Memory Systems
and Memory-Centric Computing Systems

Part 2: RowHammer

Prof. Onur Mutlu

omutlu@gmail.com

https://people.inf.ethz.ch/omutlu

7 July 2019

SAMOS Tutorial

Four Key Directions

- Fundamentally Secure/Reliable/Safe Architectures

- Fundamentally Energy-Efficient Architectures
  - Memory-centric (Data-centric) Architectures

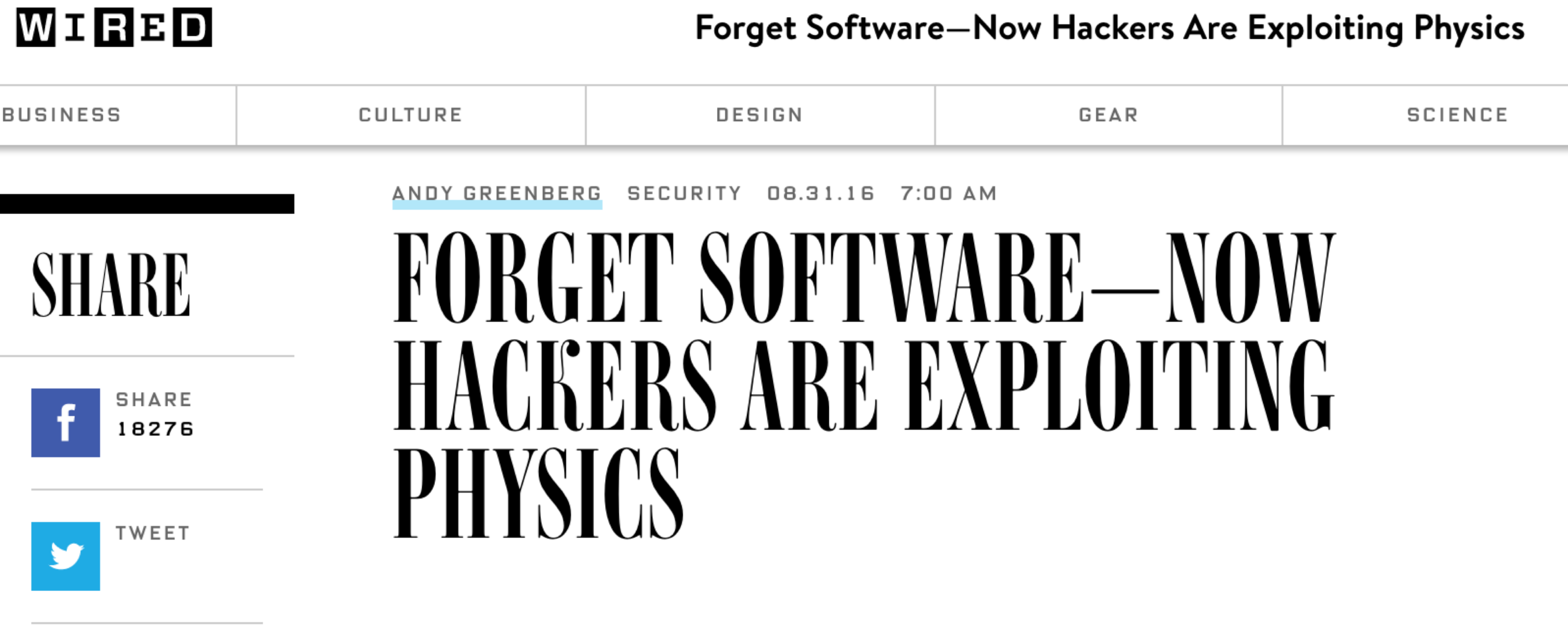
- Fundamentally Low-Latency Architectures

- Architectures for Genomics, Medicine, Health

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



Forget Software—Now Hackers Are Exploiting Physics

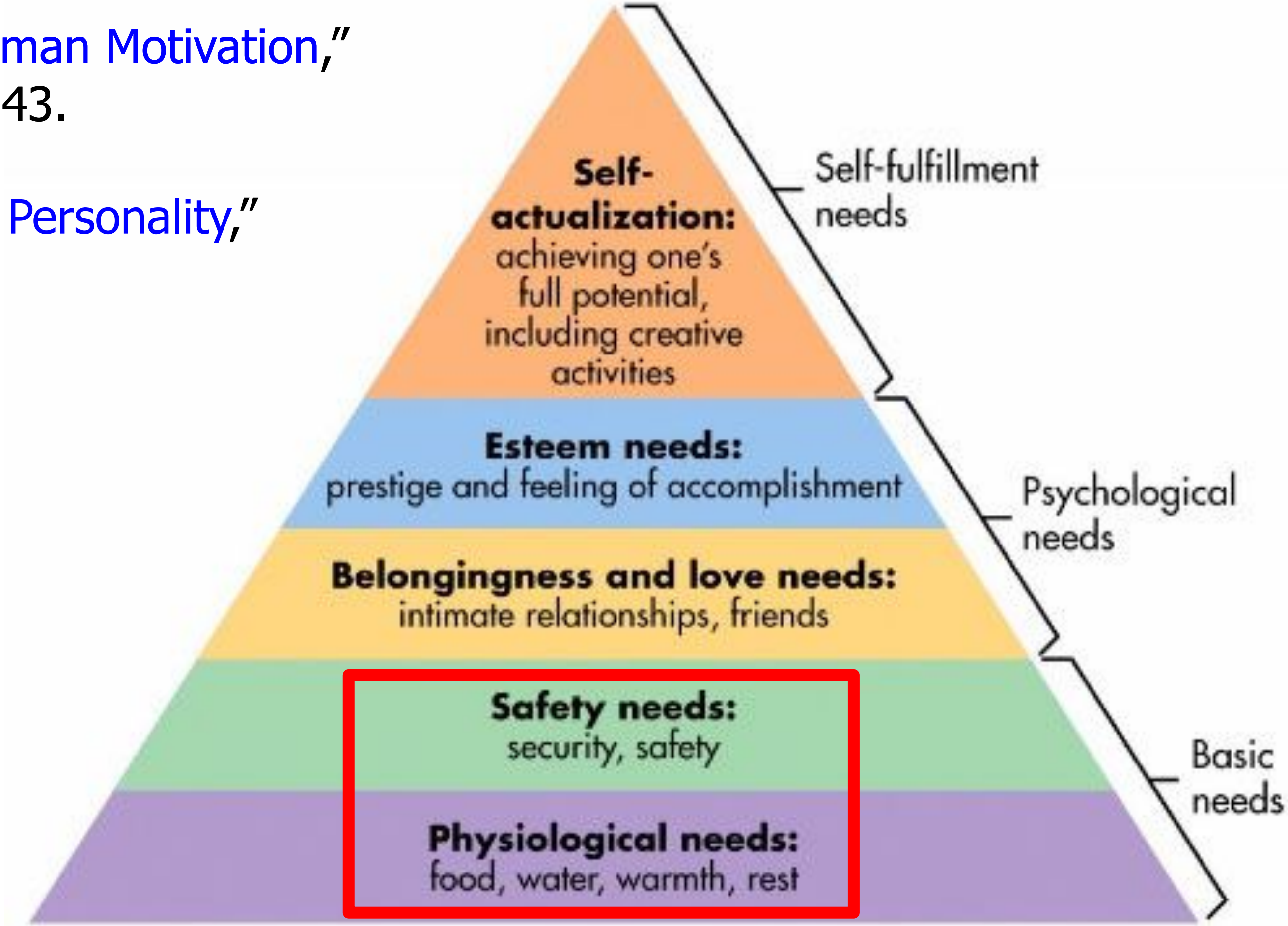
ANDY GREENBERG SECURITY 08.31.16 7:00 AM

FORGET SOFTWARE—NOW HACKERS ARE EXPLOITING PHYSICS

Maslow's (Human) Hierarchy of Needs

Maslow, "A Theory of Human Motivation," Psychological Review, 1943.

Maslow, "Motivation and Personality," Book, 1954-1970.

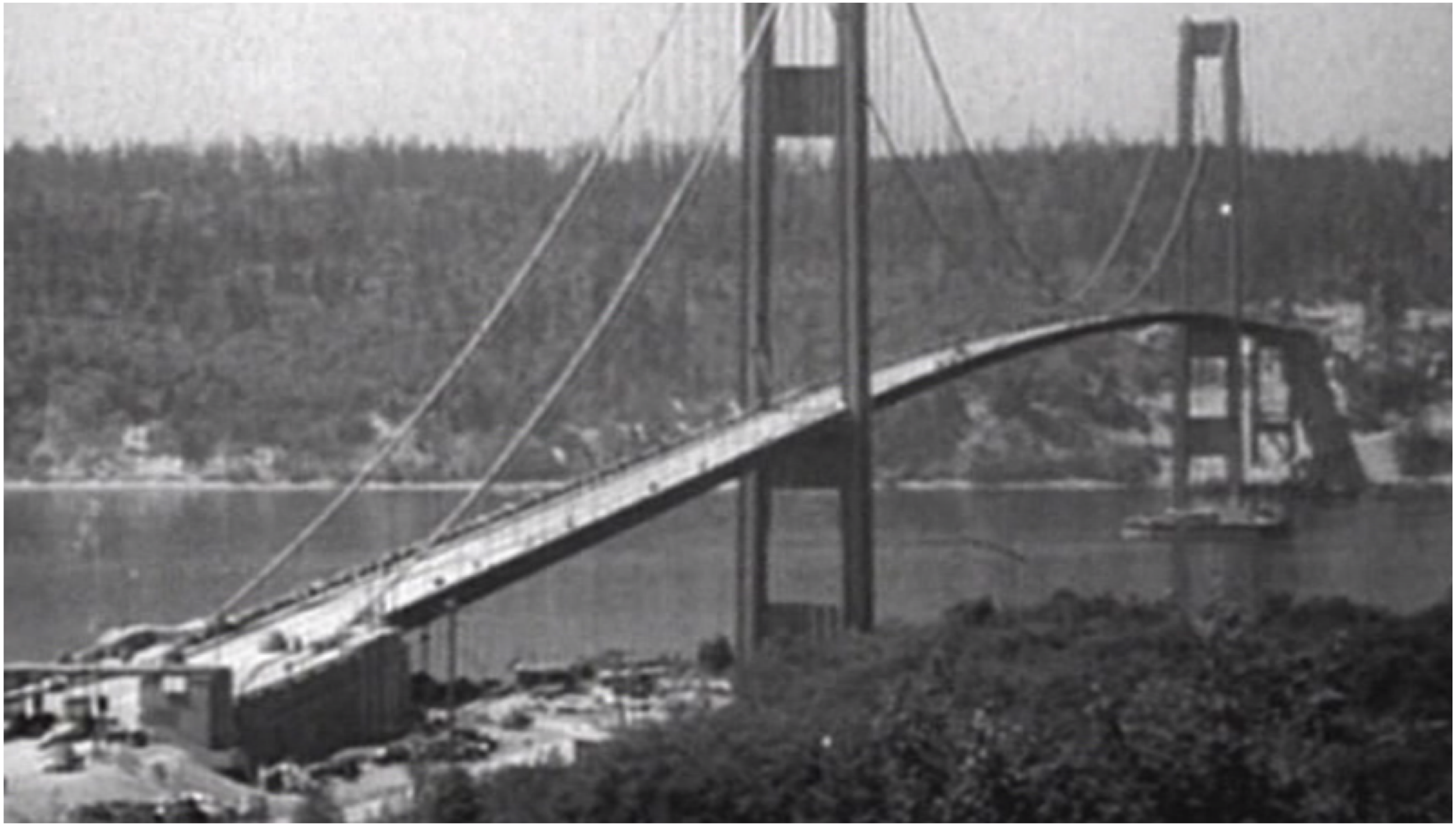


Self-actualization: achieving one's full potential, including creative activities
Esteem needs: prestige and feeling of accomplishment
Belongingness and love needs: intimate relationships, friends
Safety needs: security, safety
Physiological needs: food, water, warmth, rest
Self-fulfillment needs
Psychological needs
Basic needs

- We need to start with reliability and security…

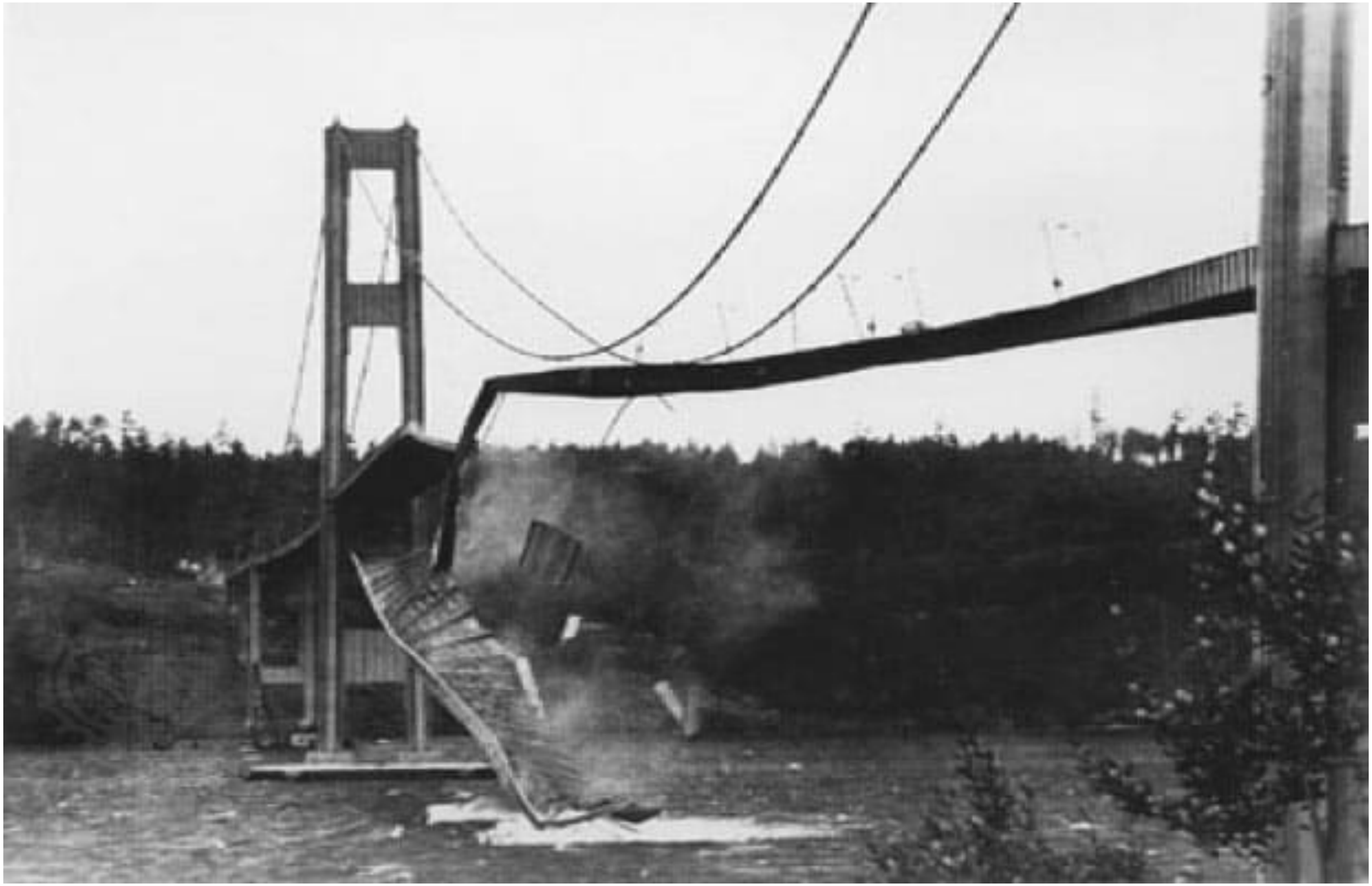
Source: https://www.simplypsychology.org/maslow.html
Source: By User:Factoryjoe - Mazlow's Hierarchy of Needs.svg, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=7964065

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

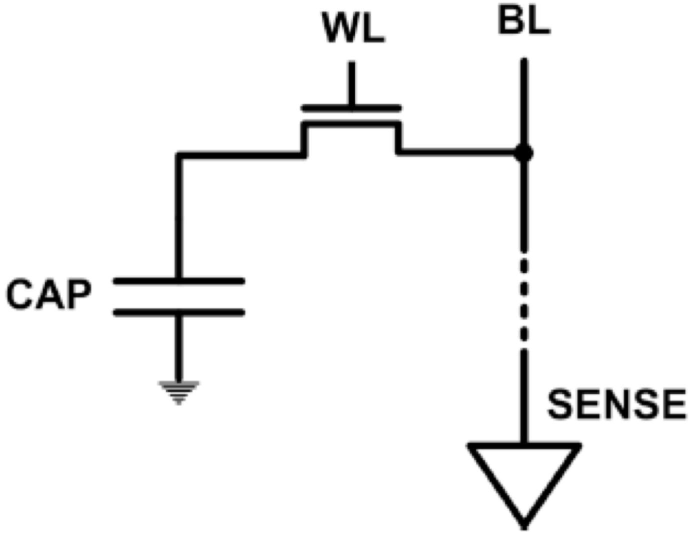
How Secure Are These People?

Security is about preventing unforeseen consequences

Source: https://s-media-cache-ak0.pinimg.com/originals/48/09/54/4809543a9c7700246a0cf8acdae27abf.jpg

The DRAM Scaling Problem

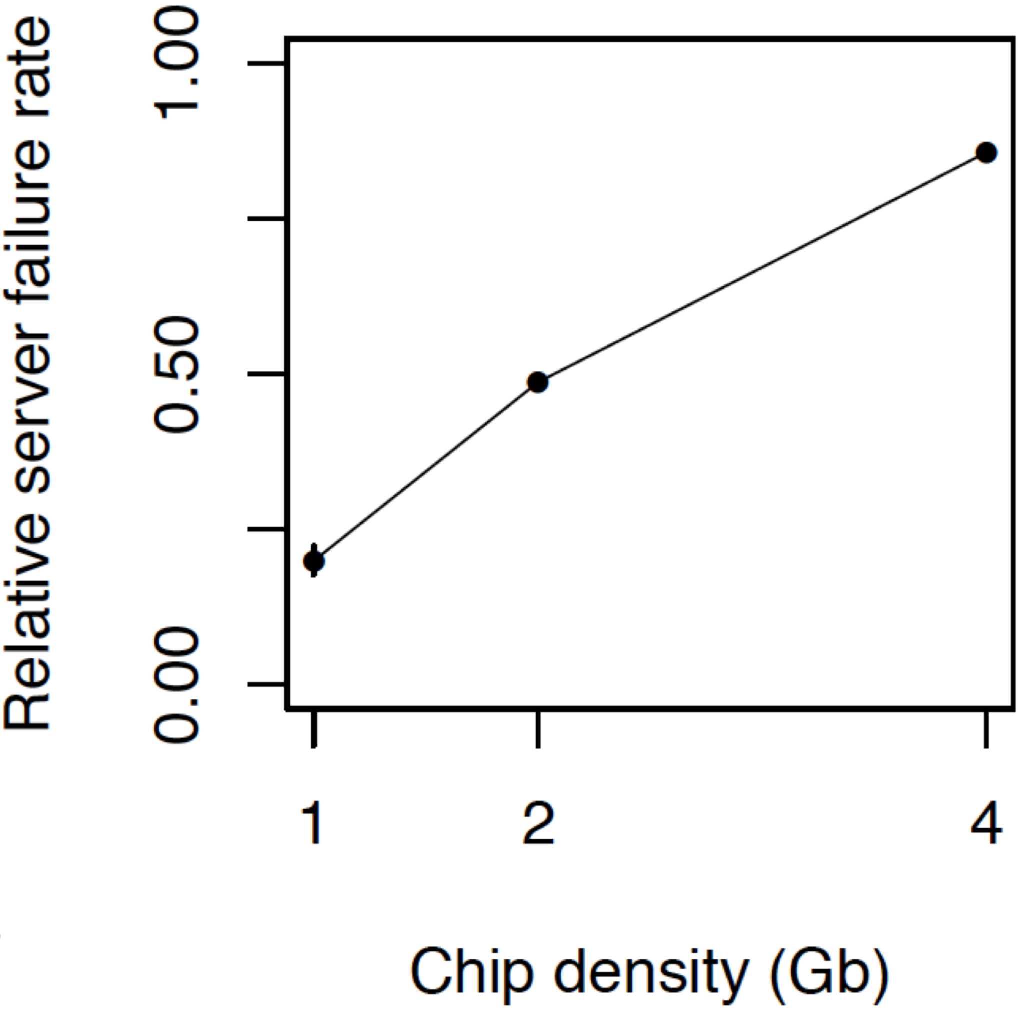
- DRAM stores charge in a capacitor (charge-based memory)
  - Capacitor must be large enough for reliable sensing
  - Access transistor should be large enough for low leakage and high retention time
  - Scaling beyond 40-35nm (2013) is challenging [ITRS, 2009]



- 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.

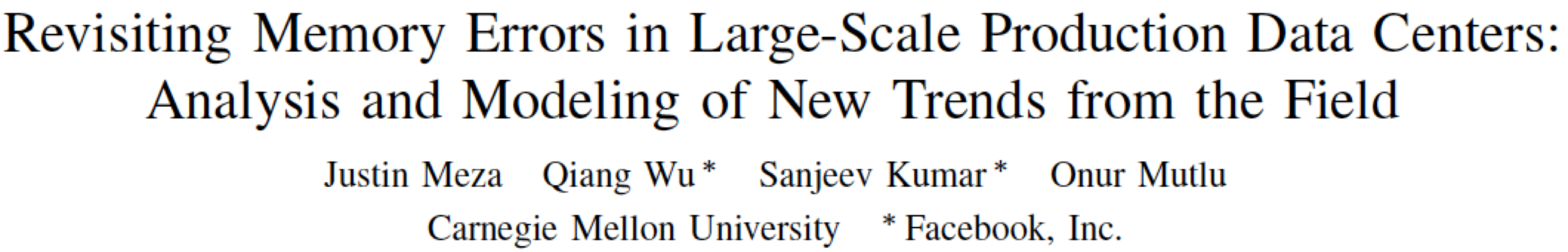


Intuition: quadratic increase in capacity

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

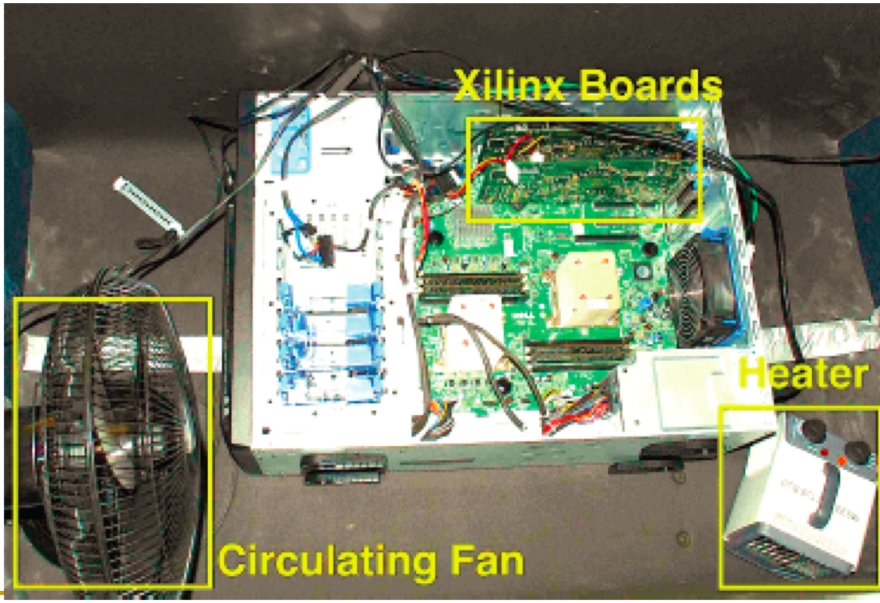
An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms (Liu et al., ISCA 2013)

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

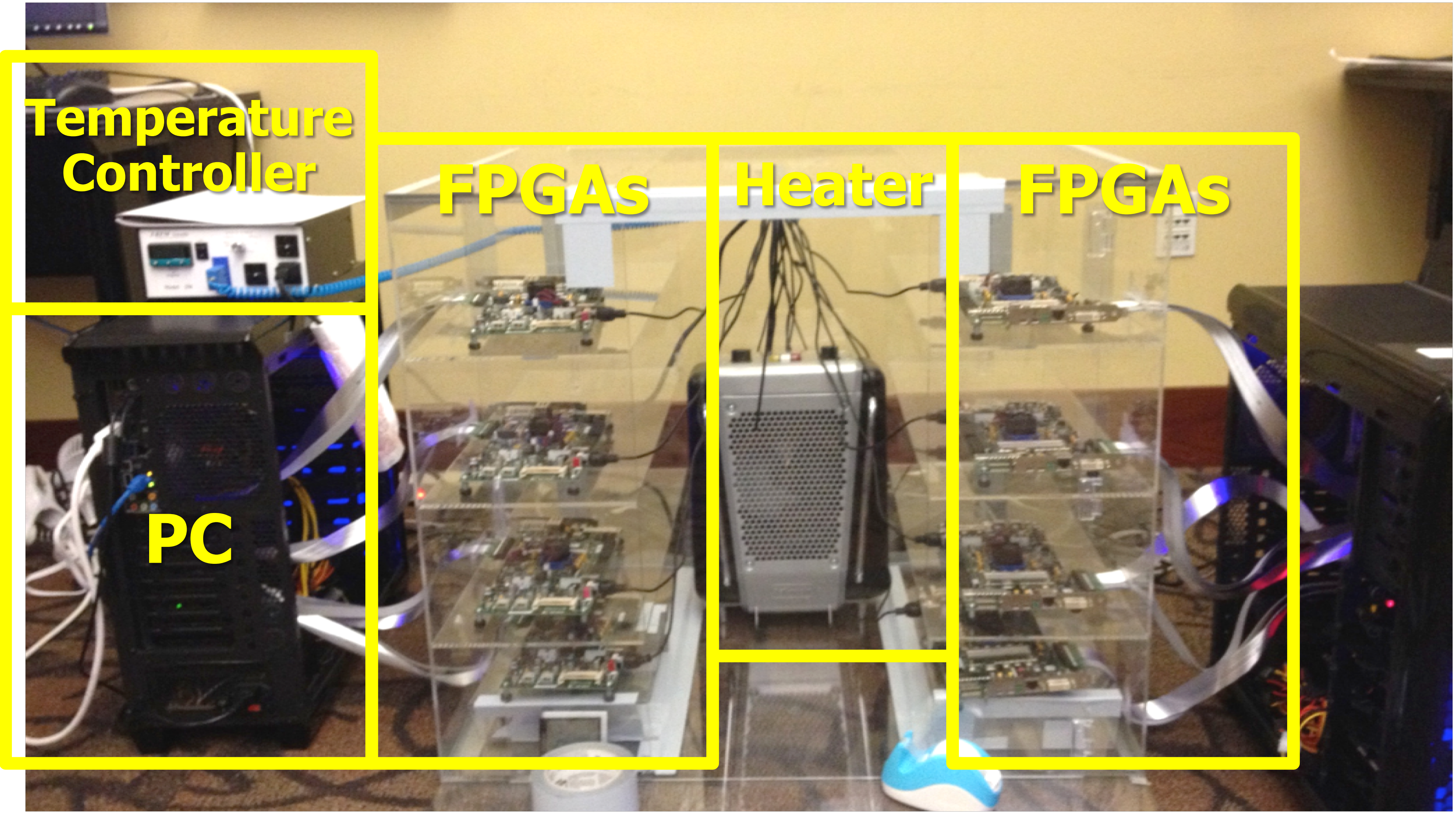
Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors (Kim et al., ISCA 2014)

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



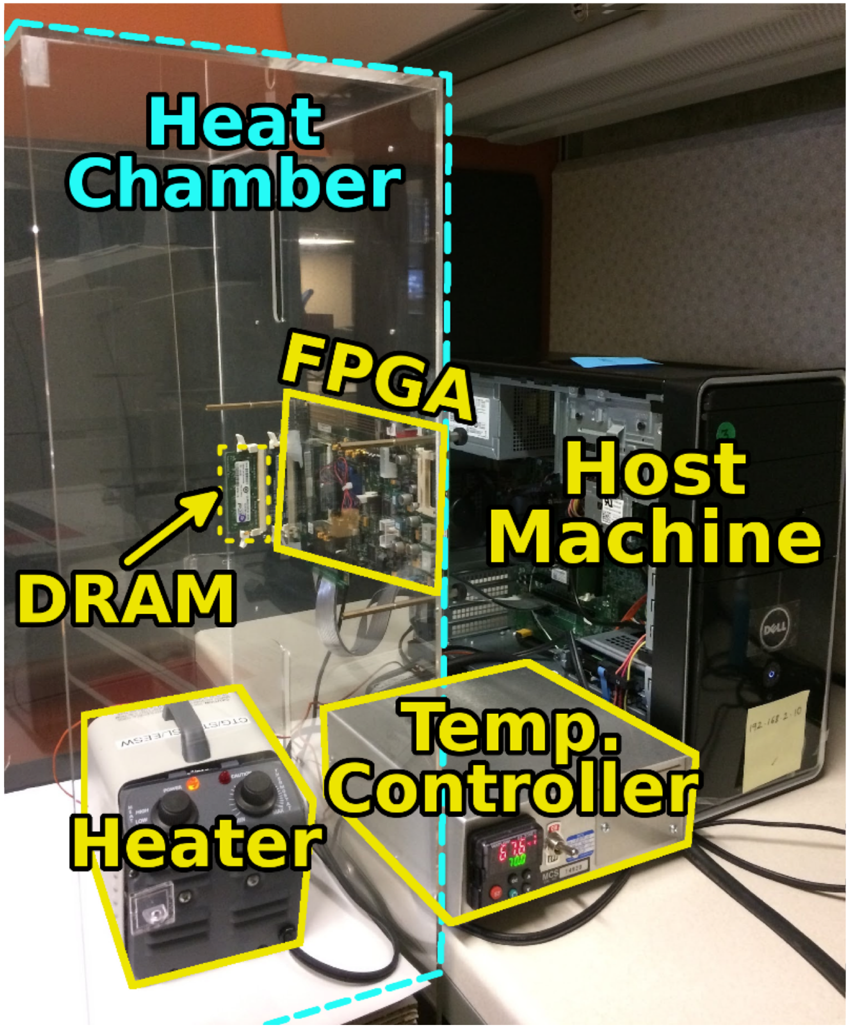
Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

SoftMC: Open Source DRAM Infrastructure

- Hasan Hassan et al., "**SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies**," HPCA 2017.

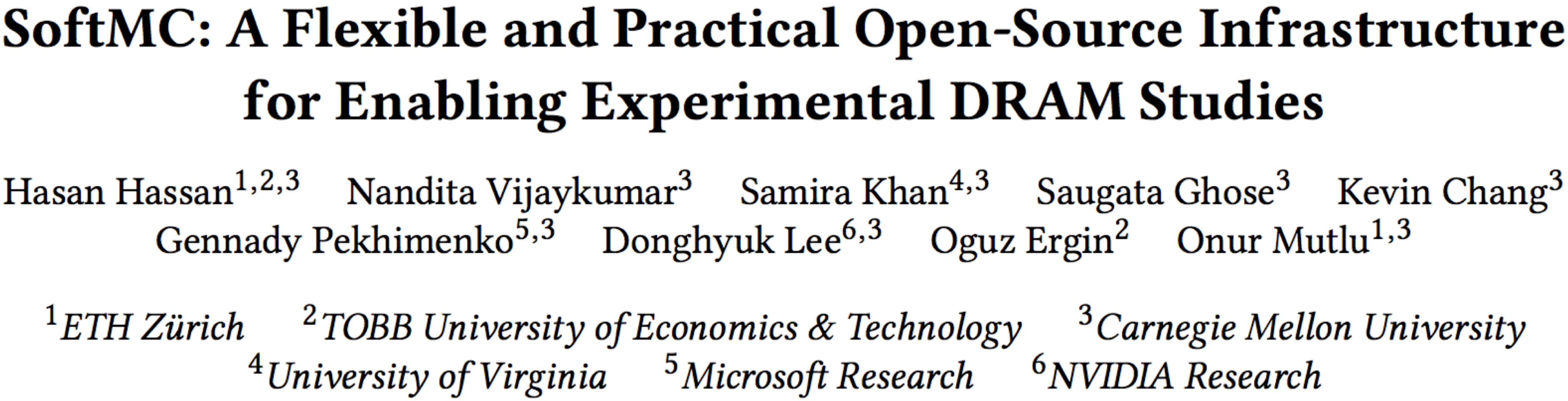
- **Flexible**
- **Easy to Use (C++ API)**
- **Open-source**

github.com/CMU-SAFARI/SoftMC



SoftMC

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



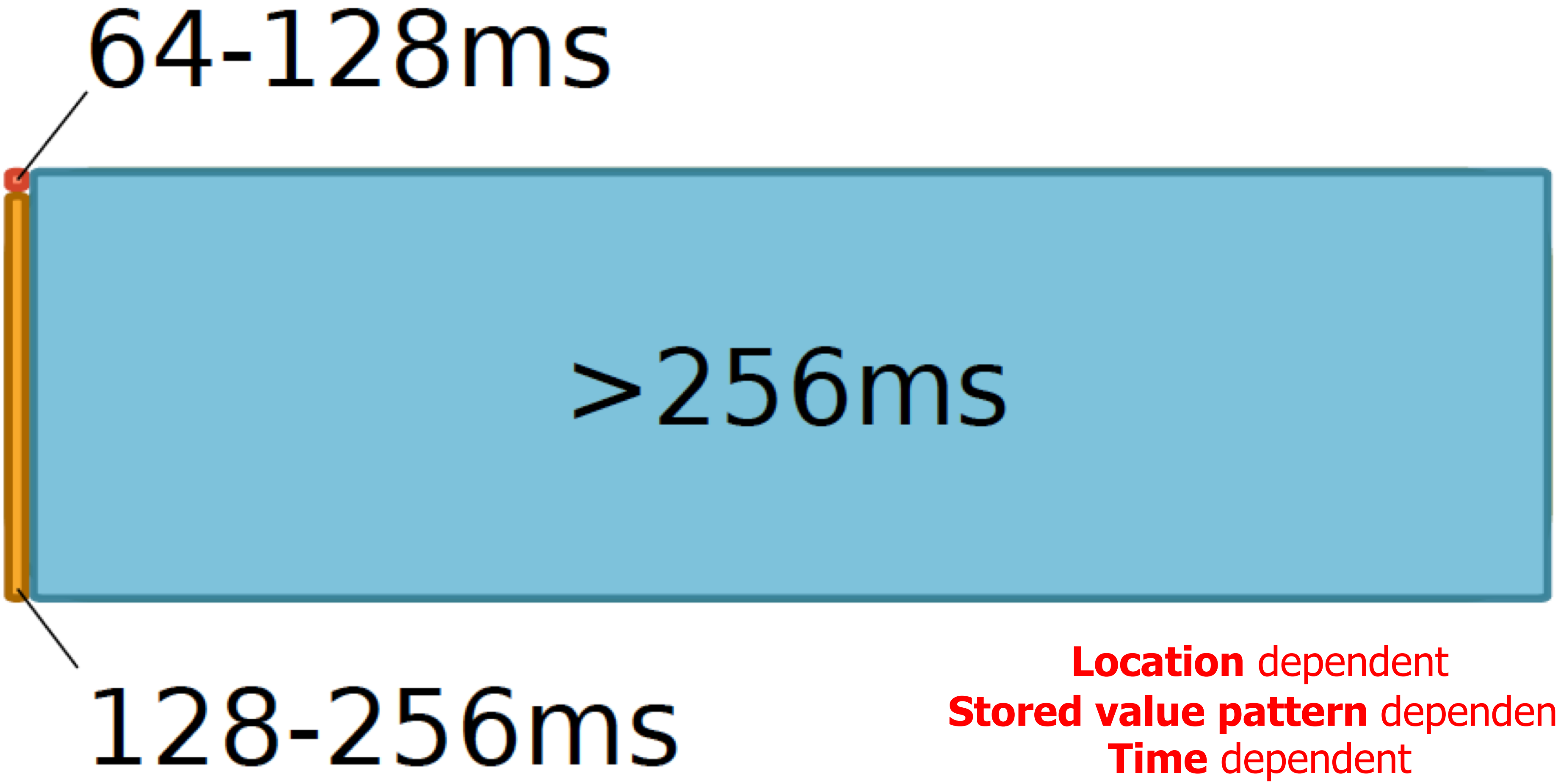
SoftMC: A Flexible and Practical Open-Source Infrastructure for Enabling Experimental DRAM Studies

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

[1]*ETH Zürich* [2]*TOBB University of Economics & Technology* [3]*Carnegie Mellon University*
[4]*University of Virginia* [5]*Microsoft Research* [6]*NVIDIA Research*

Data Retention in Memory [Liu et al., ISCA 2013]

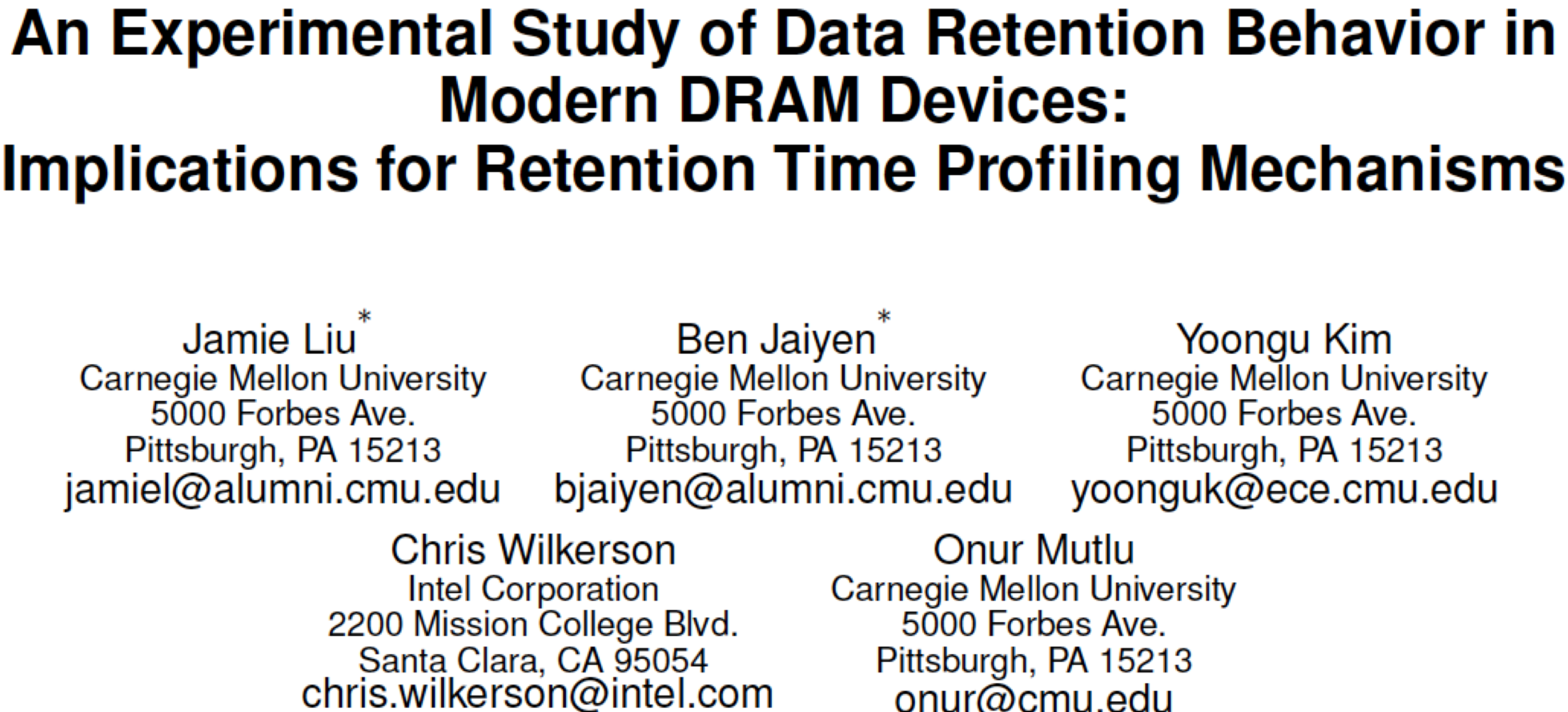
- Retention Time Profile of DRAM looks like this:



Location dependent
Stored value pattern dependent
Time dependent

Analysis of Data Retention Failures [ISCA'13]

- Jamie Liu, Ben Jaiyen, Yoongu Kim, Chris Wilkerson, and Onur Mutlu, **"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)



An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms

Jamie Liu[*]
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
jamiel@alumni.cmu.edu

Ben Jaiyen[*]
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
bjaiyen@alumni.cmu.edu

Yoongu Kim
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
yoonguk@ece.cmu.edu

Chris Wilkerson
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95054
chris.wilkerson@intel.com

Onur Mutlu
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
onur@cmu.edu

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"
  *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (**SIGMETRICS**)*, Austin, TX, June 2014. [Slides (pptx) (pdf)] [Poster (pptx) (pdf)] [Full data sets]

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

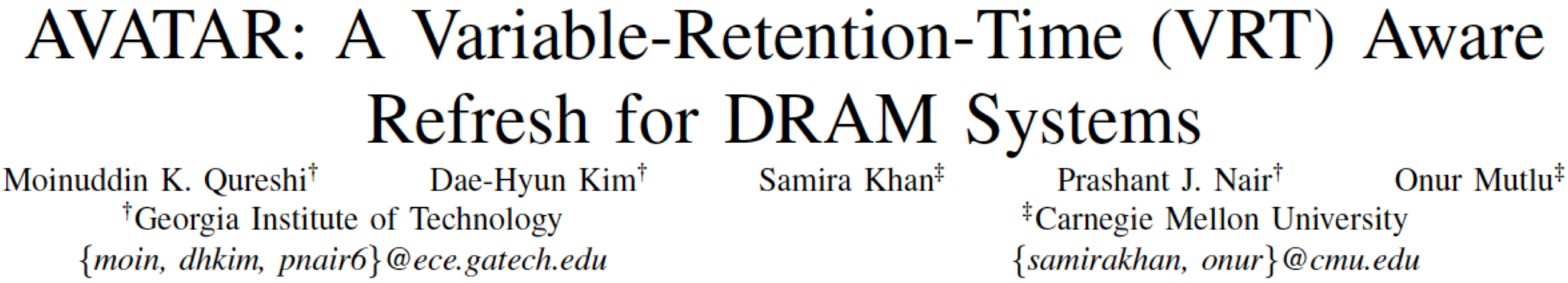
Chris Wilkerson[*]
chris.wilkerson@intel.com

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

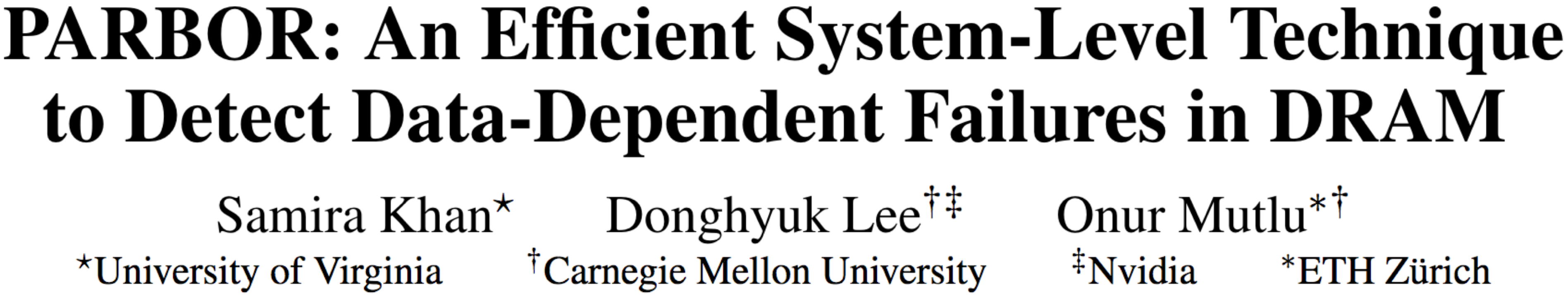
Moinuddin K. Qureshi[†] Dae-Hyun Kim[†] Samira Khan[‡] Prashant J. Nair[†] Onur Mutlu[‡]

[†]Georgia Institute of Technology
{moin, dhkim, pnair6}@ece.gatech.edu

[‡]Carnegie Mellon University
{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"** *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Toulouse, France, June 2016. [Slides (pptx) (pdf)]



