR4-1x/1y
3. Hammer Count (HC) Effects

RowHammer bit flip rates increase when going from old to new DDR4 technology node generations.

RowHammer bit flip rates (i.e., RowHammer vulnerability) increase with technology node generation.
5. First RowHammer Bit Flips per Chip

What is the minimum Hammer Count required to cause bit flips ($HC_{\text{first}}$)?

We note the different DRAM types on the x-axis: DDR3, DDR4, LPDDR4.

We focus on trends across chips of the same DRAM type to draw conclusions.
5. First RowHammer Bit Flips per Chip

Newer chips from a given DRAM manufacturer more vulnerable to RowHammer
5. First RowHammer Bit Flips per Chip

In a DRAM type, $\text{HC}_{\text{first}}$ reduces significantly from old to new chips, i.e., DDR3: 69.2k to 22.4k, DDR4: 17.5k to 10k, LPDDR4: 16.8k to 4.8k.

There are chips whose weakest cells fail after only 4800 hammers.

Newer chips from a given DRAM manufacturer are more vulnerable to RowHammer.
Key Takeaways from 1580 Chips

• Chips of newer DRAM technology nodes are more vulnerable to RowHammer.

• There are chips today whose weakest cells fail after only 4800 hammers.

• Chips of newer DRAM technology nodes can exhibit RowHammer bit flips 1) in more rows and 2) farther away from the victim row.
Mitigation Mechanism Evaluation

**PARA, ProHIT, and MRLoc** mitigate RowHammer bit flips in worst chips today with reasonable system performance (92%, 100%, 100%)
Mitigation Mechanism Evaluation

Only PARA’s design scales to low $HC_{first}$ values but has very low normalized system performance.
Mitigation Mechanism Evaluation

**Ideal mechanism is significantly better than any existing mechanism for** $\text{HC}_{\text{first}} < 1024$

**Significant opportunity for developing a RowHammer solution with low performance overhead that supports low** $\text{HC}_{\text{first}}$
Key Takeaways from Mitigation Mechanisms

• Existing RowHammer mitigation mechanisms can prevent RowHammer attacks with reasonable system performance overhead in DRAM chips today.

• Existing RowHammer mitigation mechanisms do not scale well to DRAM chips more vulnerable to RowHammer.

• There is still significant opportunity for developing a mechanism that is scalable with low overhead.
RowHammer Solutions Going Forward

Two promising directions for new RowHammer solutions:

1. **DRAM-system cooperation**
   - We believe the DRAM and system should cooperate more to provide a holistic solution can prevent RowHammer at low cost

2. **Profile-guided**
   - Accurate profile of RowHammer-susceptible cells in DRAM provides a powerful substrate for building targeted RowHammer solutions, e.g.:
     - Only increase the refresh rate for rows containing RowHammer-susceptible cells
   - A fast and accurate profiling mechanism is a key research challenge for developing low-overhead and scalable RowHammer solutions
Revisiting RowHammer
An Experimental Analysis of Modern Devices and Mitigation Techniques

Jeremie S. Kim       Minesh Patel
A. Giray Yağlıkçı   Hasan Hassan
Roknoddin Azizi      Lois Orosa    Onur Mutlu

SAFARI

ETH Zürich

Carnegie Mellon
Computer Architecture, Fall 2020, Lecture 5b

- RowHammer in 2020: Revisiting RowHammer (ETH Zürich, Fall 2020)
- https://www.youtube.com/watch?v=gR7XR-Eepcg&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=10

https://www.youtube.com/onurmutlulectures
Revisiting RowHammer in 2020 (I)

 Jeremie S. Kim, Minesh Patel, A. Giray Yaglikci, Hasan Hassan, Roknoddin Azizi, Lois Orosa, and Onur Mutlu,
 "Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques"
[Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (20 minutes)]
[Lightning Talk Video (3 minutes)]

Revisiting RowHammer: An Experimental Analysis of Modern DRAM Devices and Mitigation Techniques

 Jeremie S. Kim$†, Minesh Patel$, A. Giray Yağlıkçı$
 Hasan Hassan$, Roknoddin Azizi$, Lois Orosa$
 Onur Mutlu$†$

$ETH Zürich †Carnegie Mellon University
More RowHammer in 2020
RowHammer in 2020 (IV)

Session 1A: Security & Privacy I

5:00 PM CEST – 5:15 PM CEST
Graphene: Strong yet Lightweight Row Hammer Protection
Yeonhong Park, Woosuk Kwon, Eojin Lee, Tae Jun Ham, Jung Ho Ahn, Jae W. Lee (Seoul National University)

5:15 PM CEST – 5:30 PM CEST
Persist Level Parallelism: Streamlining Integrity Tree Updates for Secure Persistent Memory
Alexander Freij, Shougang Yuan, Huiyang Zhou (NC State University); Yan Solihin (University of Central Florida)

5:30 PM CEST – 5:45 PM CEST
PThammer: Cross-User-Kernel-Boundary Rowhammer through Implicit Accesses
Zhi Zhang (University of New South Wales and Data61, CSIRO, Australia); Yueqiang Cheng (Baidu Security); Dongxi Liu, Surya Nepal (Data61, CSIRO, Australia); Zhi Wang (Florida State University); Yuval Yarom (University of Adelaide and Data61, CSIRO, Australia)
RowHammer in 2020 (V)

Session #5: RowHammer

Session chair: Michael Franz (UC Irvine)

**RAMBleed: Reading Bits in Memory Without Accessing Them**
Andrew Kwong (University of Michigan), Daniel Genkin (University of Michigan), Daniel Gruss (Data61)

**Are We Susceptible to Rowhammer? An End-to-End Methodology for Cloud Providers**
Lucian Cojocar (Microsoft Research), Jeremie Kim (ETH Zurich, CMU), Minesh Patel (ETH Zurich, Microsoft Research), Onur Mutlu (ETH Zurich, CMU)

**Leveraging EM Side-Channel Information to Detect Rowhammer Attacks**
Zhenkai Zhang (Texas Tech University), Zihao Zhan (Vanderbilt University), Daniel Balasubramanian (Vanderbilt University), Peter Volgyesi (Vanderbilt University), Xenofon Koutsoukos (Vanderbilt University)

**TRRespass: Exploiting the Many Sides of Target Row Refresh**
Pietro Frigo (Vrije Universiteit Amsterdam, The Netherlands), Emanuele Vannacci (Vrije Universiteit Amsterdam), Victor Veen (Qualcomm Technologies, Inc.), Onur Mutlu (ETH Zürich), Cristiano Giuffrida (Vrije Universiteit Amsterdam, The Netherlands), Kaveh Razavi (Vrije Universiteit Amsterdam, The Netherlands)
DeepHammer: Depleting the Intelligence of Deep Neural Networks through Targeted Chain of Bit Flips
Fan Yao, University of Central Florida; Adnan Siraj Rakin and Deliang Fan, Arizona State University
RowHammer in 2021

HPCA 2021

The 27th IEEE International Symposium on High-Performance Computer Architecture (HPCA-27), Seoul, South Korea

| BlockHammer: Preventing RowHammer at Low Cost by Blacklisting Rapidly-Accessed DRAM Rows |
| Abdullah Giray Yaglikci, Minesh Patel (ETH Zürich); Jeremie Kim (ETH Zurich); Roknoddin AziziBarzoki (ETH Zürich); Jisung Park (ETH Zurich); Hasan Hassan, Ataberk Olgun, Lois Orosa, Konstantinos Kanellopoulos, Taha Shahroodi (ETH Zürich); Saugata Ghose (University of Illinois at Urbana-Champaign / Carnegie Mellon University); Onur Mutlu (ETH Zürich) |
More to Come…
Future Memory
Reliability/Security Challenges
Future of Main Memory Security

- DRAM is becoming less reliable → more vulnerable

- Due to difficulties in DRAM scaling, other problems may also appear (or they may be going unnoticed)

- Some errors may already be slipping into the field
  - Read disturb errors (Rowhammer)
  - Retention errors
  - Read errors, write errors
  - ...