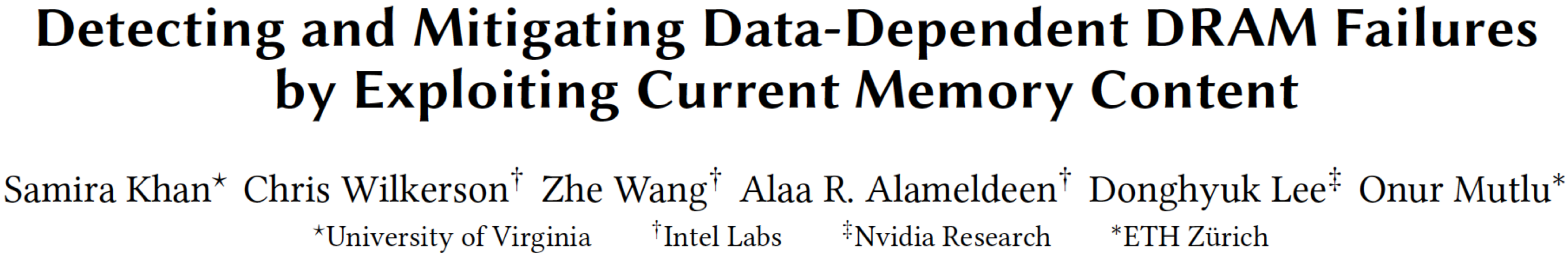
PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM

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)]



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

Samira Khan* Chris Wilkerson† Zhe Wang† Alaa R. Alameldeen† Donghyuk Lee‡ Onur Mutlu*
*University of Virginia †Intel Labs ‡Nvidia Research *ETH Zürich

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"
  *Proceedings of the 44th International Symposium on Computer Architecture (**ISCA**)*, Toronto, Canada, June 2017.
  [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

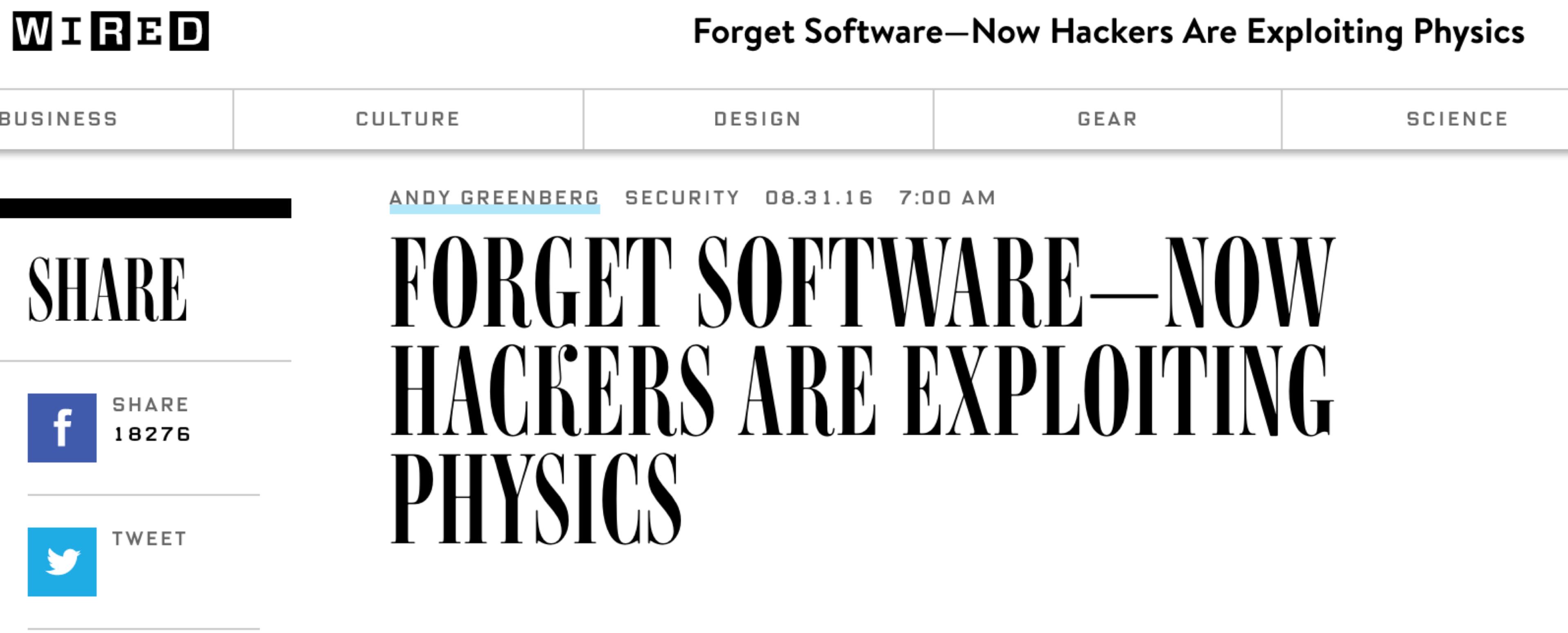
Minesh Patel[§‡] Jeremie S. Kim[‡§] Onur Mutlu[§‡]
[§]ETH Zürich [‡]Carnegie Mellon University

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

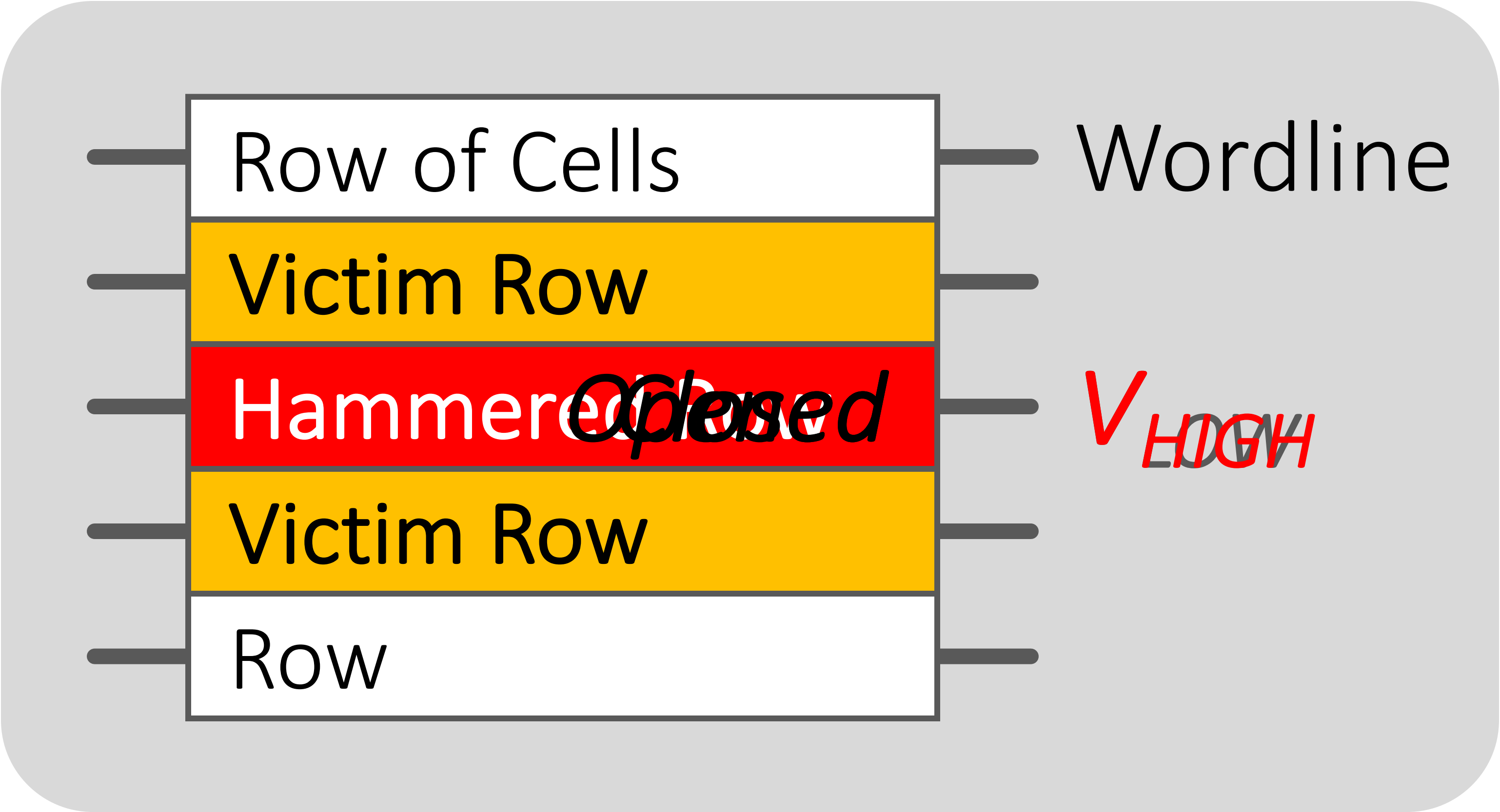


Forget Software—Now Hackers Are Exploiting Physics

ANDY GREENBERG SECURITY 08.31.16 7:00 AM

FORGET SOFTWARE—NOW HACKERS ARE EXPLOITING PHYSICS

Modern DRAM is Prone to Disturbance Errors

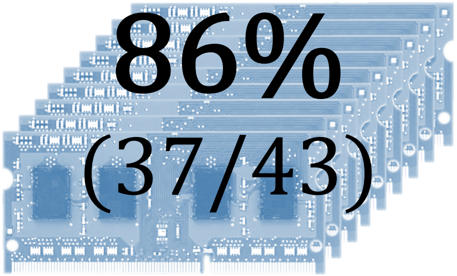


Repeatedly reading a row enough times (before memory gets refreshed) induces disturbance errors in **adjacent rows** in most real DRAM chips you can buy today

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

Most DRAM Modules Are Vulnerable

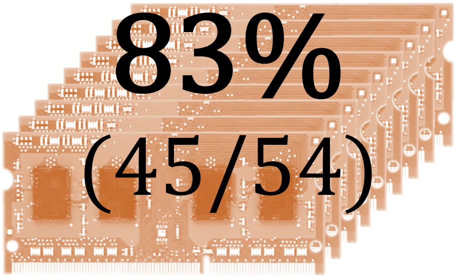
A company



86%
(37/43)

Up to
1.0×10^7
errors

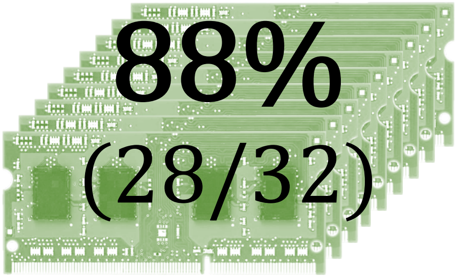
B company



83%
(45/54)

Up to
2.7×10^6
errors

C company

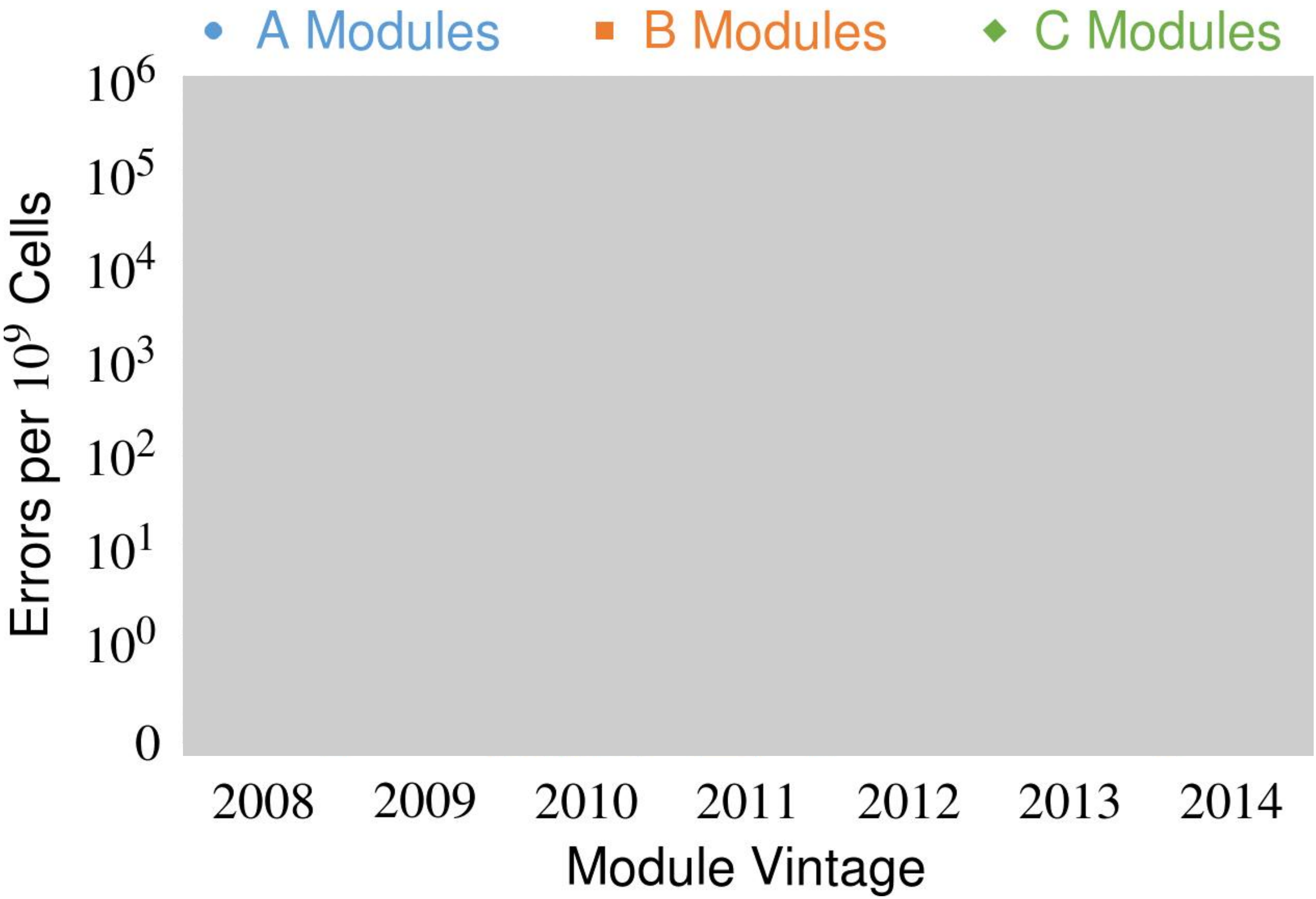


88%
(28/32)

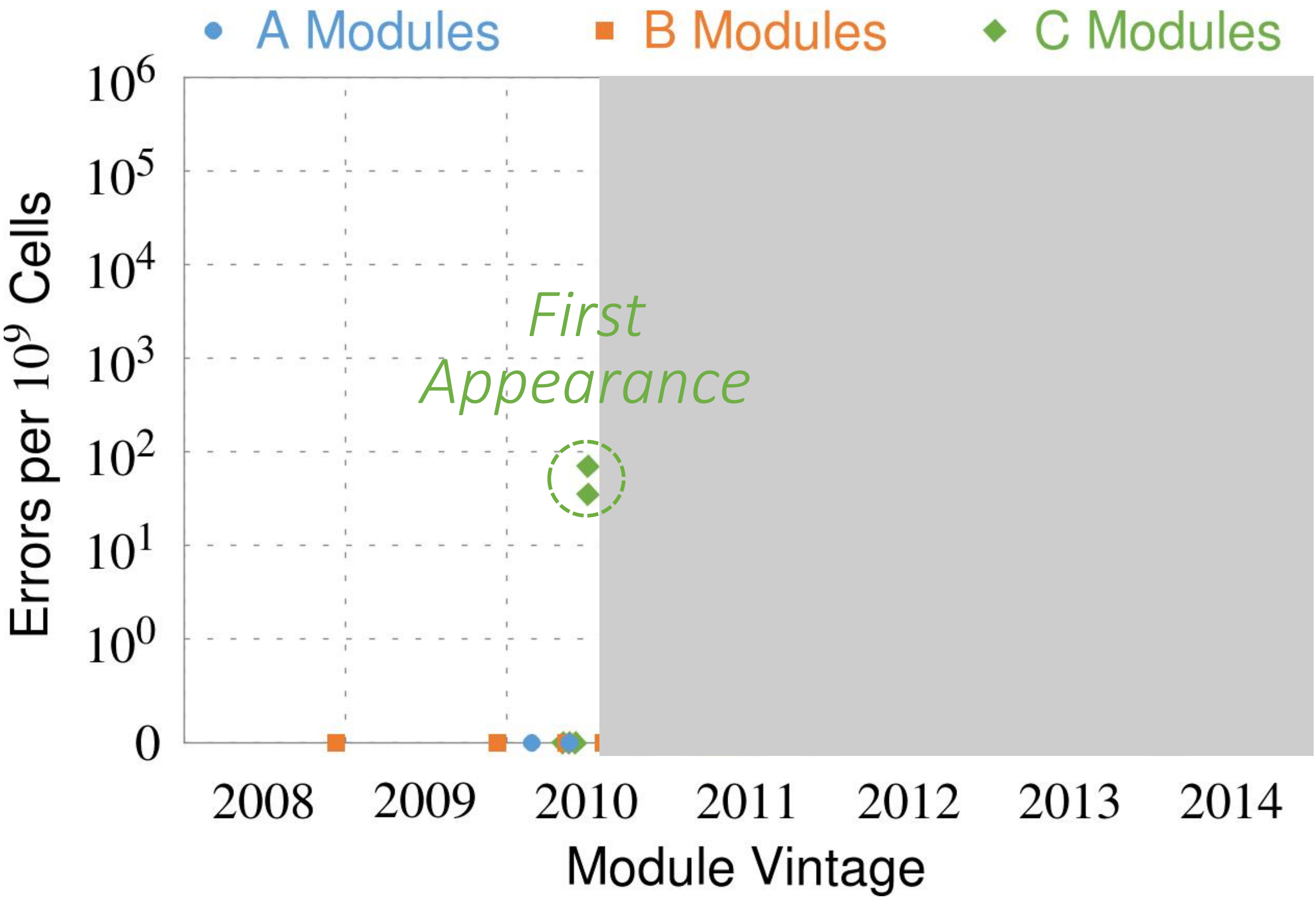
Up to
3.3×10^5
errors

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

Recent DRAM Is More Vulnerable

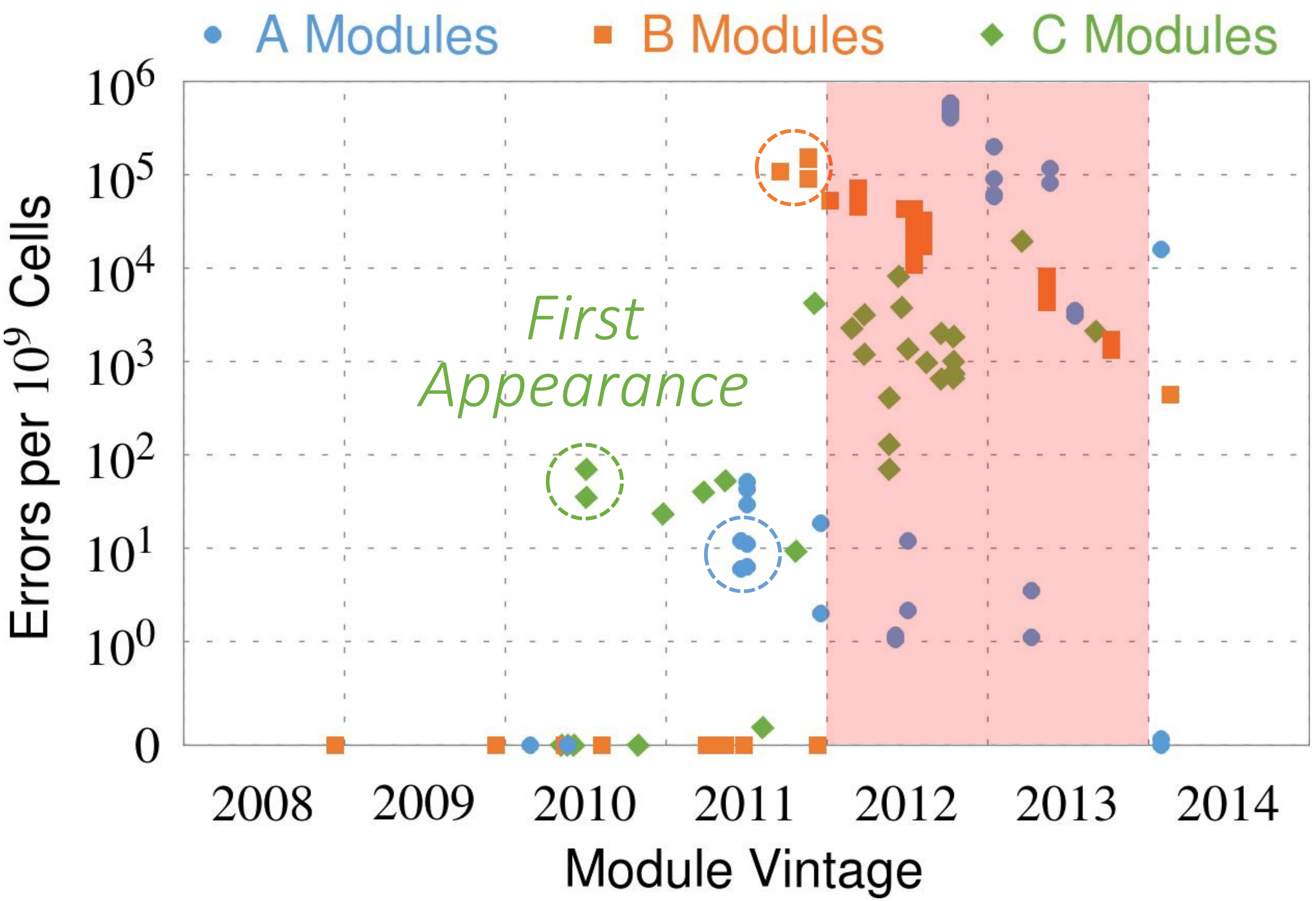


Recent DRAM Is More Vulnerable



First Appearance

Recent DRAM Is More Vulnerable



All modules from 2012–2013 are vulnerable

Why Is This Happening?

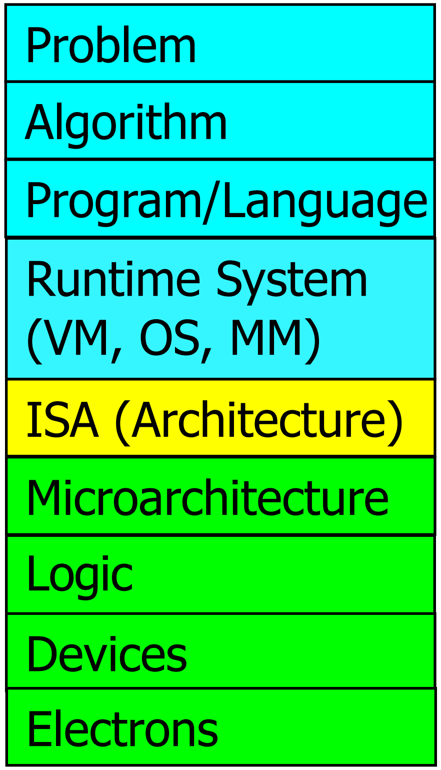
- DRAM cells are too close to each other!
  - They are not electrically isolated from each other

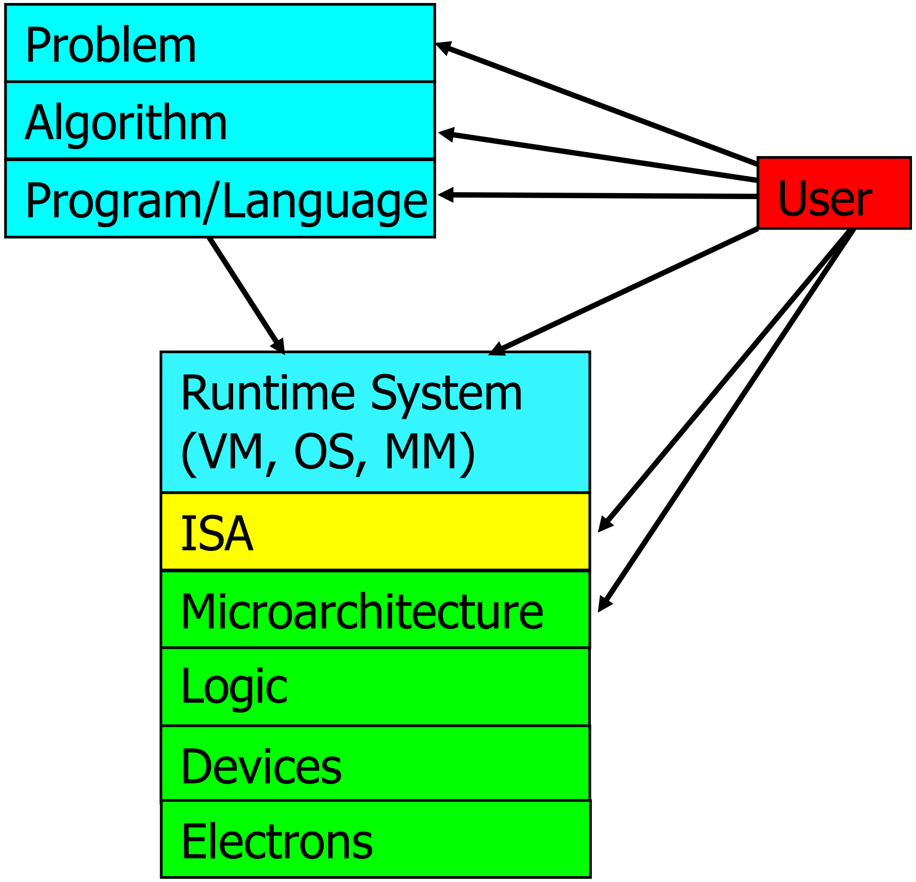
- Access to one cell affects the value in nearby cells
  - due to electrical interference between
    - the cells
    - wires used for accessing the cells
  - Also called cell-to-cell coupling/interference

- 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

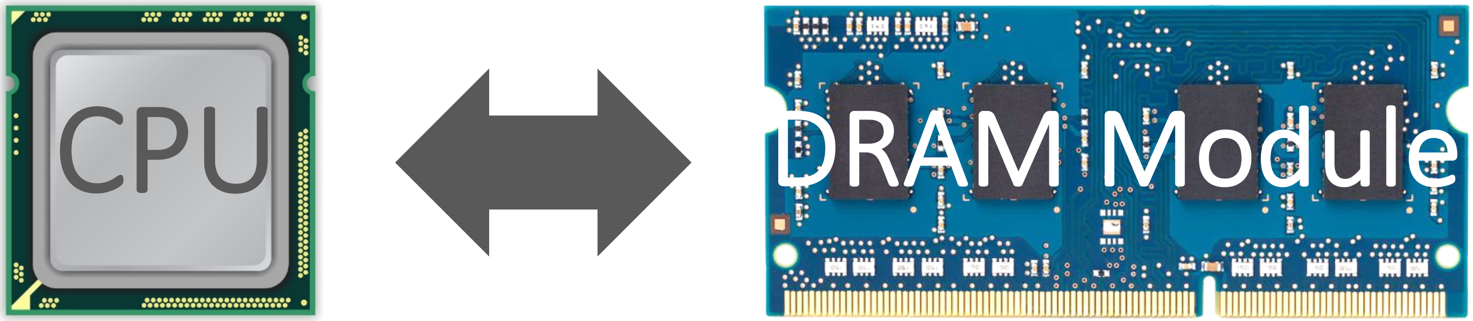
Higher-Level Implications

- This simple circuit level failure mechanism has enormous implications on upper layers of the transformation hierarchy

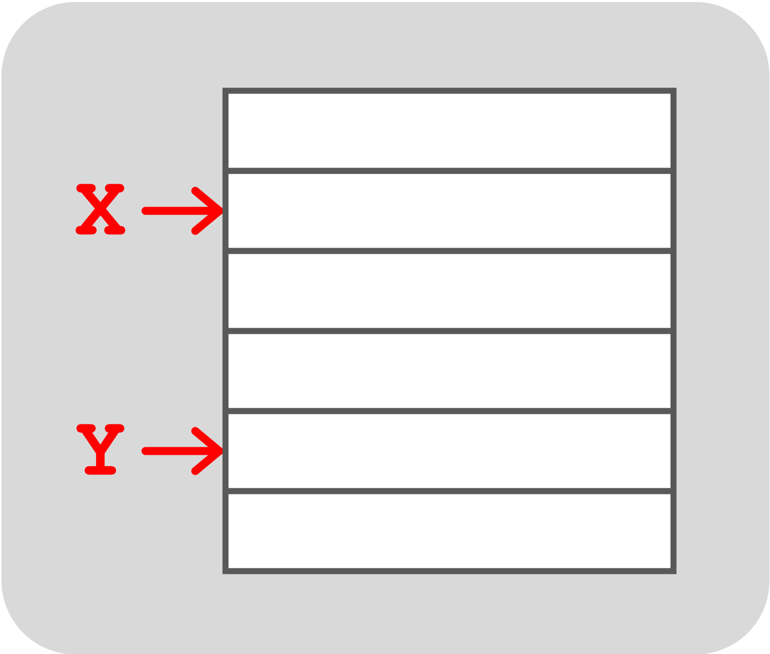




A Simple Program Can Induce Many Errors

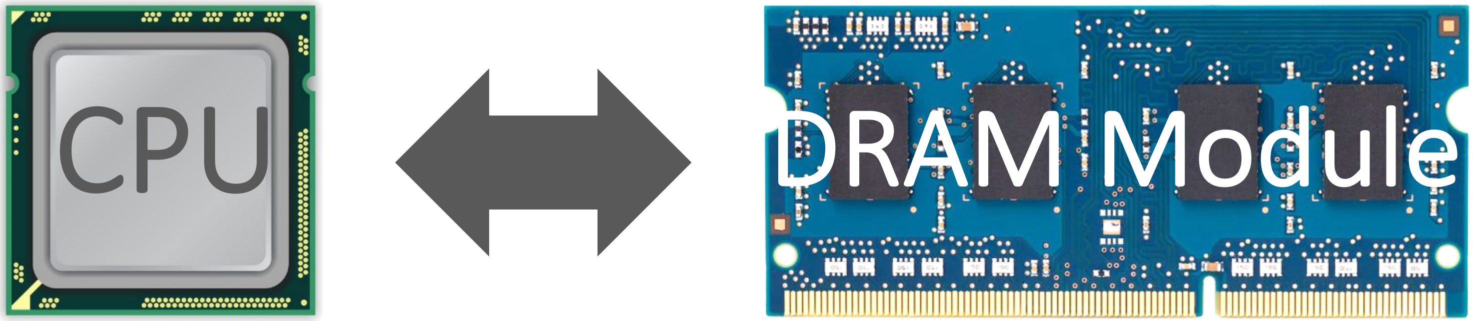


```
loop:
  mov (X), %eax
  mov (Y), %ebx
  clflush (X)
  clflush (Y)
  mfence
  jmp loop
```



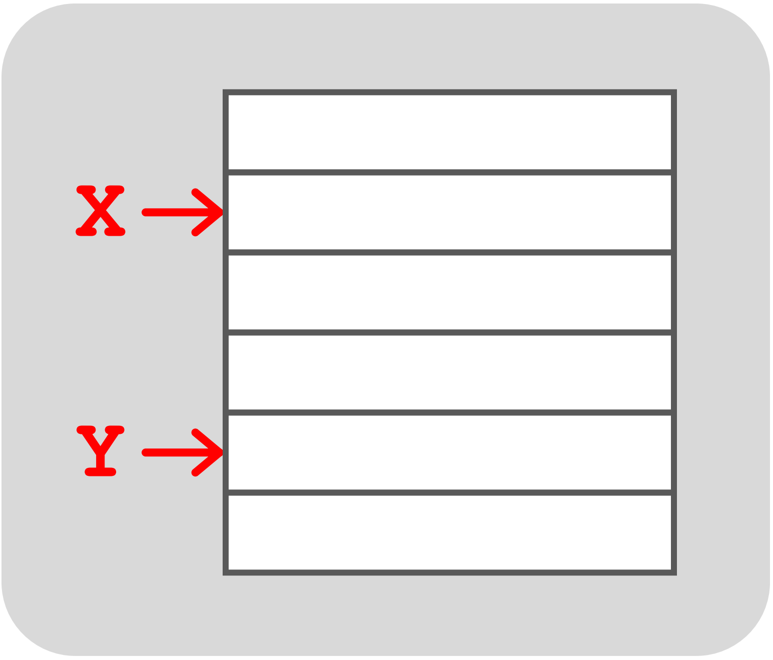
Download from: https://github.com/CMU-SAFARI/rowhammer

A Simple Program Can Induce Many Errors



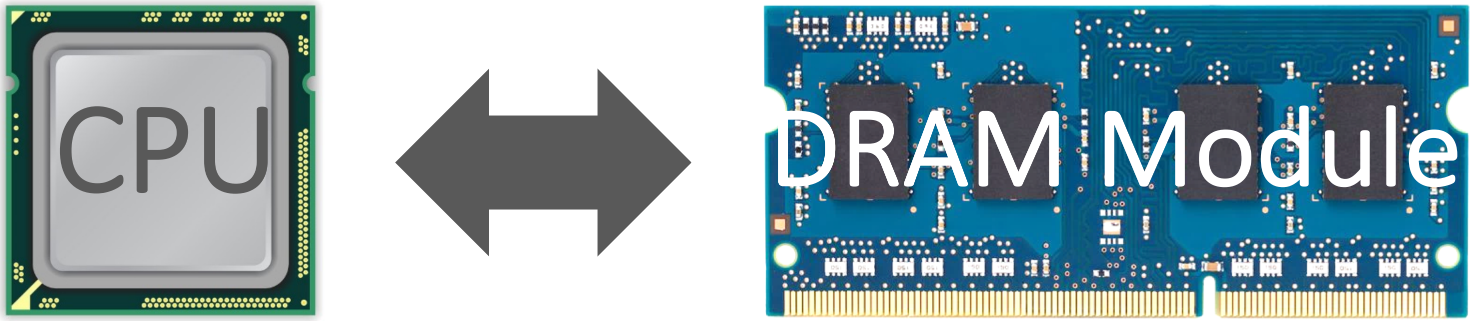
1. Avoid *cache hits*
   - Flush **X** from cache

2. Avoid *row hits* to **X**
   - Read **Y** in another row

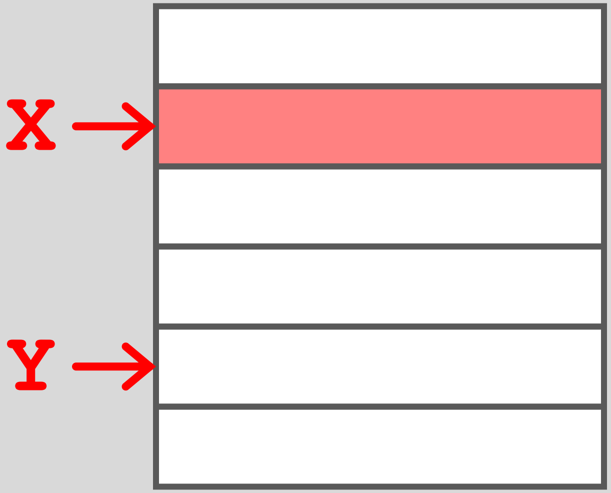


Download from: https://github.com/CMU-SAFARI/rowhammer

A Simple Program Can Induce Many Errors

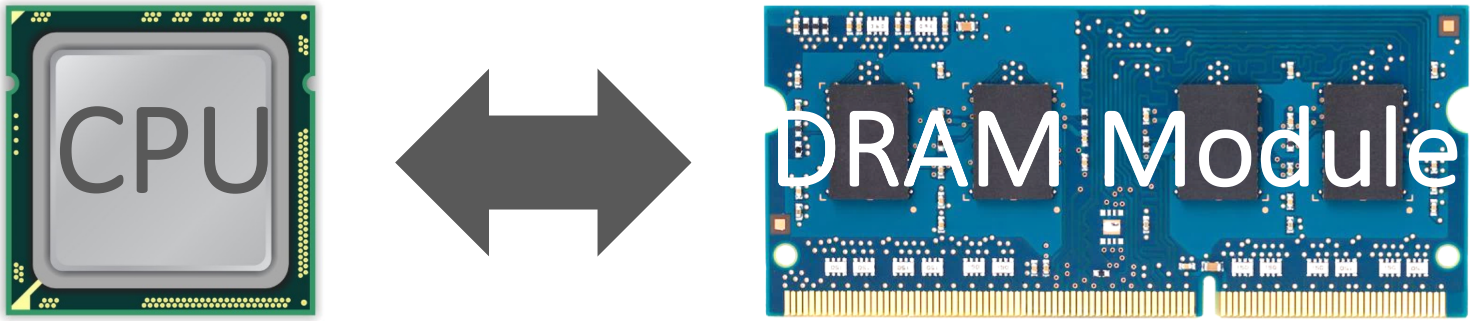


```
loop:
  mov (X), %eax
  mov (Y), %ebx
  clflush (X)
  clflush (Y)
  mfence
  jmp loop
```

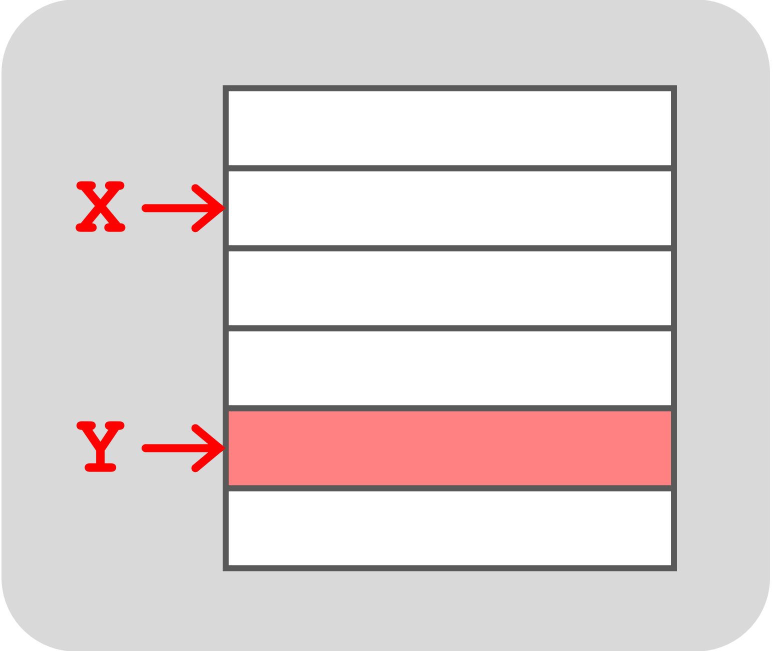


Download from: https://github.com/CMU-SAFARI/rowhammer

A Simple Program Can Induce Many Errors

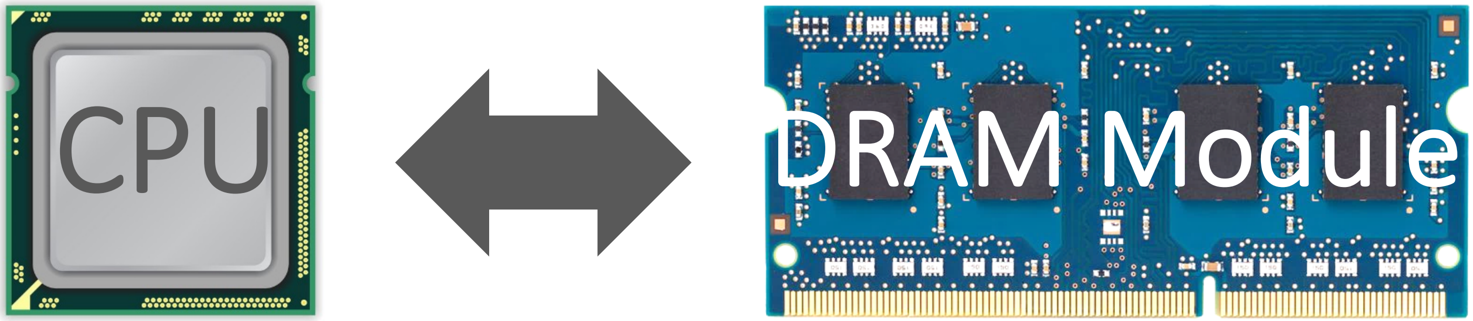


```
loop:
   mov  (X), %eax
   mov  (Y), %ebx
   clflush (X)
   clflush (Y)
   mfence
   jmp loop
```

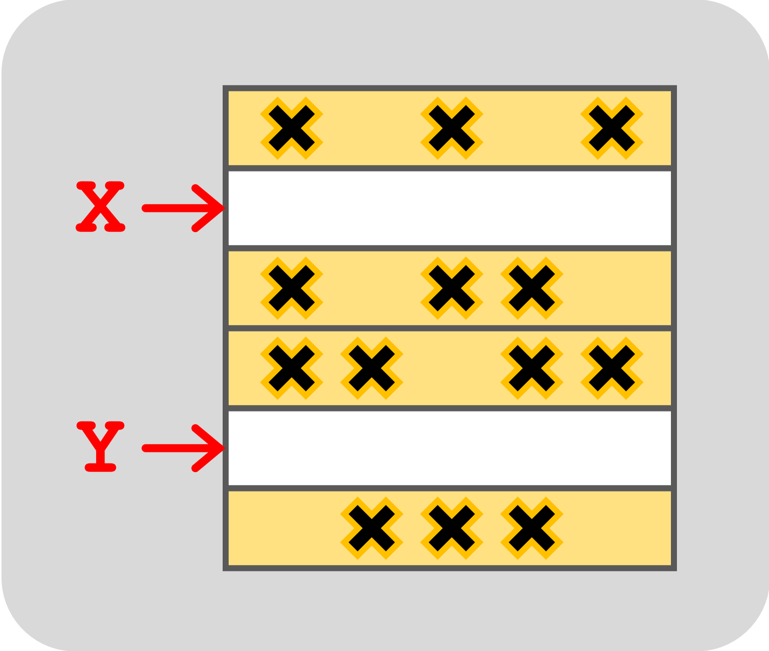


Download from: https://github.com/CMU-SAFARI/rowhammer

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

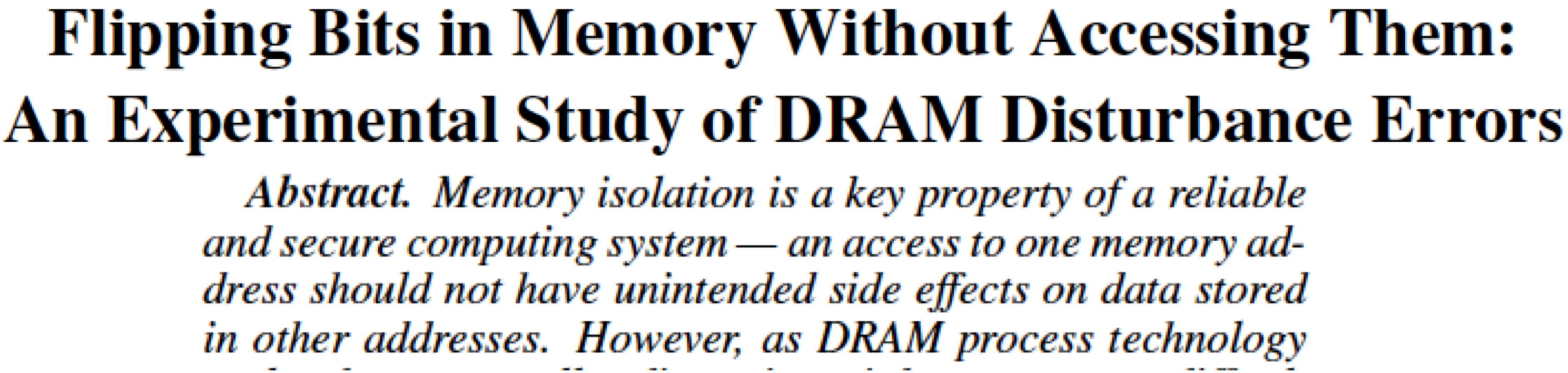
Observed Errors in Real Systems

CPU Architecture	Errors	Access-Rate
Intel Haswell (2013)	22.9K	12.3M/sec
Intel Ivy Bridge (2012)	20.7K	11.7M/sec
Intel Sandy Bridge (2011)	16.1K	11.6M/sec
AMD Piledriver (2012)	59	6.1M/sec

A real reliability & security issue

Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

One Can Take Over an Otherwise-Secure System



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

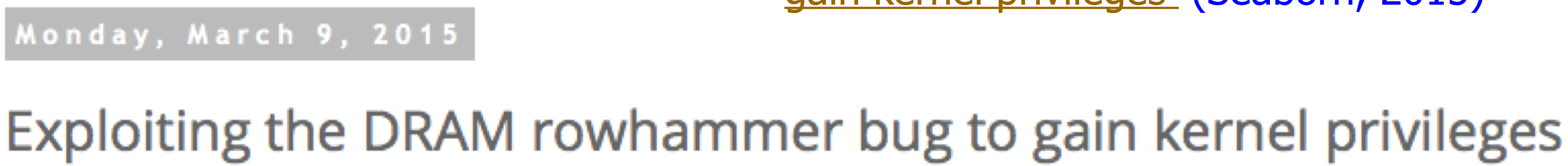
Abstract. *Memory isolation is a key property of a reliable and secure computing system — an access to one memory address should not have unintended side effects on data stored in other addresses. However, as DRAM process technology*

Project Zero

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

News and updates from the Project Zero team at Google

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



Monday, March 9, 2015

Exploiting the DRAM rowhammer bug to gain kernel privileges

RowHammer Security Attack Example

- "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

Security Implications

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)
  - 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"** *Proceedings of the 41st International Symposium on Computer Architecture (**ISCA**)*, Minneapolis, MN, June 2014. [Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Source Code and Data]

- Our Source Code to Induce Errors in Modern DRAM Chips (June 2014)
  - 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
  - **Double-sided Rowhammer**

Selected Readings on RowHammer (II)

- Remote RowHammer Attacks via JavaScript (July 2015)
  - http://arxiv.org/abs/1507.06955
  - 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)
  - https://www.ieee-security.org/TC/SP2016/papers/0824a987.pdf
  - Bosman et al., IEEE S&P 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)
  - https://www.usenix.org/system/files/conference/usenixsecurity17/sec17-brasser.pdf
  - Brasser et al., USENIX Security 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)
  - https://ieeexplore.ieee.org/document/7495576
  - 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)
  - https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_xiao.pdf
  - Xiao et al., USENIX Security 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)
    - https://link.springer.com/content/pdf/10.1007%2F978-3-662-53140-2_29.pdf
    - 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)
    - https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_pessl.pdf
    - 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)
  - https://www.usenix.org/system/files/conference/usenixsecurity16/sec16_paper_razavi.pdf
  - 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
  - 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)
  - https://web.eecs.umich.edu/~misiker/resources/HOST-2017-Misiker.pdf
  - 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
  - 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)
  - https://arxiv.org/pdf/1710.00551.pdf
  - 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
  - Van Der Veen et al., DIMVA 2018
  - Presents RAMPAGE, a DMA-based RowHammer attack against the latest Android OS

Selected Readings on RowHammer (IX)

- Grand Pwning Unit: Accelerating Microarchitectural Attacks with the GPU (May 2018)
  - https://www.vusec.net/wp-content/uploads/2018/05/glitch.pdf
  - Frigo et al., IEEE S&P 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)
  - https://www.cs.vu.nl/~herbertb/download/papers/throwhammer_atc18.pdf
  - 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**."