- These errors can also pose security vulnerabilities
The Takeaway, Reinforced

Main Memory Needs
Intelligent Controllers for Security
Keeping Future Memory Secure
How Do We Keep Memory Secure?

- DRAM
- Flash memory
- Emerging Technologies
  - Phase Change Memory
  - STT-MRAM
  - RRAM, memristors
  - ...

SAFARI
Solution Direction: Principled Designs

Design fundamentally secure computing architectures

Predict and prevent such safety issues
Architecting Future Memory for Security

- **Understand**: Methods for vulnerability modeling & discovery
  - Modeling and prediction based on real (device) data and analysis
  - Understanding vulnerabilities
  - Developing reliable metrics

- **Architect**: Principled architectures with security as key concern
  - Good partitioning of duties across the stack
  - Cannot give up performance and efficiency
  - Patch-ability in the field

- **Design & Test**: Principled design, automation, (online) testing
  - Design for security
  - High coverage and good interaction with system reliability methods
Understand and Model with Experiments (DRAM)

Understand and Model with Experiments (Flash)

HAPS-52 Mother Board

Virtex-V FPGA (NAND Controller)

USB Daughter Board

Virtex-II Pro (USB controller)

USB Jack

NAND Daughter Board

1x-nm NAND Flash

Collapse of the “Galloping Gertie” (1940)

Source: AP
http://www.wsdot.wa.gov/tnbhistory/connections/connections3.htm
Another Example (1994)
Yet Another Example (2007)

A More Recent Example (2018)

The Takeaway, Again

In-Field Patch-ability (Intelligent Memory) Can Avoid Such Failures
Final Thoughts on RowHammer
Before RowHammer (I)

Using Memory Errors to Attack a Virtual Machine

Sudhakar Govindavajhala * Andrew W. Appel
Princeton University
{sudhakar,appel}@cs.princeton.edu

We present an experimental study showing that soft memory errors can lead to serious security vulnerabilities in Java and .NET virtual machines, or in any system that relies on type-checking of untrusted programs as a protection mechanism. Our attack works by sending to the JVM a Java program that is designed so that almost any memory error in its address space will allow it to take control of the JVM. All conventional Java and .NET virtual machines are vulnerable to this attack. The technique of the attack is broadly applicable against other language-based security schemes such as proof-carrying code.

We measured the attack on two commercial Java Virtual Machines: Sun’s and IBM’s. We show that a single-bit error in the Java program’s data space can be exploited to execute arbitrary code with a probability of about 70%, and multiple-bit errors with a lower probability.

Our attack is particularly relevant against smart cards or tamper-resistant computers, where the user has physical access (to the outside of the computer) and can use various means to induce faults; we have successfully used heat. Fortunately, there are some straightforward defenses against this attack.

7 Physical fault injection

If the attacker has physical access to the outside of the machine, as in the case of a smart card or other tamper-resistant computer, the attacker can induce memory errors. We considered attacks on boxes in form factors ranging from a credit card to a palmtop to a desktop PC.

We considered several ways in which the attacker could induce errors.4

IEEE S&P 2003

Before RowHammer (II)

Using Memory Errors to Attack a Virtual Machine

Sudhakar Govindavajhala * Andrew W. Appel
Princeton University
{svdhakar,appel}@cs.princeton.edu

Figure 3. Experimental setup to induce memory errors, showing a PC built from surplus components, clip-on gooseneck lamp, 50-watt spotlight bulb, and digital thermometer. Not shown is the variable AC power supply for the lamp.
After RowHammer: Byzantine Failures

- This class of failures is known as **Byzantine failures**

- Characterized by
  - Undetected erroneous computation
  - Opposite of “fail fast (with an error or no result)”

- “erroneous” can be “malicious” (intent is the only distinction)

- Very difficult to detect and confine Byzantine failures

- **Do all you can to avoid them**

The Byzantine Generals Problem

LESLIE LAMPORT, ROBERT SHOSTAK, and MARSHALL PEASE
SRI International

Reliable computer systems must handle malfunctioning components that give conflicting information to different parts of the system. This situation can be expressed abstractly in terms of a group of generals of the Byzantine army camped with their troops around an enemy city. Communicating only by messenger, the generals must agree upon a common battle plan. However, one or more of them may be traitors who will try to confuse the others. The problem is to find an algorithm to ensure that the loyal generals will reach agreement. It is shown that, using only oral messages, this problem is solvable if and only if more than two-thirds of the generals are loyal; so a single traitor can confound two loyal generals. With unforgeable written messages, the problem is solvable for any number of generals and possible traitors. Applications of the solutions to reliable computer systems are then discussed.

Categories and Subject Descriptors: C.2.4. [Computer-Communication Networks]: Distributed Systems—network operating systems; D.4.4 [Operating Systems]: Communications Management—network communication; D.4.5 [Operating Systems]: Reliability—fault tolerance

General Terms: Algorithms, Reliability

Additional Key Words and Phrases: Interactive consistency

https://dl.acm.org/citation.cfm?id=357176
RowHammer, Revisited

- One can predictably induce bit flips in commodity DRAM chips
  - >80% of the tested DRAM chips are vulnerable

- First example of how a simple hardware failure mechanism can create a widespread system security vulnerability
RowHammer: Retrospective

- New mindset that has enabled a renewed interest in HW security attack research:
  - Real (memory) chips are vulnerable, in a simple and widespread manner → this causes real security problems
  - Hardware reliability → security connection is now mainstream discourse

- Many new RowHammer attacks...
  - Tens of papers in top security venues
  - More to come as RowHammer is getting worse (DDR4 & beyond)

- Many new RowHammer solutions...
  - Apple security release; Memtest86 updated
  - Many solution proposals in top venues (latest in ISCA 2019)
  - Principled system-DRAM co-design (in original RowHammer paper)
  - More to come...
Perhaps Most Importantly…

- RowHammer enabled a shift of mindset in mainstream security researchers
  - General-purpose hardware is fallible, in a widespread manner
  - Its problems are exploitable

- This mindset has enabled many systems security researchers to examine hardware in more depth
  - And understand HW’s inner workings and vulnerabilities

- It is no coincidence that two of the groups that discovered Meltdown and Spectre heavily worked on RowHammer attacks before
  - More to come…
Conclusion
Summary: RowHammer

- Memory reliability is reducing
- Reliability issues open up security vulnerabilities
  - Very hard to defend against

**Rowhammer is a prime example**
- First example of how a simple hardware failure mechanism can create a widespread system security vulnerability
- Its implications on system security research are tremendous & exciting

**Bad news:** RowHammer is getting worse.

**Good news:** We have a lot more to do.
- We are now fully aware hardware is easily fallible.
- We are developing both attacks and solutions.
- We are developing principled models, methodologies, solutions.
For More on RowHammer…