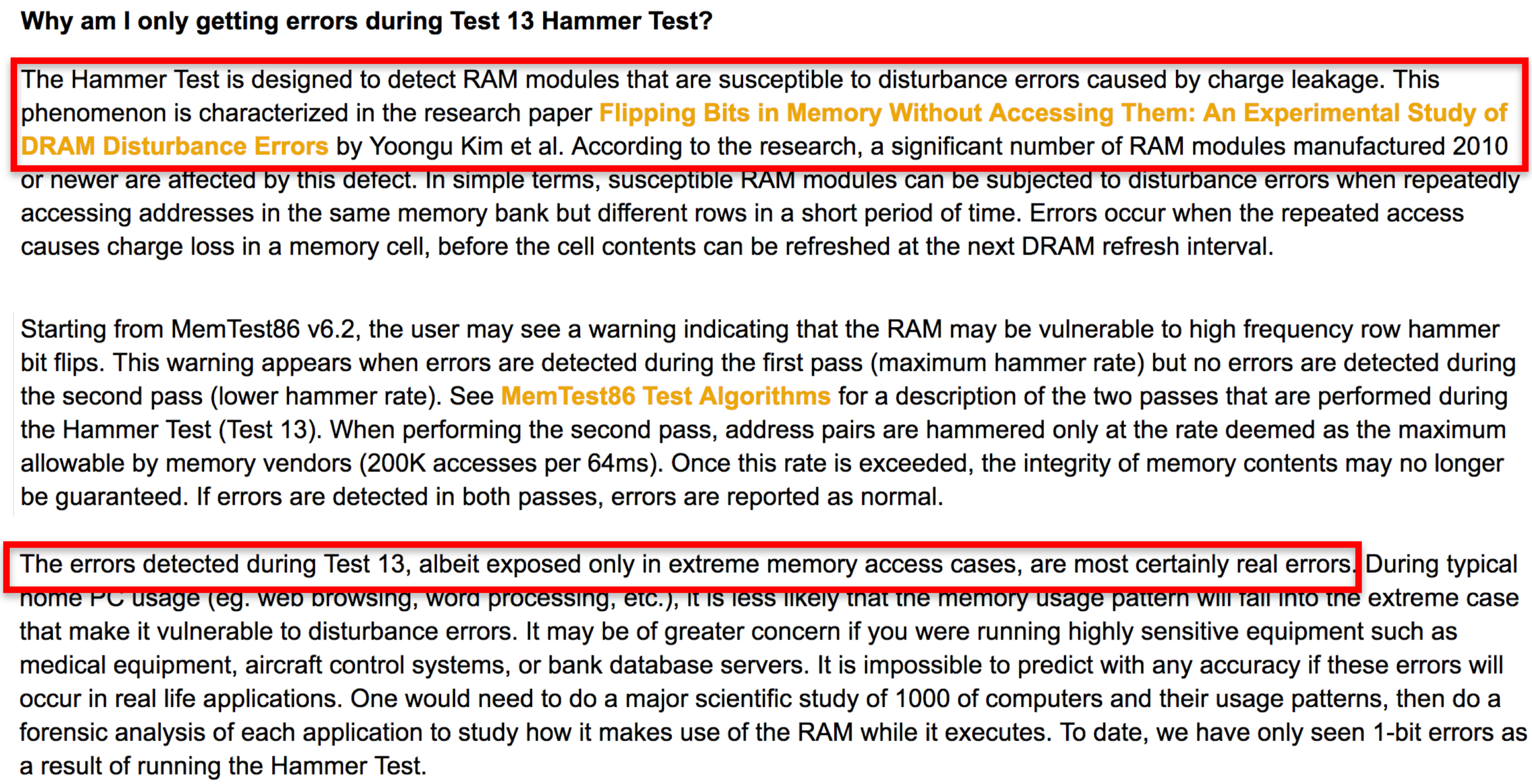
Selected Readings on RowHammer (X)

- Nethammer: Inducing Rowhammer Faults through Network Requests (July 2018)
  - https://arxiv.org/pdf/1805.04956.pdf
  - 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)
  - https://www.usenix.org/system/files/osdi18-konoth.pdf
  - 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



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

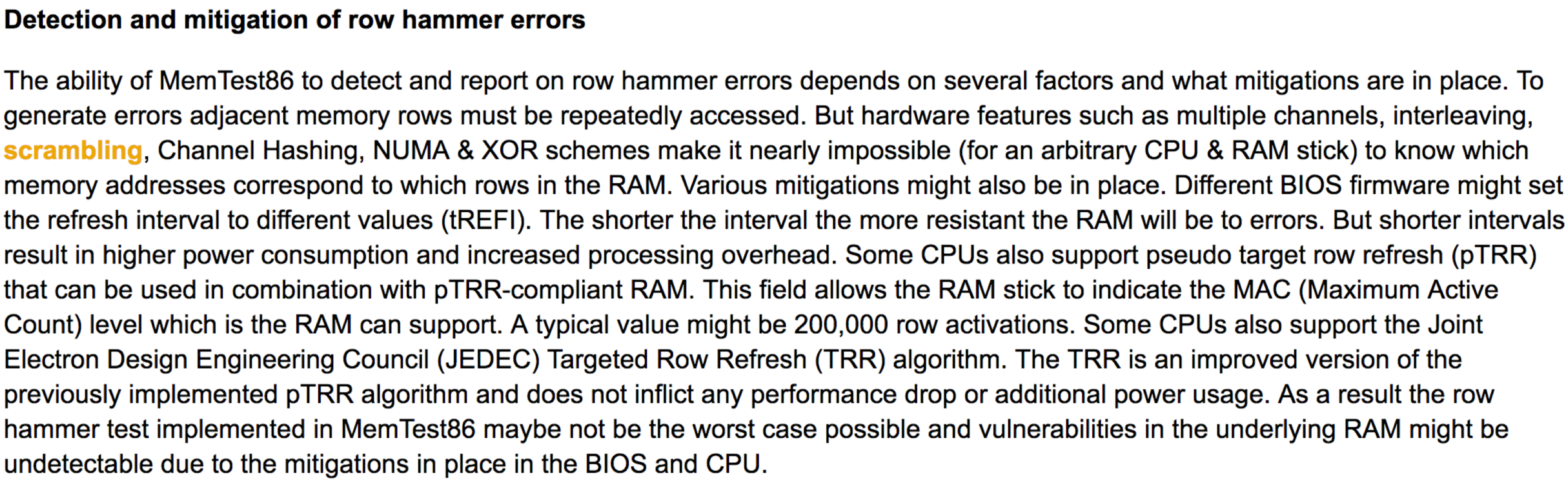
The Hammer Test is designed to detect RAM modules that are susceptible to disturbance errors caused by charge leakage. This phenomenon is characterized in the research paper **Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors** by Yoongu Kim et al. According 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 (eg. 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)

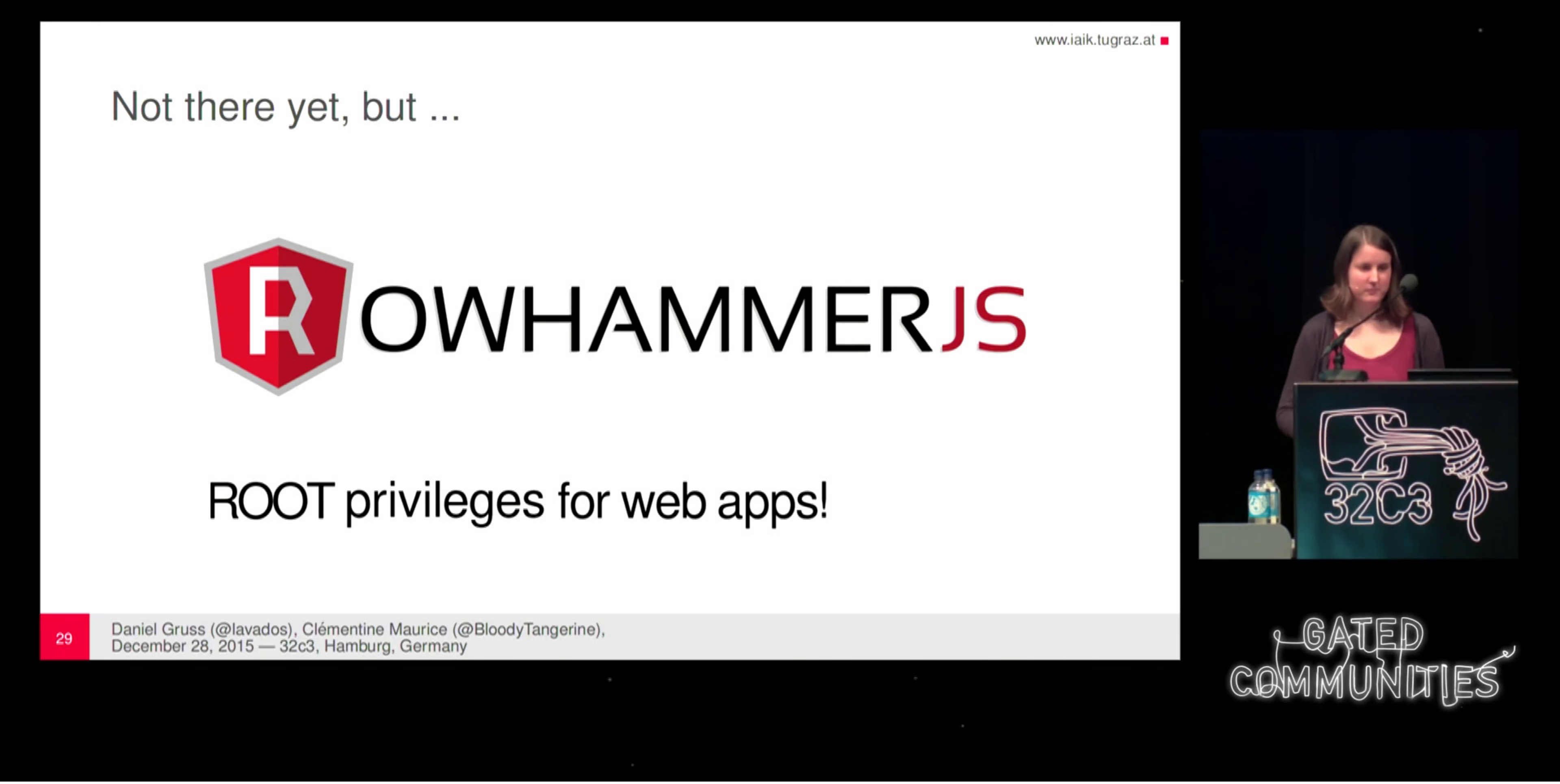
- *Breach of memory protection*
  - OS page (4KB) fits inside DRAM row (8KB)
  - Adjacent DRAM row → Different OS page

- *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."



www.iaik.tugraz.at

Not there yet, but ...

ROWHAMMERJS

ROOT privileges for web apps!

29

Daniel Gruss (@lavados), Clémentine Maurice (@BloodyTangerine),
December 28, 2015 — 32c3, Hamburg, Germany

Rowhammer.js: A Remote Software-Induced Fault Attack in JavaScript (DIMVA'16)

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

More Security Implications (II)

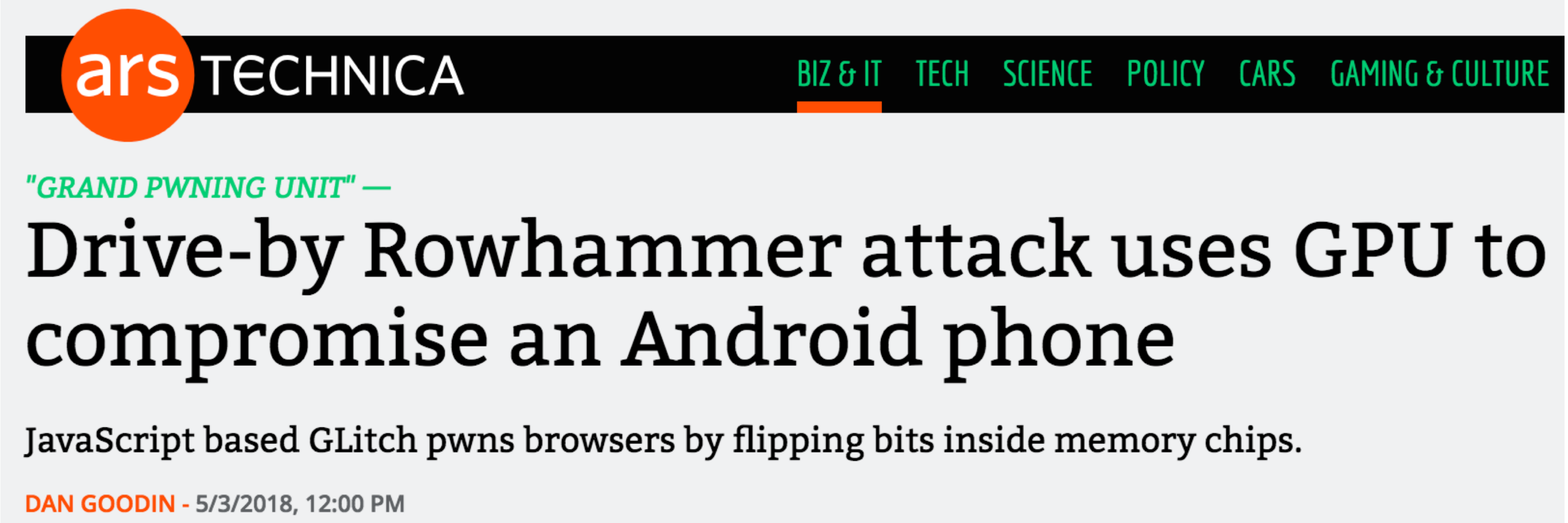
"Can gain control of a smart phone deterministically"

Drammer: Deterministic Rowhammer Attacks on Mobile Platforms, CCS'16

Source: https://fossbytes.com/drammer-rowhammer-attack-android-root-devices/

More Security Implications (III)

- Using an integrated GPU in a mobile system to remotely escalate privilege via the WebGL interface

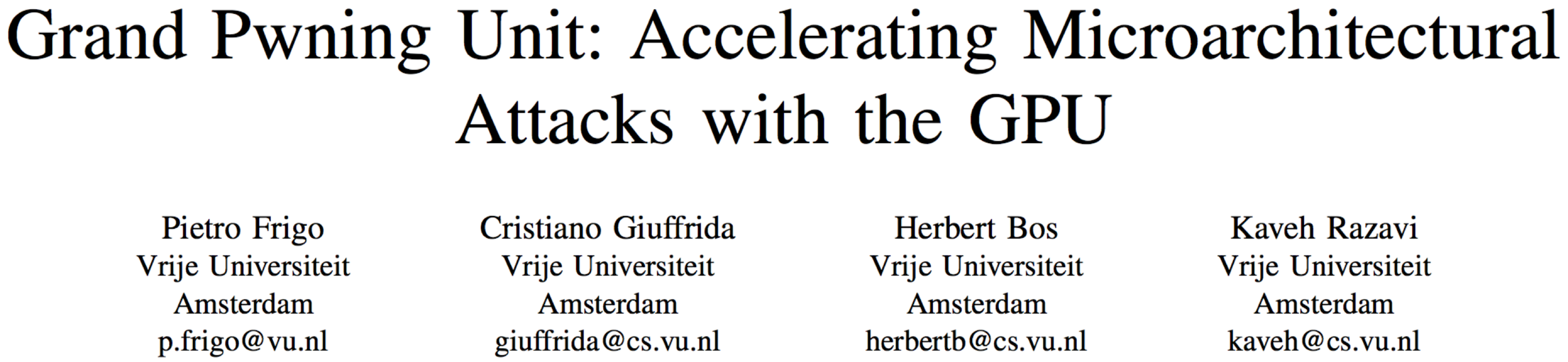


"GRAND PWNING UNIT" —

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

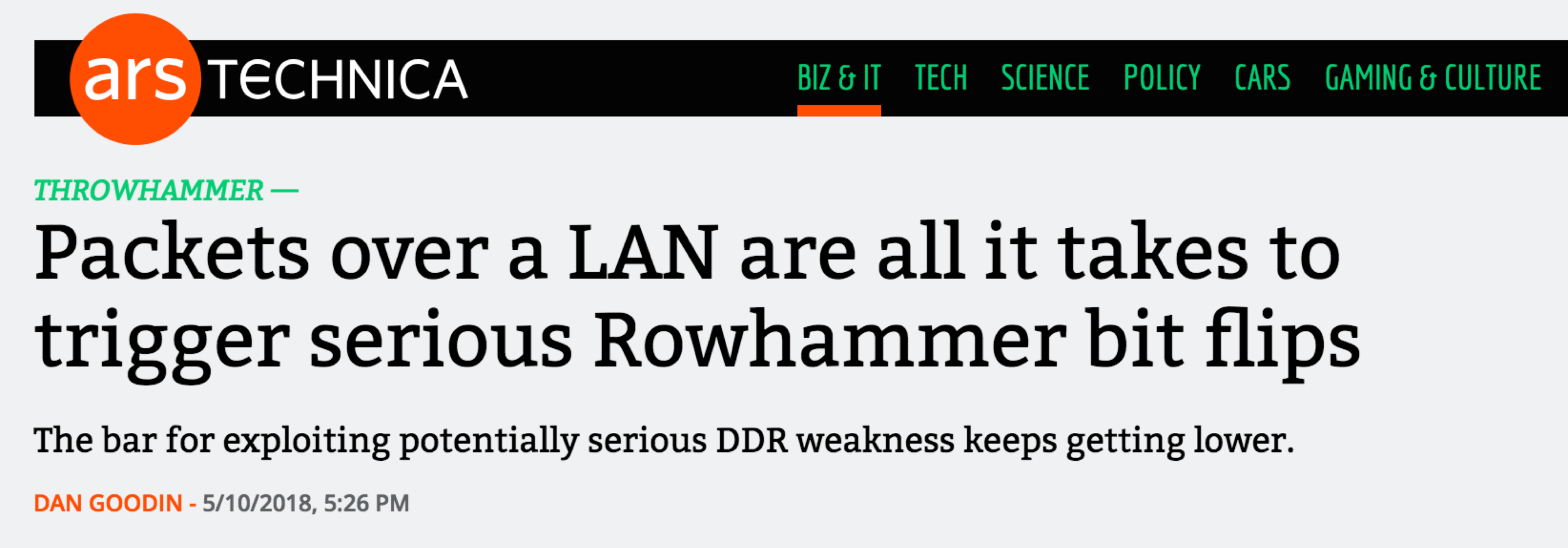
Cristiano Giuffrida
Vrije Universiteit
Amsterdam
giuffrida@cs.vu.nl

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

Kaveh Razavi
Vrije Universiteit
Amsterdam
kaveh@cs.vu.nl

More Security Implications (IV)

- Rowhammer over RDMA (I)

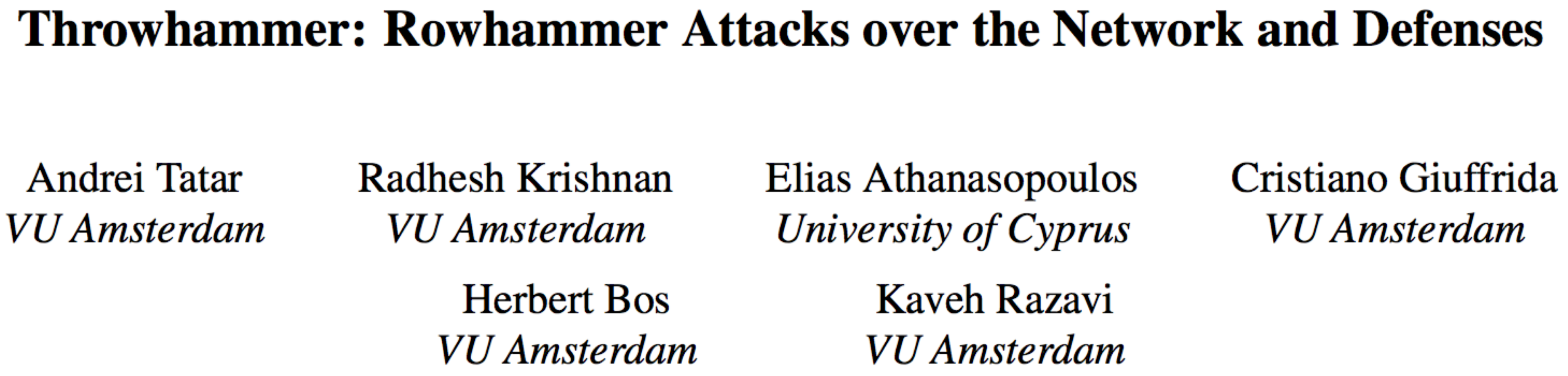


THROWHAMMER —

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.

DAN GOODIN - 5/10/2018, 5:26 PM



Throwhammer: Rowhammer Attacks over the Network and Defenses

Andrei Tatar
VU Amsterdam

Radhesh Krishnan
VU Amsterdam

Elias Athanasopoulos
University of Cyprus

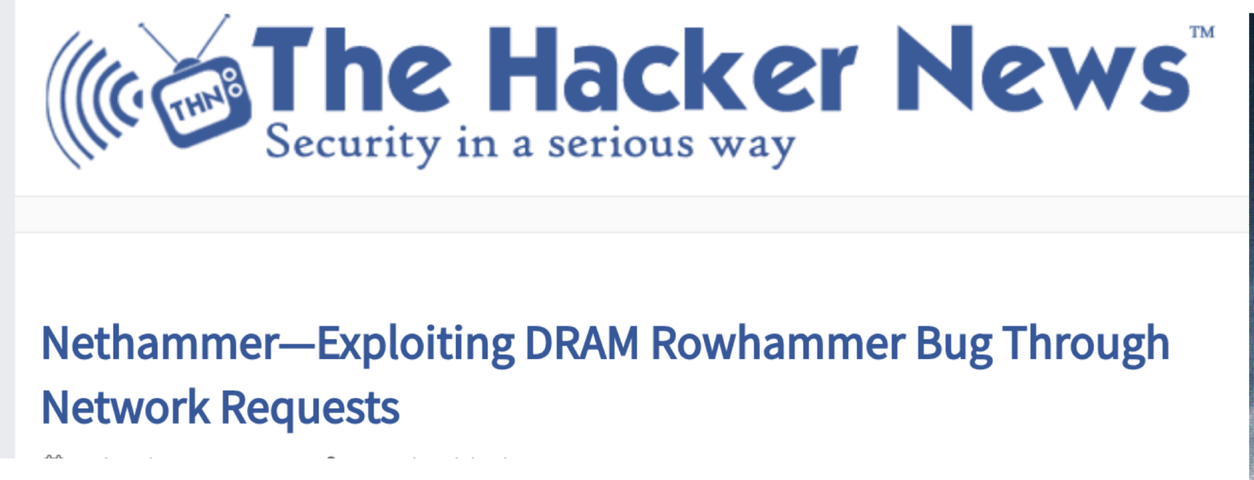
Cristiano Giuffrida
VU Amsterdam

Herbert Bos
VU Amsterdam

Kaveh Razavi
VU Amsterdam

More Security Implications (V)

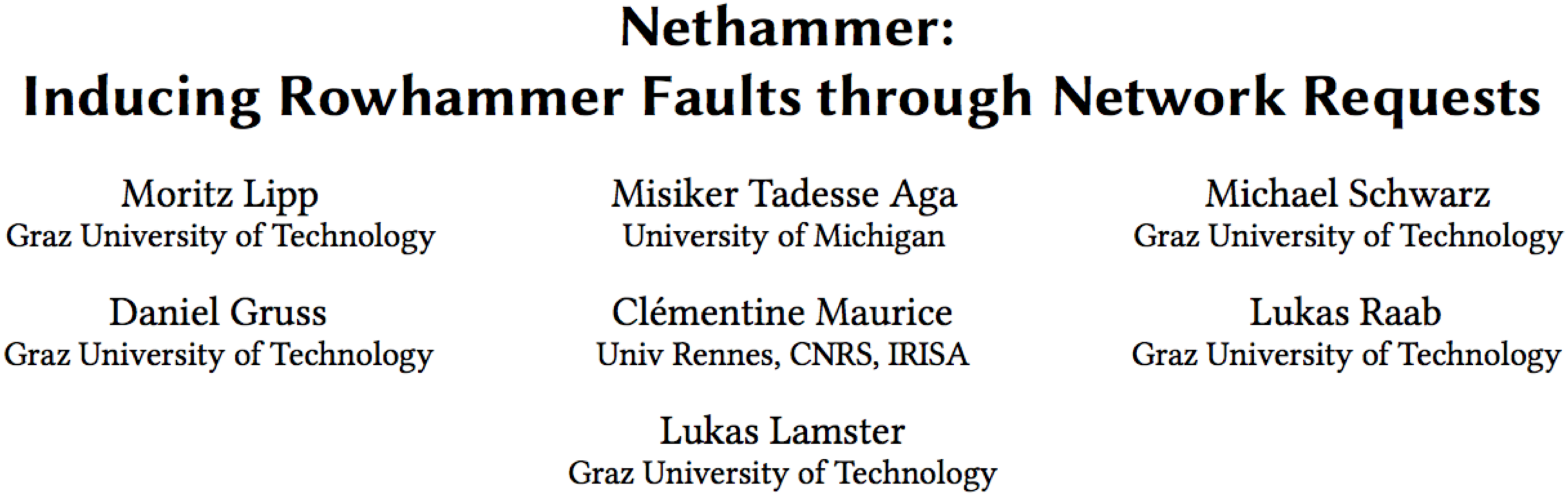
- Rowhammer over RDMA (II)



The Hacker News™

Security in a serious way

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

Michael Schwarz
Graz University of Technology

Daniel Gruss
Graz University of Technology

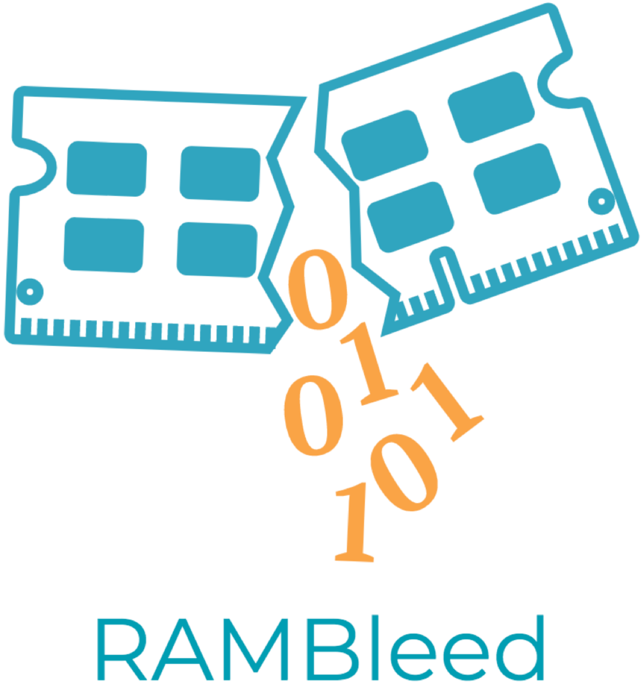
Clémentine Maurice
Univ Rennes, CNRS, IRISA

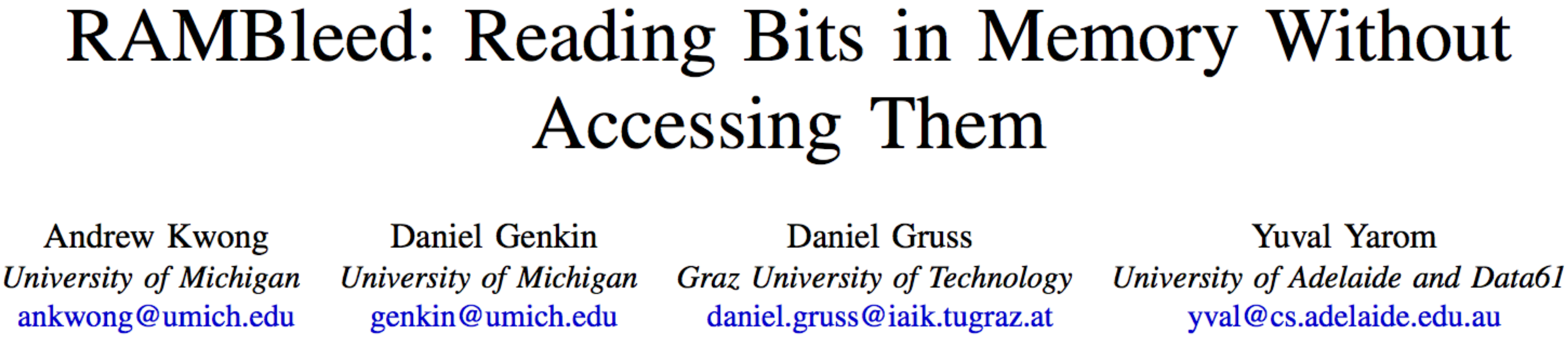
Lukas Raab
Graz University of Technology

Lukas Lamster
Graz University of Technology

More Security Implications (VI)

- IEEE S&P 2020




RAMBleed: Reading Bits in Memory Without Accessing Them

Andrew Kwong
University of Michigan
ankwong@umich.edu

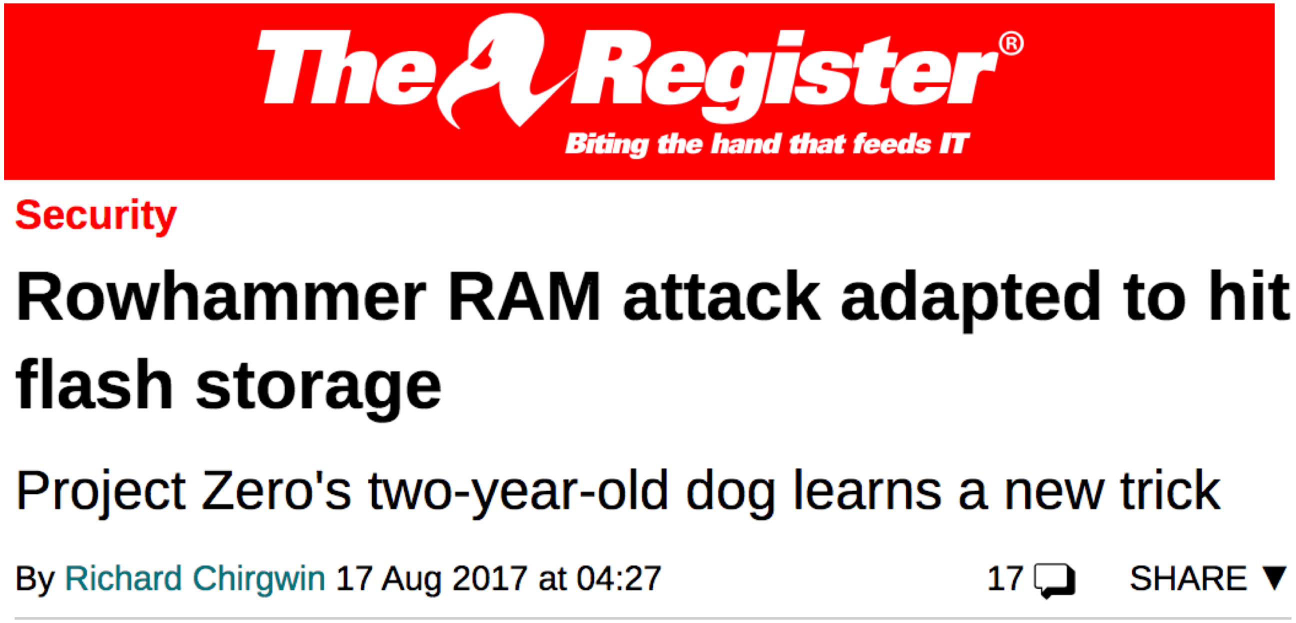
Daniel Genkin
University of Michigan
genkin@umich.edu

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)

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



The Register®

Biting the hand that feeds IT

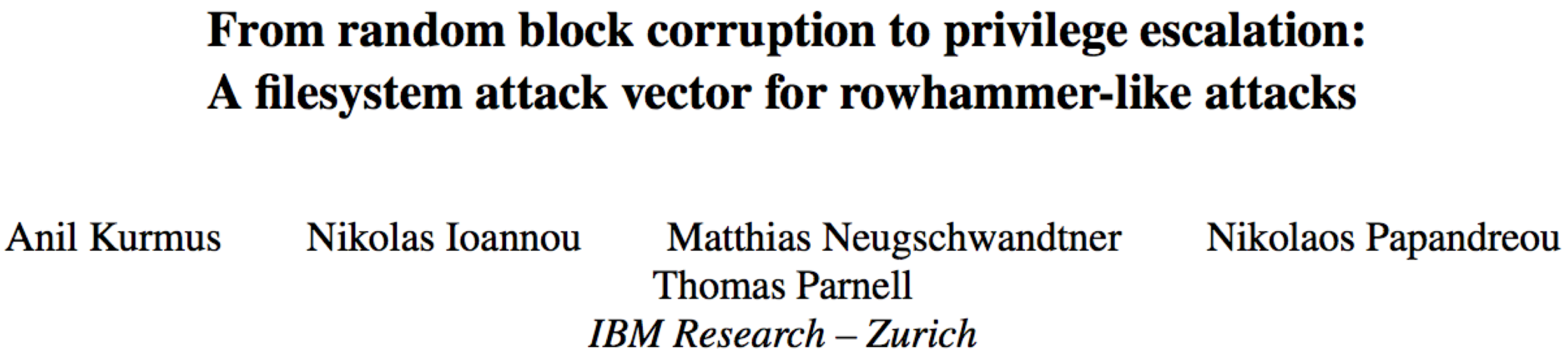
Security

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

17 SHARE ▼



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?

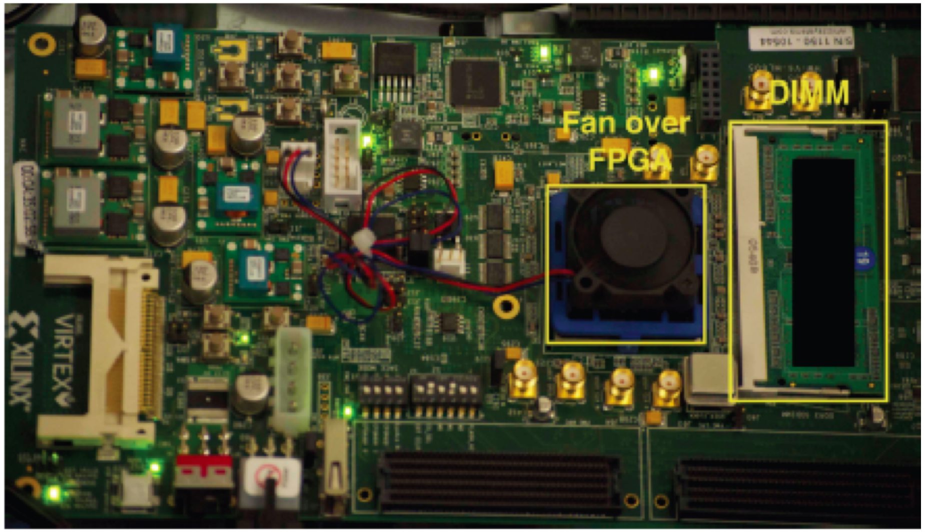
Understanding RowHammer

Root Causes of Disturbance Errors

- *Cause 1: Electromagnetic coupling*
  - Toggling the wordline voltage briefly increases the voltage of adjacent wordlines
  - Slightly opens adjacent rows → Charge leakage
- *Cause 2: Conductive bridges*
- *Cause 3: Hot-carrier injection*

Confirmed by at least one manufacturer

Experimental DRAM Testing Infrastructure



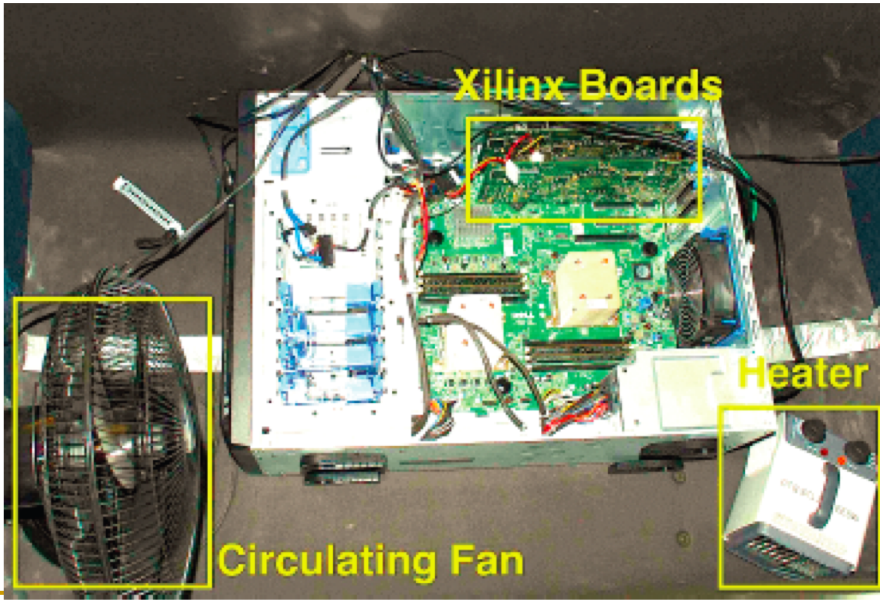
An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms (Liu et al., ISCA 2013)

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

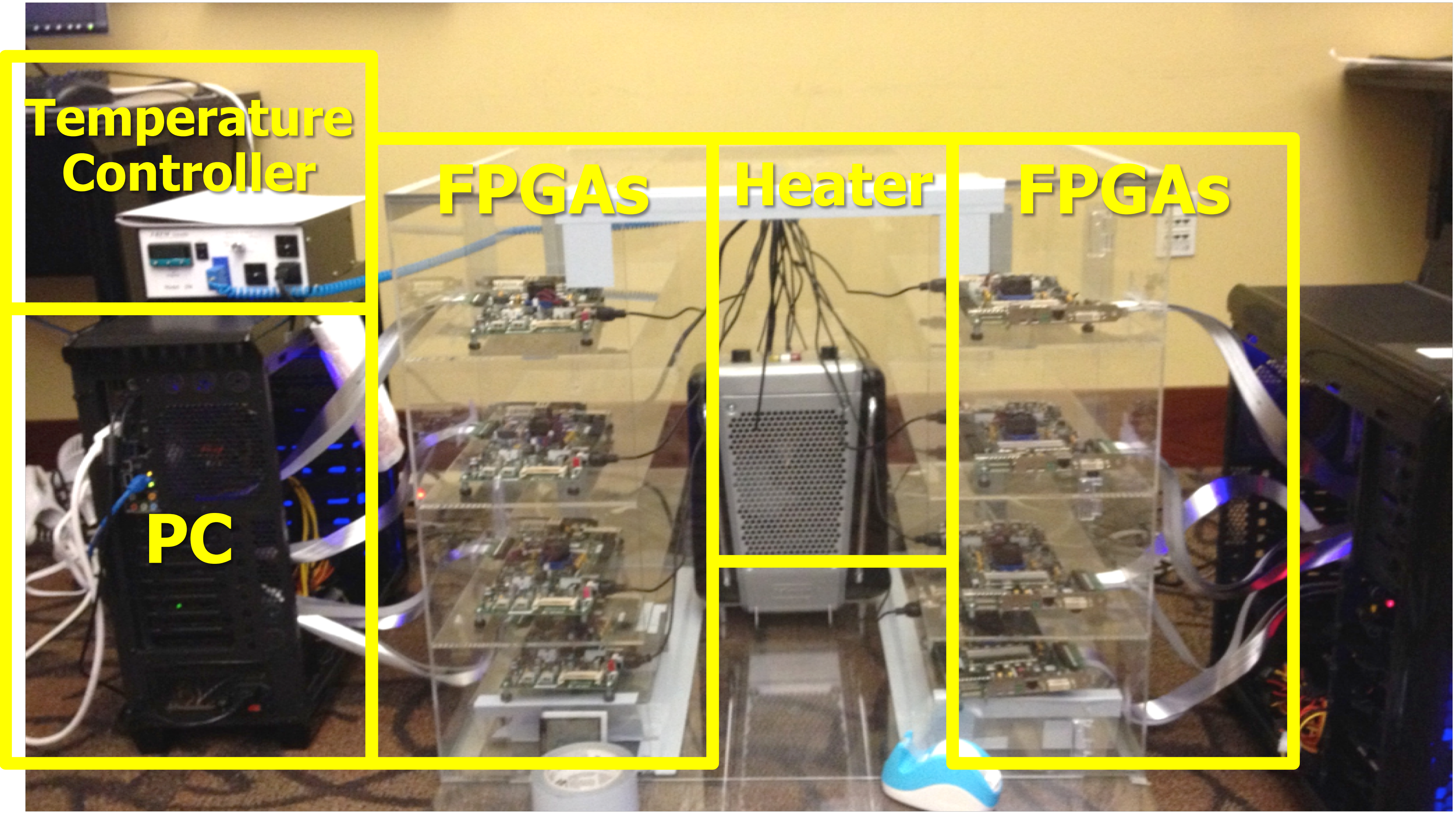
Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors (Kim et al., ISCA 2014)

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)



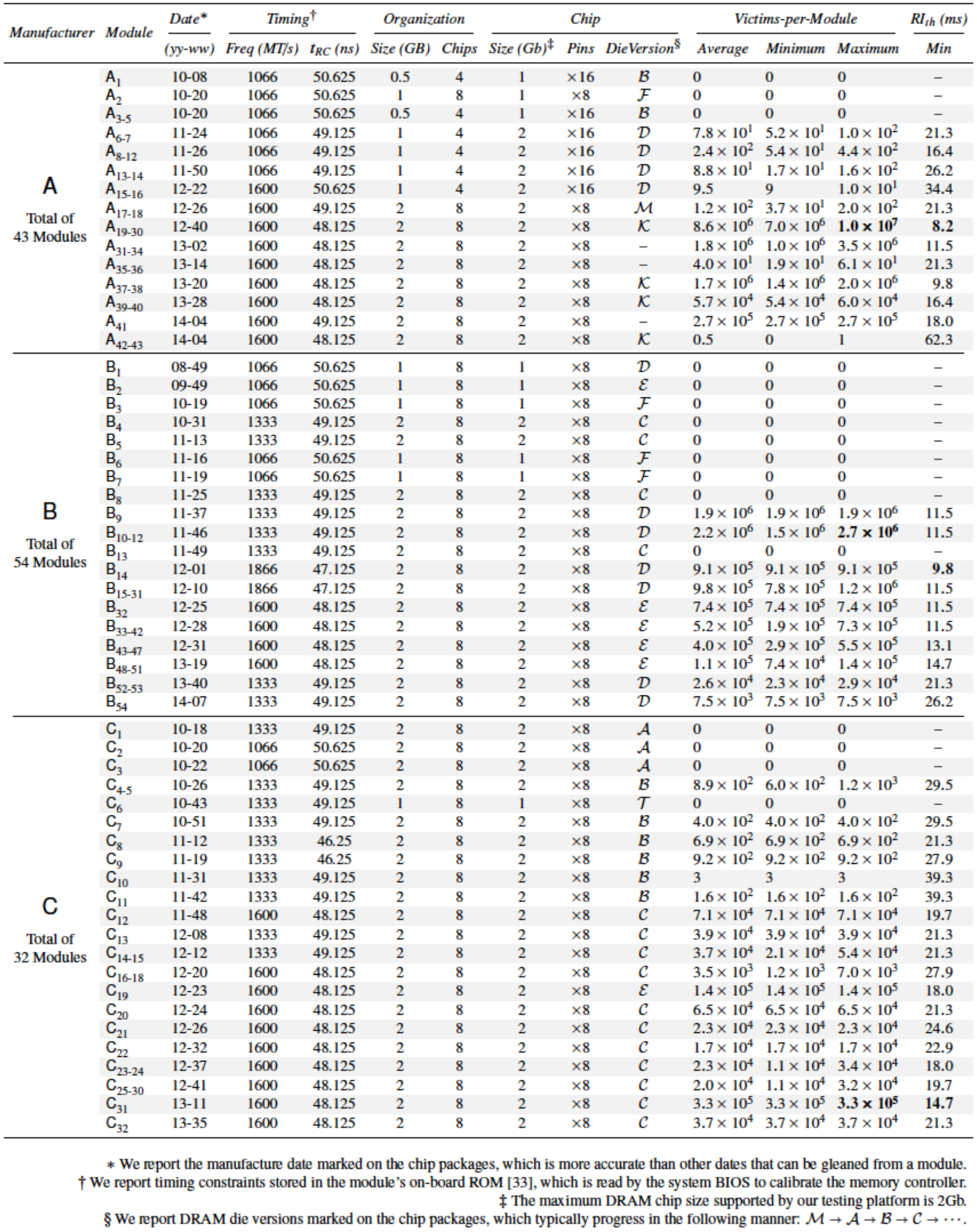
Experimental DRAM Testing Infrastructure



Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

Tested DRAM Modules

(129 total)



Manufacturer	Module	*Date**	*Timing*†		*Organization*		*Chip*			*Victims-per-Module*			RI_{th} *(ms)*
		(yy-ww)	*Freq (MT/s)*	t_{RC} *(ns)*	*Size (GB)*	*Chips*	*Size (Gb)*‡	*Pins*	*DieVersion*§	*Average*	*Minimum*	*Maximum*	*Min*
A Total of 43 Modules	A_{1}	10-08	1066	50.625	0.5	4	1	×16	$\mathcal{B}$	0	0	0	–
	A_{2}	10-20	1066	50.625	1	8	1	×8	$\mathcal{F}$	0	0	0	–
	$A_{3\text{-}5}$	10-20	1066	50.625	0.5	4	1	×16	$\mathcal{B}$	0	0	0	–
	$A_{6\text{-}7}$	11-24	1066	49.125	1	4	2	×16	$\mathcal{D}$	7.8×10^{1}	5.2×10^{1}	1.0×10^{2}	21.3
	$A_{8\text{-}12}$	11-26	1066	49.125	1	4	2	×16	$\mathcal{D}$	2.4×10^{2}	5.4×10^{1}	4.4×10^{2}	16.4
	$A_{13\text{-}14}$	11-50	1066	49.125	1	4	2	×16	$\mathcal{D}$	8.8×10^{1}	1.7×10^{1}	1.6×10^{2}	26.2
	$A_{15\text{-}16}$	12-22	1600	50.625	1	4	2	×16	$\mathcal{D}$	9.5	9	1.0×10^{1}	34.4
	$A_{17\text{-}18}$	12-26	1600	49.125	2	8	2	×8	$\mathcal{M}$	1.2×10^{2}	3.7×10^{1}	2.0×10^{2}	21.3
	$A_{19\text{-}30}$	12-40	1600	48.125	2	8	2	×8	$\mathcal{K}$	8.6×10^{6}	7.0×10^{6}	$\mathbf{1.0 \times 10^{7}}$	**8.2**
	$A_{31\text{-}34}$	13-02	1600	48.125	2	8	2	×8	–	1.8×10^{6}	1.0×10^{6}	3.5×10^{6}	11.5
	$A_{35\text{-}36}$	13-14	1600	48.125	2	8	2	×8	–	4.0×10^{1}	1.9×10^{1}	6.1×10^{1}	21.3
	$A_{37\text{-}38}$	13-20	1600	48.125	2	8	2	×8	$\mathcal{K}$	1.7×10^{6}	1.4×10^{6}	2.0×10^{6}	9.8
	$A_{39\text{-}40}$	13-28	1600	48.125	2	8	2	×8	$\mathcal{K}$	5.7×10^{4}	5.4×10^{4}	6.0×10^{4}	16.4
	A_{41}	14-04	1600	49.125	2	8	2	×8	–	2.7×10^{5}	2.7×10^{5}	2.7×10^{5}	18.0
	$A_{42\text{-}43}$	14-04	1600	48.125	2	8	2	×8	$\mathcal{K}$	0.5	0	1	62.3
B Total of 54 Modules	B_{1}	08-49	1066	50.625	1	8	1	×8	$\mathcal{D}$	0	0	0	–
	B_{2}	09-49	1066	50.625	1	8	1	×8	$\mathcal{E}$	0	0	0	–
	B_{3}	10-19	1066	50.625	1	8	1	×8	$\mathcal{F}$	0	0	0	–
	B_{4}	10-31	1333	49.125	2	8	2	×8	$\mathcal{C}$	0	0	0	–
	B_{5}	11-13	1333	49.125	2	8	2	×8	$\mathcal{C}$	0	0	0	–
	B_{6}	11-16	1066	50.625	1	8	1	×8	$\mathcal{F}$	0	0	0	–
	B_{7}	11-19	1066	50.625	1	8	1	×8	$\mathcal{F}$	0	0	0	–
	B_{8}	11-25	1333	49.125	2	8	2	×8	$\mathcal{C}$	0	0	0	–
	B_{9}	11-37	1333	49.125	2	8	2	×8	$\mathcal{D}$	1.9×10^{6}	1.9×10^{6}	1.9×10^{6}	11.5
	$B_{10\text{-}12}$	11-46	1333	49.125	2	8	2	×8	$\mathcal{D}$	2.2×10^{6}	1.5×10^{6}	$\mathbf{2.7 \times 10^{6}}$	11.5
	B_{13}	11-49	1333	49.125	2	8	2	×8	$\mathcal{C}$	0	0	0	–
	B_{14}	12-01	1866	47.125	2	8	2	×8	$\mathcal{D}$	9.1×10^{5}	9.1×10^{5}	9.1×10^{5}	**9.8**
	$B_{15\text{-}31}$	12-10	1866	47.125	2	8	2	×8	$\mathcal{D}$	9.8×10^{5}	7.8×10^{5}	1.2×10^{6}	11.5
	B_{32}	12-25	1600	48.125	2	8	2	×8	$\mathcal{E}$	7.4×10^{5}	7.4×10^{5}	7.4×10^{5}	11.5
	$B_{33\text{-}42}$	12-28	1600	48.125	2	8	2	×8	$\mathcal{E}$	5.2×10^{5}	1.9×10^{5}	7.3×10^{5}	11.5
	$B_{43\text{-}47}$	12-31	1600	48.125	2	8	2	×8	$\mathcal{E}$	4.0×10^{5}	2.9×10^{5}	5.5×10^{5}	13.1
	$B_{48\text{-}51}$	13-19	1600	48.125	2	8	2	×8	$\mathcal{E}$	1.1×10^{5}	7.4×10^{4}	1.4×10^{5}	14.7
	$B_{52\text{-}53}$	13-40	1333	49.125	2	8	2	×8	$\mathcal{D}$	2.6×10^{4}	2.3×10^{4}	2.9×10^{4}	21.3
	B_{54}	14-07	1333	49.125	2	8	2	×8	$\mathcal{D}$	7.5×10^{3}	7.5×10^{3}	7.5×10^{3}	26.2
C Total of 32 Modules	C_{1}	10-18	1333	49.125	2	8	2	×8	$\mathcal{A}$	0	0	0	–
	C_{2}	10-20	1066	50.625	2	8	2	×8	$\mathcal{A}$	0	0	0	–
	C_{3}	10-22	1066	50.625	2	8	2	×8	$\mathcal{A}$	0	0	0	–
	$C_{4\text{-}5}$	10-26	1333	49.125	2	8	2	×8	$\mathcal{B}$	8.9×10^{2}	6.0×10^{2}	1.2×10^{3}	29.5
	C_{6}	10-43	1333	49.125	1	8	1	×8	$\mathcal{T}$	0	0	0	–
	C_{7}	10-51	1333	49.125	2	8	2	×8	$\mathcal{B}$	4.0×10^{2}	4.0×10^{2}	4.0×10^{2}	29.5
	C_{8}	11-12	1333	46.25	2	8	2	×8	$\mathcal{B}$	6.9×10^{2}	6.9×10^{2}	6.9×10^{2}	21.3
	C_{9}	11-19	1333	46.25	2	8	2	×8	$\mathcal{B}$	9.2×10^{2}	9.2×10^{2}	9.2×10^{2}	27.9
	C_{10}	11-31	1333	49.125	2	8	2	×8	$\mathcal{B}$	3	3	3	39.3
	C_{11}	11-42	1333	49.125	2	8	2	×8	$\mathcal{B}$	1.6×10^{2}	1.6×10^{2}	1.6×10^{2}	39.3
	C_{12}	11-48	1600	48.125	2	8	2	×8	$\mathcal{C}$	7.1×10^{4}	7.1×10^{4}	7.1×10^{4}	19.7
	C_{13}	12-08	1333	49.125	2	8	2	×8	$\mathcal{C}$	3.9×10^{4}	3.9×10^{4}	3.9×10^{4}	21.3
	$C_{14\text{-}15}$	12-12	1333	49.125	2	8	2	×8	$\mathcal{C}$	3.7×10^{4}	2.1×10^{4}	5.4×10^{4}	21.3
	$C_{16\text{-}18}$	12-20	1600	48.125	2	8	2	×8	$\mathcal{C}$	3.5×10^{3}	1.2×10^{3}	7.0×10^{3}	27.9
	C_{19}	12-23	1600	48.125	2	8	2	×8	$\mathcal{E}$	1.4×10^{5}	1.4×10^{5}	1.4×10^{5}	18.0
	C_{20}	12-24	1600	48.125	2	8	2	×8	$\mathcal{C}$	6.5×10^{4}	6.5×10^{4}	6.5×10^{4}	21.3
	C_{21}	12-26	1600	48.125	2	8	2	×8	$\mathcal{C}$	2.3×10^{4}	2.3×10^{4}	2.3×10^{4}	24.6
	C_{22}	12-32	1600	48.125	2	8	2	×8	$\mathcal{C}$	1.7×10^{4}	1.7×10^{4}	1.7×10^{4}	22.9
	$C_{23\text{-}24}$	12-37	1600	48.125	2	8	2	×8	$\mathcal{C}$	2.3×10^{4}	1.1×10^{4}	3.4×10^{4}	18.0
	$C_{25\text{-}30}$	12-41	1600	48.125	2	8	2	×8	$\mathcal{C}$	2.0×10^{4}	1.1×10^{4}	3.2×10^{4}	19.7
	C_{31}	13-11	1600	48.125	2	8	2	×8	$\mathcal{C}$	3.3×10^{5}	3.3×10^{5}	$\mathbf{3.3 \times 10^{5}}$	**14.7**
	C_{32}	13-35	1600	48.125	2	8	2	×8	$\mathcal{C}$	3.7×10^{4}	3.7×10^{4}	3.7×10^{4}	21.3

∗ 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: $\mathcal{M} \rightarrow \mathcal{A} \rightarrow \mathcal{B} \rightarrow \mathcal{C} \rightarrow \cdots$.

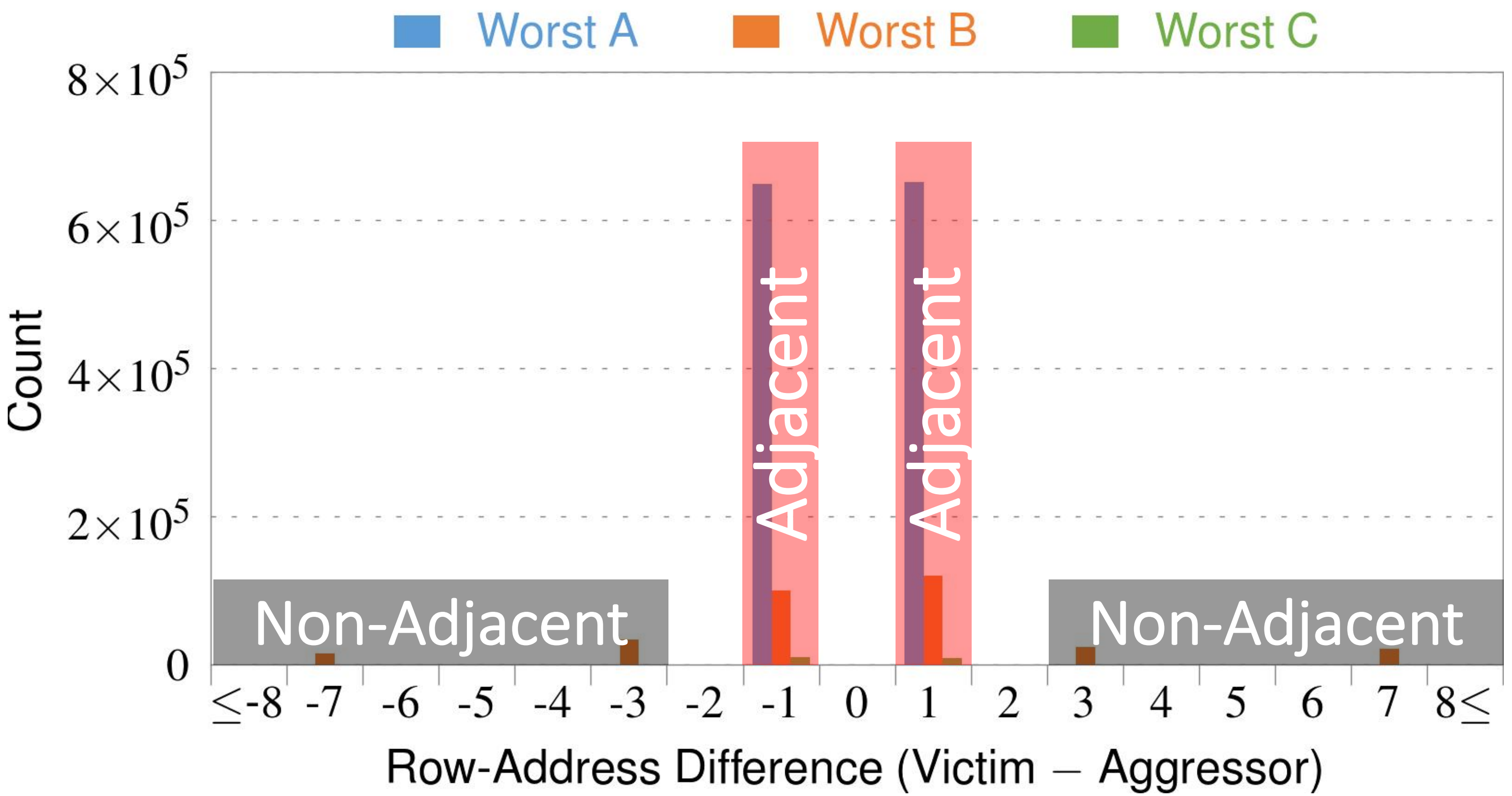
Table 3. Sample population of 129 DDR3 DRAM modules, categorized by manufacturer and sorted by manufacture date

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)

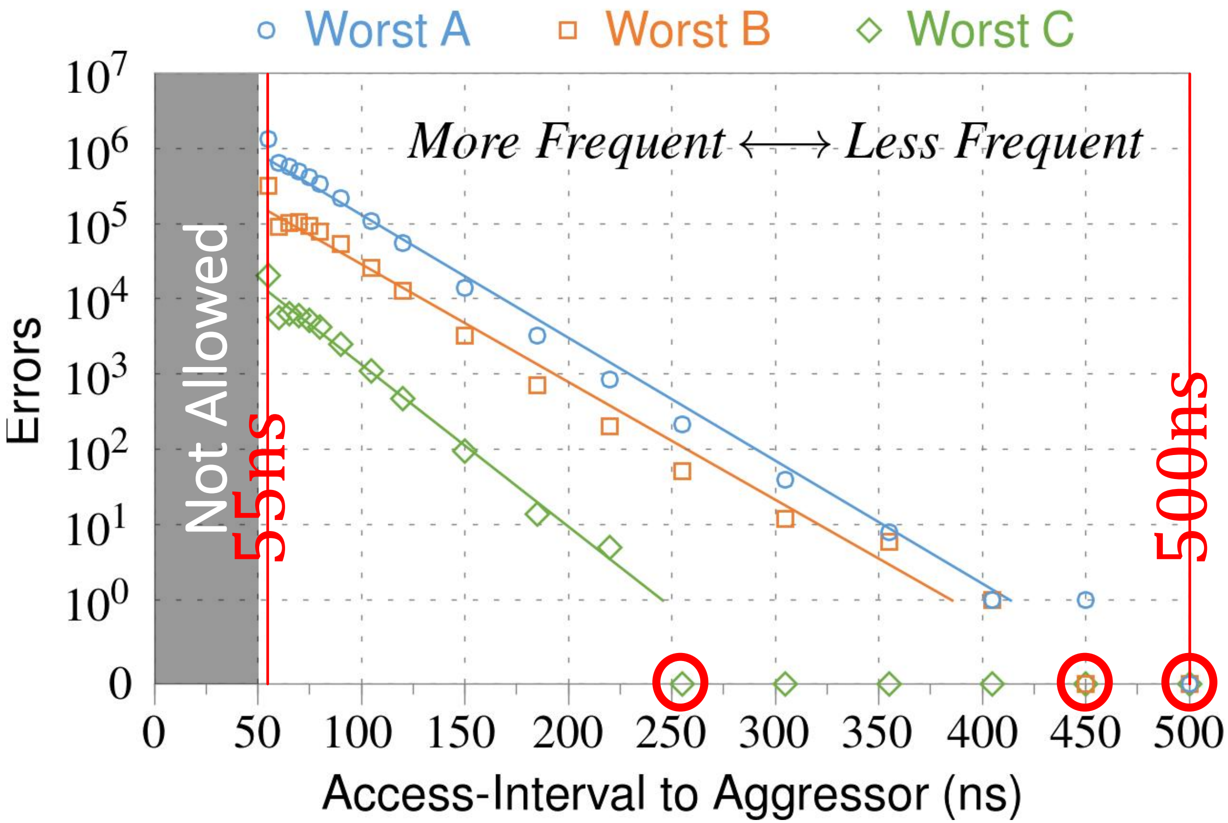
4. Adjacency: Aggressor & Victim



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

Most aggressors & victims are adjacent

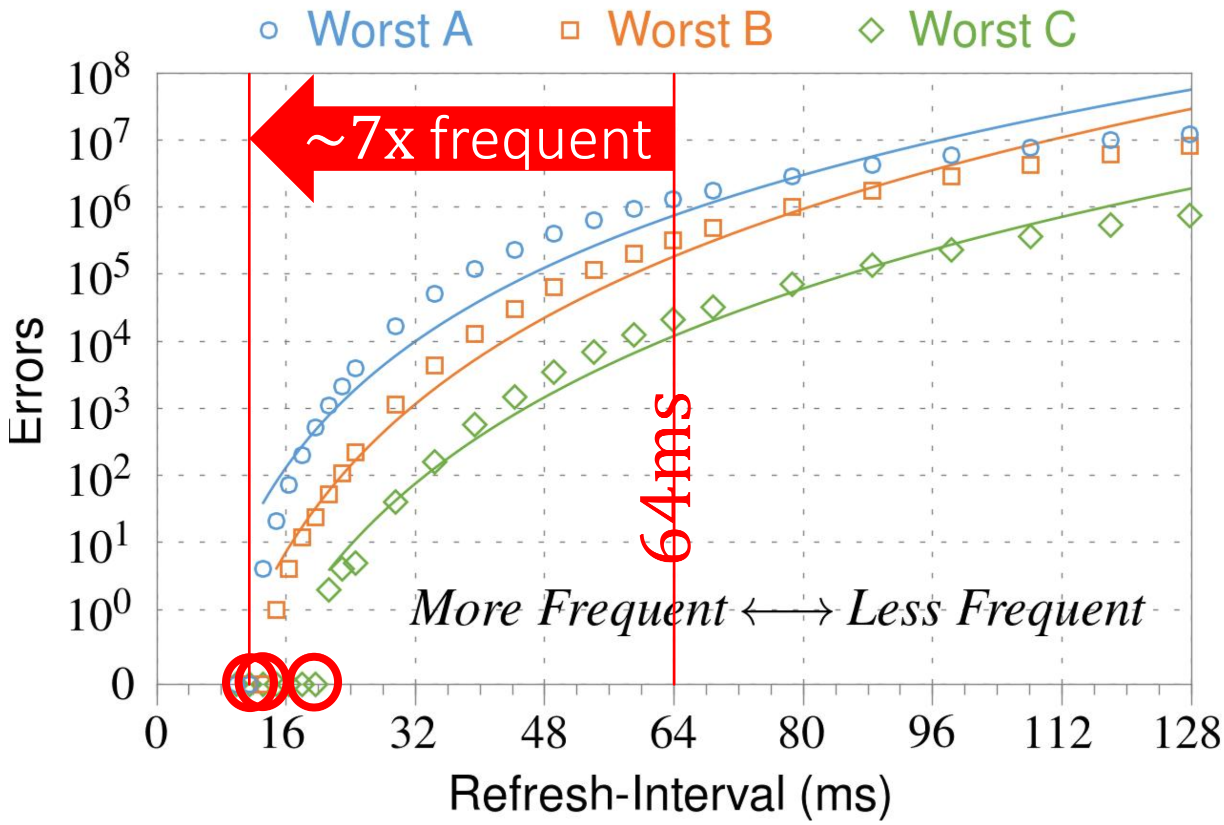
❶ Access Interval (Aggressor)



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

Less frequent accesses → Fewer errors

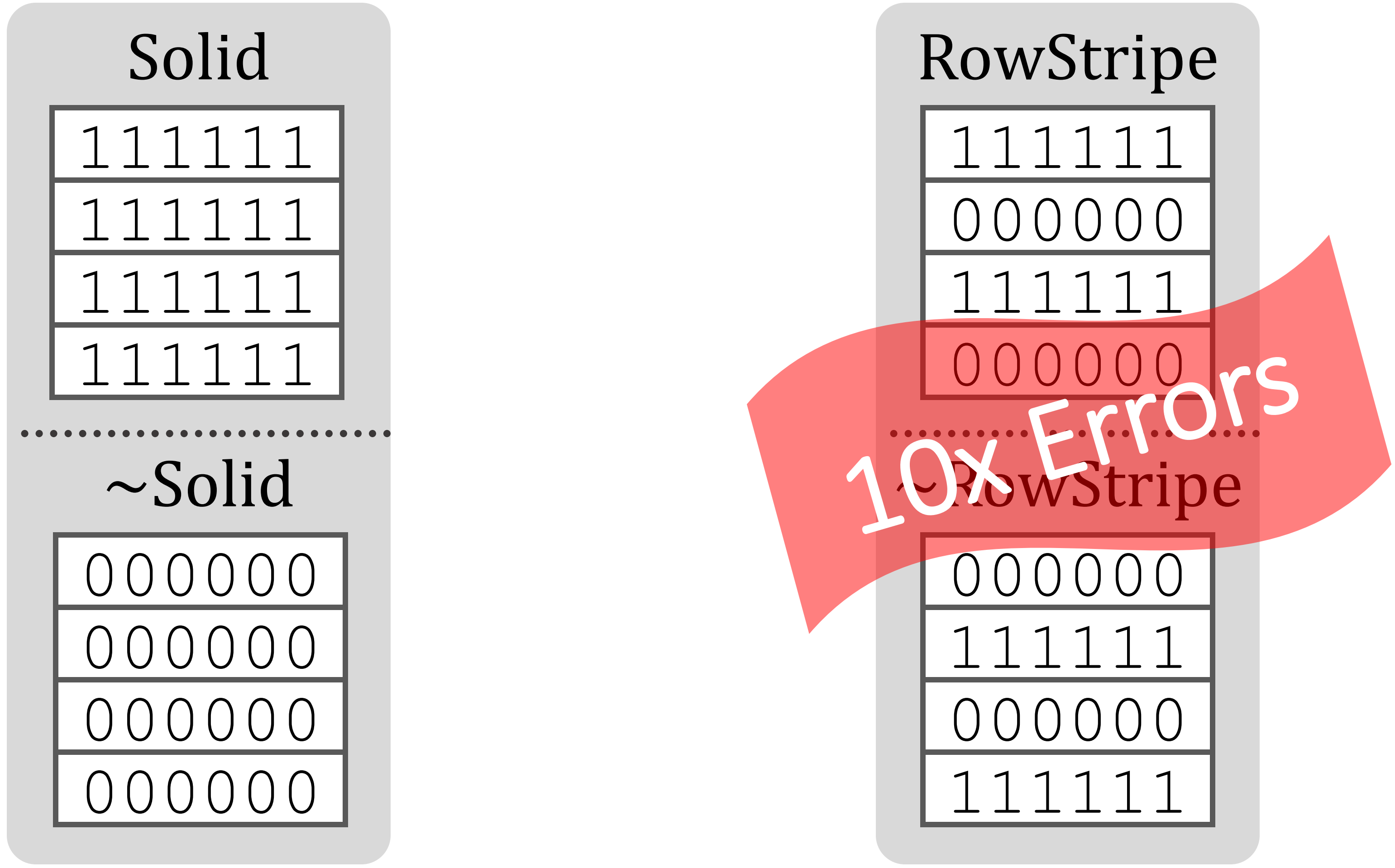
❷ Refresh Interval



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

More frequent refreshes → Fewer errors

❸ Data Pattern



Solid

111111
111111
111111
111111

~Solid

000000
000000
000000
000000

RowStripe

111111
000000
111111
000000

~RowStripe

000000
111111
000000
111111

10x Errors

Errors affected by data stored in other cells

6. Other Results (in Paper)

- *Victim Cells ≠ Weak Cells (i.e., leaky cells)*
  - Almost no overlap between them

- *Errors not strongly affected by temperature*
  - Default temperature: $50°\mathrm{C}$
  - At $30°\mathrm{C}$ and $70°\mathrm{C}$, 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

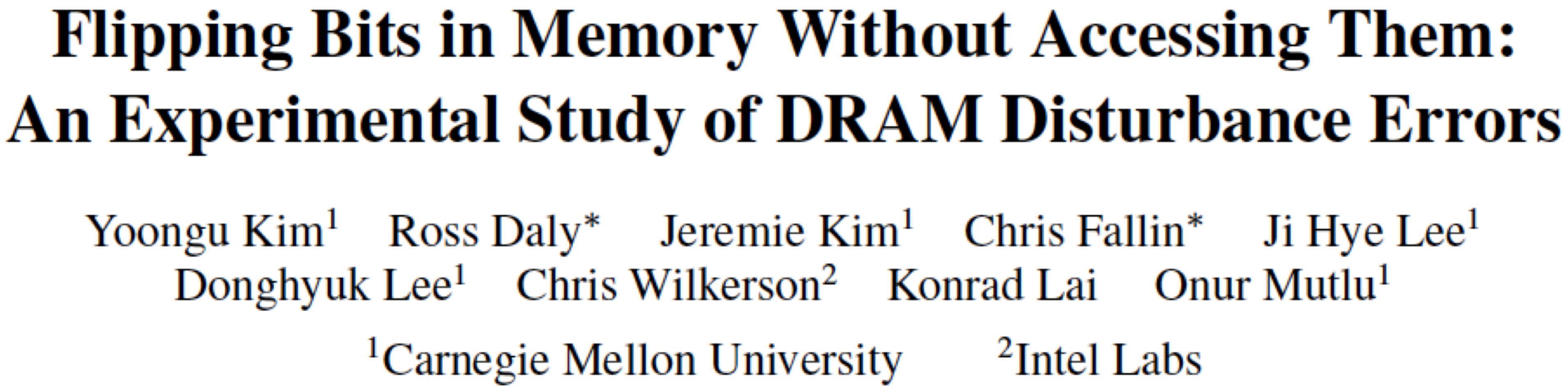
- *As many as 4 errors per cache-line*
  - Simple ECC (e.g., SECDED) cannot prevent all errors

- *Number of cells & rows affected by aggressor*
  - Victims cells per aggressor: ≤ 110
  - Victims rows per aggressor: ≤ 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

More on 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"
Proceedings of the 41st International Symposium on Computer Architecture (***ISCA***), Minneapolis, MN, June 2014.
[Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)] [Source Code and Data]



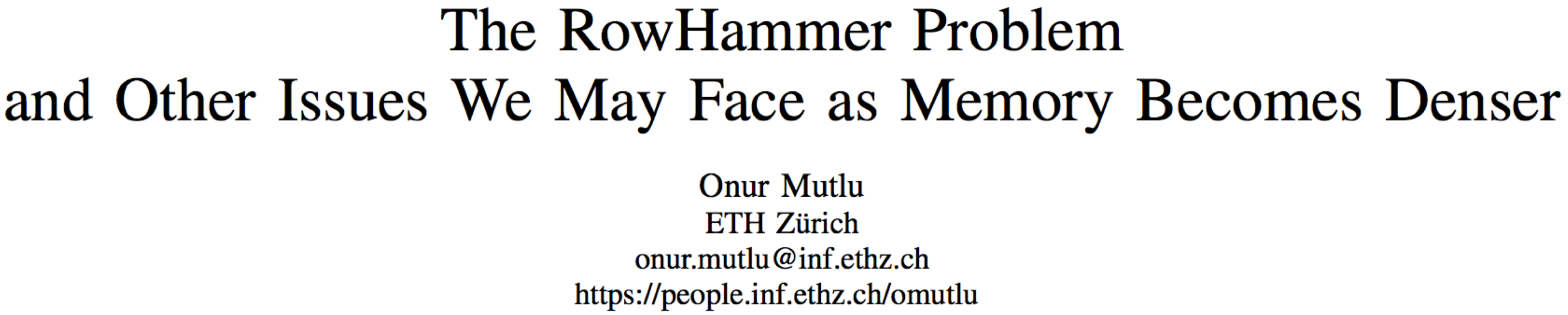
Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors

Yoongu Kim[1] Ross Daly* Jeremie Kim[1] Chris Fallin* Ji Hye Lee[1]
Donghyuk Lee[1] Chris Wilkerson[2] Konrad Lai Onur Mutlu[1]

[1]Carnegie Mellon University [2]Intel Labs

Retrospective on RowHammer & Future

- Onur Mutlu,
  "The RowHammer Problem and Other Issues We May Face as Memory Becomes Denser"
  *Invited Paper in Proceedings of the Design, Automation, and Test in Europe Conference (**DATE**)*, Lausanne, Switzerland, March 2017.
  [Slides (pptx) (pdf)]



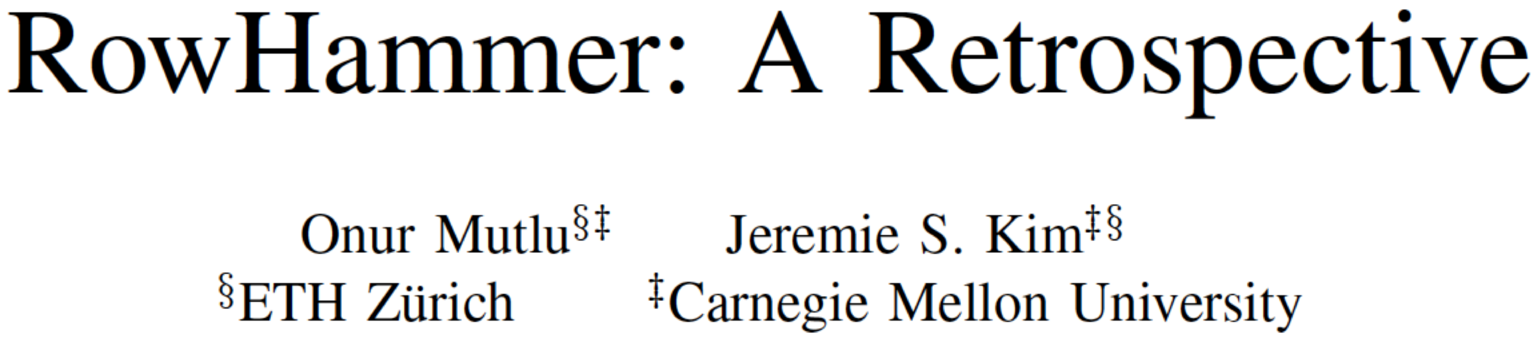
The RowHammer Problem
and Other Issues We May Face as Memory Becomes Denser

Onur Mutlu
ETH Zürich
onur.mutlu@inf.ethz.ch
https://people.inf.ethz.ch/omutlu

https://people.inf.ethz.ch/omutlu/pub/rowhammer-and-other-memory-issues_date17.pdf

A More Recent RowHammer Retrospective

- Onur Mutlu and Jeremie Kim,
  "RowHammer: A Retrospective"
  *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (**TCAD**) Special Issue on Top Picks in Hardware and Embedded Security*, 2019.
  [Preliminary arXiv version]



RowHammer: A Retrospective

Onur Mutlu[§‡] Jeremie S. Kim[‡§]
[§]ETH Zürich [‡]Carnegie Mellon University

RowHammer Solutions

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

• Make better DRAM chips	Cost
• Refresh frequently	Power, Performance
• Sophisticated ECC	Cost, Power
• Access counters	Cost, Power, Complexity

Naive Solutions

1 Throttle accesses to same row

- Limit access-interval: $\geq 500\text{ns}$
- Limit number of accesses: $\leq 128\text{K}$ $(=64\text{ms}/500\text{ns})$

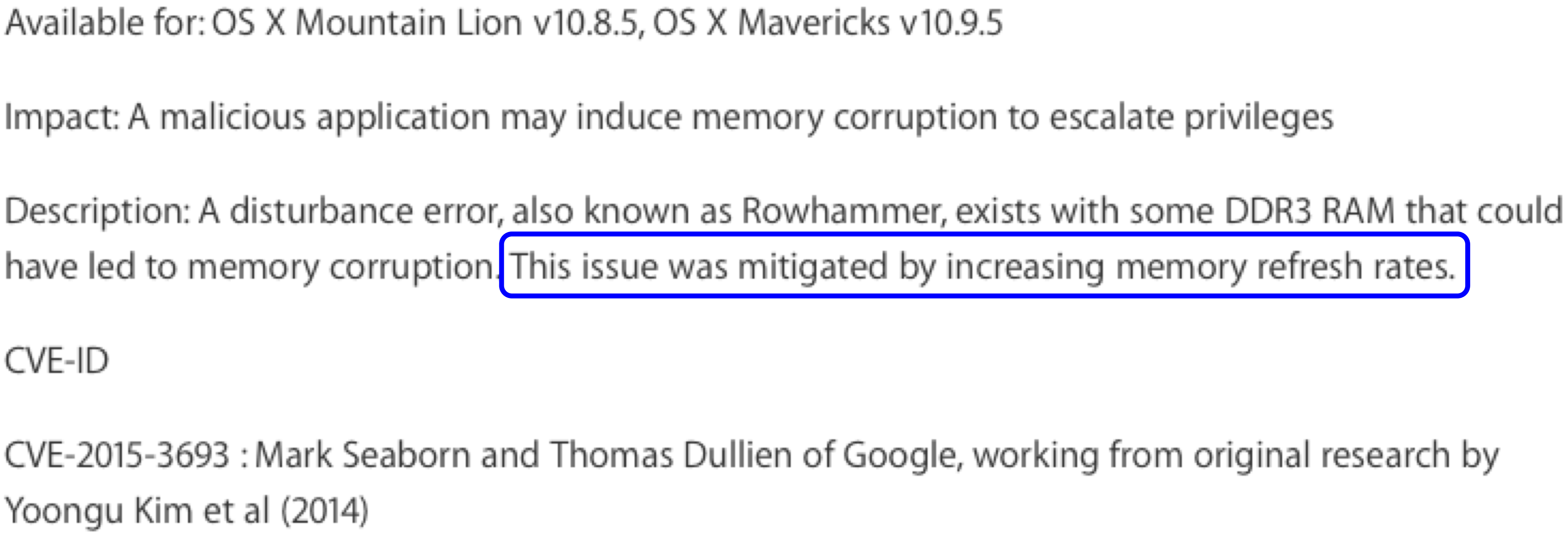
2 Refresh more frequently

- Shorten refresh-interval by $\sim 7\text{x}$

Both naive solutions introduce significant overhead in performance and power

Apple's Patch for RowHammer

- https://support.apple.com/en-gb/HT204934



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 other vendors released similar patches

Our 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: 9.4×10^{-14}
  - By adjusting the value of p, we can vary the strength of protection against errors

Advantages of PARA

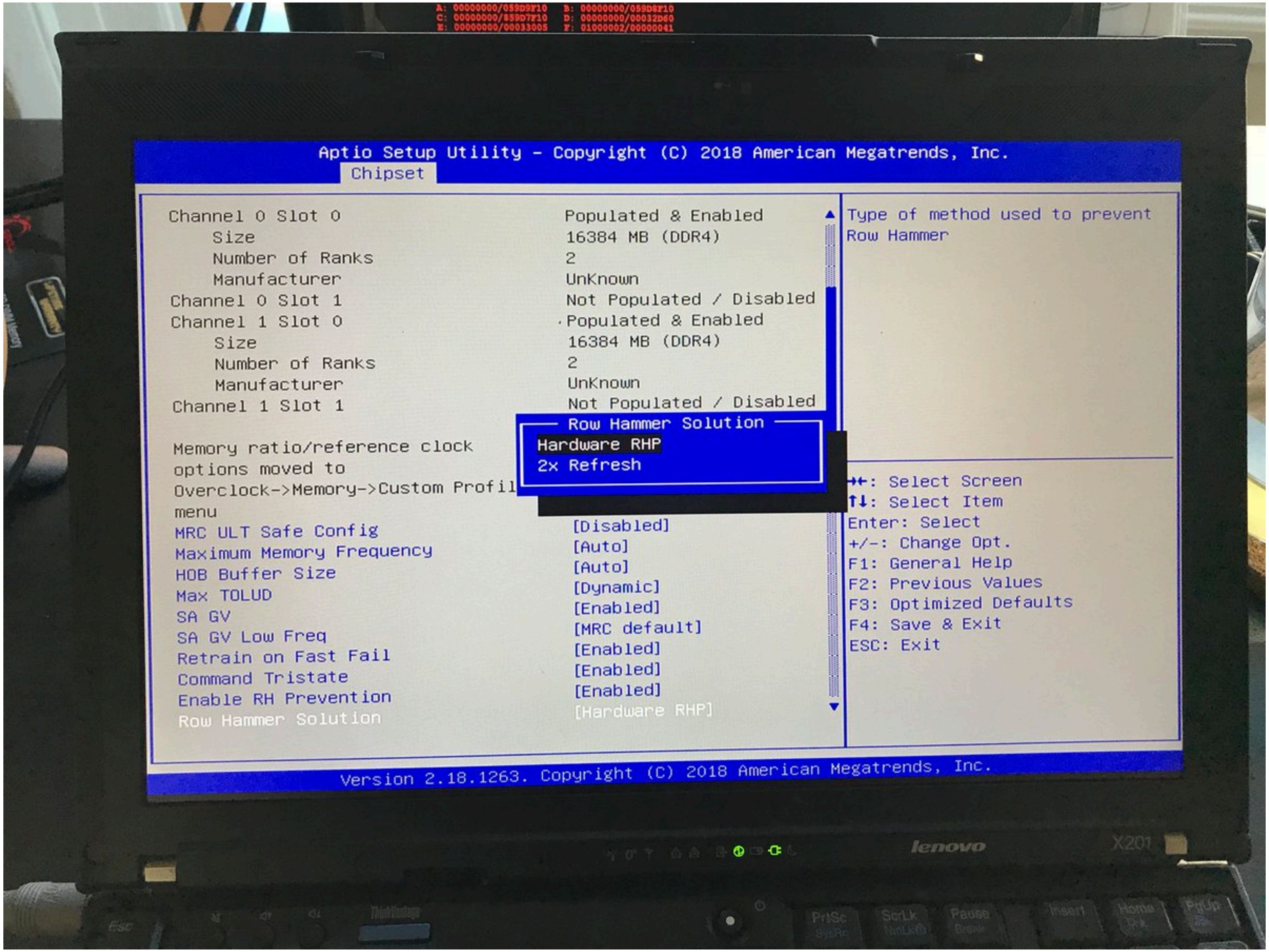
- *PARA refreshes rows infrequently*
  - Low power
  - Low performance-overhead
    - Average slowdown: 0.20% (for 29 benchmarks)
    - Maximum slowdown: 0.75%
- *PARA is stateless*
  - Low cost
  - Low complexity

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

Requirements for PARA

- If implemented in DRAM chip
  - Enough slack in timing parameters
  - Plenty of slack today:
    - Lee et al., “Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common Case,” HPCA 2015.
    - 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.
    - Ghose et al., “What Your DRAM Power Models Are Not Telling You: Lessons from a Detailed Experimental Study,” SIGMETRICS 2018.
- 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)

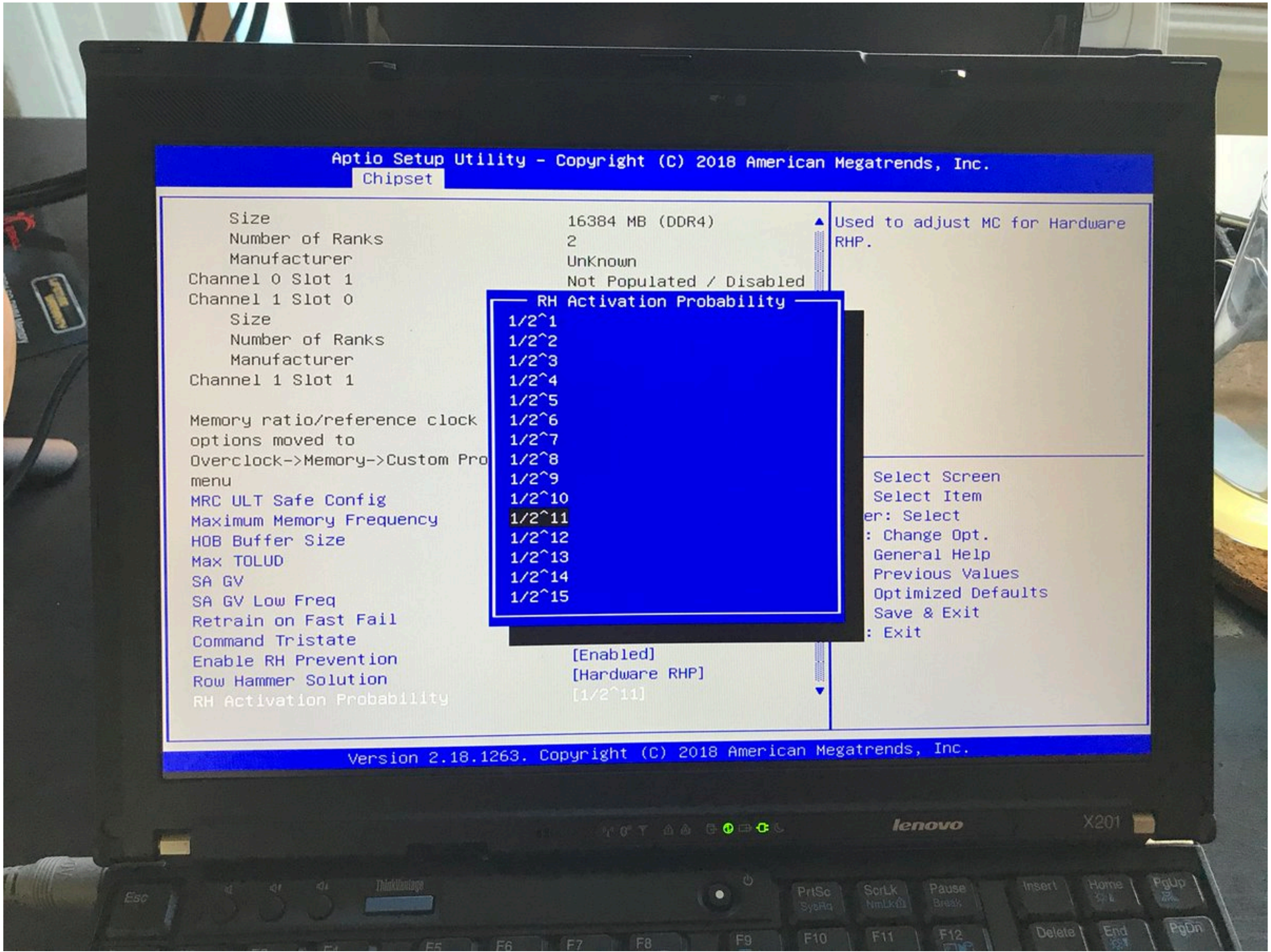


```
A: 00000000/059D9F10   B: 00000000/059D8F10
C: 00000000/859D7F10   D: 00000000/00032D60
E: 00000000/00033005   F: 01000002/00000041
```

```
          Aptio Setup Utility - Copyright (C) 2018 American Megatrends, Inc.
             Chipset

 Channel 0 Slot 0                        Populated & Enabled       Type of method used to prevent
     Size                                16384 MB (DDR4)           Row Hammer
     Number of Ranks                     2
     Manufacturer                        UnKnown
 Channel 0 Slot 1                        Not Populated / Disabled
 Channel 1 Slot 0                        Populated & Enabled
     Size                                16384 MB (DDR4)
     Number of Ranks                     2
     Manufacturer                        UnKnown
 Channel 1 Slot 1                        Not Populated / Disabled
                                      ─ Row Hammer Solution ─
 Memory ratio/reference clock          Hardware RHP
 options moved to                      2x Refresh                  →←: Select Screen
 Overclock->Memory->Custom Profil                                  ↑↓: Select Item
 menu                                                              Enter: Select
 MRC ULT Safe Config                      [Disabled]               +/-: Change Opt.
 Maximum Memory Frequency                 [Auto]                   F1: General Help
 HOB Buffer Size                          [Auto]                   F2: Previous Values
 Max TOLUD                                [Dynamic]                F3: Optimized Defaults
 SA GV                                    [Enabled]                F4: Save & Exit
 SA GV Low Freq                           [MRC default]            ESC: Exit
 Retrain on Fast Fail                     [Enabled]
 Command Tristate                         [Enabled]
 Enable RH Prevention                     [Enabled]
 Row Hammer Solution                      [Hardware RHP]

          Version 2.18.1263. Copyright (C) 2018 American Megatrends, Inc.
```

https://twitter.com/isislovecruft/status/1021939922754723841

Probabilistic Activation in Real Life (II)

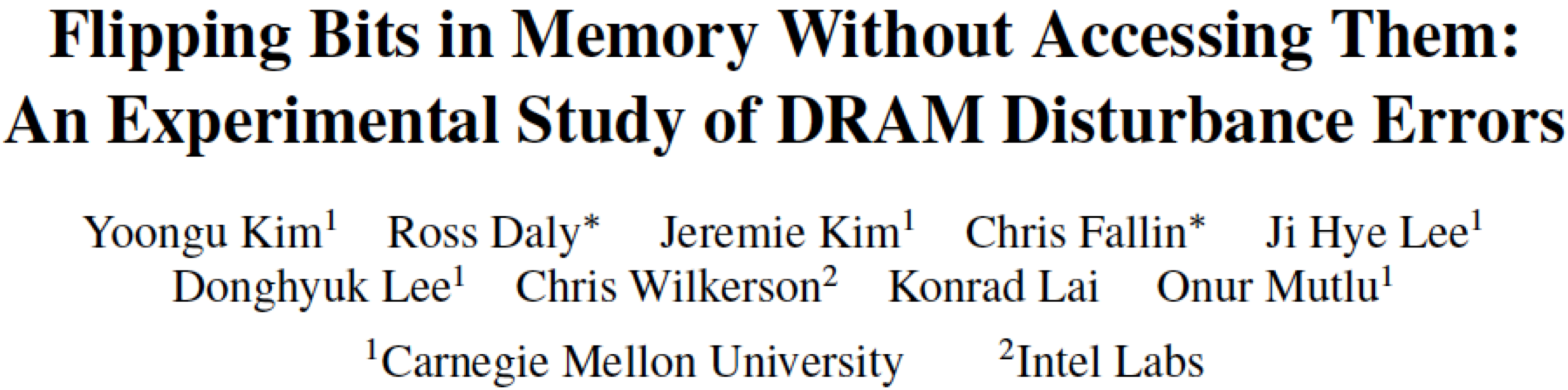


```
Aptio Setup Utility - Copyright (C) 2018 American Megatrends, Inc.
             Chipset

    Size                              16384 MB (DDR4)          Used to adjust MC for Hardware
    Number of Ranks                   2                        RHP.
    Manufacturer                      UnKnown
Channel 0 Slot 1                      Not Populated / Disabled
Channel 1 Slot 0               ┌ RH Activation Probability ┐
    Size                       │ 1/2^1                     │
    Number of Ranks            │ 1/2^2                     │
    Manufacturer               │ 1/2^3                     │
Channel 1 Slot 1               │ 1/2^4                     │
                               │ 1/2^5                     │
Memory ratio/reference clock   │ 1/2^6                     │
options moved to               │ 1/2^7                     │
Overclock->Memory->Custom Pro  │ 1/2^8                     │
menu                           │ 1/2^9                     │   Select Screen
MRC ULT Safe Config            │ 1/2^10                    │   Select Item
Maximum Memory Frequency       │ 1/2^11                    │ er: Select
HOB Buffer Size                │ 1/2^12                    │ : Change Opt.
Max TOLUD                      │ 1/2^13                    │   General Help
SA GV                          │ 1/2^14                    │   Previous Values
SA GV Low Freq                 │ 1/2^15                    │   Optimized Defaults
Retrain on Fast Fail           └───────────────────────────┘   Save & Exit
Command Tristate                                             : Exit
Enable RH Prevention                  [Enabled]
Row Hammer Solution                   [Hardware RHP]
RH Activation Probability             [1/2^11]

       Version 2.18.1263. Copyright (C) 2018 American Megatrends, Inc.
```

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"
Proceedings of the 41st International Symposium on Computer Architecture (***ISCA***), Minneapolis, MN, June 2014.
[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 Kim[1] Ross Daly* Jeremie Kim[1] Chris Fallin* Ji Hye Lee[1]
Donghyuk Lee[1] Chris Wilkerson[2] Konrad Lai Onur Mutlu[1]

[1]Carnegie Mellon University [2]Intel Labs

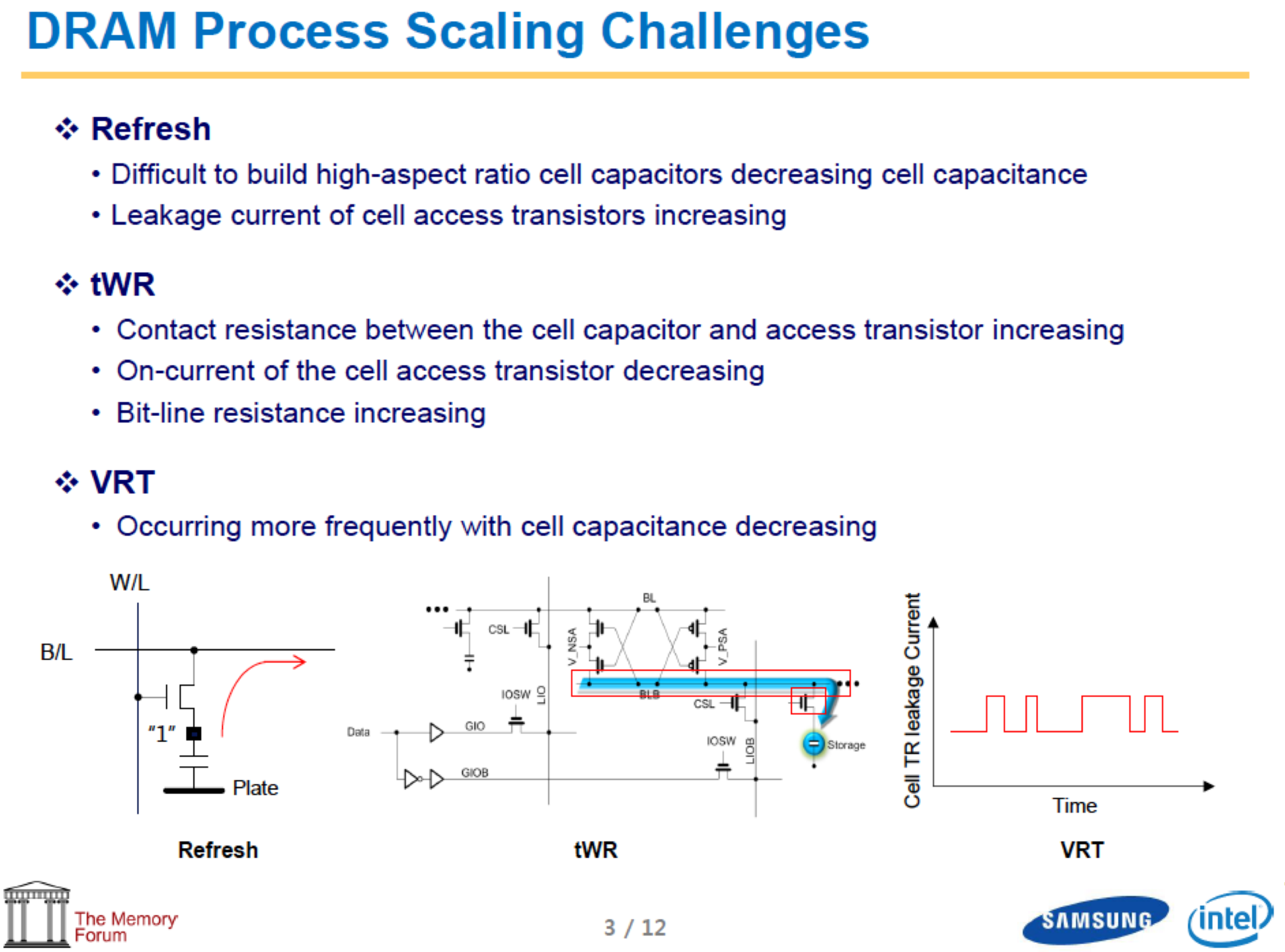
A Takeaway

Main Memory Needs

Intelligent Controllers

for Security

Industry Is Writing Papers About It, Too

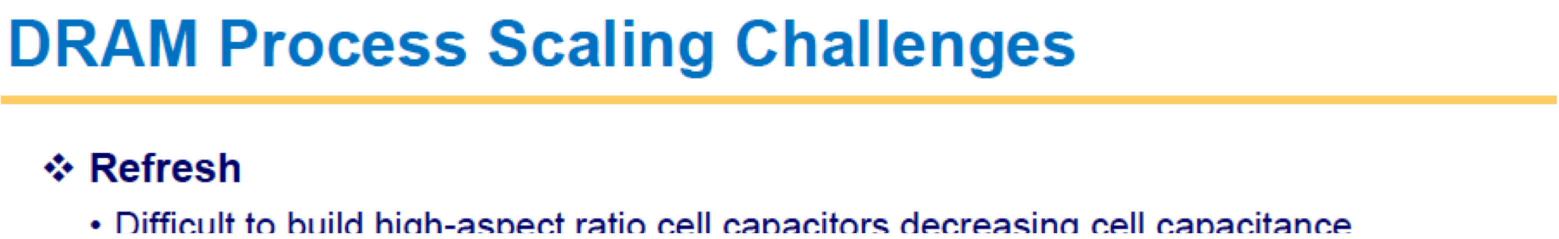


DRAM Process Scaling Challenges

- **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

Refresh **tWR** **VRT**

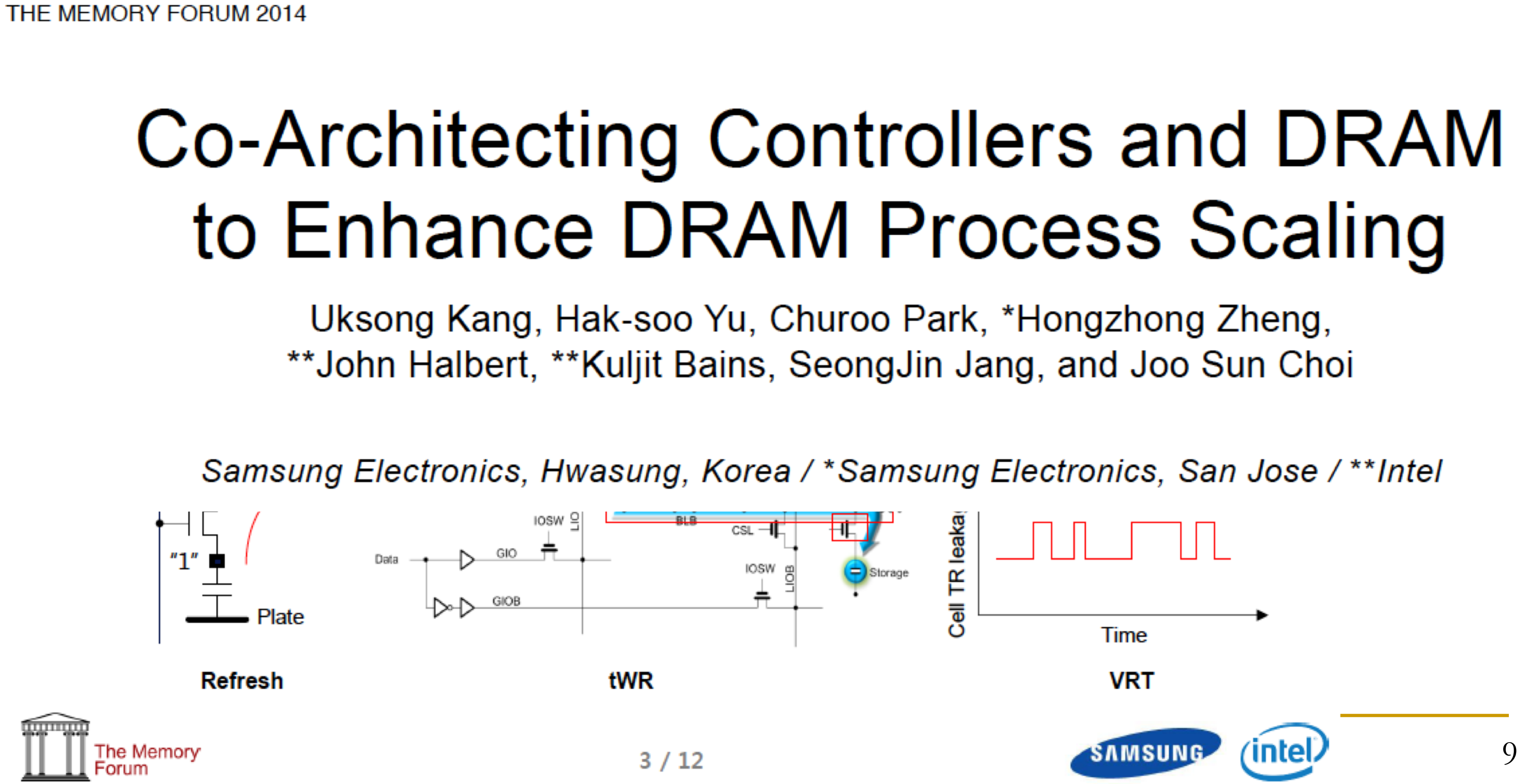
Call for Intelligent Memory Controllers



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*

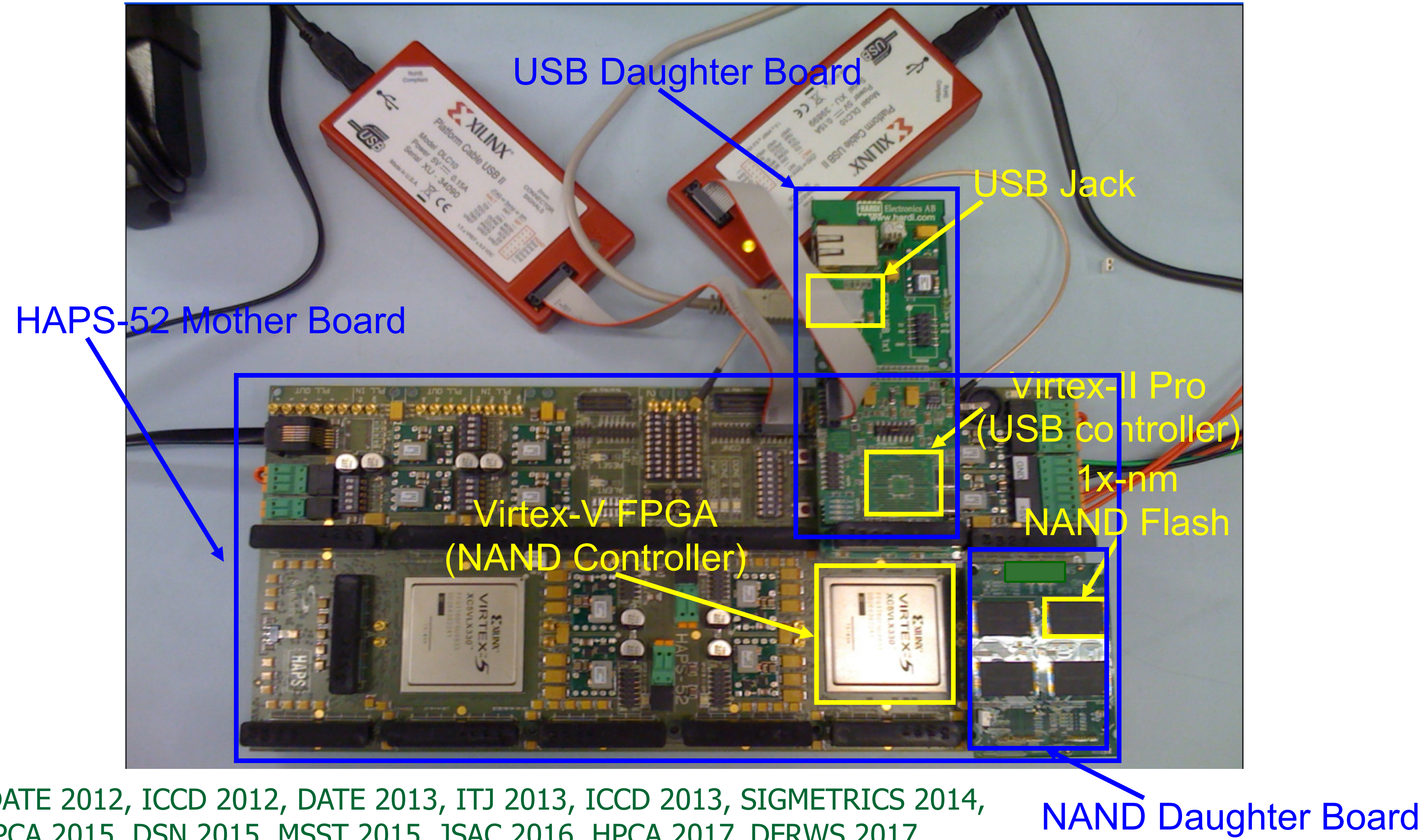
"1"
Plate
IOSW
LIO
BLB
CSL
Data
GIO
GIOB
IOSW
LIOB
Storage
Cell TR leaka
Time

Refresh **tWR** **VRT**

The Memory Forum

3 / 12

Aside: Intelligent Controller for NAND Flash

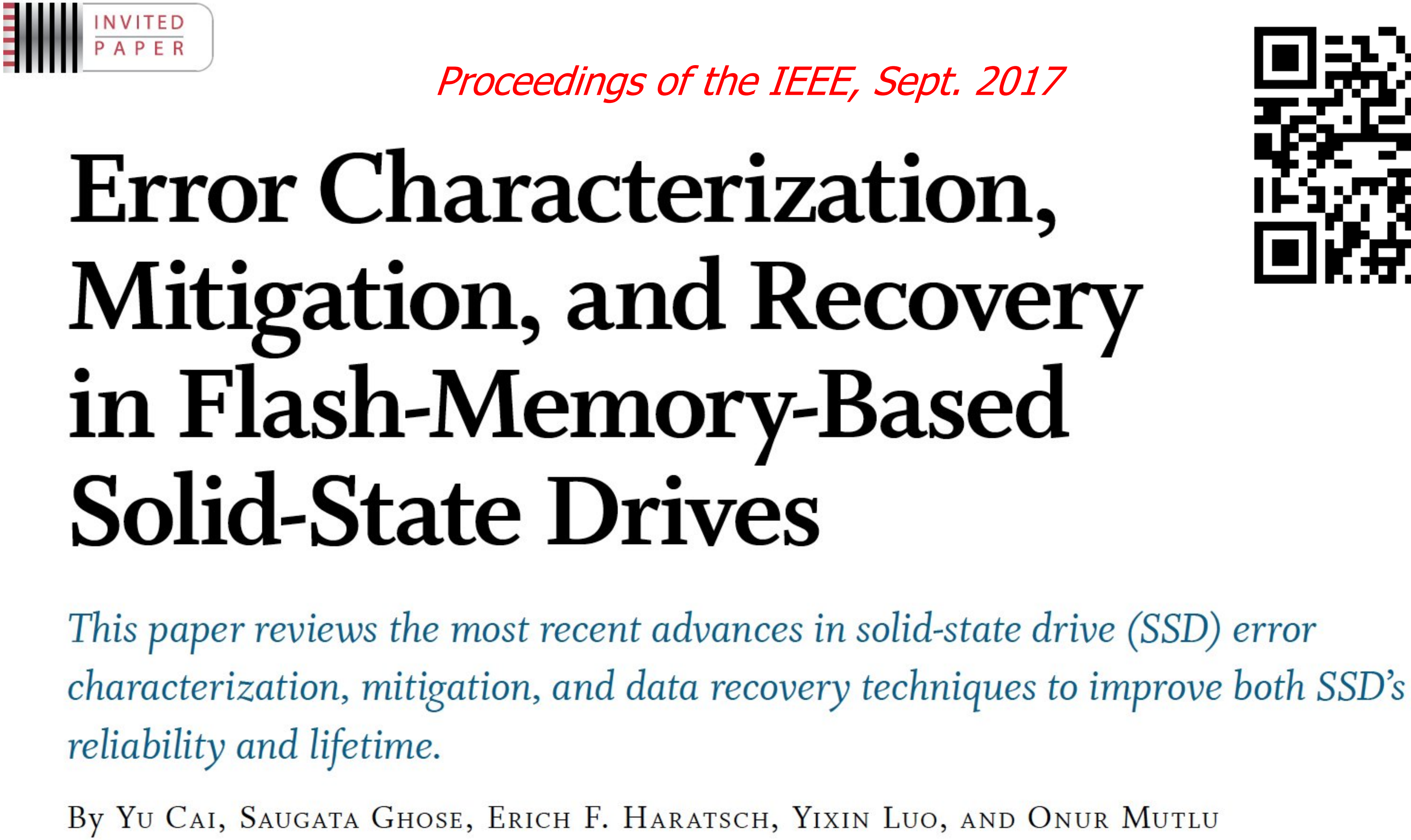


[DATE 2012, ICCD 2012, DATE 2013, ITJ 2013, ICCD 2013, SIGMETRICS 2014, HPCA 2015, DSN 2015, MSST 2015, JSAC 2016, HPCA 2017, DFRWS 2017, PIEEE 2017, HPCA 2018, SIGMETRICS 2018]

Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017.

Aside: Intelligent Controller for NAND Flash

Proceedings of the IEEE, Sept. 2017



INVITED PAPER

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

A Key Takeaway

Main Memory Needs

Intelligent Controllers

Future Memory Reliability/Security Challenges

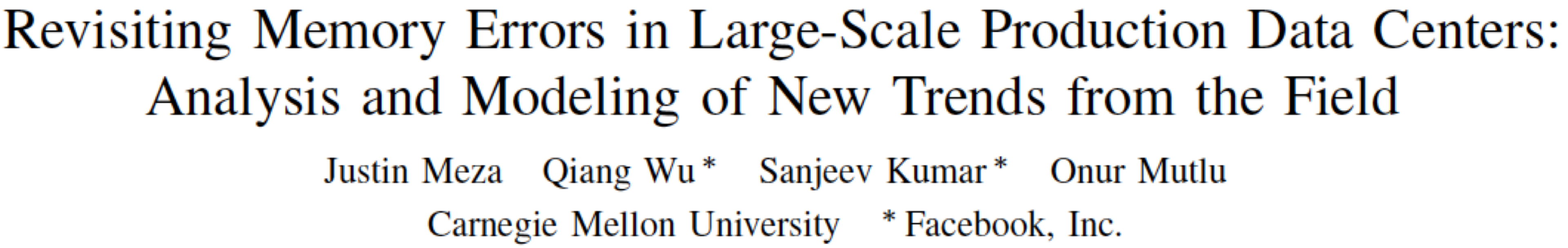
Future of Main Memory

- DRAM is becoming less reliable → more vulnerable

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]

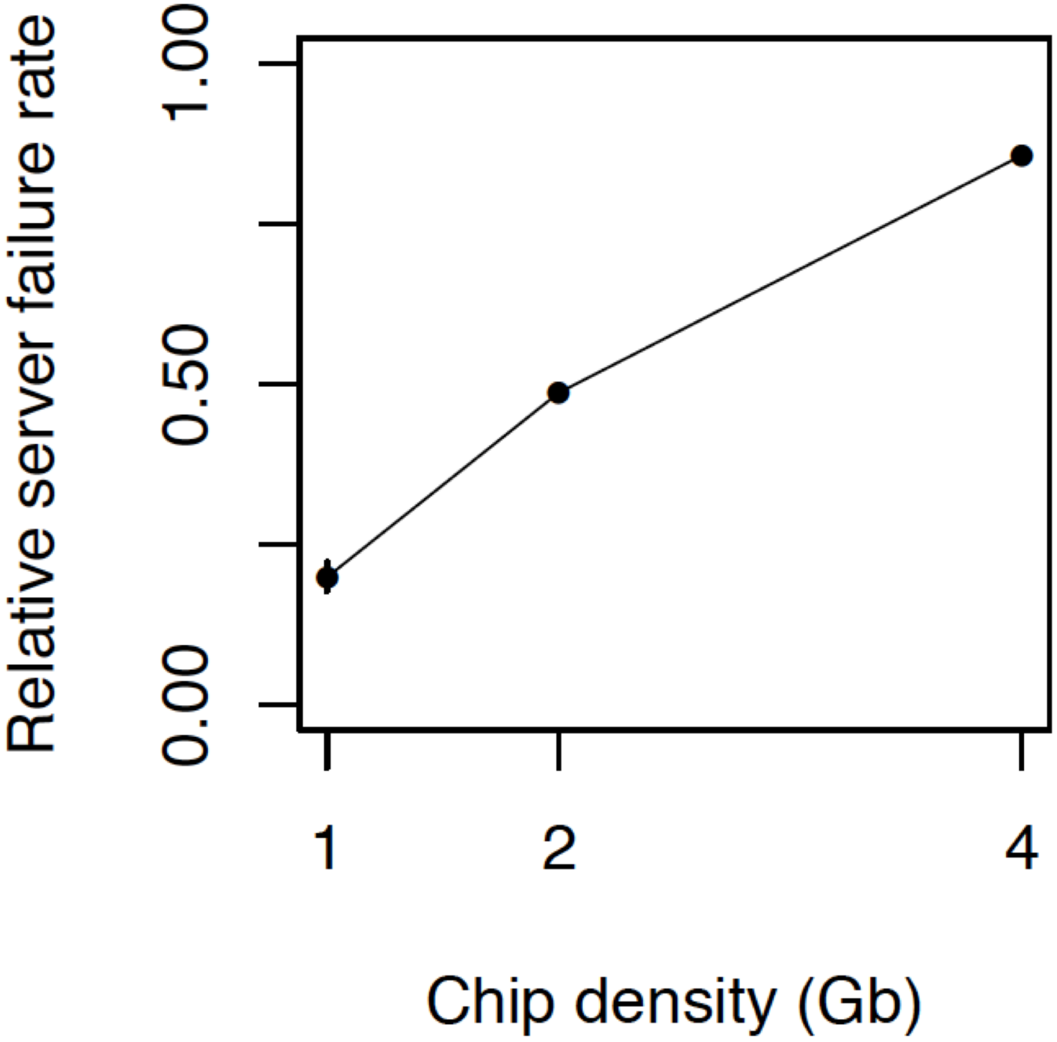


Revisiting Memory Errors in Large-Scale Production Data Centers: Analysis and Modeling of New Trends from the Field

Justin Meza Qiang Wu * Sanjeev Kumar * Onur Mutlu

Carnegie Mellon University * Facebook, Inc.

DRAM Reliability Reducing



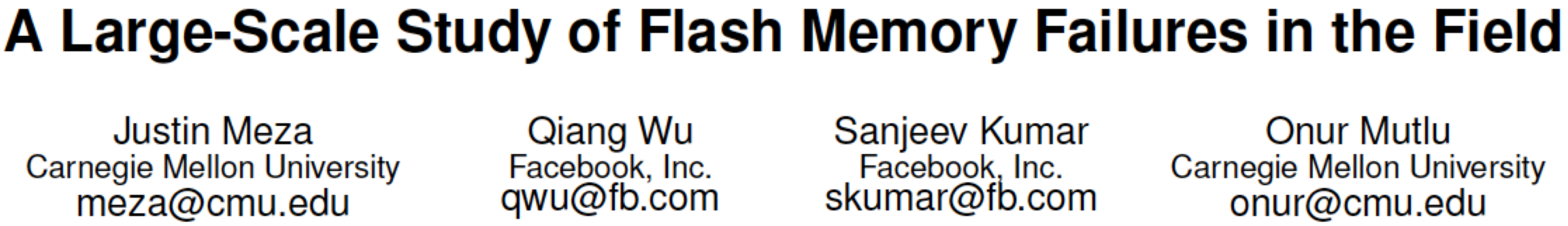
Intuition: quadratic increase in capacity

Meza+, "Revisiting Memory Errors in Large-Scale Production Data Centers," DSN'15.

Aside: SSD Error Analysis in the Field

- First large-scale field study of flash memory errors

- Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu,
"A Large-Scale Study of Flash Memory Errors in the Field"
Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (***SIGMETRICS***), Portland, OR, June 2015.
[Slides (pptx) (pdf)] [Coverage at ZDNet]



A Large-Scale Study of Flash Memory Failures in the Field

Justin Meza
Carnegie Mellon University
meza@cmu.edu

Qiang Wu
Facebook, Inc.
qwu@fb.com

Sanjeev Kumar
Facebook, Inc.
skumar@fb.com

Onur Mutlu
Carnegie Mellon University
onur@cmu.edu

Future of Main Memory

- 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

DRAM Data Retention Time Failures

- Determining the data retention time of a cell/row is getting more difficult

- Retention failures may already be slipping into the field

Keeping Future Memory Secure

How Do We Keep Memory Secure?

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

Challenge and Opportunity for Future

Fundamentally

Secure, Reliable, Safe

Computing Architectures

Solution Direction: Principled Designs

Design fundamentally secure computing architectures

Predict and prevent such safety issues

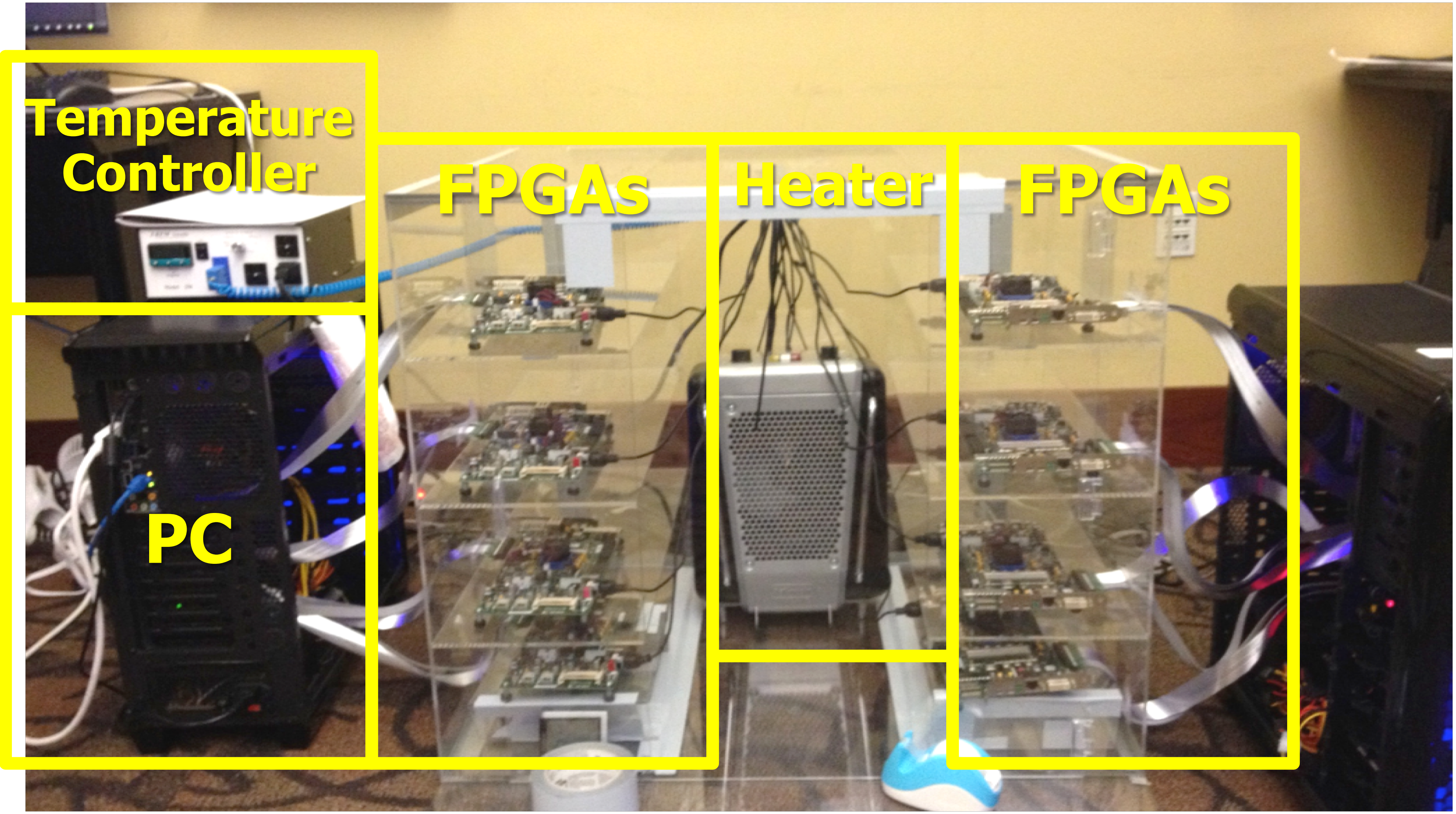
Architecting for Security

- **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

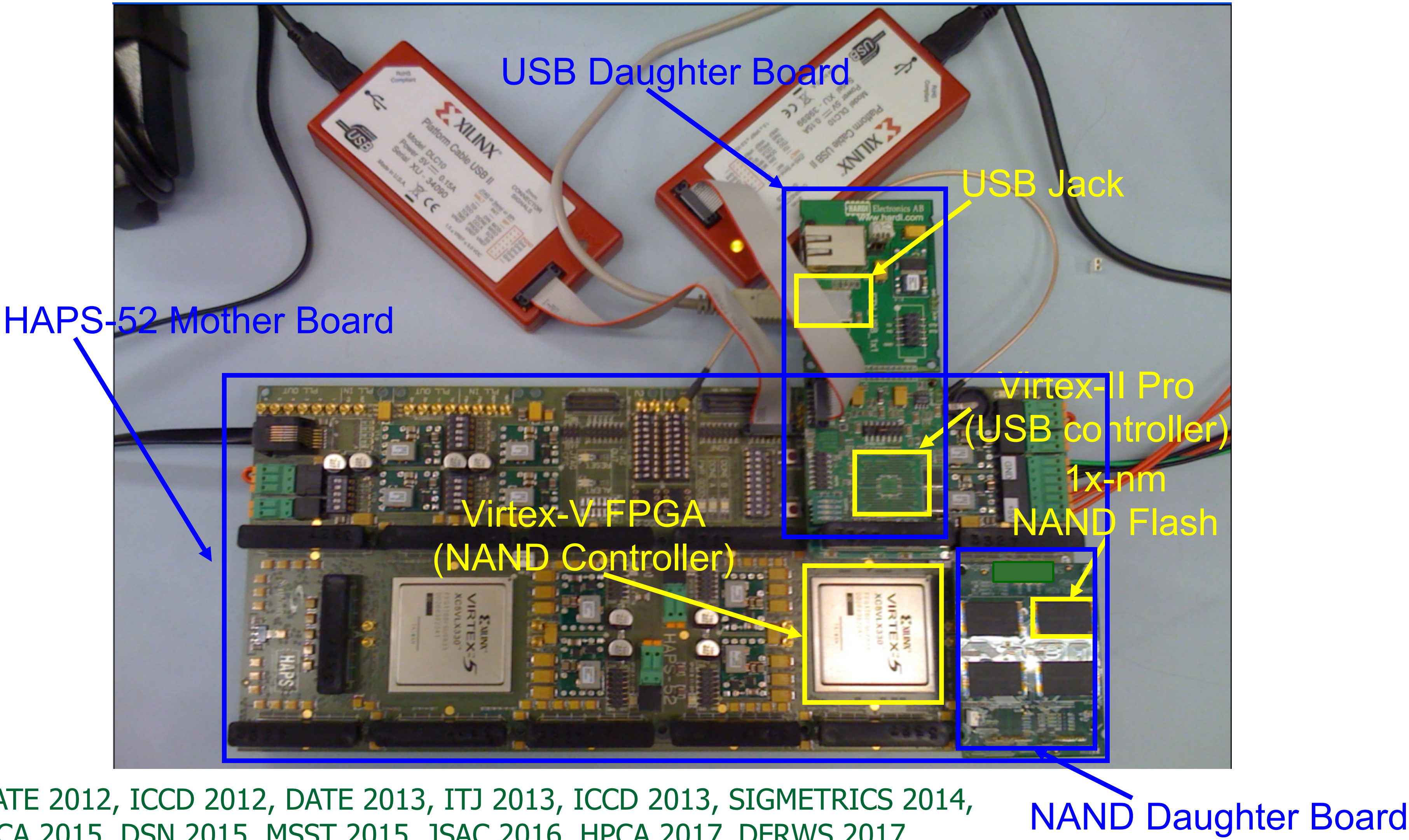
- **Design & Test**: Principled design, automation, (online) testing
  - Design for security
  - High coverage and good interaction with system reliability methods

Understand and Model with Experiments (DRAM)



Kim+, "Flipping Bits in Memory Without Accessing Them: An Experimental Study of DRAM Disturbance Errors," ISCA 2014.

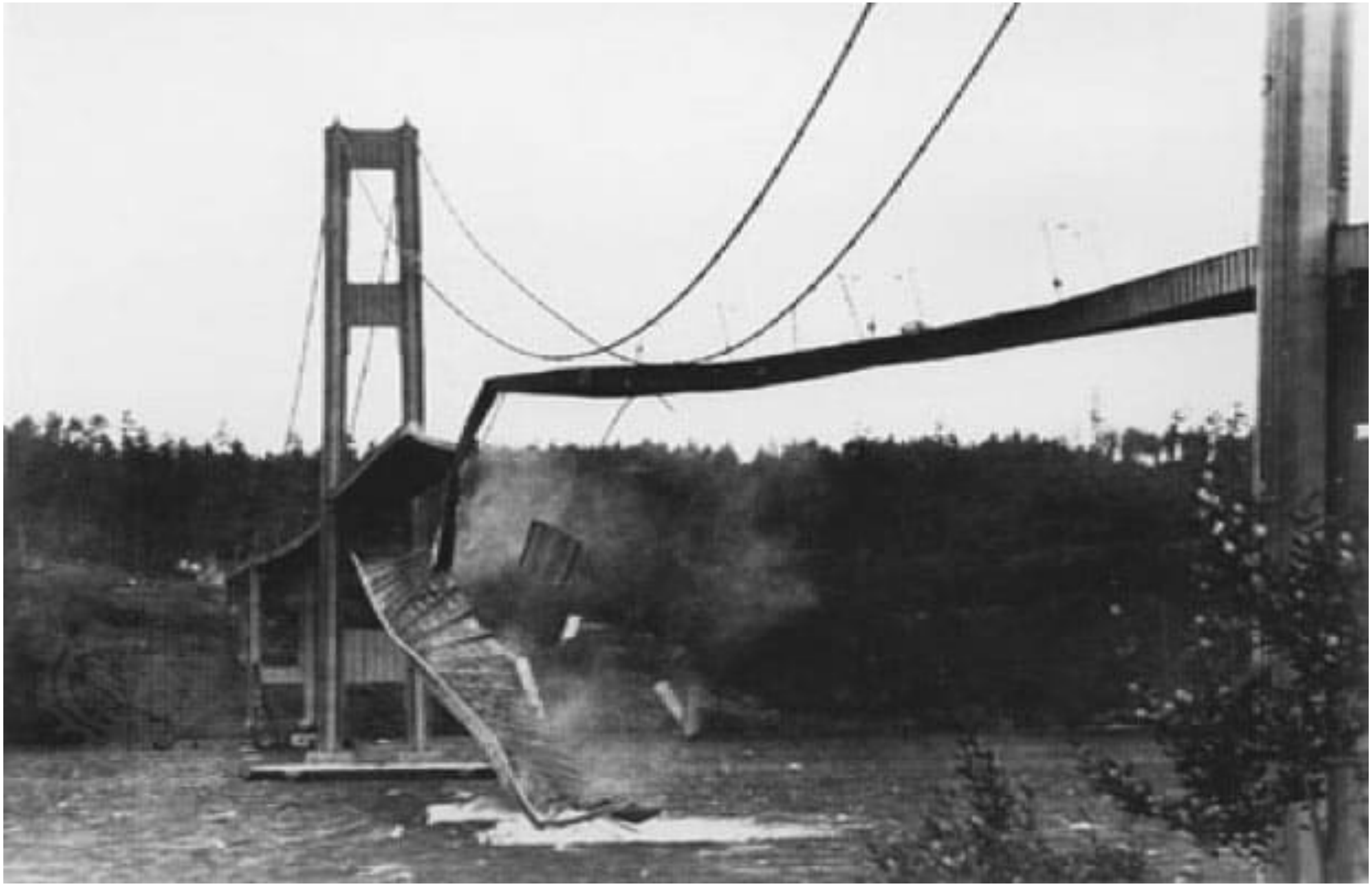
Understand and Model with Experiments (Flash)



[DATE 2012, ICCD 2012, DATE 2013, ITJ 2013, ICCD 2013, SIGMETRICS 2014, HPCA 2015, DSN 2015, MSST 2015, JSAC 2016, HPCA 2017, DFRWS 2017, PIEEE 2017, HPCA 2018, SIGMETRICS 2018]

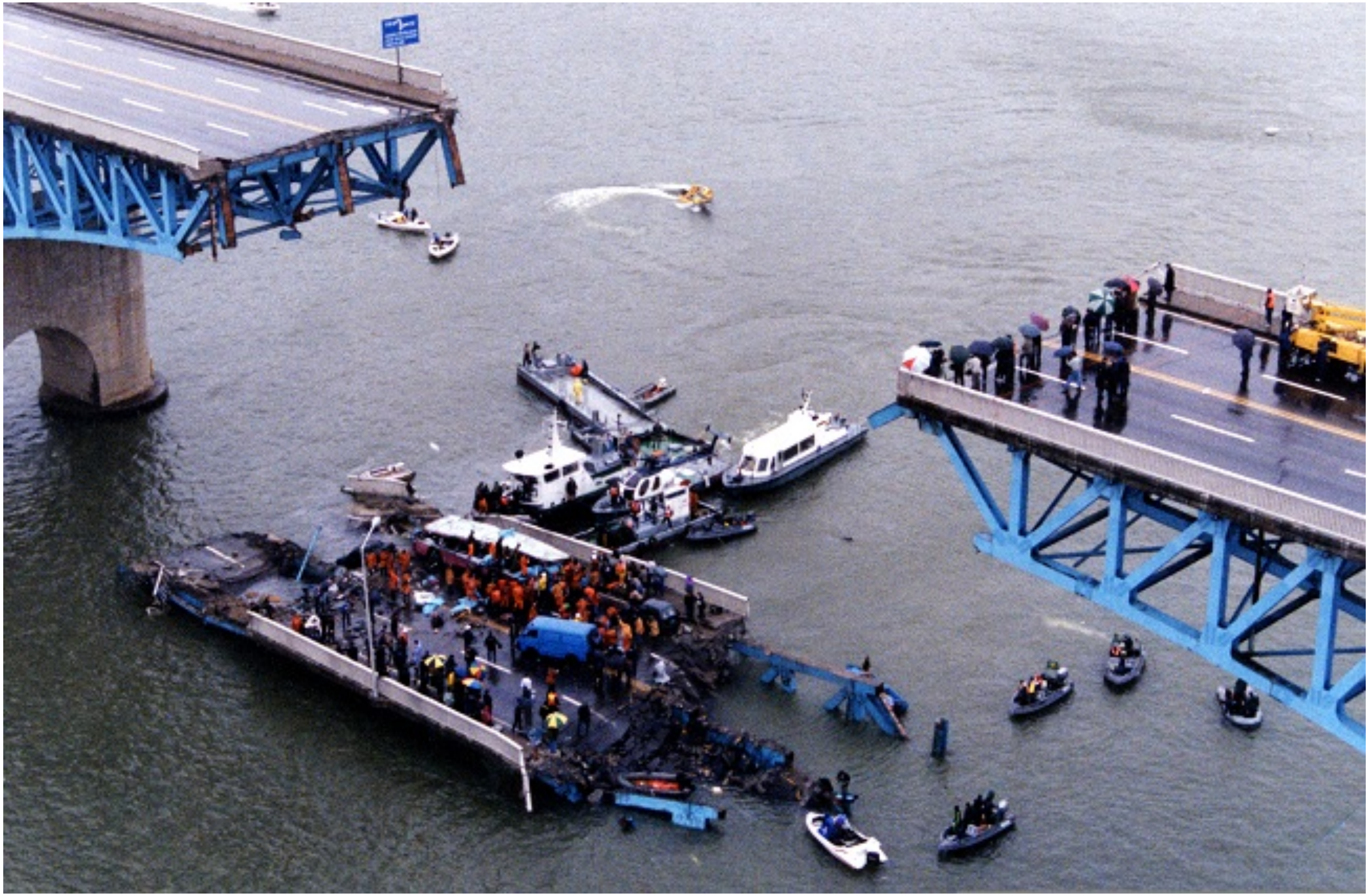
Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017.