- Onur Mutlu and Jeremie Kim, "RowHammer: A Retrospective"
  [Preliminary arXiv version]
  [Slides from COSADE 2019 (pptx)]
  [Slides from VLSI-SOC 2020 (pptx) (pdf)]
  [Talk Video (30 minutes)]

---

**RowHammer: A Retrospective**

Onur Mutlu$^§$†‡
§ETH Zürich

Jeremie S. Kim$^‡$§
†Carnegie Mellon University
Revisiting RowHammer: An Experimental Analysis of Modern Devices and Mitigation Techniques

Jeremie S. Kim, Minesh Patel, A. Giray Yaglikci, Hasan Hassan, Roknoddin Azizi, Lois Orosa, and Onur Mutlu,


[Slides (pptx) (pdf)]
[Lightning Talk Slides (pptx) (pdf)]
[Talk Video (20 minutes)]
[Lightning Talk Video (3 minutes)]
TRRespass: Exploiting the Many Sides of Target Row Refresh

Pietro Frigo*, Emanuele Vannacci*, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi

*Vrije Universiteit Amsterdam
†Microsoft Research
§ETH Zürich
¶Qualcomm Technologies Inc.
RowHammer in 2020 (III)

- Lucian Cojocar, Jeremie Kim, Minesh Patel, Lillian Tsai, Stefan Saroiu, Alec Wolman, and Onur Mutlu,

"Are We Susceptible to Rowhammer? An End-to-End Methodology for Cloud Providers"


[Slides (pptx) (pdf)]
[Talk Video (17 minutes)]

Are We Susceptible to Rowhammer?
An End-to-End Methodology for Cloud Providers

Lucian Cojocar, Jeremie Kim§†, Minesh Patel§, Lillian Tsai‡, Stefan Saroiu, Alec Wolman, and Onur Mutlu§†
Microsoft Research, §ETH Zürich, †CMU, ‡MIT
Detailed Lectures on RowHammer

- **Computer Architecture, Fall 2020, Lecture 4b**
  - RowHammer (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=KDy632z23UE&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=8](https://www.youtube.com/watch?v=KDy632z23UE&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=8)

- **Computer Architecture, Fall 2020, Lecture 5a**
  - RowHammer in 2020: TRRespass (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=pwRw7QqK_qA&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=9](https://www.youtube.com/watch?v=pwRw7QqK_qA&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=9)

- **Computer Architecture, Fall 2020, Lecture 5b**
  - RowHammer in 2020: Revisiting RowHammer (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=qR7XR-Eepcg&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=10](https://www.youtube.com/watch?v=qR7XR-Eepcg&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=10)

- **Computer Architecture, Fall 2020, Lecture 5c**
  - Secure and Reliable Memory (ETH Zürich, Fall 2020)
  - [https://www.youtube.com/watch?v=HvswnsfG3oQ&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=11](https://www.youtube.com/watch?v=HvswnsfG3oQ&list=PL5Q2soXY2Zi9xidyIgBxUz7xRPS-wisBN&index=11)

**SAFARI**

[https://www.youtube.com/onurmutlulectures](https://www.youtube.com/onurmutlulectures)
Funding Acknowledgments

- Alibaba, AMD, ASML, Google, Facebook, Hi-Silicon, HP Labs, Huawei, IBM, Intel, Microsoft, Nvidia, Oracle, Qualcomm, Rambus, Samsung, Seagate, VMware
- NSF
- NIH
- GSRC
- SRC
- CyLab
Acknowledgments

- **My current and past students and postdocs**
  - Rachata Ausavarungnirun, Abhishek Bhowmick, Amirali Boroumand, Rui Cai, Yu Cai, Kevin Chang, Saugata Ghose, Kevin Hsieh, Tyler Huberty, Ben Jaiyen, Samira Khan, Jeremie Kim, Yoongu Kim, Yang Li, Jamie Liu, Lavanya Subramanian, Donghyuk Lee, Yixin Luo, Justin Meza, Gennady Pekhimenko, Vivek Seshadri, Lavanya Subramanian, Nandita Vijaykumar, HanBin Yoon, Jishen Zhao, ...

- **My collaborators**
  - Can Alkan, Chita Das, Phil Gibbons, Sriram Govindan, Norm Jouppi, Mahmut Kandemir, Mike Kozuch, Konrad Lai, Ken Mai, Todd Mowry, Yale Patt, Moinuddin Qureshi, Partha Ranganathan, Bikash Sharma, Kushagra Vaid, Chris Wilkerson, ...
Acknowledgments

SAFARI
SAFARI Research Group
safari.ethz.ch

Think BIG, Aim HIGH!

https://safari.ethz.ch
Onur Mutlu’s SAFARI Research Group

Computer architecture, HW/SW, systems, bioinformatics, security, memory

https://safari.ethz.ch/safari-newsletter-january-2021/

38+ Researchers

Think BIG, Aim HIGH!

SAFARI

https://safari.ethz.ch
Dear SAFARI friends,

2019 and the first three months of 2020 have been very positive and eventful times for SAFARI.
Dear SAFARI friends,

Happy New Year! We are excited to share our group highlights with you in this second edition of the SAFARI newsletter (You can find the first edition from April 2020 here). 2020 has
The Story of RowHammer

Onur Mutlu
omutlu@gmail.com
https://people.inf.ethz.ch/omutlu

19 January 2021
SEHAS Keynote @ HiPEAC
Backup Slides for Further Info
Research & Teaching: Some Overview Talks

https://www.youtube.com/onurmutlulectures

- **Future Computing Architectures**
  - https://www.youtube.com/watch?v=kgiZI5OcGFM&list=PL5Q2soXY2Zi8d_5MGV6EnXEJHnV2YFBjl&index=1

- **Enabling In-Memory Computation**
  - https://www.youtube.com/watch?v=njX_14584Jw&list=PL5Q2soXY2Zi8d_5MGV6EnXEJHnV2YFBjl&index=16

- **Accelerating Genome Analysis**
  - https://www.youtube.com/watch?v=hPnSmfwu2-A&list=PL5Q2soXY2Zi8d_5MGV6EnXEJHnV2YFBjl&index=9

- **Rethinking Memory System Design**
  - https://www.youtube.com/watch?v=F7xZLNMIY1E&list=PL5Q2soXY2Zi8d_5MGV6EnXEJHnV2YFBjl&index=3

- **Intelligent Architectures for Intelligent Machines**
  - https://www.youtube.com/watch?v=n8Aj_A0WSq8&list=PL5Q2soXY2Zi8d_5MGV6EnXEJHnV2YFBjl&index=22

- **Revisiting RowHammer**
  - https://www.youtube.com/watch?v=B58YT9hZM4g&list=PL5Q2soXY2Zi8d_5MGV6EnXEJHnV2YFBjl&index=25
An Interview on Research and Education

- Computing Research and Education (@ ISCA 2019)
  - https://www.youtube.com/watch?v=8ffSEKZhmv0&list=PL5Q2soXY2Zi_4oP9LdL3cc8G6NIjD2Ydz

- Maurice Wilkes Award Speech (10 minutes)
  - https://www.youtube.com/watch?v=tcQ3z3JpuA&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=15
More Thoughts and Suggestions

- Onur Mutlu, "Some Reflections (on DRAM)"
  Award Speech for ACM SIGARCH Maurice Wilkes Award, at the ISCA Awards Ceremony, Phoenix, AZ, USA, 25 June 2019.
  [Slides (pptx) (pdf)]
  [Video of Award Acceptance Speech (Youtube; 10 minutes) (Youku; 13 minutes)]
  [Video of Interview after Award Acceptance (Youtube; 1 hour 6 minutes) (Youku; 1 hour 6 minutes)]
  [News Article on "ACM SIGARCH Maurice Wilkes Award goes to Prof. Onur Mutlu"]