Recall: Collapse of the “Galloping Gertie”



Source: AP
http://www.wsdot.wa.gov/tnbhistory/connections/connections3.htm

Another Example (1994)

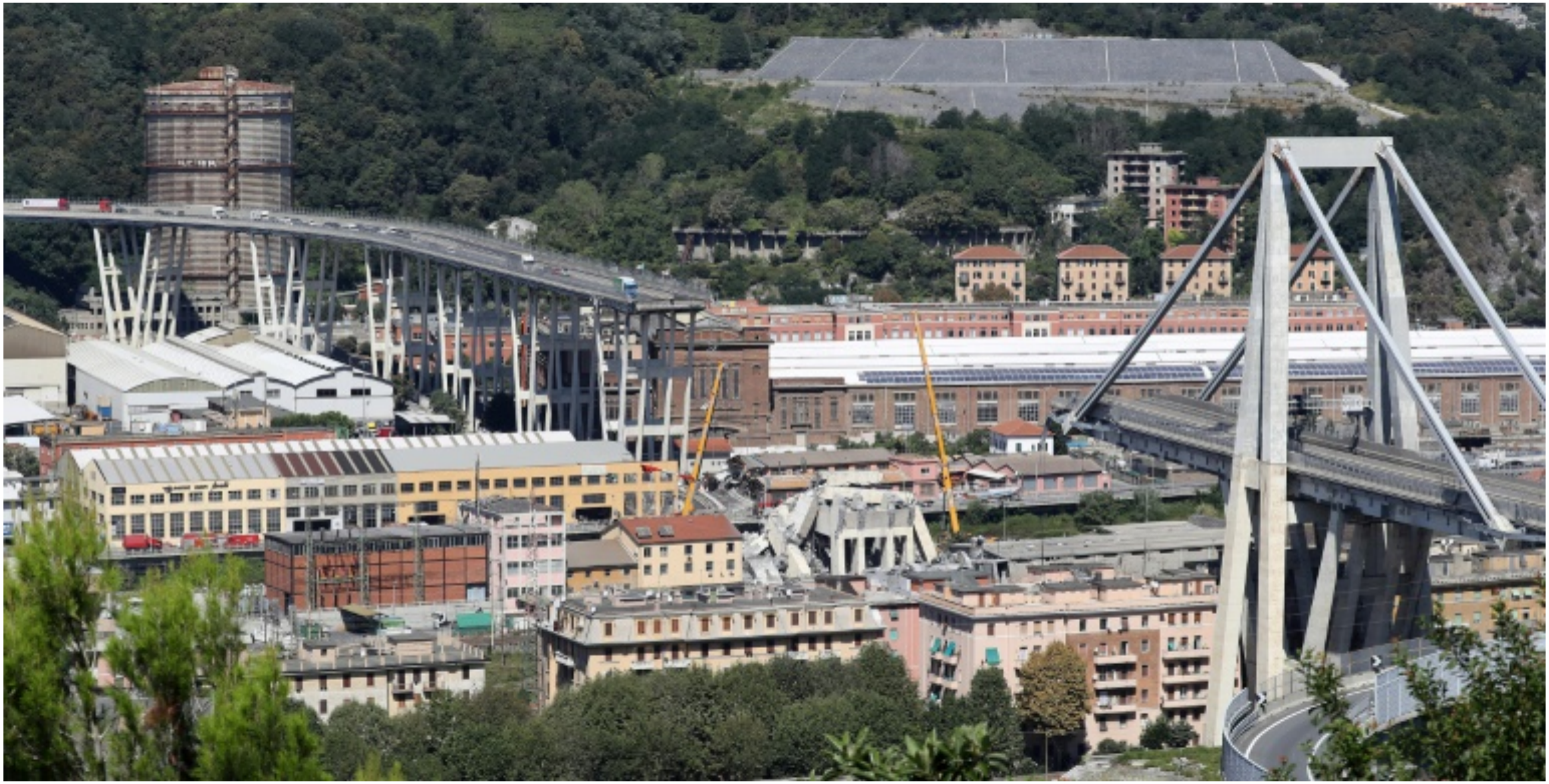


Source: By 최광모 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=35197984

Yet Another Example (2007)

Source: Morry Gash/AP, https://www.npr.org/2017/08/01/540669701/10-years-after-bridge-collapse-america-is-still-crumbling?t=1535427165809

A More Recent Example (2018)



Source: AFP / Valery HACHE, https://www.capitalfm.co.ke/news/2018/08/genoa-bridge-collapse-what-we-know/

The Takeaway, Again

In-Field Patch-ability

(Intelligent Memory)

Can Avoid Such Failures

Final Thoughts on RowHammer

Some Thoughts on RowHammer

- A simple hardware failure mechanism can create a widespread system security vulnerability

- How to exploit and fix the vulnerability requires a strong understanding across the transformation layers
  - And, a strong understanding of tools available to you

- Fixing needs to happen for two types of chips
  - Existing chips (already in the field)
  - Future chips
- Mechanisms for fixing are different between the two types

Aside: 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

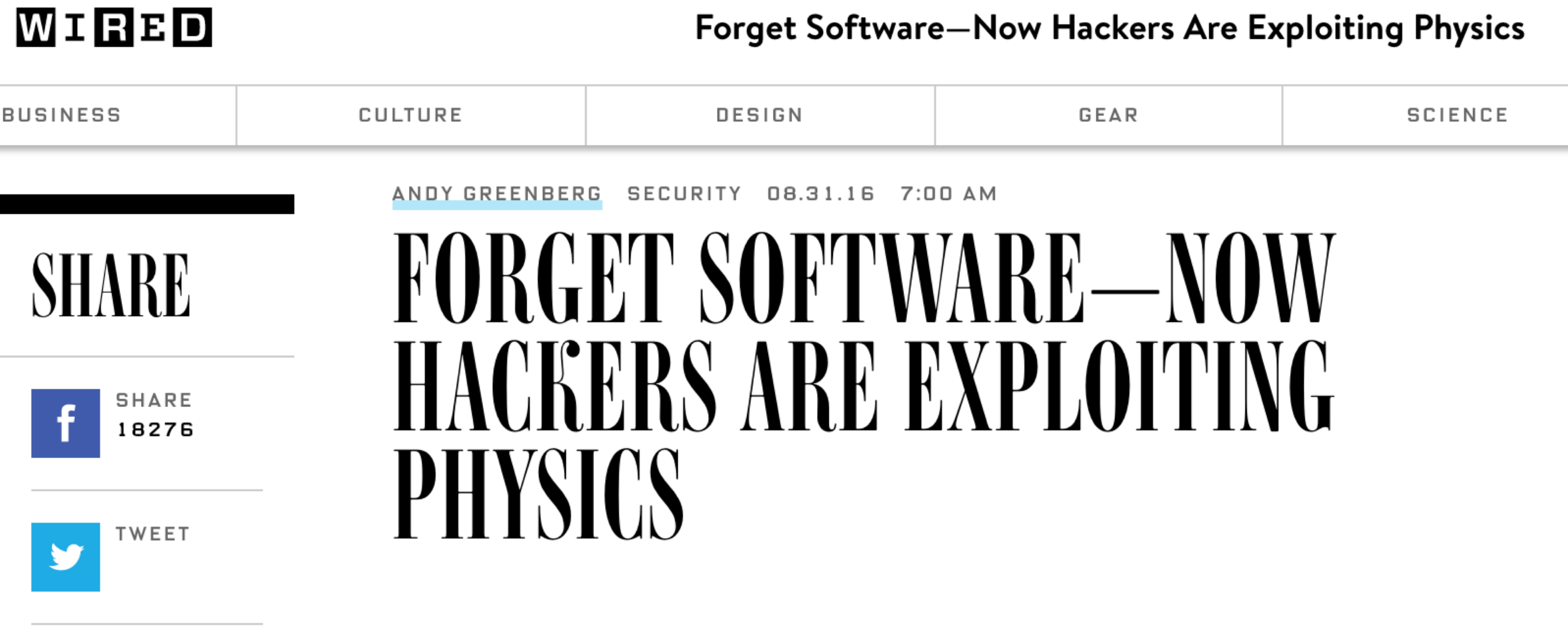
- Lamport et al., “The Byzantine Generals Problem,” ACM TOPLAS 1982.

Slide credit: Mahadev Satyanarayanan, CMU, 15-440, Spring 2015

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



WIRED

Forget Software—Now Hackers Are Exploiting Physics

ANDY GREENBERG SECURITY 08.31.16 7:00 AM

FORGET SOFTWARE—NOW HACKERS ARE EXPLOITING PHYSICS

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

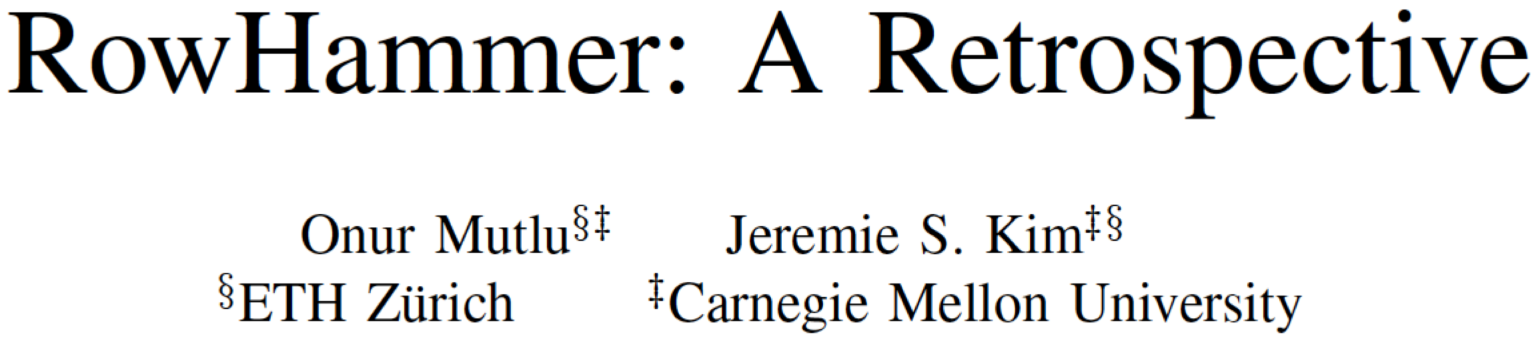
- It is no coincidence that two of the groups that discovered Meltdown and Spectre heavily worked on RowHammer attacks before
  - **More to come…**

Summary: RowHammer

- DRAM 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"
IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (***TCAD***) *Special Issue on Top Picks in Hardware and Embedded Security*, 2019.
[Preliminary arXiv version]



RowHammer: A Retrospective

Onur Mutlu[§‡] Jeremie S. Kim[‡§]
[§]ETH Zürich [‡]Carnegie Mellon University

Rowhammer

Memory Systems
and Memory-Centric Computing Systems

Part 2: RowHammer

Prof. Onur Mutlu

omutlu@gmail.com

https://people.inf.ethz.ch/omutlu

7 July 2019

SAMOS Tutorial

SAFARI ETH zürich Carnegie Mellon

Future Memory Reliability/Security Challenges

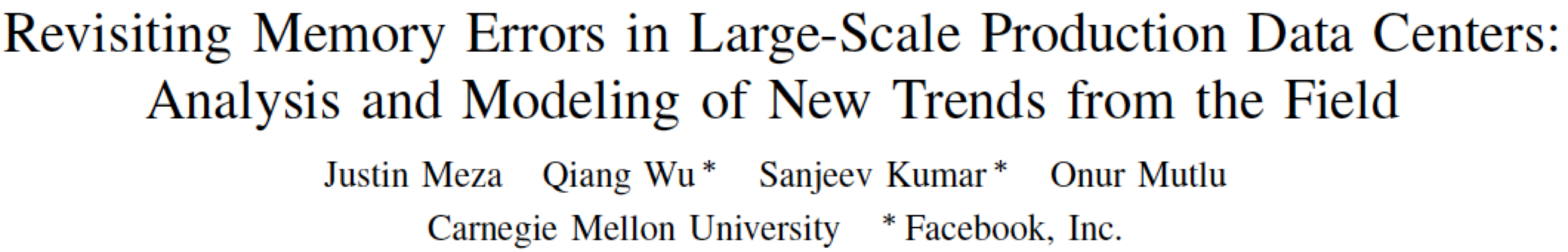
Future of Main Memory

- DRAM is becoming less reliable → more vulnerable

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]

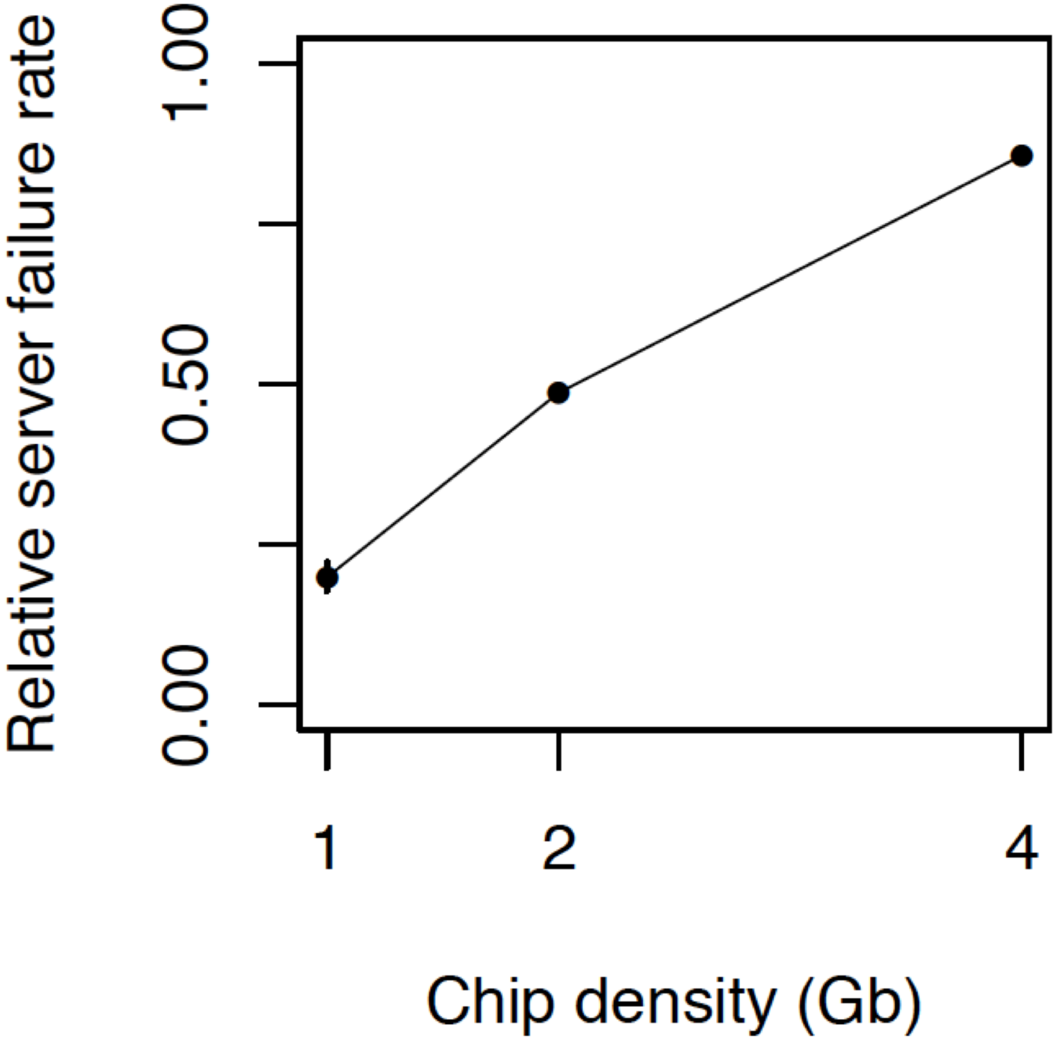


Revisiting Memory Errors in Large-Scale Production Data Centers: Analysis and Modeling of New Trends from the Field

Justin Meza Qiang Wu * Sanjeev Kumar * Onur Mutlu

Carnegie Mellon University * Facebook, Inc.

DRAM Reliability Reducing



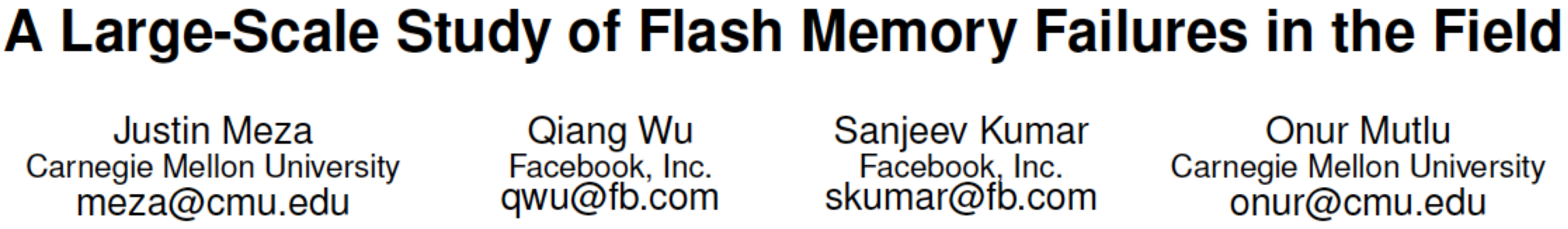
Intuition: quadratic increase in capacity

Meza+, "Revisiting Memory Errors in Large-Scale Production Data Centers," DSN'15.

Aside: SSD Error Analysis in the Field

- First large-scale field study of flash memory errors

- Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu,
"A Large-Scale Study of Flash Memory Errors in the Field"
Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems
*(**SIGMETRICS**)*, Portland, OR, June 2015.
[Slides (pptx) (pdf)] [Coverage at ZDNet]



A Large-Scale Study of Flash Memory Failures in the Field

Justin Meza
Carnegie Mellon University
meza@cmu.edu

Qiang Wu
Facebook, Inc.
qwu@fb.com

Sanjeev Kumar
Facebook, Inc.
skumar@fb.com

Onur Mutlu
Carnegie Mellon University
onur@cmu.edu

Future of Main Memory

- 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

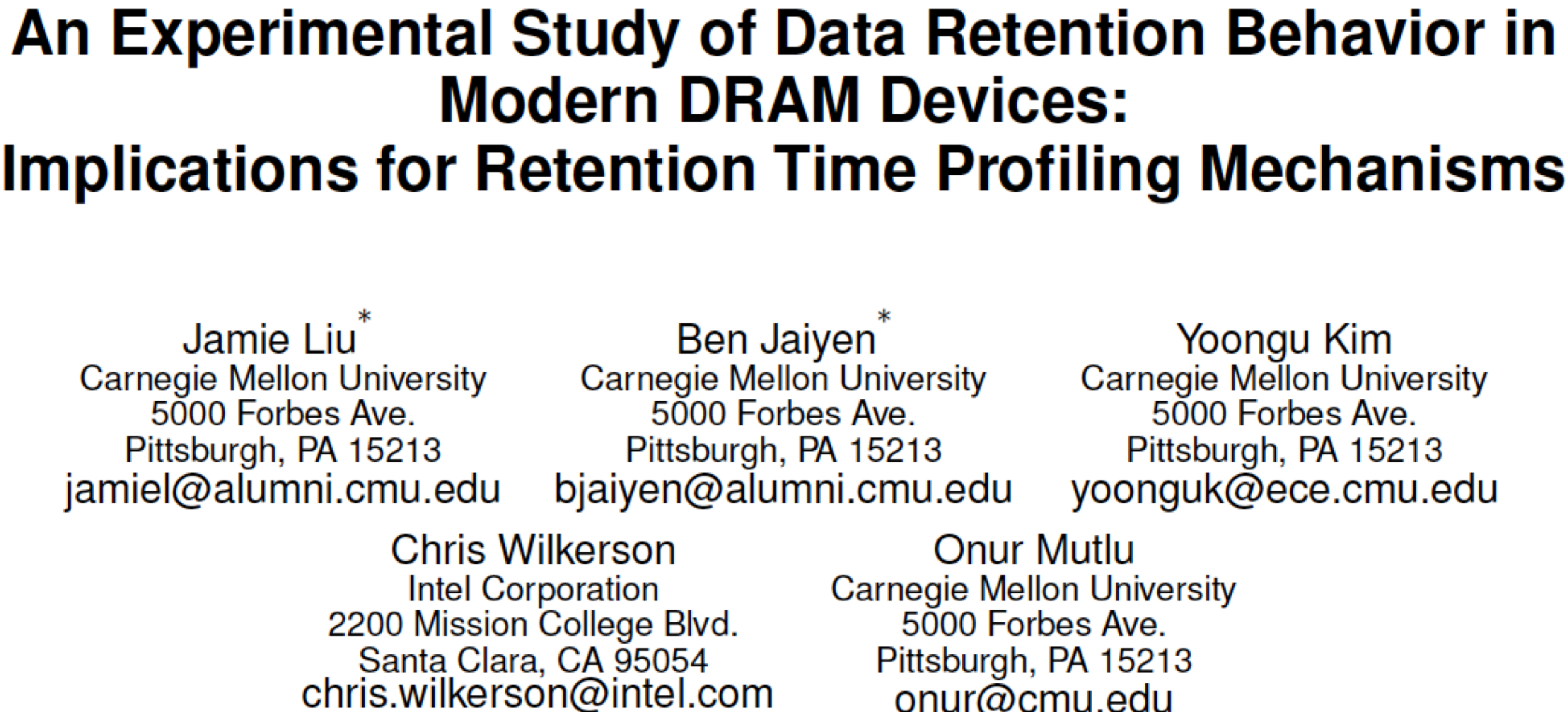
DRAM Data Retention Time Failures

- Determining the data retention time of a cell/row is getting more difficult

- Retention failures may already be slipping into the field

Analysis of Data Retention Failures **[ISCA'13]**

- Jamie Liu, Ben Jaiyen, Yoongu Kim, Chris Wilkerson, and Onur Mutlu, **"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)



An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms

Jamie Liu[*]
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
jamiel@alumni.cmu.edu

Ben Jaiyen[*]
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
bjaiyen@alumni.cmu.edu

Yoongu Kim
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
yoonguk@ece.cmu.edu

Chris Wilkerson
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95054
chris.wilkerson@intel.com

Onur Mutlu
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
onur@cmu.edu

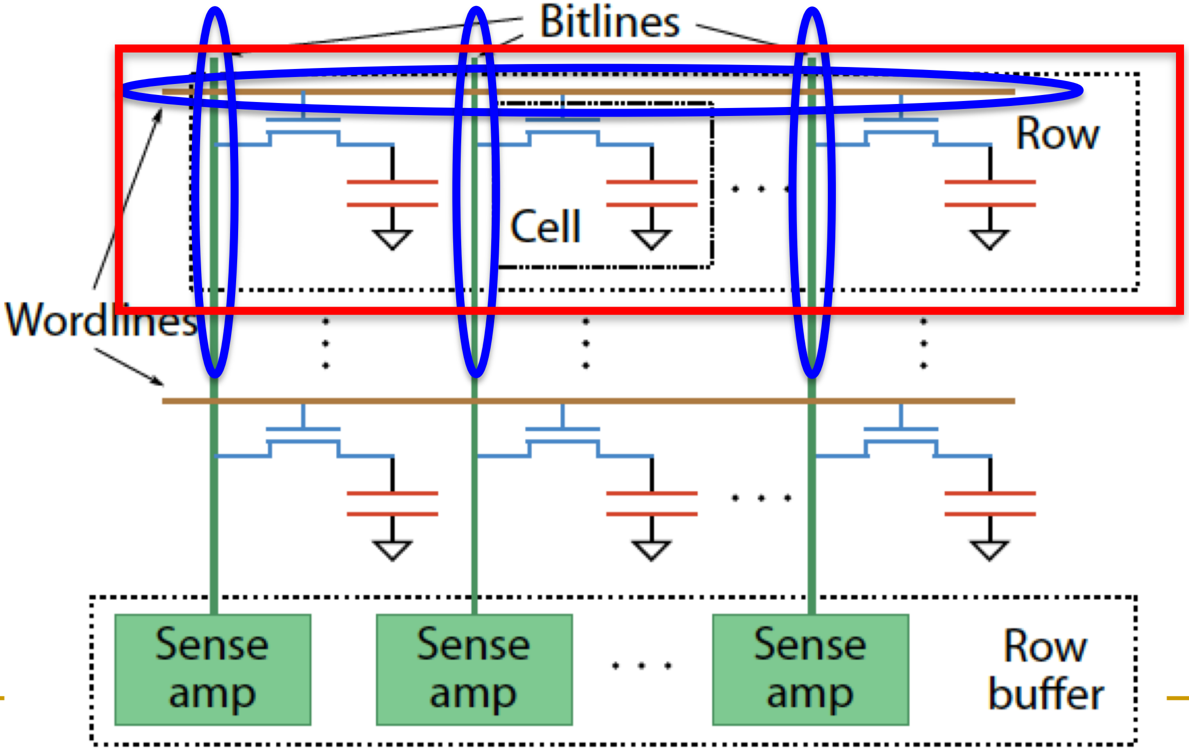
Two Challenges to Retention Time Profiling

- Data Pattern Dependence (DPD) of retention time

- Variable Retention Time (VRT) phenomenon

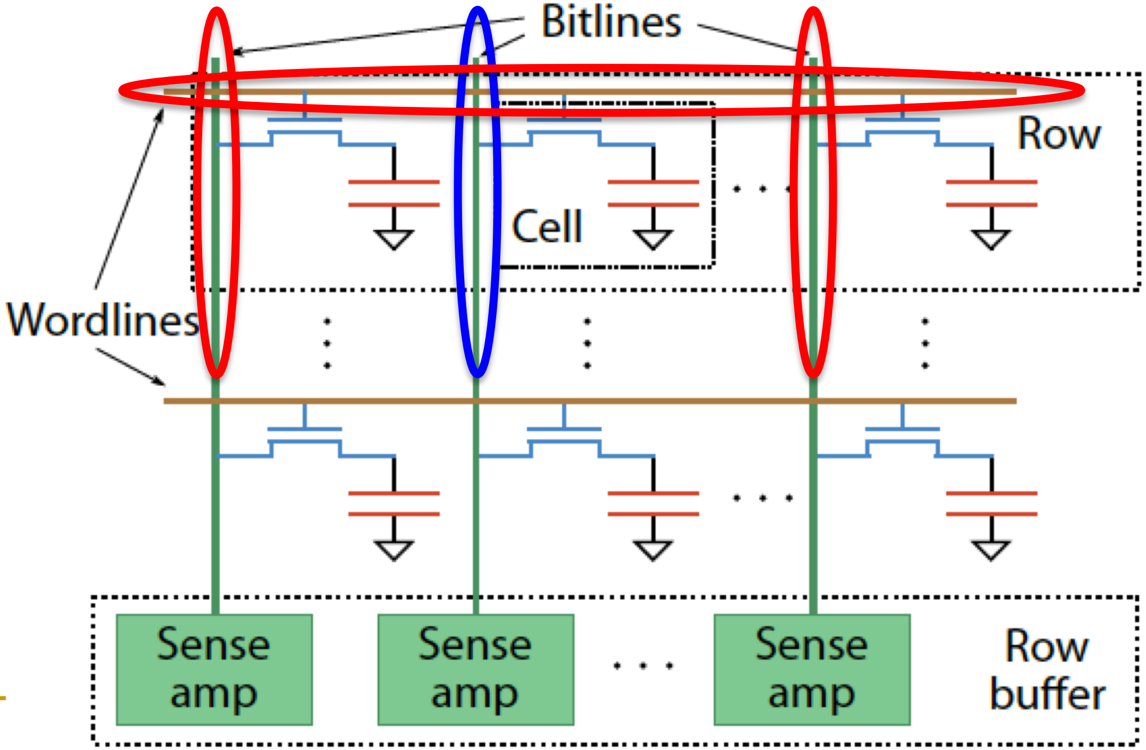
Two Challenges to Retention Time Profiling

- Challenge 1: Data Pattern Dependence (DPD)
  - Retention time of a DRAM cell depends on its value and the values of cells nearby it
  - When a row is activated, all bitlines are perturbed simultaneously



Data Pattern Dependence

- Electrical noise on the bitline affects reliable sensing of a DRAM cell
- The magnitude of this noise is affected by values of nearby cells via
  - Bitline-bitline coupling → electrical coupling between adjacent bitlines
  - Bitline-wordline coupling → electrical coupling between each bitline and the activated wordline



Data Pattern Dependence

- Electrical noise on the bitline affects reliable sensing of a DRAM cell
- The magnitude of this noise is affected by values of nearby cells via
  - Bitline-bitline coupling → electrical coupling between adjacent bitlines
  - Bitline-wordline coupling → electrical coupling between each bitline and the activated wordline

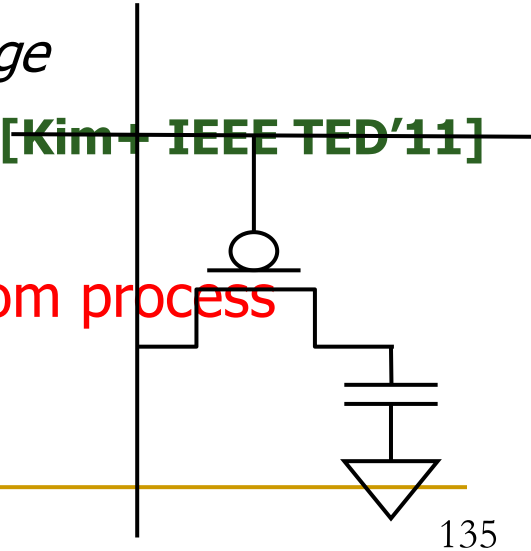
- Retention time of a cell depends on data patterns stored in nearby cells

  → need to find the worst data pattern to find worst-case retention time

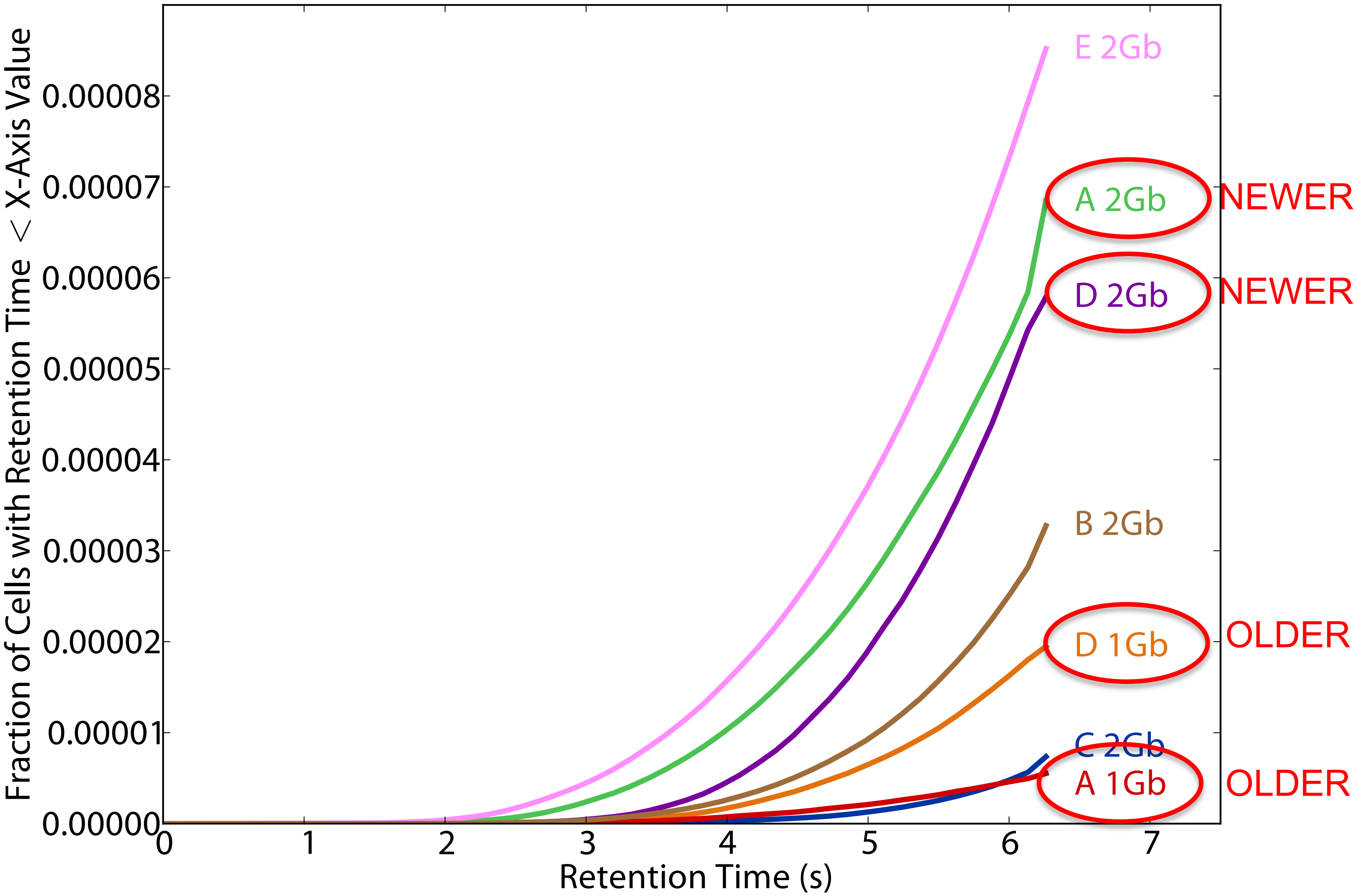
  → this pattern is location dependent

Two Challenges to Retention Time Profiling

- Challenge 2: Variable Retention Time (VRT)
  - Retention time of a DRAM cell changes randomly over time
    - a cell alternates between multiple retention time states
  - Leakage current of a cell changes sporadically due to a charge trap in the gate oxide of the DRAM cell access transistor
  - When the trap becomes occupied, charge leaks more readily from the transistor's drain, leading to a short retention time
    - Called *Trap-Assisted Gate-Induced Drain Leakage*
  - This process appears to be a random process **[Kim+ IEEE TED'11]**
  - Worst-case retention time depends on a random process → need to find the worst case despite this

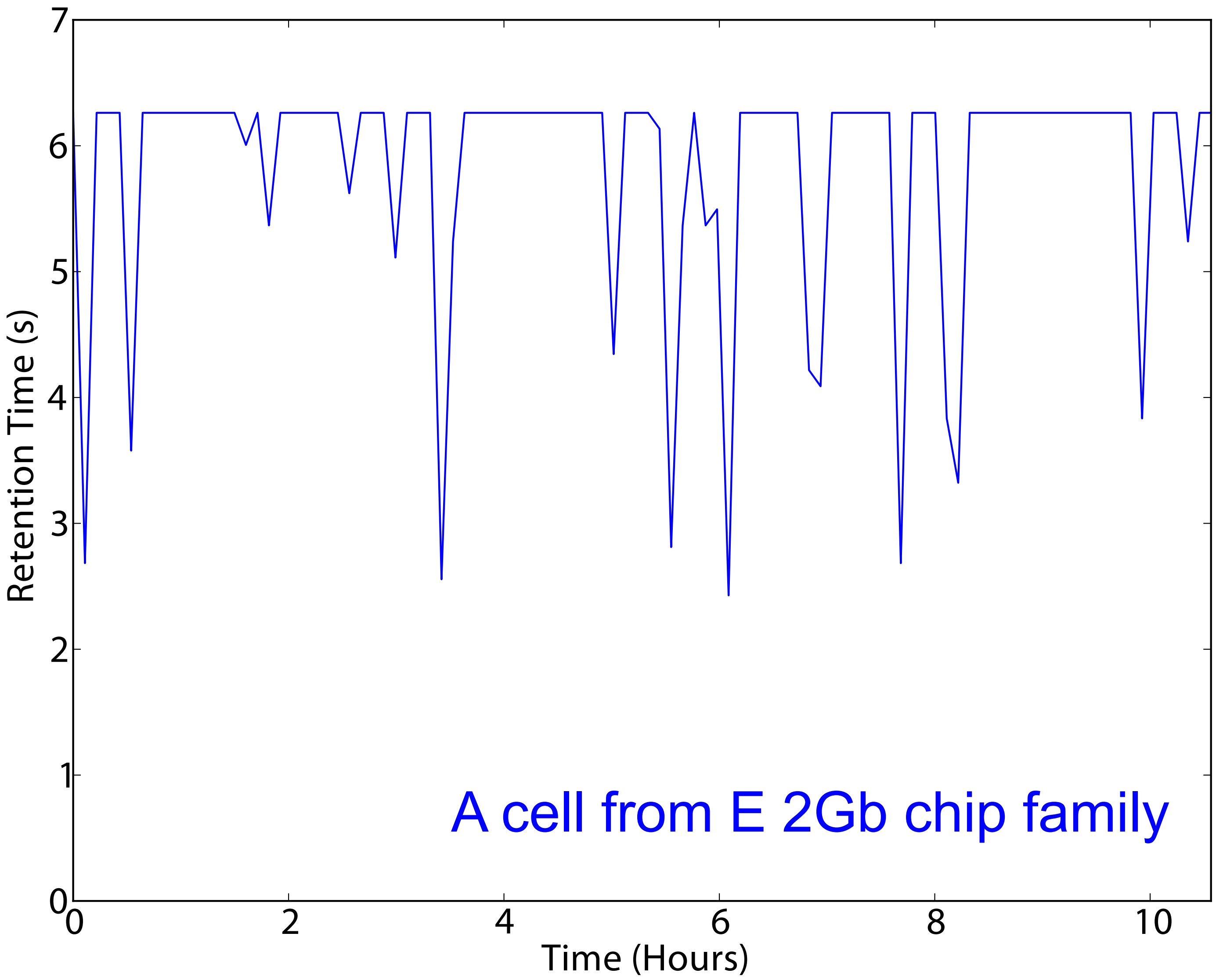


Modern DRAM Retention Time Distribution



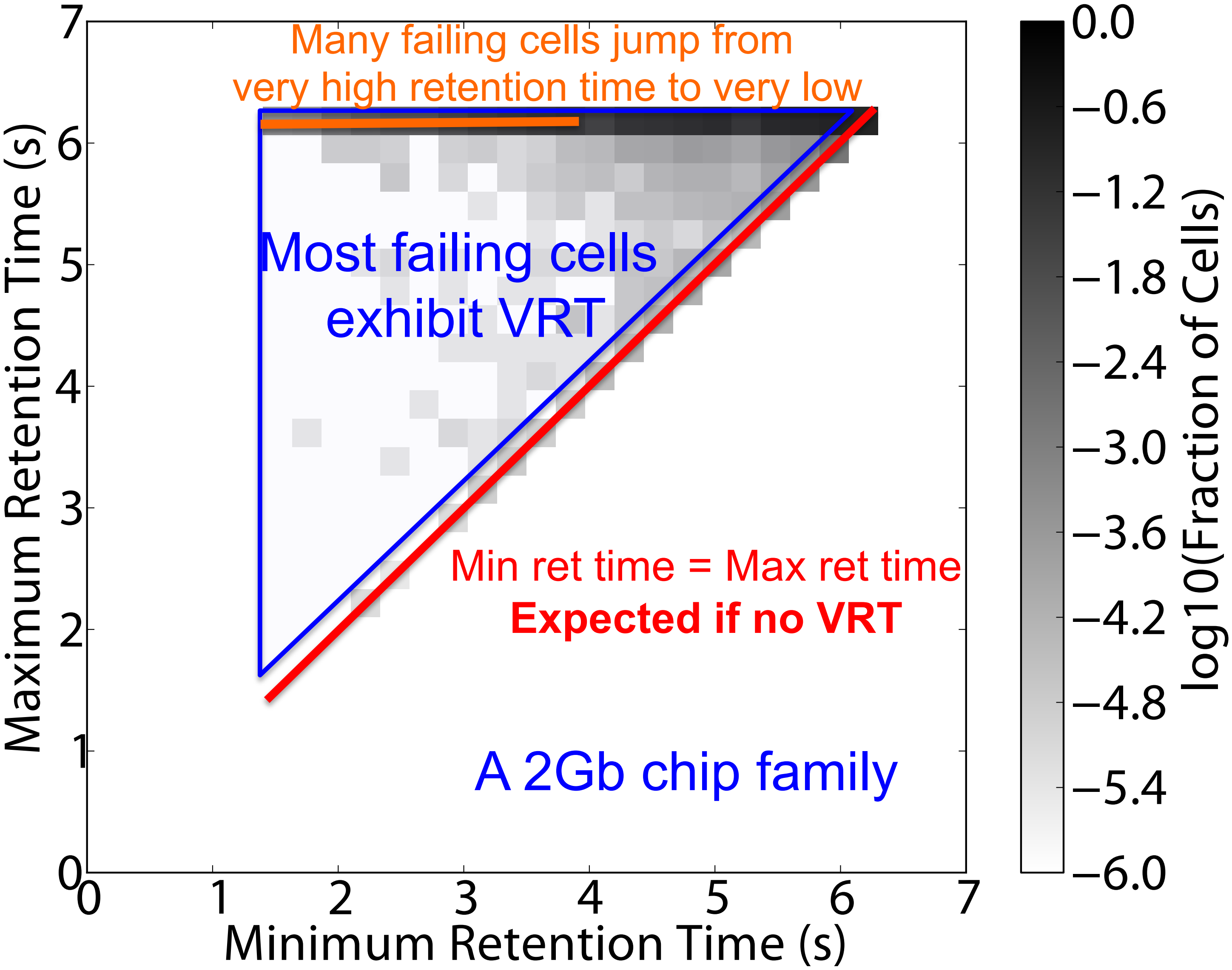
Newer device families have more weak cells than older ones
Likely a result of technology scaling

An Example VRT Cell



A cell from E 2Gb chip family

Variable Retention Time



Many failing cells jump from very high retention time to very low

Most failing cells exhibit VRT

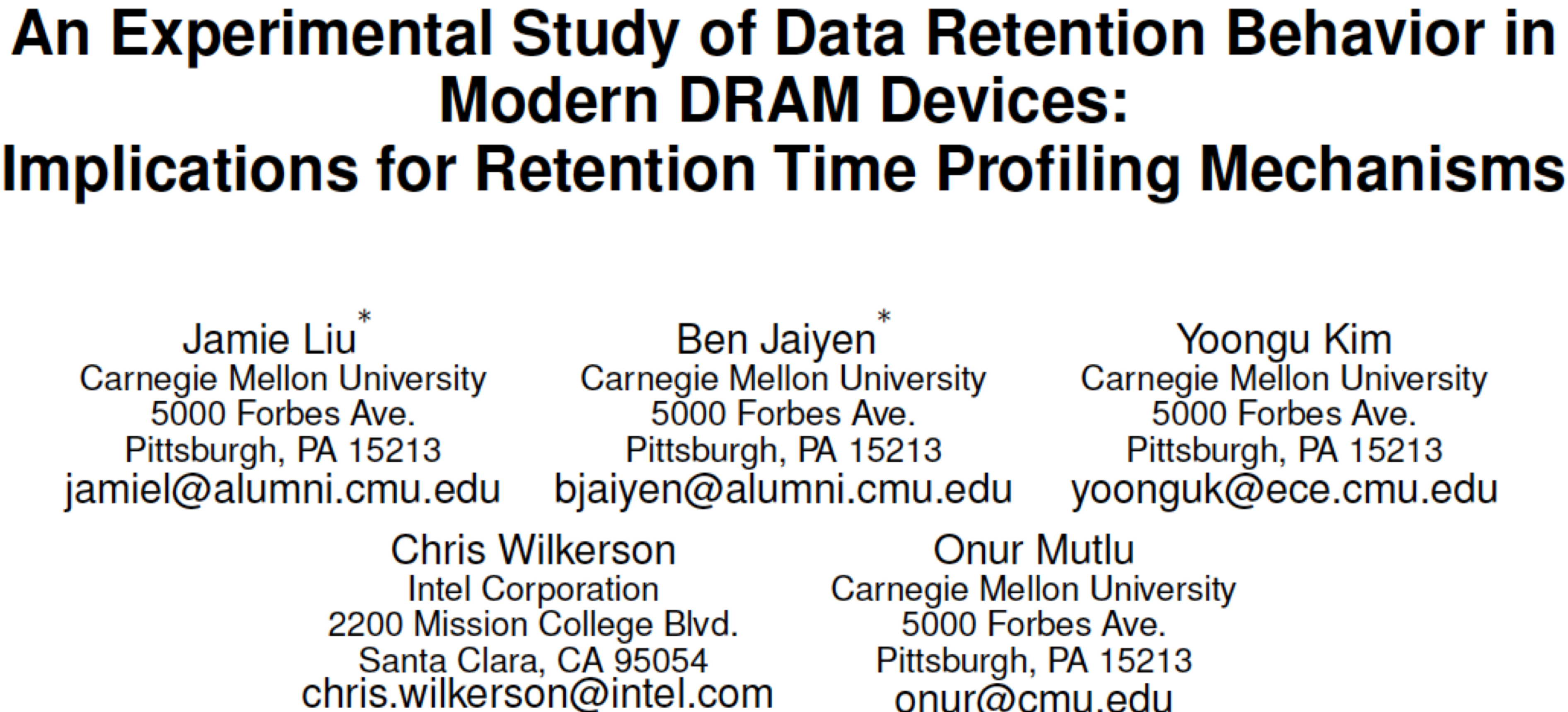
Min ret time = Max ret time

Expected if no VRT

A 2Gb chip family

More on Data Retention Failures [ISCA'13]

- Jamie Liu, Ben Jaiyen, Yoongu Kim, Chris Wilkerson, and Onur Mutlu, **"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)



An Experimental Study of Data Retention Behavior in Modern DRAM Devices: Implications for Retention Time Profiling Mechanisms

Jamie Liu[*]
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
jamiel@alumni.cmu.edu

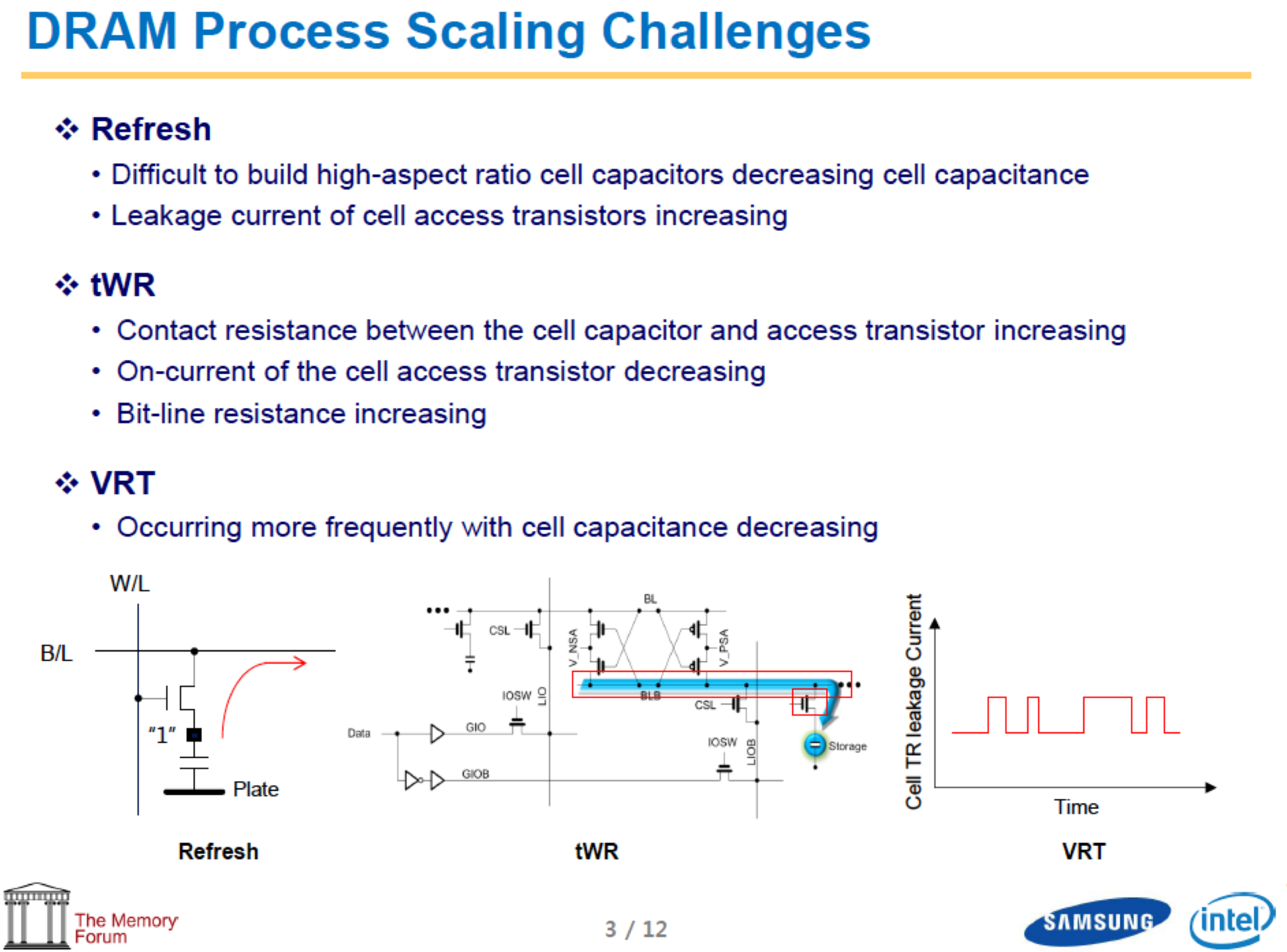
Ben Jaiyen[*]
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
bjaiyen@alumni.cmu.edu

Yoongu Kim
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
yoonguk@ece.cmu.edu

Chris Wilkerson
Intel Corporation
2200 Mission College Blvd.
Santa Clara, CA 95054
chris.wilkerson@intel.com

Onur Mutlu
Carnegie Mellon University
5000 Forbes Ave.
Pittsburgh, PA 15213
onur@cmu.edu

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

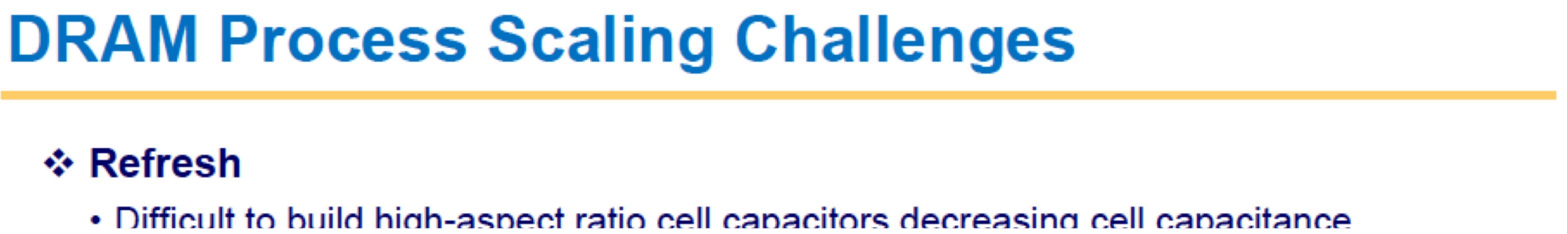
Refresh **tWR** **VRT**

The Memory Forum

3 / 12

SAMSUNG intel

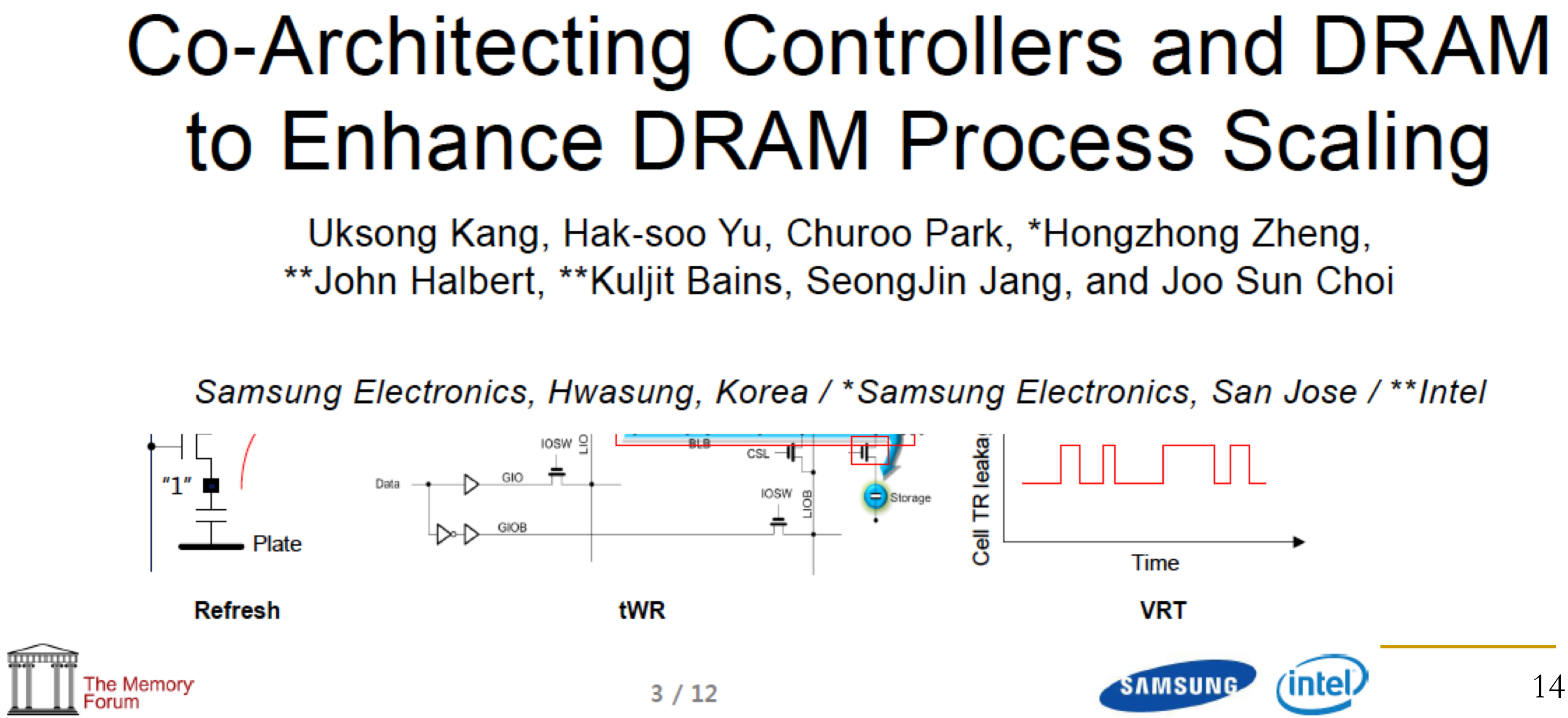
Industry Is Writing Papers About It, Too



DRAM Process Scaling Challenges

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

THE MEMORY FORUM 2014



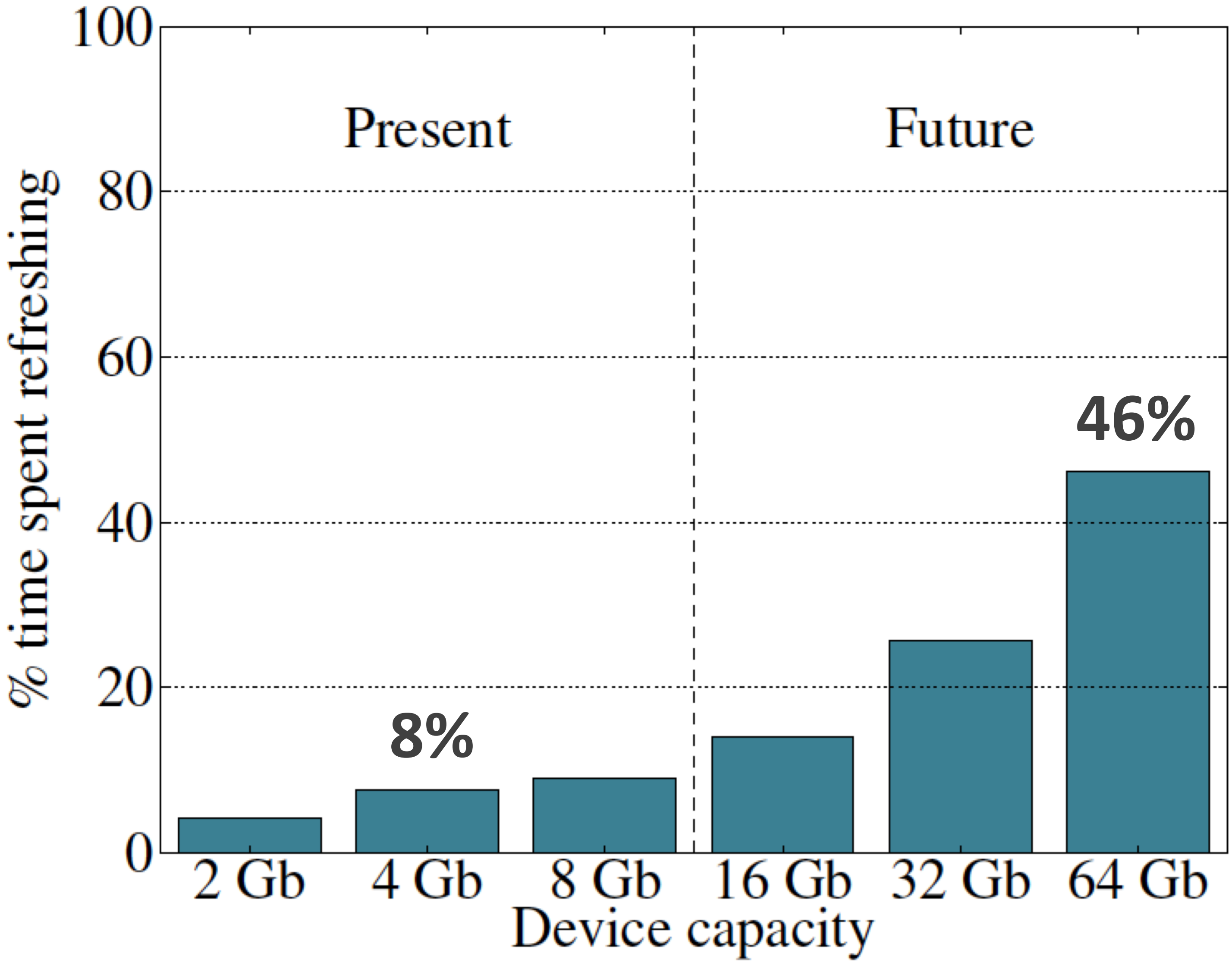
Co-Architecting Controllers and DRAM to Enhance DRAM Process Scaling

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*

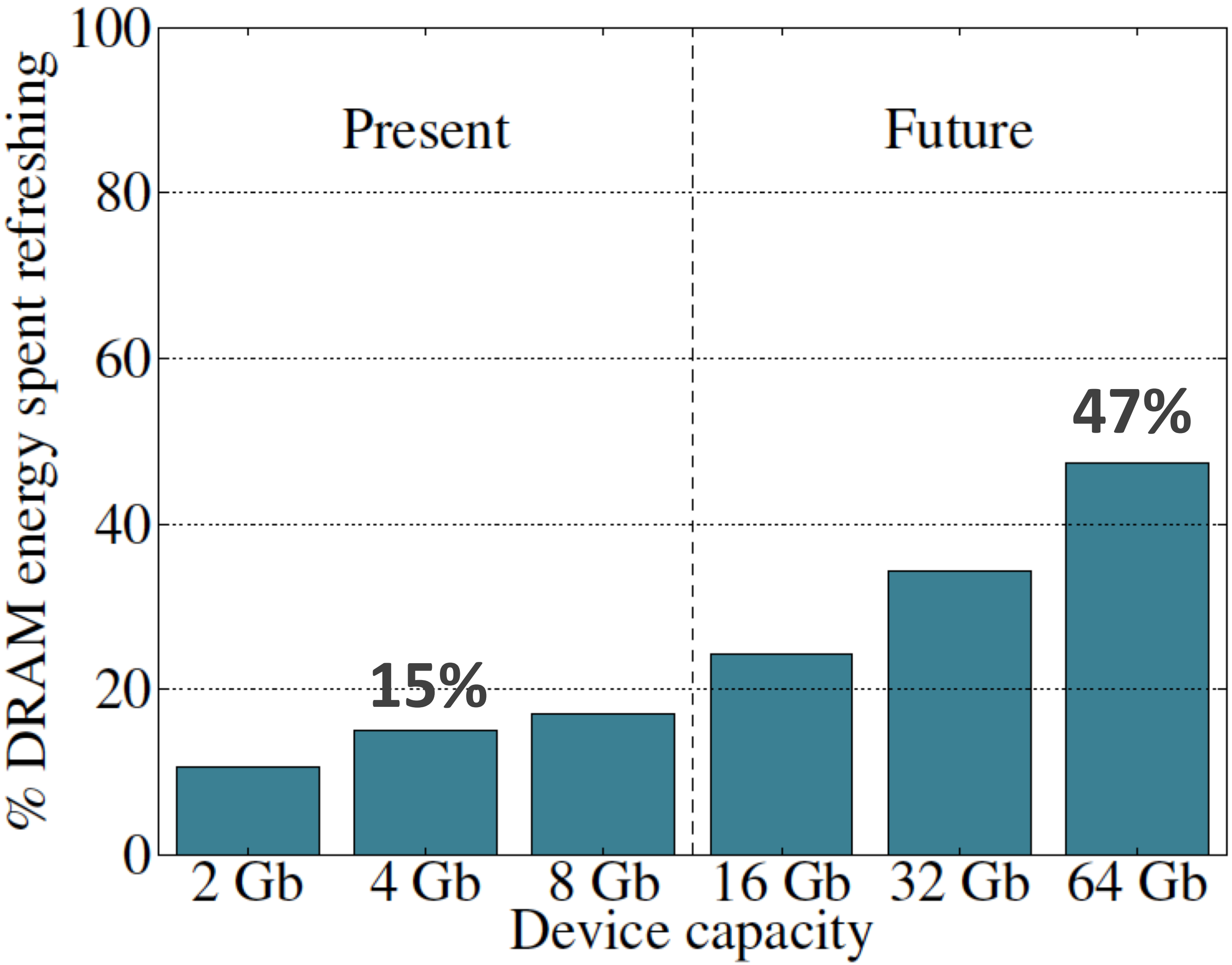
3 / 12

Refresh Overhead: Performance



Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

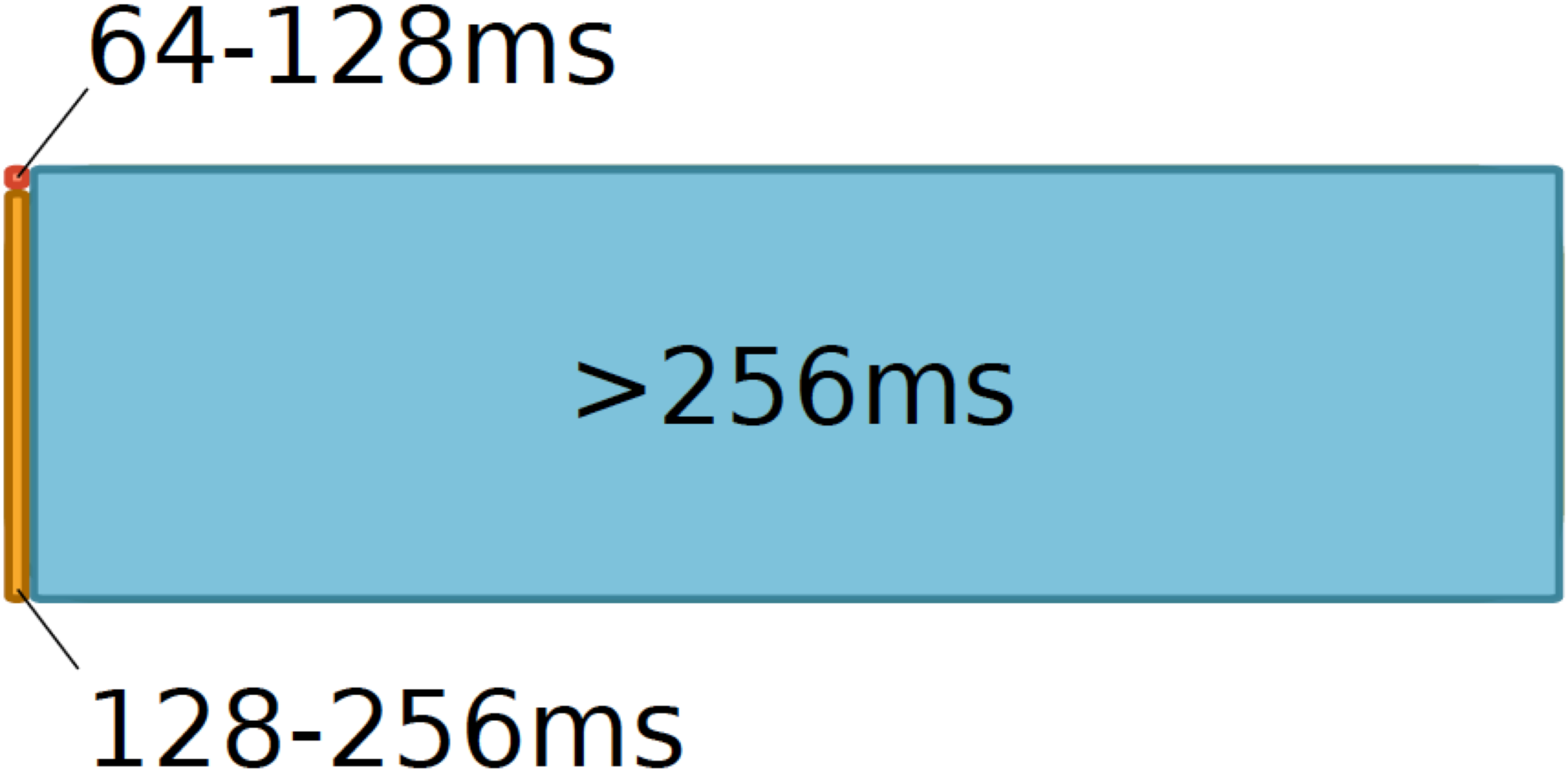
Refresh Overhead: Energy



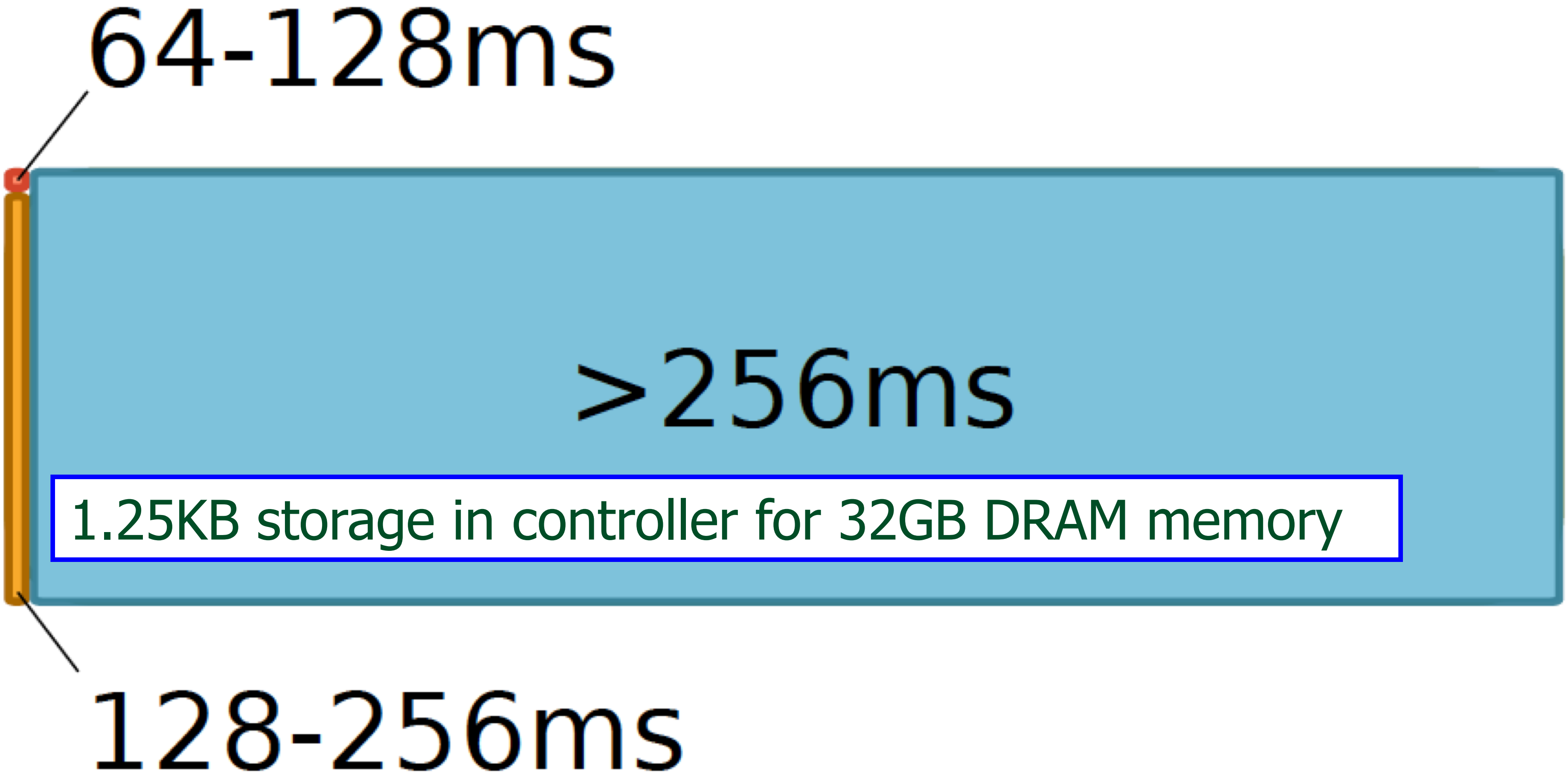
Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

Most Refreshes Are Unnecessary

- Retention Time Profile of DRAM looks like this:



RAIDR: Eliminating Unnecessary Refreshes



64-128ms

>256ms

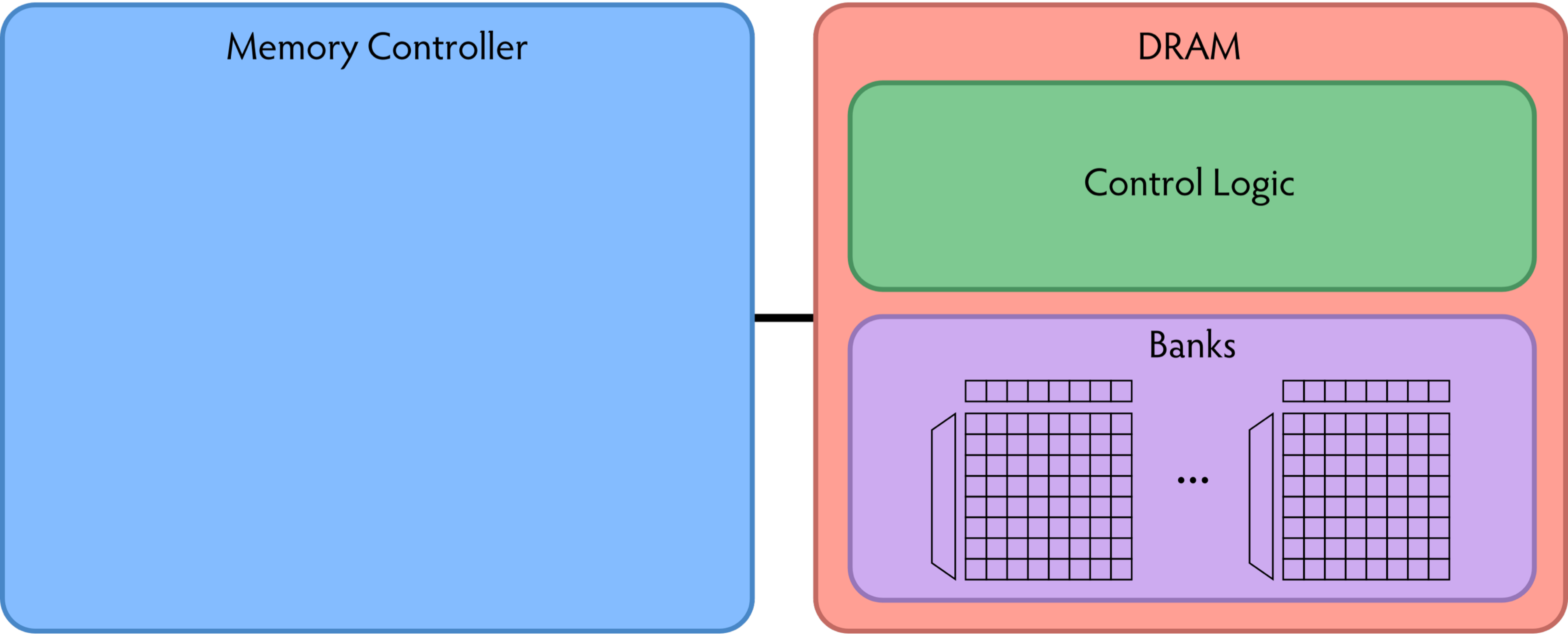
1.25KB storage in controller for 32GB DRAM memory

128-256ms

Can reduce refreshes by ~75%
→ reduces energy consumption and improves performance

SAFARI Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

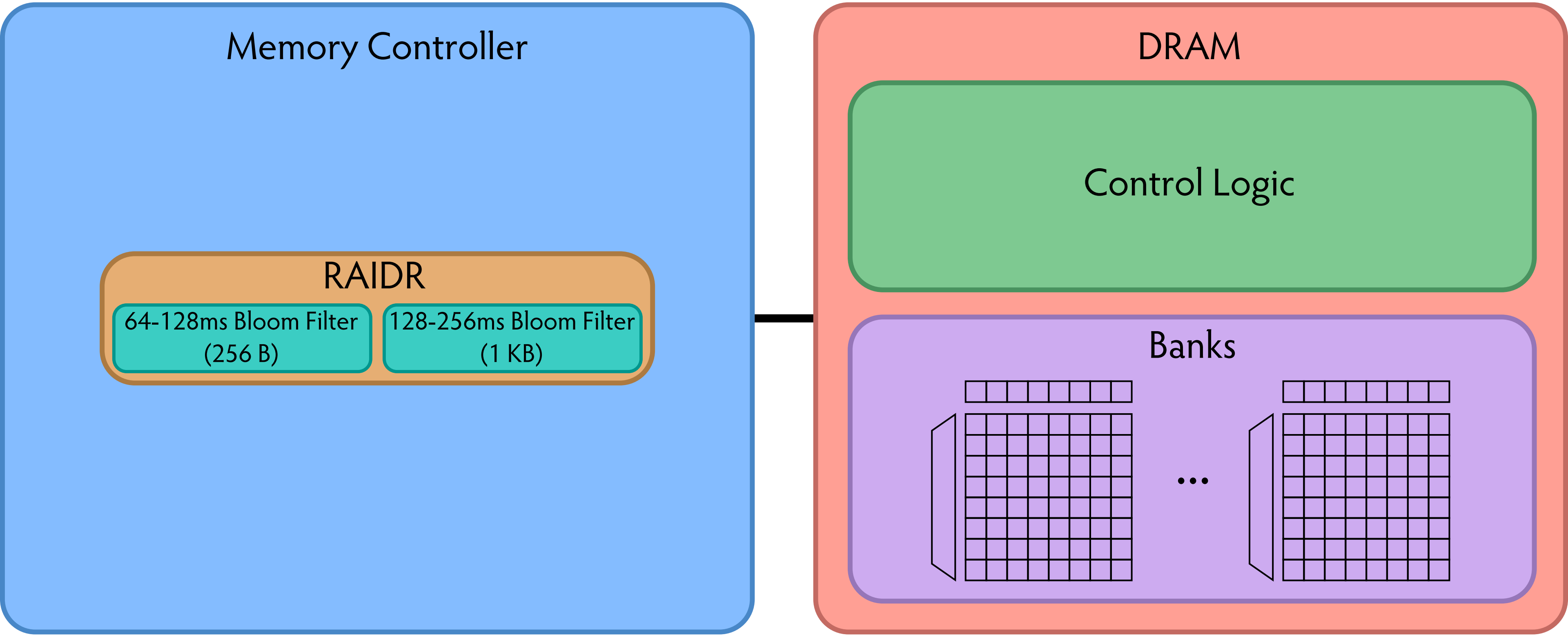
RAIDR: Baseline Design



Refresh control is in DRAM in today’s auto-refresh systems

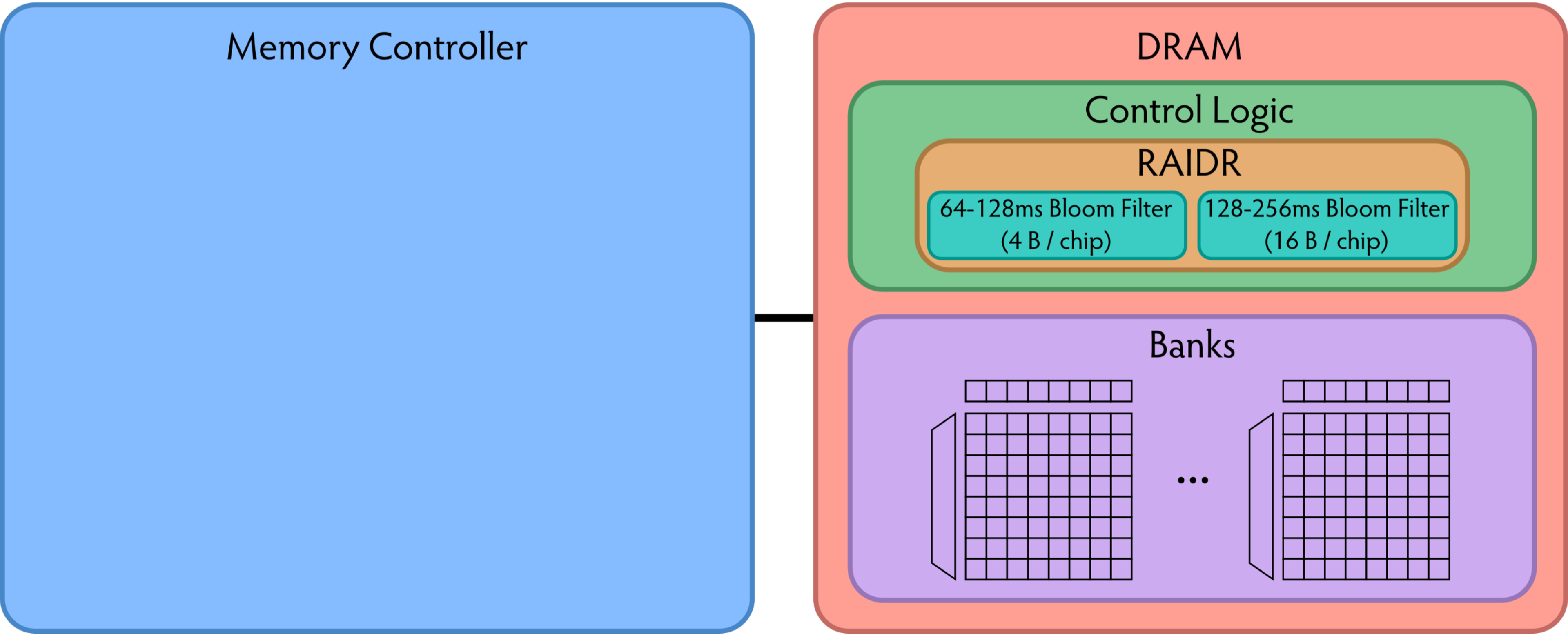
RAIDR can be implemented in either the controller or DRAM

RAIDR in Memory Controller: Option 1



Overhead of RAIDR in DRAM controller:
1.25 KB Bloom Filters, 3 counters, additional commands issued for per-row refresh (all accounted for in evaluations)

RAIDR in DRAM Chip: Option 2



Overhead of RAIDR in DRAM chip:

Per-chip overhead: 20B Bloom Filters, 1 counter (4 Gbit chip)

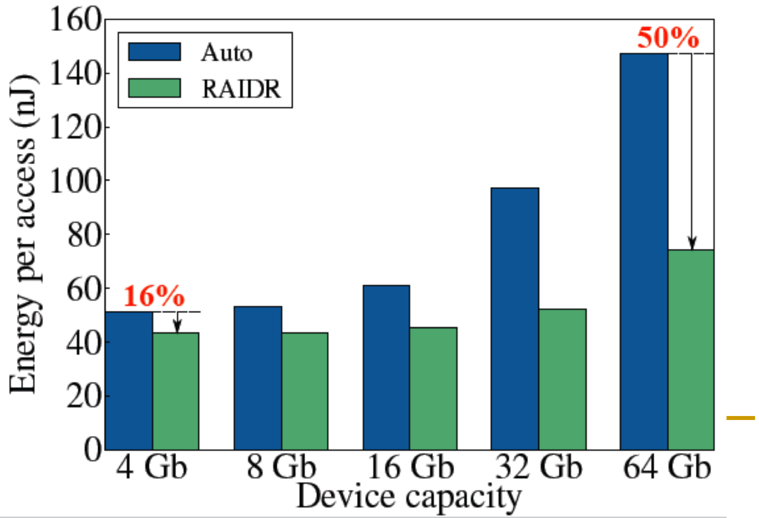
Total overhead: 1.25KB Bloom Filters, 64 counters (32 GB DRAM)

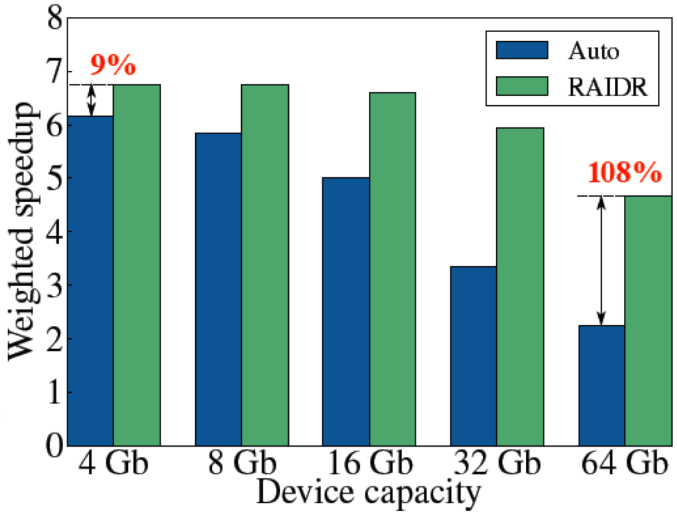
RAIDR: Results and Takeaways

- System: 32GB DRAM, 8-core; SPEC, TPC-C, TPC-H workloads

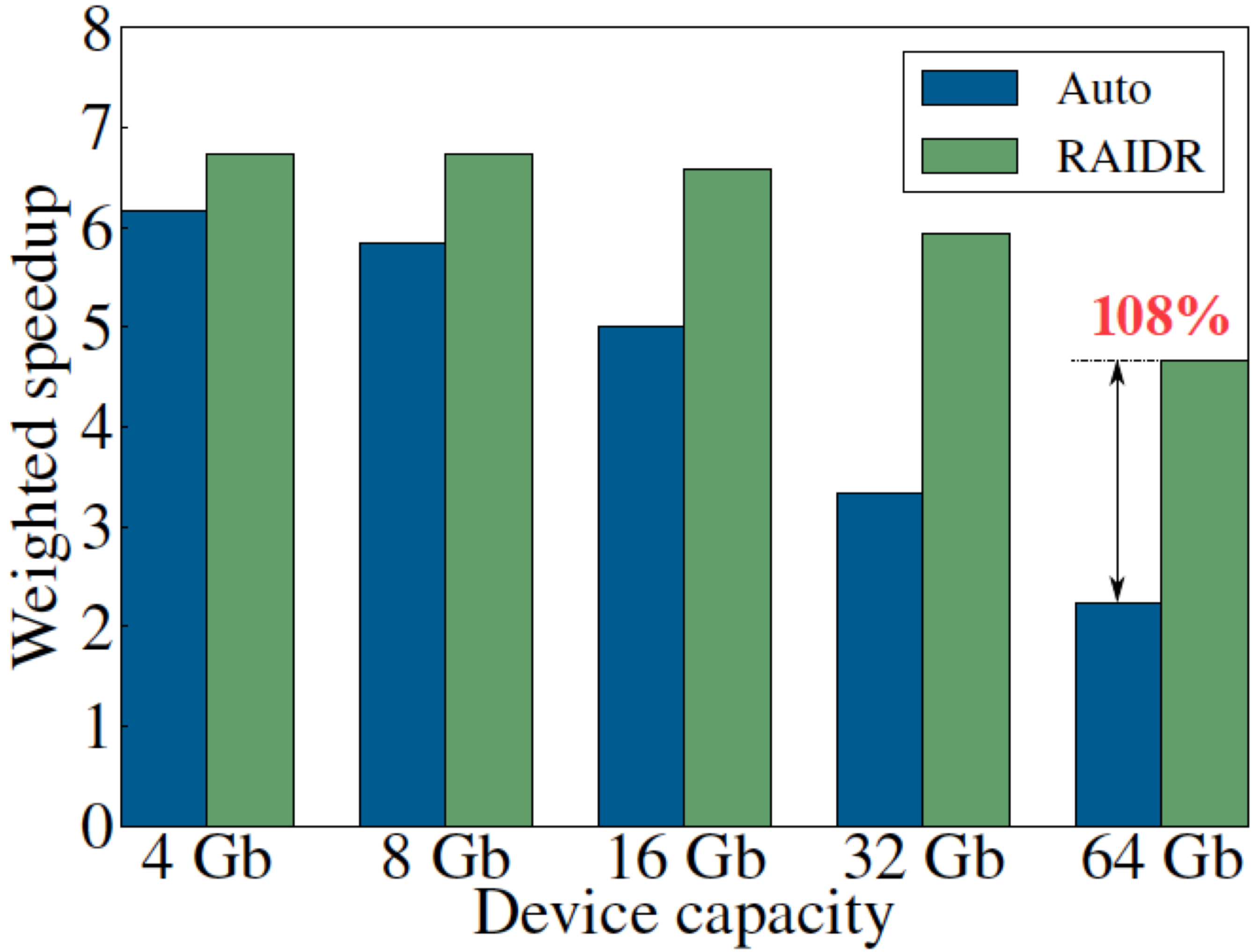
- RAIDR hardware cost: 1.25 kB (2 Bloom filters)
- Refresh reduction: 74.6%
- Dynamic DRAM energy reduction: 16%
- Idle DRAM power reduction: 20%
- Performance improvement: 9%

- Benefits increase as DRAM scales in density





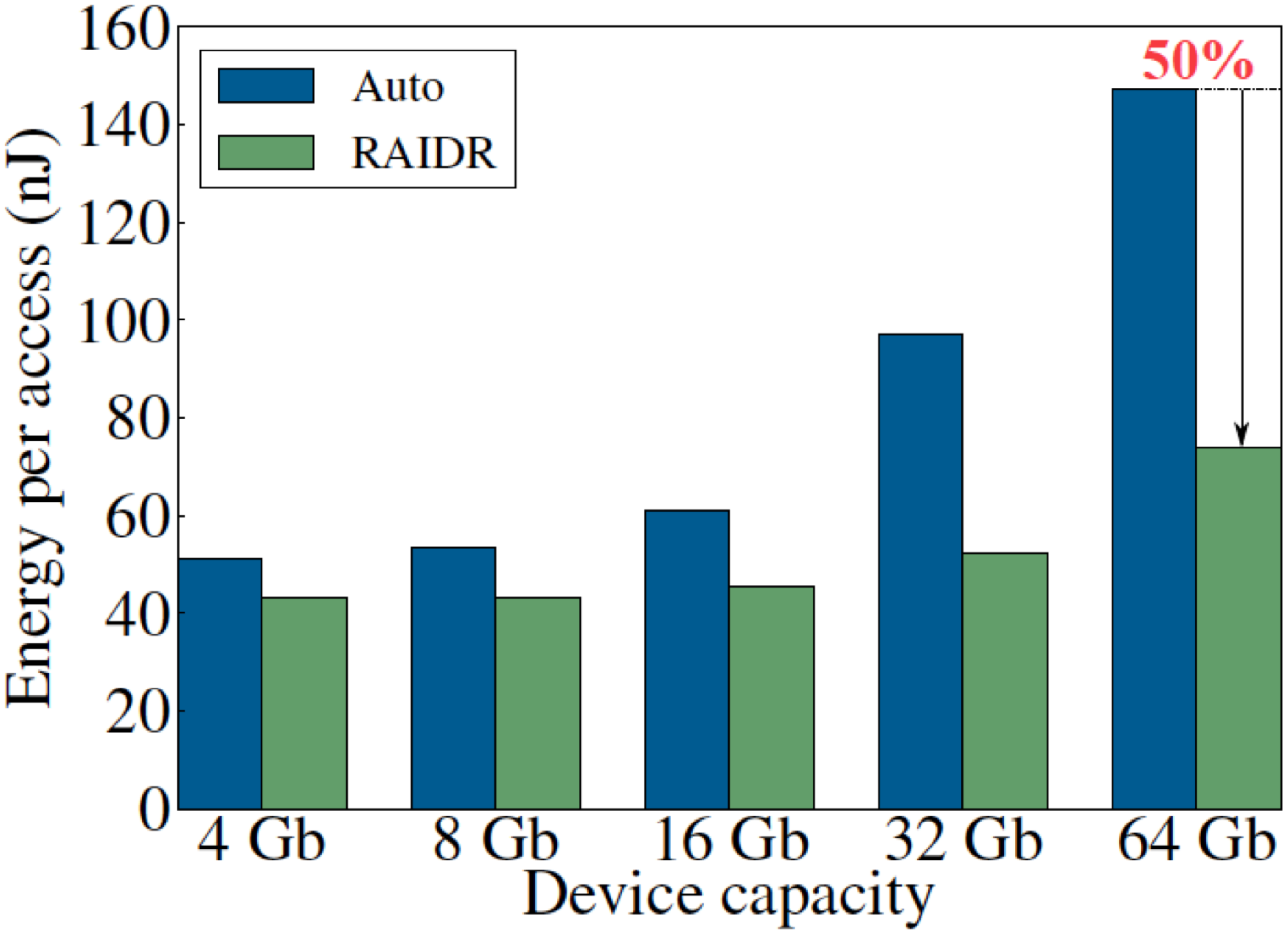
DRAM Device Capacity Scaling: Performance



RAIDR performance benefits increase with DRAM chip capacity

Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

DRAM Device Capacity Scaling: Energy

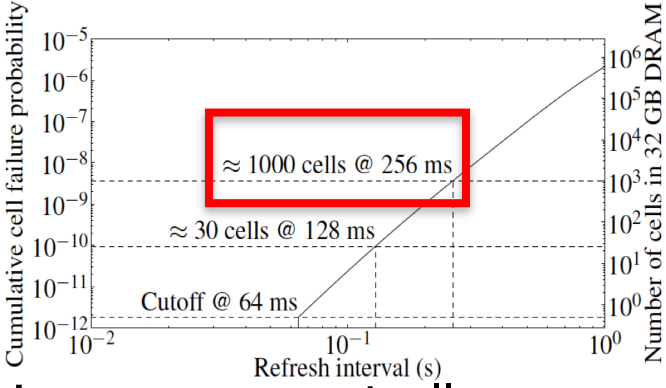


RAIDR energy benefits increase with DRAM chip capacity

SAFARI Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

RAIDR: Eliminating Unnecessary Refreshes

- Observation: Most DRAM rows can be refreshed much less often without losing data **[Kim+, EDL'09][Liu+ ISCA'13]**



- Key idea: Refresh rows containing weak cells more frequently, other rows less frequently

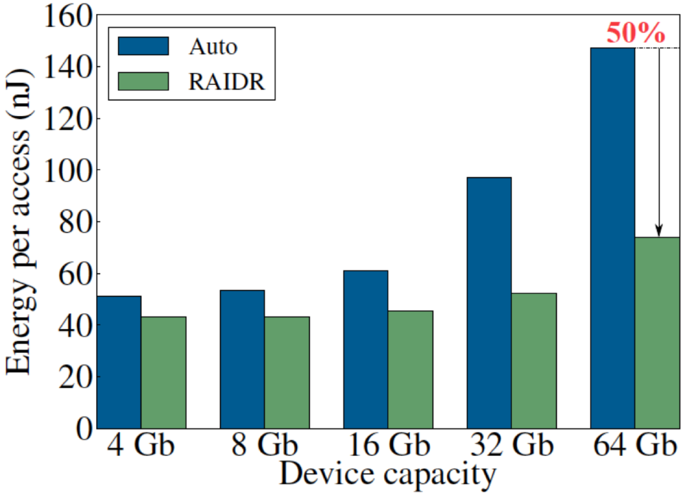
  1. Profiling: Profile retention time of all rows

  2. Binning: Store rows into bins by retention time in memory controller

     Efficient storage with Bloom Filters (only 1.25KB for 32GB memory)

  3. Refreshing: Memory controller refreshes rows in different bins at different rates

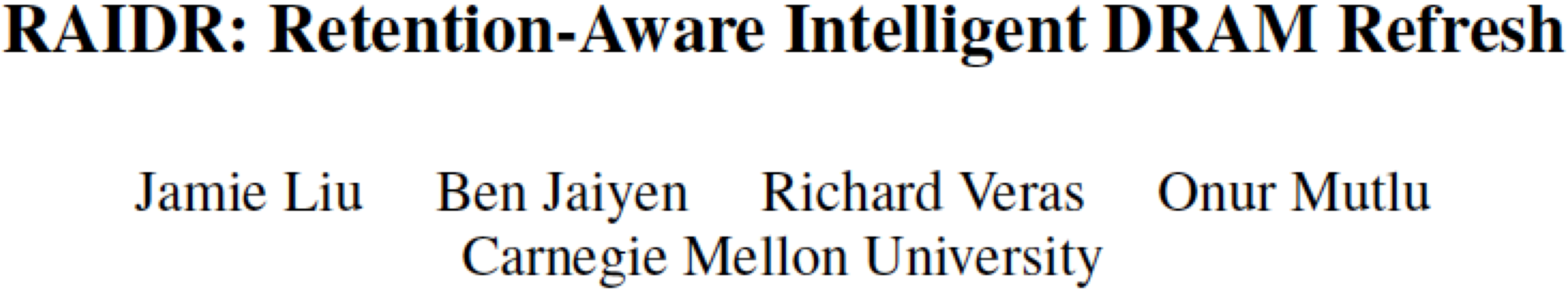
- Results: 8-core, 32GB, SPEC, TPC-C, TPC-H
  - 74.6% refresh reduction @ 1.25KB storage
  - ~16%/20% DRAM dynamic/idle power reduction
  - ~9% performance improvement
  - Benefits increase with DRAM capacity



Liu et al., "RAIDR: Retention-Aware Intelligent DRAM Refresh," ISCA 2012.