- Onur Mutlu, "How to Build an Impactful Research Group"
  57th Design Automation Conference Early Career Workshop (DAC), Virtual, 19 July 2020.
  [Slides (pptx) (pdf)]
RowHammer Solutions
Naive Solutions

1. **Throttle accesses to same row**
   - Limit access-interval: $\geq 500\text{ns}$
   - Limit number of accesses: $\leq 128\text{K} (=64\text{ms}/500\text{ns})$

2. **Refresh more frequently**
   - Shorten refresh-interval by $\sim 7x$

Both naive solutions introduce significant overhead in performance and power
Industry Is Writing Papers About It, Too

DRAM Process Scaling Challenges

- **Refresh**
  - Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance
  - Leakage current of cell access transistors increasing

- **tWR**
  - Contact resistance between the cell capacitor and access transistor increasing
  - On-current of the cell access transistor decreasing
  - Bit-line resistance increasing

- **VRT**
  - Occurring more frequently with cell capacitance decreasing
Industry Is Writing Papers About It, Too

DRAM Process Scaling Challenges

- Refresh
  - Difficult to build high-aspect ratio cell capacitors decreasing cell capacitance

THE MEMORY FORUM 2014

Co-Architecting Controllers and DRAM to Enhance DRAM Process Scaling

Uksong Kang, Hak-soo Yu, Churoo Park, *Hongzhong Zheng, **John Halbert, **Kuljit Bains, SeongJin Jang, and Joo Sun Choi

Samsung Electronics, Hwasung, Korea / *Samsung Electronics, San Jose / **Intel
Revisiting RowHammer in 2020
Executive Summary

- **Motivation**: Denser DRAM chips are more vulnerable to RowHammer but no characterization-based study demonstrates how vulnerability scales

- **Problem**: Unclear if existing mitigation mechanisms will remain viable for future DRAM chips that are likely to be more vulnerable to RowHammer

- **Goal**:
  1. Experimentally demonstrate how vulnerable modern DRAM chips are to RowHammer and study how this vulnerability will scale going forward
  2. Study viability of existing mitigation mechanisms on more vulnerable chips

- **Experimental Study**: First rigorous RowHammer characterization study across a broad range of DRAM chips
  - 1580 chips of different DRAM {types, technology node generations, manufacturers}
  - We find that RowHammer vulnerability worsens in newer chips

- **RowHammer Mitigation Mechanism Study**: How five state-of-the-art mechanisms are affected by worsening RowHammer vulnerability
  - Reasonable performance loss (8% on average) on modern DRAM chips
  - Scale poorly to more vulnerable DRAM chips (e.g., 80% performance loss)

- **Conclusion**: it is critical to research more effective solutions to RowHammer for future DRAM chips that will likely be even more vulnerable to RowHammer

*SAFARI*
Motivation

- Denser DRAM chips are more vulnerable to RowHammer.

- Three prior works [Kim+, ISCA’14], [Park+, MR’16], [Park+, MR’16], over the last six years provide RowHammer characterization data on real DRAM.

- However, there is no comprehensive experimental study that demonstrates how vulnerability scales across DRAM types and technology node generations.

- It is unclear whether current mitigation mechanisms will remain viable for future DRAM chips that are likely to be more vulnerable to RowHammer.
Goal

1. Experimentally demonstrate how vulnerable modern DRAM chips are to RowHammer and predict how this vulnerability will scale going forward

2. Examine the viability of current mitigation mechanisms on more vulnerable chips
Effective RowHammer Characterization

To characterize our DRAM chips at worst-case conditions, we:

1. **Prevent sources of interference during core test loop**
   - We disable:
     - **DRAM refresh**: to avoid refreshing victim row
     - **DRAM calibration events**: to minimize variation in test timing
     - **RowHammer mitigation mechanisms**: to observe circuit-level effects
   - Test for less than refresh window (32ms) to avoid retention failures

2. **Worst-case access sequence**
   - We use worst-case access sequence based on prior works’ observations
   - For each row, repeatedly access the two directly physically-adjacent rows as fast as possible

[More details in the paper]
Worst-case RowHammer Access Sequence.

Using that for our LPDDR4-1x chips from Manufacturer B, when key observation that repeatedly accessing an arbitrary row undocumented activations is limited by the DRAM timing parameter retention failures [53, 82, 97] so that we do not con.

First, we want to ensure that we have control over how our test. We can then use this mapping information to quickly deduce the address mappings for each type of chip that we target physical rows.

Testing Methodology

<table>
<thead>
<tr>
<th>Row 0</th>
<th>Aggressor Row</th>
</tr>
</thead>
<tbody>
<tr>
<td>REFRESH</td>
<td>Row 1</td>
</tr>
<tr>
<td>Row 2</td>
<td>Aggressor Row</td>
</tr>
<tr>
<td>Row 3</td>
<td>Row</td>
</tr>
<tr>
<td>Row 4</td>
<td>Row</td>
</tr>
<tr>
<td>Row 5</td>
<td>Row</td>
</tr>
</tbody>
</table>

Algorithm 1:

```python
DRAM_RowHammer_Characterization():
    foreach row in DRAM:
        set victim_row to row
        set aggressor_row1 to victim_row – 1
        set aggressor_row2 to victim_row + 1
        Disable DRAM refresh
        Refresh victim_row
        for n = 1 → HC: // core test loop
            activate aggressor_row1
            activate aggressor_row2
        Enable DRAM refresh
        Record RowHammer bit flips to storage
        Restore bit flips to original values
```

This indicates a row address remapping that is internal to the DRAM chip such that every pair of consecutive rows share the same internal wordline. To account for this DRAM-internal effects of RowHammer on our DRAM chips.

To directly observe effectiveness of we leverage stable ambient temperature of 50°C. We test the effects of changing the data, Checkered1 (CH1: 0xAA) or Rowstripe1 (RS1: 0xFF), in physical rows.

- stripe0 (CO0: 0x55), Colstripe1 (CO1: 0xAA) [53, 82, 97]. In this end, to the best of our knowledge, we disable all DRAM-physically-adjacent rows as fast as possible.
- the retention time violations. The core loop of our RowHammer test runs for less than 64 ms (i.e., the refresh pattern in any way. Therefore, during the core loop of each RowHammer test (i.e., when activations are issued at +1 for a given row we repeatedly access a single row within two consecutive – 1).
- Second, a we repeatedly access an arbitrary row undocument number of times we access (i.e., activate) a victim row's.

SAFARI
Testing Methodology

<table>
<thead>
<tr>
<th>closed</th>
<th>Row 0</th>
<th>Aggressor Row</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Row 1</td>
<td>Aggressor Row</td>
</tr>
<tr>
<td></td>
<td>Row 2</td>
<td>Row</td>
</tr>
<tr>
<td></td>
<td>Row 3</td>
<td>Aggressor Row</td>
</tr>
<tr>
<td></td>
<td>Row 4</td>
<td>Victim Row</td>
</tr>
<tr>
<td></td>
<td>Row 5</td>
<td>Aggressor Row</td>
</tr>
</tbody>
</table>

**DRAM_RowHammer_Characterization()**:

```python
foreach row in DRAM:
    set victim_row to row
    set aggressor_row1 to victim_row - 1
    set aggressor_row2 to victim_row + 1
    Disable DRAM refresh
    Refresh victim_row
    for n = 1 to HC: // core test loop
        activate aggressor_row1
        activate aggressor_row2
        Enable DRAM refresh
        Record RowHammer bit flips to storage
        Restore bit flips to original values
```

- Disable refresh to **prevent interruptions** in the core loop of our test from refresh operations
- Induce RowHammer bit flips on a **fully charged row**
- Core test loop where we alternate accesses to adjacent rows
- **1 Hammer (HC) = two accesses**
- Prevent further retention failures
- Record bit flips for analysis
1. RowHammer Vulnerability

Q. Can we induce RowHammer bit flips in all of our DRAM chips?

All chips are vulnerable, except many DDR3 chips

- A total of 1320 out of all 1580 chips (84%) are vulnerable
- Within DDR3-old chips, only 12% of chips (24/204) are vulnerable
- Within DDR3-new chips, 65% of chips (148/228) are vulnerable

Newer DRAM chips are more vulnerable to RowHammer
2. Data Pattern Dependence

Q. Are some data patterns more effective in inducing RowHammer bit flips?

- We test several data patterns typically examined in prior work to identify the worst-case data pattern.

- The worst-case data pattern is consistent across chips of the same manufacturer and DRAM type-node configuration.

- We use the worst-case data pattern per DRAM chip to characterize each chip at worst-case conditions and minimize the extensive testing time.

[More detail and figures in paper]
3. Hammer Count (HC) Effects

Q. How does the Hammer Count affect the number of bit flips induced?

Hammer Count = 2 Accesses, one to each adjacent row of victim
4. Spatial Effects: Row Distance

Q. Where do RowHammer bit flips occur relative to aggressor rows?

The number of RowHammer bit flips that occur in a given row decreases as the distance from the victim row (row 0) increases.
4. Spatial Effects: Row Distance

We normalize data by inducing a bit flip rate of $10^{-6}$ in each chip.

Chips of newer DRAM technology nodes can exhibit RowHammer bit flips 1) in more rows and 2) farther away from the victim row.
4. Spatial Effects: Row Distance

We plot this data for each DRAM type-node configuration per manufacturer:

[More analysis in the paper]
4. Spatial Distribution of Bit Flips

Q. How are RowHammer bit flips spatially distributed across a chip?

We normalize data by inducing a bit flip rate of $10^{-6}$ in each chip.

The distribution of RowHammer bit flip density per word changes significantly in LPDDR4 chips from other DRAM types.

At a bit flip rate of $10^{-6}$, a 64-bit word can contain up to 4 bit flips. Even at this very low bit flip rate, a very strong ECC is required.
4. Spatial Distribution of Bit Flips

We plot this data for each DRAM type-node configuration per manufacturer.

[More analysis in the paper]
5. First RowHammer Bit Flips per Chip

What is the minimum Hammer Count required to cause bit flips ($HC_{first}$)?

![Box plot showing hammer count needed for the first bit flip for different memory types.](image)

- **Whisker**
- **Q3: 75% point**
- **Median: 50%**
- **Q1: 25% point**
- **Whisker**
Evaluation Methodology

- **Cycle-level simulator:** Ramulator [Kim+, CAL’15]
  [https://github.com/CMU-SAFARI/ramulator](https://github.com/CMU-SAFARI/ramulator)
  - 4GHz, 4-wide, 128 entry instruction window
  - 48 8-core workload mixes randomly drawn from SPEC CPU2006 (10 < MPKI < 740)

- **Metrics to evaluate mitigation mechanisms**
  1. **DRAM Bandwidth Overhead:** fraction of total system DRAM bandwidth consumption from mitigation mechanism
  2. **Normalized System Performance:** normalized weighted speedup to a 100% baseline
Evaluation Methodology

• We evaluate **five state-of-the-art mitigation mechanisms**:  
  - **Increased Refresh Rate** [Kim+, ISCA’14]  
  - **PARA** [Kim+, ISCA’14]  
  - **ProHIT** [Son+, DAC’17]  
  - **MRLoc** [You+, DAC’19]  
  - **TWiCe** [Lee+, ISCA’19]

• and **one ideal refresh-based mitigation mechanism**:  
  - **Ideal**

• More detailed descriptions in the paper on:  
  - Descriptions of mechanisms in our paper and the original publications  
  - How we scale each mechanism to more vulnerable DRAM chips (lower $HC_{first}$)
Mitigation Mech. Eval. (Increased Refresh)

Substantial overhead for high $HC_{\text{first}}$ values.

This mechanism does not support $HC_{\text{first}} < 32k$ due to the prohibitively high refresh rates required.
Mitigation Mechanism Evaluation (PARA)

- **Low Performance Overhead**
  - **High Performance Overhead**

$H_{C_{\text{first}}}$ (number of hammers required to induce first RowHammer bit flip)

Normalized System Performance (%)

- **Mfr. A**
- **Mfr. B**
- **Mfr. C**

80% performance loss
Mitigation Mechanism Evaluation (ProHIT)

Normalized System Performance (%) vs. $HC_{\text{first}}$ (number of hammers required to induce first RowHammer bit flip)

- Increased Refresh Rate
- PARA
- ProHIT

SAFARI
Mitigation Mechanism Evaluation (MRLoc)

Models for scaling ProHIT and MRLoc for $HC_{\text{first}} < 2k$ are not provided and how to do so is not intuitive.
Mitigation Mechanism Evaluation (TWiCe)

We evaluate an ideal scalable version (TWiCe-ideal) assuming it solves two critical design issues.
Mitigation Mechanism Evaluation (Ideal)

- **Para**
- **MFR A**
- **MFR B**
- **MFR C**

**HC** (number of hammers required to induce first RowHammer bit flip)

**Increased Refresh Rate**

**ProHit**

**MR Loc**

**TWC e**

**TWC e-ideal**

**Ideal**

**6% performance loss**

**Ideal mechanism** issues a refresh command to a row only right before the row can potentially experience a RowHammer bit flip.
Additional Details in the Paper

• Single-cell RowHammer bit flip probability

• More details on our data pattern dependence study

• Analysis of Error Correcting Codes (ECC) in mitigating RowHammer bit flips

• Additional observations on our data

• Methodology details for characterizing DRAM

• Further discussion on comparing data across different infrastructures

• Discussion on scaling each mitigation mechanism
RowHammer Reviews
Initial RowHammer Reviews

Disturbance Errors in DRAM: Demonstration, Characterization, and Prevention

Rejected (R2) 863kB Friday 31 May 2013 2:00:53pm PDT

b9bf06021da54cddf4cd0b3565558a181868b972

You are an author of this paper.

+ ABSTRACT

<table>
<thead>
<tr>
<th>Review #66A</th>
<th>Review #66B</th>
<th>Review #66C</th>
<th>Review #66D</th>
<th>Review #66E</th>
<th>Review #66F</th>
</tr>
</thead>
<tbody>
<tr>
<td>OveMer</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>Nov</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>WriQua</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>RevExp</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>
Missing the Point  Reviews from Micro 2013

**Paper Weaknesses**

This is an excellent test methodology paper, but there is no micro-architectural or architectural content.

**Paper Weaknesses**

- Whereas they show disturbance may happen in DRAM array, authors don't show it can be an issue in realistic DRAM usage scenario.
- Lacks architectural/microarchitectural impact on the DRAM disturbance analysis.

**Paper Weaknesses**

The mechanism investigated by the authors is one of many well known disturb mechanisms. The paper does not discuss the root causes to sufficient depth and the importance of this mechanism compared to others. Overall the length of the sections restating known information is much too long in relation to new work.
More…

Reviews from ISCA 2014

**Paper Weaknesses**

1) The disturbance error (a.k.a coupling or cross-talk noise induced error) is a known problem to the DRAM circuit community.

2) What you demonstrated in this paper is so called DRAM row hammering issue - you can even find a Youtube video showing this! - [http://www.youtube.com/watch?v=i3-qQSnBcdo](http://www.youtube.com/watch?v=i3-qQSnBcdo)

- The architectural contribution of this study is too insignificant.

**Paper Weaknesses**

- Row Hammering appears to be well-known, and solutions have already been proposed by industry to address the issue.

- The paper only provides a qualitative analysis of solutions to the problem. A more robust evaluation is really needed to know whether the proposed solution is necessary.
Final RowHammer Reviews

Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

Accepted 639kB 21 Nov 2013 10:53:11pm CST | f039be2735313b39304ae1c6296523867a485610

You are an author of this paper.

<table>
<thead>
<tr>
<th>Review #41A</th>
<th>OveMer</th>
<th>Nov</th>
<th>WriQua</th>
<th>RevConAnd</th>
</tr>
</thead>
<tbody>
<tr>
<td>Review #41B</td>
<td>8</td>
<td>4</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>Review #41C</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Review #41D</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>Review #41E</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>Review #41F</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
Some More History
Some More Historical Perspective

- RowHammer is the first example of a circuit-level failure mechanism causing a widespread system security vulnerability.

- It led to a large body of work in security attacks, mitigations, architectural solutions, ...

- Work building on RowHammer still continues

- Initially, RowHammer was dismissed by some reviewers
  - Rejected from MICRO 2013 conference
#66 Disturbance Errors in DRAM: Demonstration, Characterization, and Prevention

**Rejected (R2)** 863kB Friday 31 May 2013 2:00:53pm PDT

b9bf06021da54cddf4cd0b3565558a1818688b972

You are an **author** of this paper.

**+ ABSTRACT**
We demonstrate the vulnerability of commodity DRAM chips to disturbance errors. By repeatedly reading from one DRAM address, we show that it is possible to corrupt the data stored [more]

**+ AUTHORS**
Y. Kim, R. Daly, J. Lee, J. Kim, C. Fallin, C. Wilkerson, O. Mutlu
[details]

**KEYWORDS**: DRAM; errors

**+ TOPICS**

<table>
<thead>
<tr>
<th>Review #66A</th>
<th>Nov</th>
<th>WriQua</th>
<th>RevExp</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Review #66B</td>
<td>5</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>Review #66C</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Review #66D</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Review #66E</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Review #66F</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>
Reviewer A

**Review #66A** Modified Friday 5 Jul 2013 3:59:18am PDT

**OVERALL MERIT (?)**

1. Reject

**PAPER SUMMARY**

This work tests and studies the disturbance problem in DRAM arrays in isolation.

**PAPER STRENGTHS**

+ Many results and observations.
+ Insights on how the may happen

**PAPER WEAKNESSES**

- Whereas they show disturbance may happen in DRAM array, authors don't show it can be an issue in realistic DRAM usage scenario
- Lacks architectural/microarchitectural impact on the DRAM disturbance analysis

**NOVELTY (?)**


**WRITING QUALITY (?)**

4. Well-written
Reviewer A -- Security is Not “Realistic”

Comments for authors
I found the paper very well written and organized, easy to understand. The topic is interesting and relevant. However, I'm not fully convinced that the disturbance problem is going to be an issue in a realistic DRAM usage scenario (main memory with caches). In that scenario the 64ms refresh interval might be enough. Overall, the work presented, the experimentation and the results are not enough to justify/claim that disturbance may be an issue for future systems, and that microarchitectural solutions are required.

I really encourage the authors to address this issue, to run the new set of experiments; if the results are positive, the work is great and will be easily accepted in a top notch conference. Test scenario in the paper (open-read-close a row many times consecutively) that is used to create disturbances is not likely to show up in a realistic usage scenario (check also rebuttal question).
WILL IT AFFECT REAL WORKLOADS ON REAL SYSTEMS?
(A, E)

Malicious workloads and pathological access-patterns can bypass/thrash the cache and access the same DRAM row a very large number of times. While these workloads may not be common, they are just as real. Using non-temporal
Reviewer A -- Demands

To make sure that correct information and messages are given to the research community, it would be good if the conclusions drawn in the paper were verified with the actual DRAM manufacturers, although I see that it can be difficult to do. In addition, knowing the technology node of each tested DRAM would make the paper stronger and would avoid speculative guesses.

Reviewer expertise (?)

4. Expert in area, with highest confidence in review.
Reviewer C

**Review #66C**  Modified Friday 12 Jul 2013 7:38:57am PDT

**OVERALL MERIT (?)**

2. Weak reject

**PAPER SUMMARY**

This paper presents a rigorous study of DRAM module errors which are observed to be caused through repeated access to the same address in the DRAMs.

**PAPER STRENGTHS**

The paper's measurement methodology is outstanding, and the authors very thoroughly dive into different test scenarios, to isolate the circumstances under which the observed errors take place.

**PAPER WEAKNESSES**

This is an excellent test methodology paper, but there is no micro-architectural or architectural content.

**NOVELTY (?)**

3. Incremental improvement.

**WRITING QUALITY (?)**

5. Outstanding

**QUESTIONS TO ADDRESS IN THE REBUTTAL**

My primary concern with this paper is that it doesn't have (micro-)architectural content, and may not spur on future work.
Reviewer C -- Leave It to DRAM Vendors

COMMENTS FOR AUTHORS
This is an extremely well-written analysis of DRAM behavior, and the authors are to be commended on establishing a robust and flexible characterization platform and methodology.

That being said, disturb errors have occurred repeatedly over the course of DRAM's history (which the authors do acknowledge). History has shown that particular disturbances, and in particular hammer errors, are short-lived, and are quickly solved by DRAM manufacturers. Historically, once these these types of errors occur at a particular lithography node/DRAM density, they must be solved by the DRAM manufacturers, because even if a solution for a systemic problem could be asserted for particular markets (e.g., server, where use of advanced coding techniques, extra chips, etc. is acceptable), there will always be significant DRAM chip volume in single-piece applications (e.g., consumer devices, etc.) where complex architectural solutions aren't an option. The authors have identified a contemporary disturb sensitivity in DRAMs, but as non-technologists, our community can generally only observe, not correct, such problems.

REVIEWER EXPERTISE (?)
4. Expert in area, with highest confidence in review.
Reviewer D -- Nothing New in RowHammer

Review #66D  Modified Thursday 18 Jul 2013 12:51pm

OVERALL MERIT (?)

1. Reject

REVIEWER EXPERTISE (?)

4. Expert in area, with highest confidence in review.

PAPER SUMMARY
The authors demonstrate that repeated activate-precharge operations on one wordline of a DRAM can disturb a few cells on adjacent wordlines. They showed that such behavior can be caused for most DRAMs and all DRAMs of recent manufacture they tested.

PAPER STRENGTHS
DRAM errors are getting more likely with newer generations and it is necessary to investigate their cause and mitigation in computer systems, as such the paper addresses a subtopic of a relevant problem.

PAPER WEAKNESSES
The mechanism investigated by the authors is one of many well known disturb mechanisms. The paper does not discuss the root causes to sufficient depth and the importance of this mechanism compared to others. Overall the length of the sections restating known information is much too long in relation to new work.

NOVELTY (?)

2. Insignificant novelty. Virtually all of the ideas are published or known.

WRITING QUALITY (?)

3. Adequate
#41 Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

You are an author of this paper.

**Abstract**

Memory isolation is a key property of a reliable and secure computing system --- an access to one memory address should not have unintended side effects on data stored in other.

**Authors**

Y. Kim, R. Daly, J. Kim, J. Lee, C. Fallin, C. Wilkerson, O. Mutlu

**Topics**

OveMer, Nov, WriQua, RevConAnd

<table>
<thead>
<tr>
<th>Review</th>
<th>OveMer</th>
<th>Nov</th>
<th>WriQua</th>
<th>RevConAnd</th>
</tr>
</thead>
<tbody>
<tr>
<td>#41A</td>
<td>8</td>
<td>4</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>#41B</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>#41C</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>#41D</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>#41E</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>#41F</td>
<td>7</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>
Review #41D  Modified 19 Feb 2014 8:47:24pm

**OVERALL MERIT (?)**

2. Reject

**PAPER SUMMARY**
The authors
1) characterize disturbance error in commodity DRAM
2) identify the root cause such errors (but it's already a well know problem in DRAM community).
3) propose a simple architectural technique to mitigate such errors.

**PAPER STRENGTHS**
The authors demonstrated the problem using the real systems

**PAPER WEAKNESSES**
1) The disturbance error (a.k.a coupling or cross-talk noise induced error) is a known problem to the DRAM circuit community.
2) What you demonstrated in this paper is so called DRAM row hammering issue - you can even find a Youtube video showing this! - [http://www.youtube.com/watch?v=i3-qOSnBcdo](http://www.youtube.com/watch?v=i3-qOSnBcdo)
2) The architectural contribution of this study is too insignificant.
Novelty (1)
2. Insignificant novelty. Virtually all of the ideas are published or known.

Writing Quality (1)
5. Outstanding

Reviewer Confidence and Expertise (1)
4. Expert in area, with highest confidence in review.

Questions for Authors
1. There are other sources of disturbance errors. How can you guarantee the errors observed by you are not from such errors?

2. You did your best on explaining why we have much fewer 1->0 error but not quite satisfied. Any other explanation?

3. Can you elaborate why we have more disturbed cells over rounds while you claim that disturbed cells are not weak cells? I'm sure this is related to device again issues.

Detailed Comments
This is a well-written and executed paper (in particular using real systems), but I have many concerns:

1) this is a well-known problem to the DRAM community (so no novelty there); in DRAM community people use

Safari
Reviewer D Continued...

2) what you did to incur disturbance is is so called "row hammering" issues - please see http://www.youtube.com/watch?v=i3-qQSnBcdo - a demonstration video for capturing this problem...

3) the relevance of this paper to ISCA. I feel that this paper (most part) is more appropriate to conferences like International Test Conference (ITC) or VLSI Test Symposium or Dependable Systems and Networks (DSN) at most. This is because the authors mainly dedicated the effort to the DRAM circuit characterization and test method in my view while the architectural contribution is very weak - I'm not even sure this can be published to these venues since it's a well known problem! I also assume techniques proposed to minimize disturbance error in STT-RAM and other technology can be employed here as well.
Rebuttal to Reviewer D

1. As we acknowledge in the paper, it is true that different types of DRAM coupling phenomena have been known to the DRAM circuits/testing community. However, there is a clear distinction between circuits/testing techniques confined to the *foundry* versus characterization/solution of a problem out in the *field*. The three citations (from 10+ years ago) do *not* demonstrate that disturbance errors exist in DIMMs sold then or now. They do *not* provide any real data (only simulated ones), let alone a large-scale characterization across many DIMMs from multiple manufacturers. They do *not* construct an attack on real systems, and they do *not* provide any solutions. Finally, our paper *already* references all three citations, or their more relevant equivalents. (The second/third citations provided by the reviewer are on bitline-coupling, whereas we cite works from the same authors on wordline-coupling [2, 3, 37].)

2. We were aware of the video from Teledyne (a test equipment company) and have *already* referenced slides from the same company [36]. In terms of their content regarding "row hammer", the video and the slides are identical: all they mention is that "aggressive row activations can corrupt adjacent rows". (They then advertise how their test equipment is able to capture a timestamped DRAM access trace, which can then be post-processed to identify when the number of activations exceeds a user-set threshold.) Both the video and slides do *not* say that this is a real problem affecting DIMMs on the market now. They do *not* provide any quantitative data, *nor* real-system demonstration, *nor* solution.
Reviewer E

OVERALL MERIT (?)
3. Weak Reject

PAPER SUMMARY
This paper studies the row disturbance problem in DRAMs. The paper includes a thorough quantitative characterization of the problem and a qualitative discussion of the source of the problem and potential solutions.

PAPER STRENGTHS
+ The paper provides a detailed quantitative characterization of the “row hammering” problem in memories.

PAPER WEAKNESSES
- Row Hammering appears to be well-known, and solutions have already been proposed by industry to address the issue.
- The paper only provides a qualitative analysis of solutions to the problem. A more robust evaluation is really needed to know whether the proposed solution is necessary.

NOVELTY (?)
2. Insignificant novelty. Virtually all of the ideas are published or known.

WRITING QUALITY (?)
3. Adequate

REVIEWER CONFIDENCE AND EXPERTISE (?)
3. Knowledgeable in area, and significant confidence in reviewing paper.
but there are numerous mentions of hammering in the literature, and clearly industry has studied this problem for many years. In particular, Intel has a patent application on a memory controller technique that addresses this exact problem, with priority date June 2012:


The patent application details sound very similar to solution 6 in this paper, so a more thorough comparison with solution 7 seems mandatory.

My overall feeling is that while the reliability characterization is important and interesting, a better target audience for the characterization work would be in a testing/reliability venue. The most interesting part of this paper from the ISCA point of view are the proposed solutions, but all of these are discussed in a very qualitative manner. My preference would be to see a much shorter characterization section with a much stronger and quantitative evaluation and comparison of the proposed solutions.
Rebuttal to Reviewer E

*Nevertheless*, we were able to induce a large number of DRAM disturbance errors on all the latest Intel/AMD platforms that we tested: Haswell, Ivy Bridge, Sandy Bridge, and Piledriver. (At the time of submission, we had tested only Sandy Bridge.) Importantly, the patents do *not* provide quantitative characterization *nor* real-system demonstration.

[R1] "Row Hammer Refresh Command." US20140006703 A1
[R2] "Row Hammer Condition Monitoring." US20140006704 A1

After our paper was submitted, two patents that had been filed by Intel were made public (one is mentioned by the reviewer [R1]). Together, the two patents describe what we posed as the *sixth* potential solution in our paper (Section 8). Essentially, the memory controller maintains a table of counters to track the number of activations to recently activated rows [R2]. And if one of the counters exceeds a certain threshold, the memory controller notifies the DRAM chips using a special command [R1]. The DRAM chips would then refresh an entire "region" of rows that includes both the aggressor and its victim(s) [R1]. For the patent [R1] to work, DRAM manufacturers must cooperate and implement this special command. (It is a convenient way of circumventing the opacity in the logical-physical mapping. If implemented, the same command can also be used for our *seventh* solution.) The limitation of this *sixth* solution is the storage overhead of the counters and the extra power required to associatively search through them on every activation (Section 8). That is why we believe our *seventh* solution to be more attractive. We will cite the patents and include a more concrete comparison between the two solutions.
Top Pick Reviews

**Review #54D**  Modified 1 Jan 2015 4:13:18pm PST

**SHORT PAPER SUMMARY**
This paper observes through experimental measurements that DRAM cells in a row can flip if a neighboring row is repeatedly open and closed. One of the solutions proposed is: every time a row is open and closed, refresh a neighboring row with a certain probability.

**CHANCE OF IMPACT (?)**

3. Minor impact

**OVERALL MERIT (?)**

2. Weak reject (Happy to discuss but unlikely to be chosen.)

**COMMENTS FOR AUTHOR**
Interesting paper for those interested in DRAM issues. I wonder if it is possible to gain an insight into why this happens.

I seem to remember that, during the presentation at ISCA, it was pointed out that DRAM manufacturers have already fixed the problem. So where is the novelty and long term impact?
Another Top Pick Review

CHANCE OF IMPACT (?) OVERALL MERIT (?)
3. Minor impact 3. Weak accept (Would consider for an honorable mention.)

COMMENTS FOR AUTHOR
This is a cute paper that explores DRAM errors in rows caused by accessing nearby rows. The results are certainly a bit surprising.

I can see this being a problem for PCs, where there is no ECC. Even if PCs, there are certain pieces that are...
SHRINT PAPER SUMMARY

This paper makes the observation that when a DRAM row is opened (activated) and closed (precharged) repeatedly, it introduces disturbance errors in adjacent DRAM rows. The paper tests 129 DRAM modules from three manufacturers providing a wealth of information: 110 of the tested DRAM modules exhibit disturbance errors, and the trend seems to be increasing over the years. The paper then introduces a mechanism to prevent DRAM disturbance errors using a probabilistic approach. The paper also includes an FPGA-based testbed to analyze DRAM chips.

CHANCE OF IMPACT (?)
3. Minor impact

OVERALL MERIT (?)
4. Accept (Would argue for at least honorable mention.)

COMMENTS FOR AUTHOR

This is a great piece of work. It makes the point that disturbance errors occur in real DRAM chips, and that the problem is consistent across DRAM manufacturers and is getting worse over time. The paper characterizes a large number of real DRAM chips, clearly demonstrating the problem. The paper provides an FPGA-based testbed, and proposes a probabilistic mechanism to prevent DRAM disturbance errors. This is a very well executed piece of work overall.

While this is the first piece of work in the scientific literature to describe and characterize the problem, the problem of DRAM disturbance errors seems to be well-known to industry (as acknowledged in the paper). This somewhat reduces the significance of the work for consideration as a Top Pick.
Suggestions to Reviewers

- Be fair; you do not know it all
- Be open-minded; you do not know it all
- Be accepting of diverse research methods: there is no single way of doing research
- Be constructive, not destructive
- Do not have double standards...

Do not block or delay scientific progress for non-reasons
An Interview on Research and Education

- Computing Research and Education (@ ISCA 2019)
  - https://www.youtube.com/watch?v=8ffSEKZhmvo&list=PL5Q2soXY2Zi_4oP9LdL3cc8G6NIjD2Ydz

- Maurice Wilkes Award Speech (10 minutes)
  - https://www.youtube.com/watch?v=tcQ3zZ3JpuA&list=PL5Q2soXY2Zi8D_5MGV6EnXEJHnV2YFBJl&index=15
More Thoughts and Suggestions

- Onur Mutlu,
  "Some Reflections (on DRAM)"
  Award Speech for ACM SIGARCH Maurice Wilkes Award, at the ISCA Awards Ceremony, Phoenix, AZ, USA, 25 June 2019.
  [Slides (pptx) (pdf)]
  [Video of Award Acceptance Speech (Youtube; 10 minutes) (Youku; 13 minutes)]
  [Video of Interview after Award Acceptance (Youtube; 1 hour 6 minutes) (Youku; 1 hour 6 minutes)]
  [News Article on "ACM SIGARCH Maurice Wilkes Award goes to Prof. Onur Mutlu"]

- Onur Mutlu,
  "How to Build an Impactful Research Group"
  57th Design Automation Conference Early Career Workshop (DAC), Virtual, 19 July 2020.
  [Slides (pptx) (pdf)]
Aside: A Recommended Book

Even if the performance analysis is correctly done and presented, it may not be enough to persuade your audience—the decision makers—to follow your recommendations. The list shown in Box 10.2 is a compilation of reasons for rejection heard at various performance analysis presentations. You can use the list by presenting it immediately and pointing out that the reason for rejection is not new and that the analysis deserves more consideration. Also, the list is helpful in getting the competing proposals rejected!

There is no clear end of an analysis. Any analysis can be rejected simply on the grounds that a proposal needs more analysis. This is the first reason listed in Box 10.2. The second most common reason for rejection of an analysis and for endless debate is the workload. Since workloads are always based on past measurements, their applicability to the current or future environment can always be questioned. Actually workload is one of the four areas of discussion that lead a performance presentation into an endless debate. These “rat holes” and their relative sizes in terms of time consumed are shown in Figure 10.26. Presenting this cartoon at the beginning of a presentation helps to avoid these areas.

Box 10.2 Reasons for Not Accepting the Results of an Analysis

1. This needs more analysis.
2. You need a better understanding of the workload.
3. It improves performance only for long I/O’s, packets, jobs, and files, and most of the I/O’s, packets, jobs, and files are short.
4. It improves performance only for short I/O’s, packets, jobs, and files, but who cares for the performance of short I/O’s, packets, jobs, and files; it’s the long ones that impact the system.
5. It needs too much memory/CPU/bandwidth and memory/CPU/bandwidth isn’t free.
6. It only saves us memory/CPU/bandwidth and memory/CPU/bandwidth is cheap.
7. There is no point in making the networks (similarly, CPUs/disks/…) faster; our CPUs/disks (any component other than the one being discussed) aren’t fast enough to use them.
8. It improves the performance by a factor of $x$, but it doesn’t really matter at the user level because everything else is so slow.
9. It is going to increase the complexity and cost.
10. Let us keep it simple stupid (and your idea is not stupid).
11. It is not simple. (Simplicity is in the eyes of the beholder.)
12. It requires too much state.
13. Nobody has ever done that before. (You have a new idea.)
14. It is not going to raise the price of our stock by even an eighth. (Nothing ever does, except rumors.)
15. This will violate the IEEE, ANSI, CCITT, or ISO standard.
16. It may violate some future standard.
17. The standard says nothing about this and so it must not be important.
18. Our competitors don’t do it. If it was a good idea, they would have done it.
19. Our competition does it this way and you don’t make money by copying others.
20. It will introduce randomness into the system and make debugging difficult.
21. It is too deterministic; it may lead the system into a cycle.
22. It’s not interoperable.
23. This impacts hardware.
24. That’s beyond today’s technology.
25. It is not self-stabilizing.
26. Why change—it’s working OK.
A Fun Reading: Food for Thought


A similar process of professionalization has transformed other parts of the scientific landscape. (Central Press/Getty Images)

THE WALL STREET JOURNAL.

Could Einstein get published today?

3 min read . Updated: 25 Sep 2020, 11:51 AM IST
The Wall Street Journal

Scientific journals and institutions have become more professionalized over the last century, leaving less room for individual style