More on RAIDR: Perf+Energy Perspective

- Jamie Liu, Ben Jaiyen, Richard Veras, and Onur Mutlu, **"RAIDR: Retention-Aware Intelligent DRAM Refresh"** *Proceedings of the 39th International Symposium on Computer Architecture* (***ISCA***), Portland, OR, June 2012. Slides (pdf)



RAIDR: Retention-Aware Intelligent DRAM Refresh

Jamie Liu Ben Jaiyen Richard Veras Onur Mutlu
Carnegie Mellon University

Finding DRAM Retention Failures

- How can we reliably find the retention time of all DRAM cells?

- Goals: so that we can
  - Make DRAM reliable and secure
  - Make techniques like RAIDR work

    → improve performance and energy

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"
  *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (**SIGMETRICS**)*, Austin, TX, June 2014. [Slides (pptx) (pdf)] [Poster (pptx) (pdf)] [Full data sets]

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

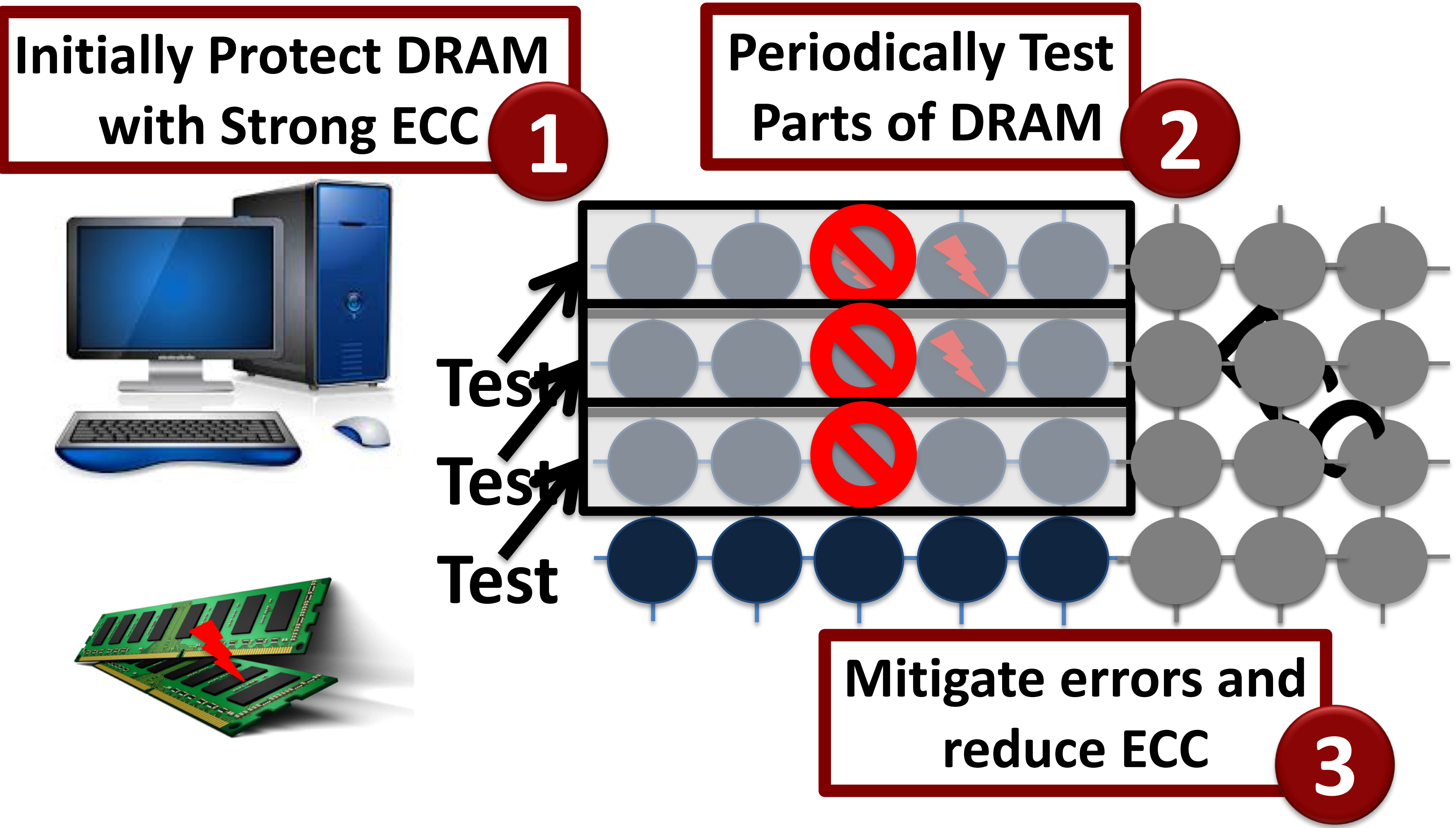
Towards an Online Profiling System

Key Observations:

- **Testing alone cannot detect all possible failures**
- **Combination of ECC and other mitigation techniques is much more effective**
  - **But degrades performance**
- **Testing can help to reduce the ECC strength**
  - **Even when starting with a higher strength ECC**

Khan+, "The Efficacy of Error Mitigation Techniques for DRAM Retention Failures: A Comparative Experimental Study," SIGMETRICS 2014.

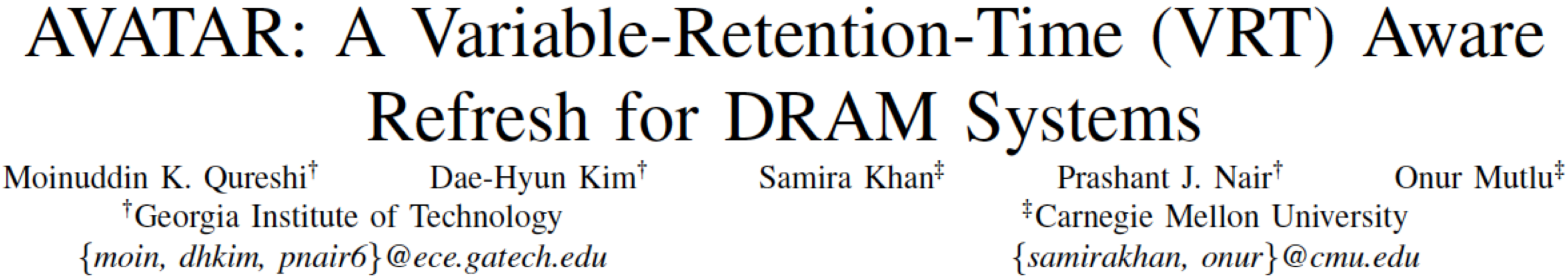
Towards an Online Profiling System


Initially Protect DRAM with Strong ECC
1
Periodically Test Parts of DRAM
2
Test
Test
Test
Mitigate errors and reduce ECC
3

Run tests periodically after a short interval at smaller regions of memory

Handling Variable Retention Time [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[‡]

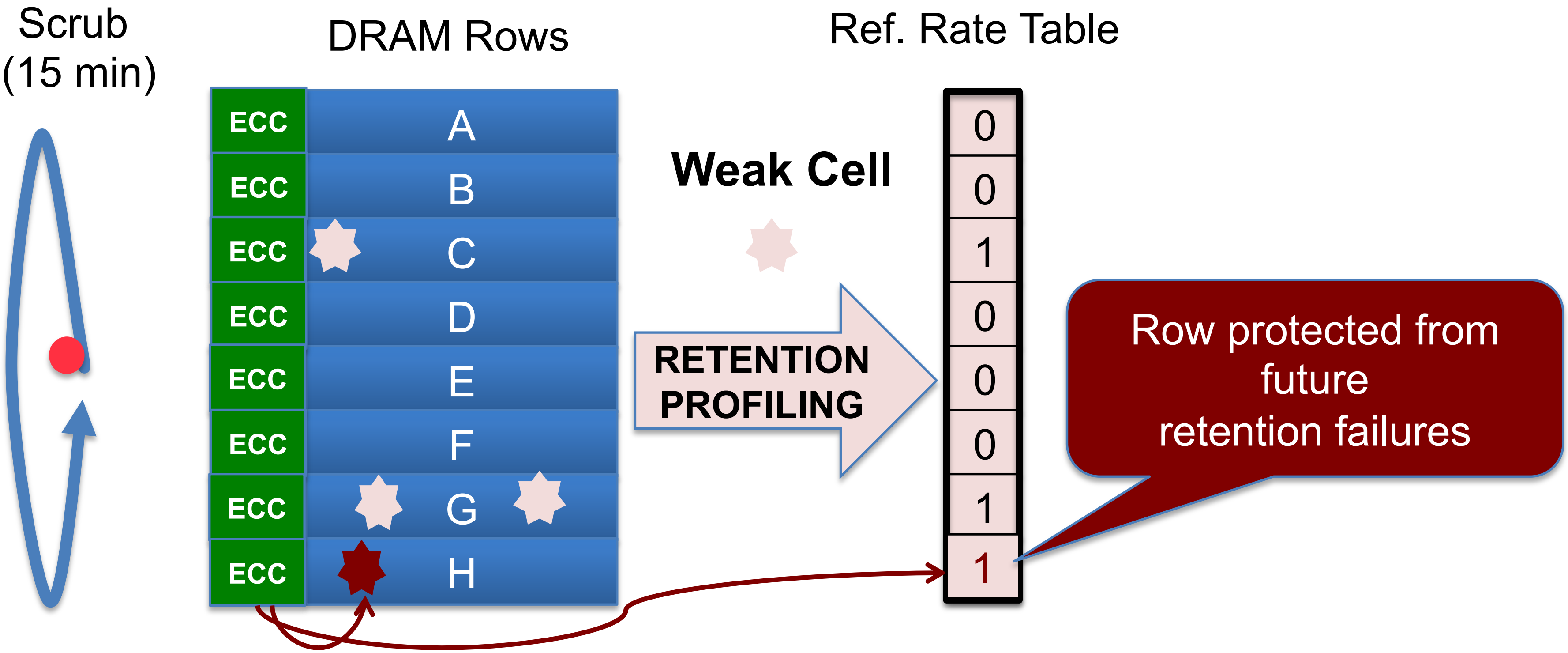
[†]Georgia Institute of Technology
{*moin, dhkim, pnair6*}*@ece.gatech.edu*

[‡]Carnegie Mellon University
{*samirakhan, onur*}*@cmu.edu*

AVATAR

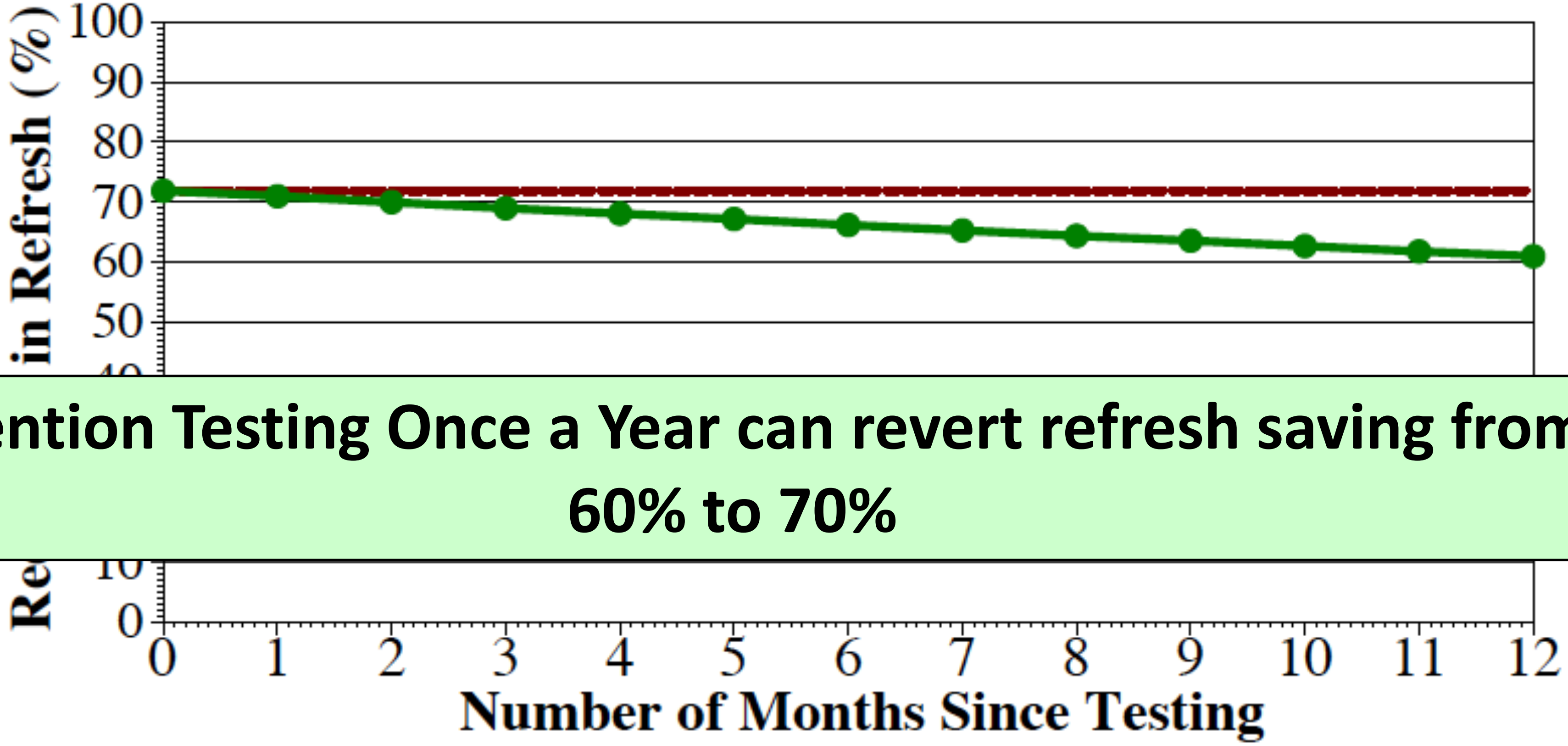
Insight: Avoid retention failures ➔ Upgrade row on ECC error

Observation: Rate of VRT >> Rate of soft error (50x-2500x)



AVATAR mitigates VRT by increasing refresh rate on error

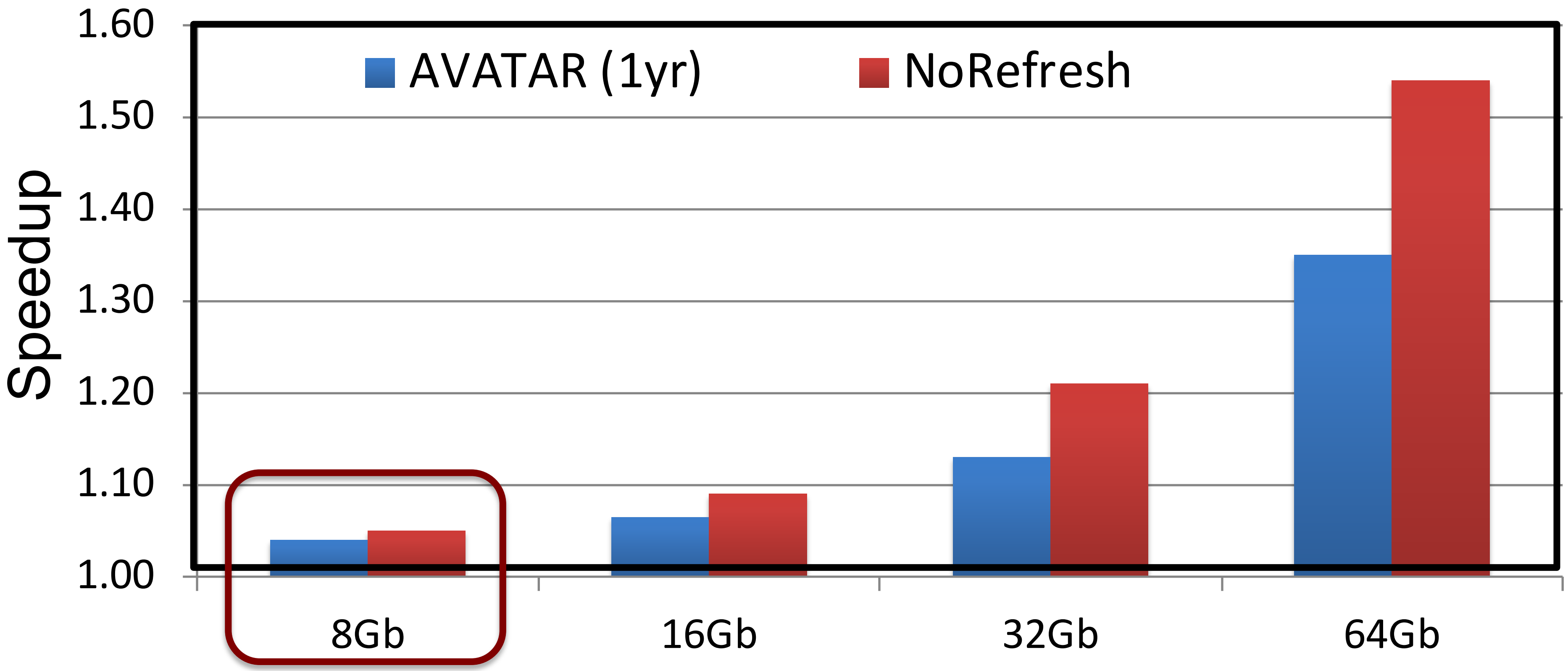
RESULTS: REFRESH SAVINGS



Retention Testing Once a Year can revert refresh saving from 60% to 70%

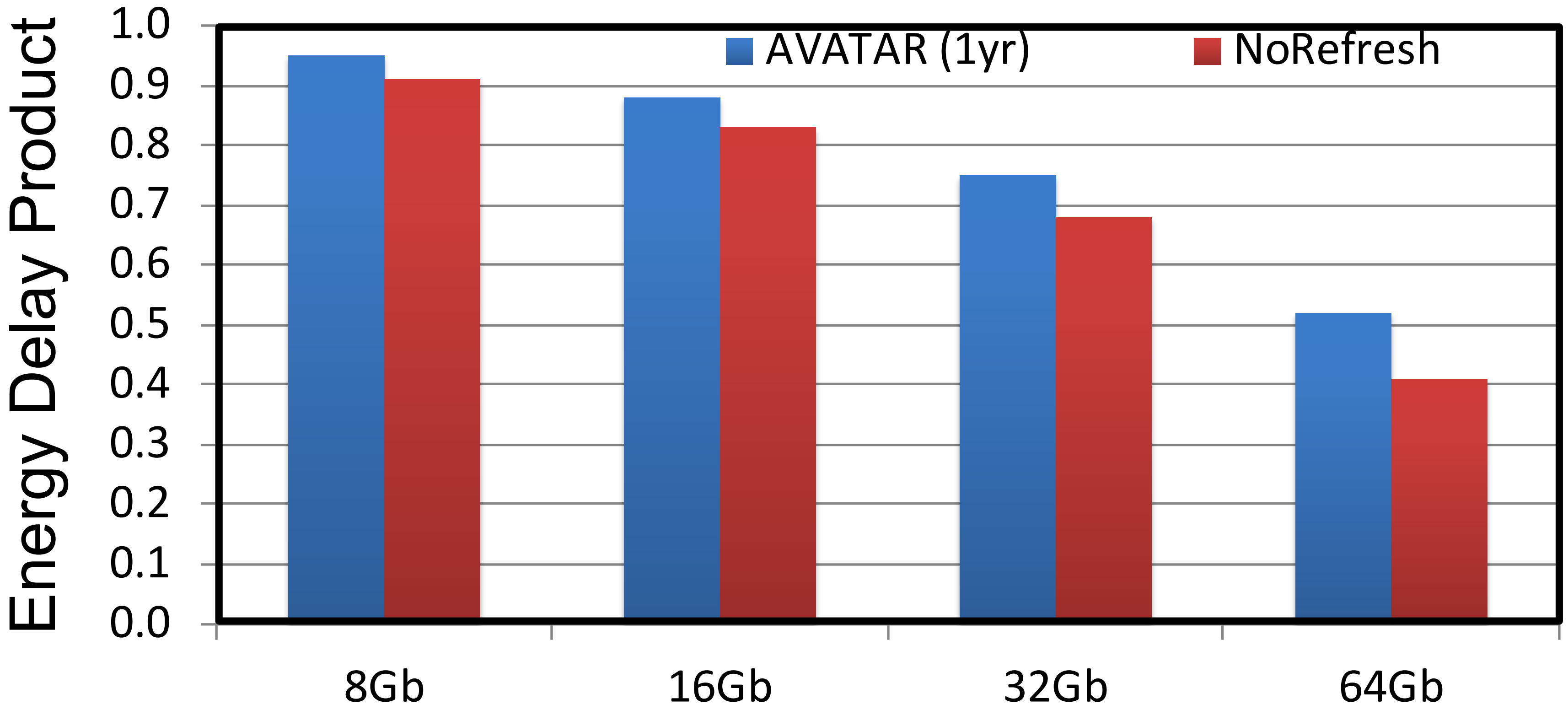
AVATAR reduces refresh by 60%-70%, similar to multi rate refresh but with VRT tolerance

SPEEDUP



AVATAR gets 2/3rd the performance of NoRefresh. More gains at higher capacity nodes

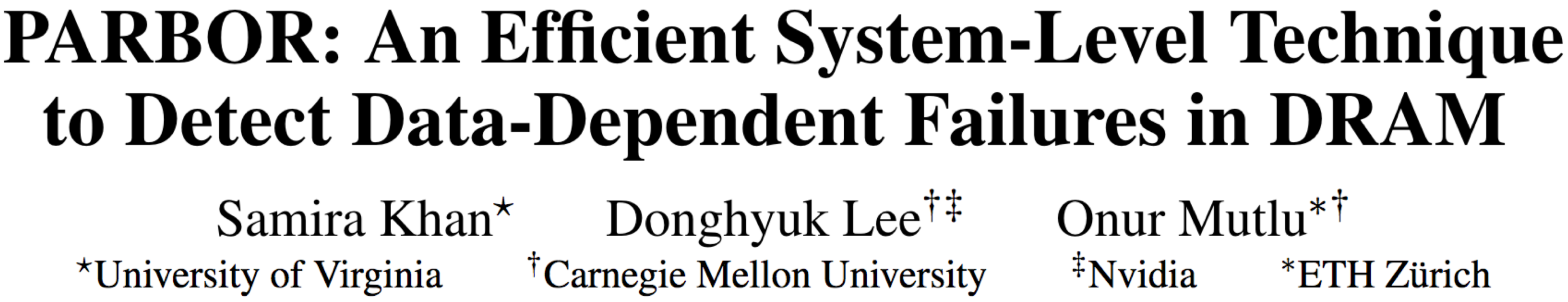
ENERGY DELAY PRODUCT



AVATAR reduces EDP,
Significant reduction at higher capacity nodes

Handling Data-Dependent Failures [DSN'16]

- Samira Khan, Donghyuk Lee, and Onur Mutlu,
  "PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM"
  *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Toulouse, France, June 2016.
  [Slides (pptx) (pdf)]



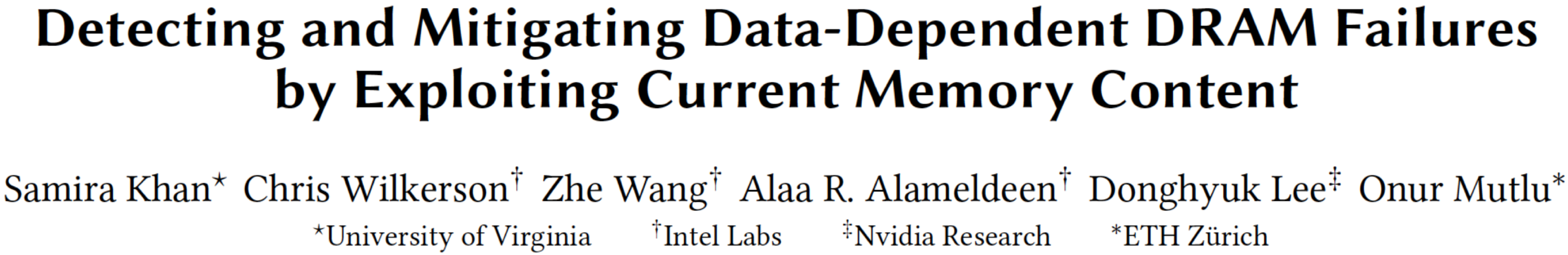
PARBOR: An Efficient System-Level Technique to Detect Data-Dependent Failures in DRAM

Samira Khan[*] Donghyuk Lee[†‡] Onur Mutlu[*†]

[*]University of Virginia [†]Carnegie Mellon University [‡]Nvidia [*]ETH Zürich

Handling Data-Dependent Failures [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)]



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

Samira Khan[⋆] Chris Wilkerson[†] Zhe Wang[†] Alaa R. Alameldeen[†] Donghyuk Lee[‡] Onur Mutlu[*]
[⋆]University of Virginia [†]Intel Labs [‡]Nvidia Research [*]ETH Zürich

Handling Both DPD and VRT [ISCA'17]

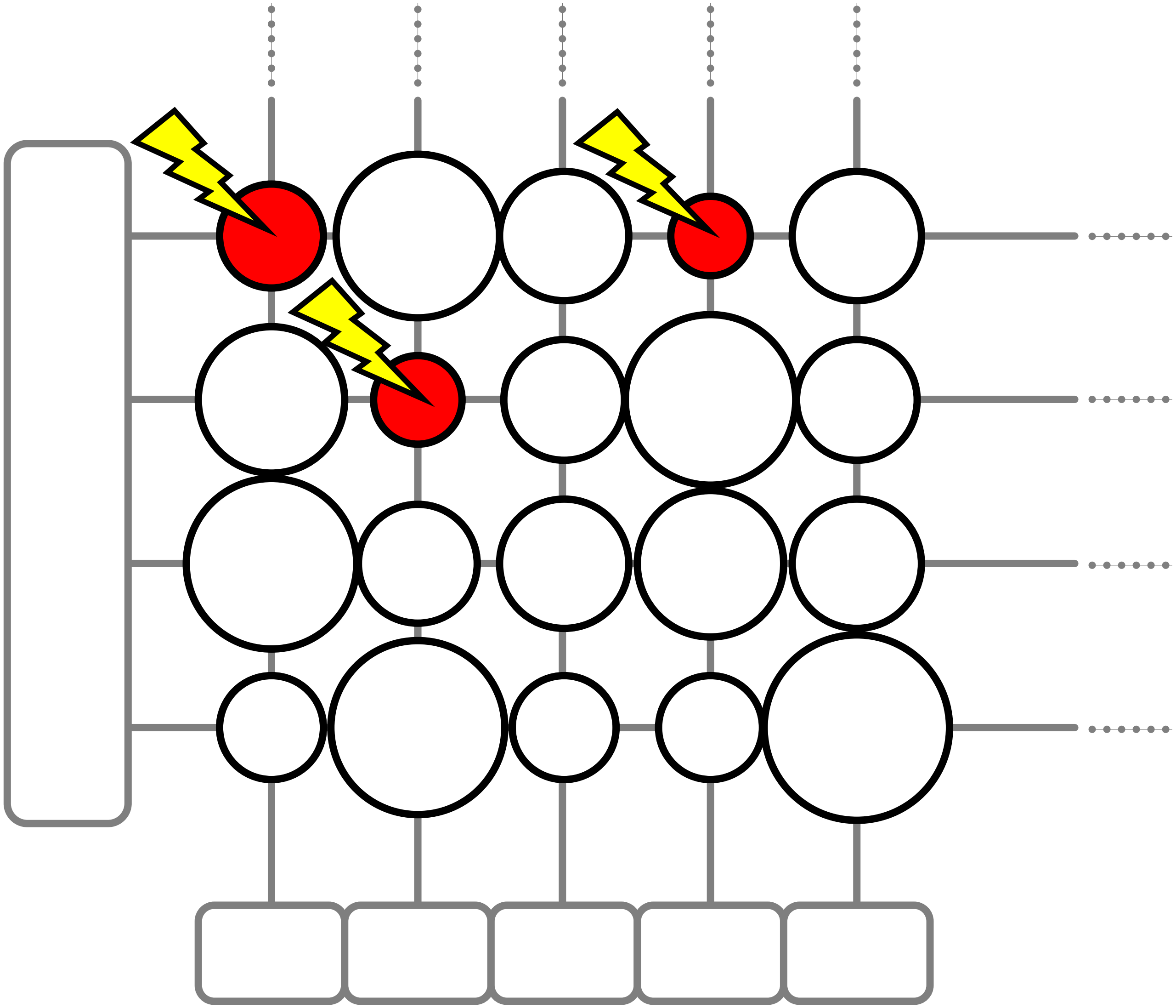
- Minesh Patel, Jeremie S. Kim, and Onur Mutlu,
  "The Reach Profiler (REAPER): Enabling the Mitigation of DRAM Retention Failures via Profiling at Aggressive Conditions"
  *Proceedings of the 44th International Symposium on Computer Architecture (**ISCA**)*, Toronto, Canada, June 2017.
  [Slides (pptx) (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

Goal: find *all* retention failures for a refresh interval *T* > default (64ms)



Process, voltage, temperature

Variable retention time

Data pattern dependence

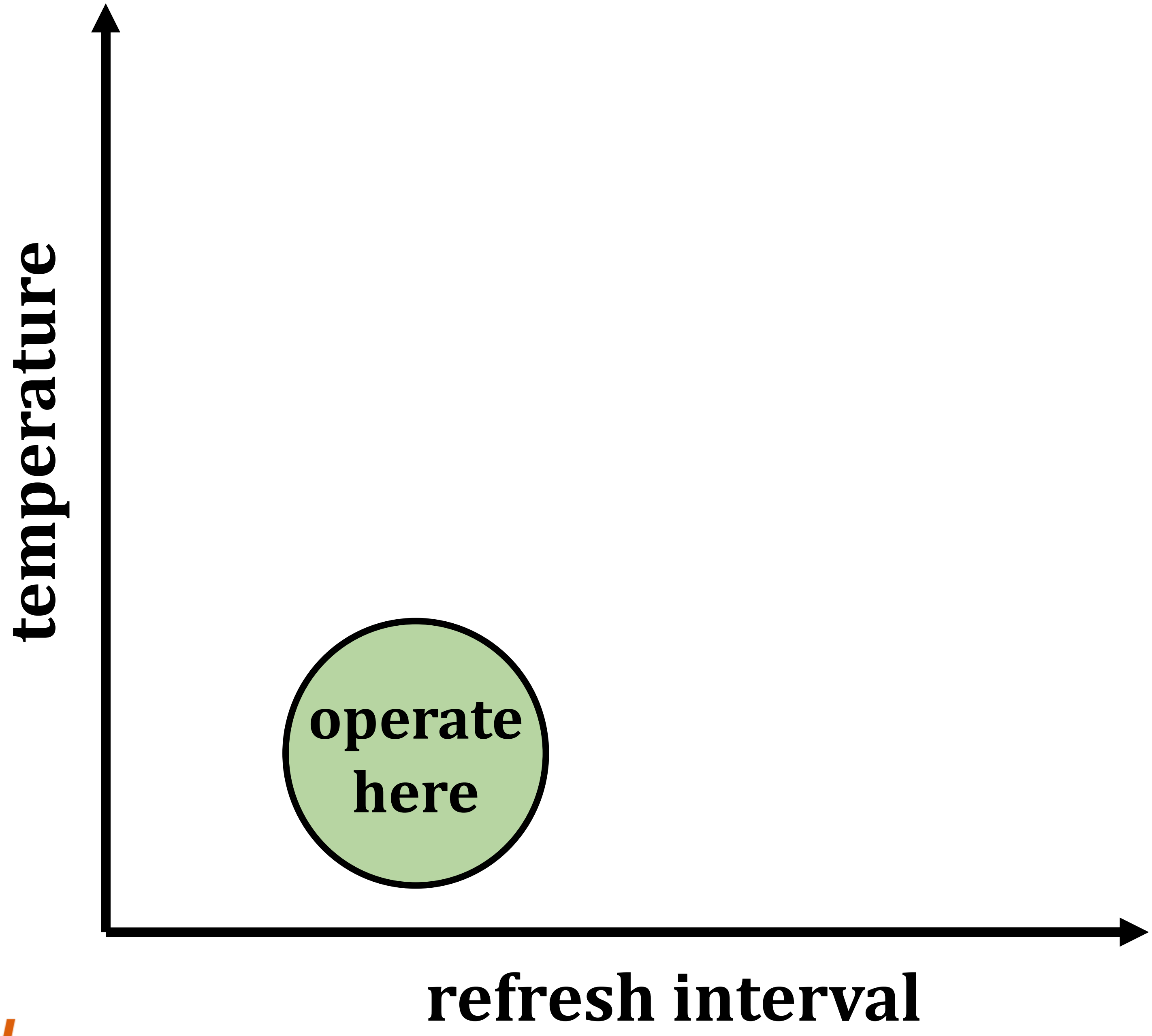
Characterization of 368 LPDDR4 DRAM Chips

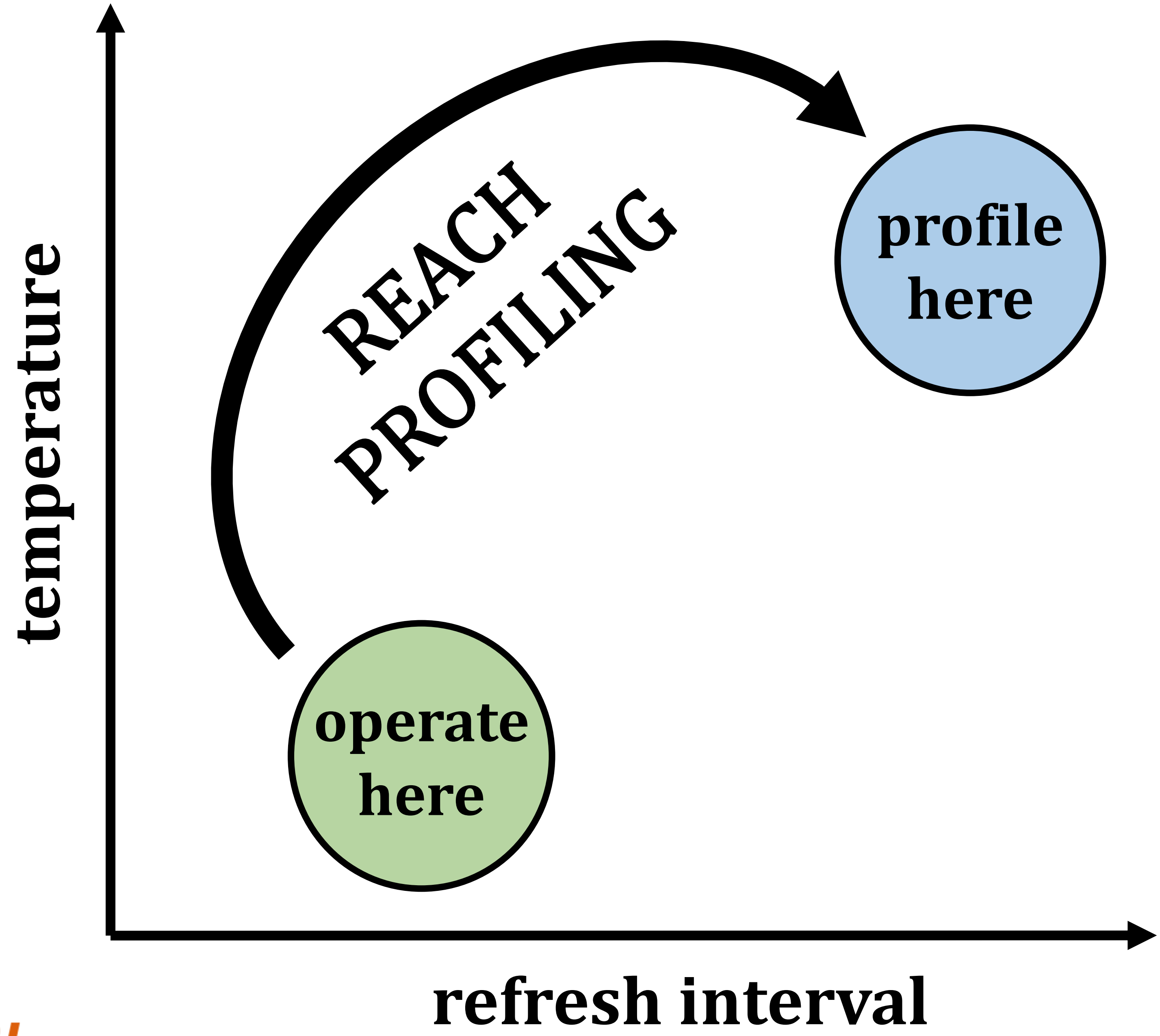
①

Cells are **more likely to fail** at an **increased (refresh interval | temperature)**

②

Complex tradeoff space between profiling **(speed & coverage & false positives)**


temperature
operate
here
refresh interval


temperature
REACH PROFILING
profile here
operate here
refresh interval

Reach Profiling

A new DRAM retention failure profiling methodology

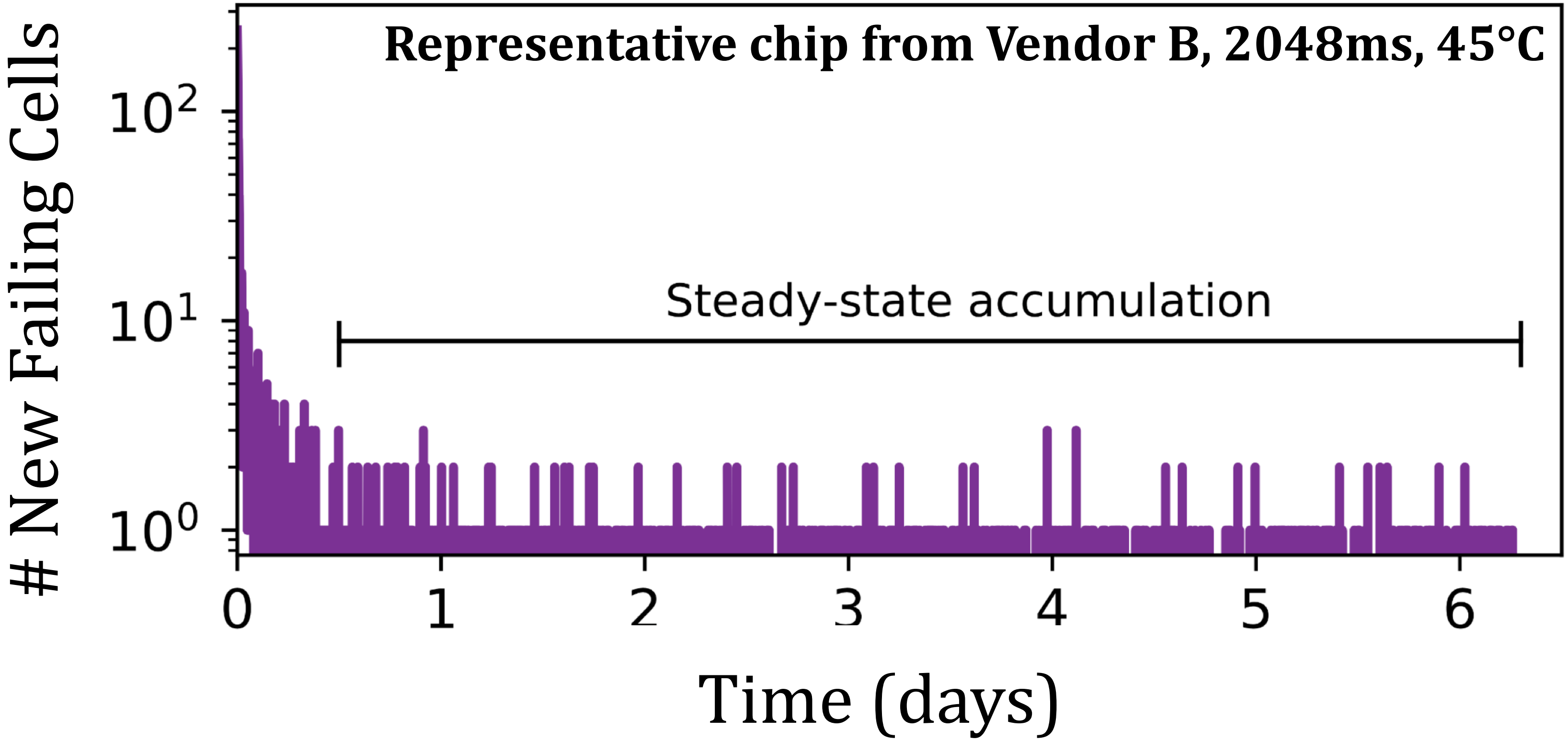
+ **Faster** and **more reliable** than current approaches

+ Enables **longer refresh intervals**

LPDDR4 Studies

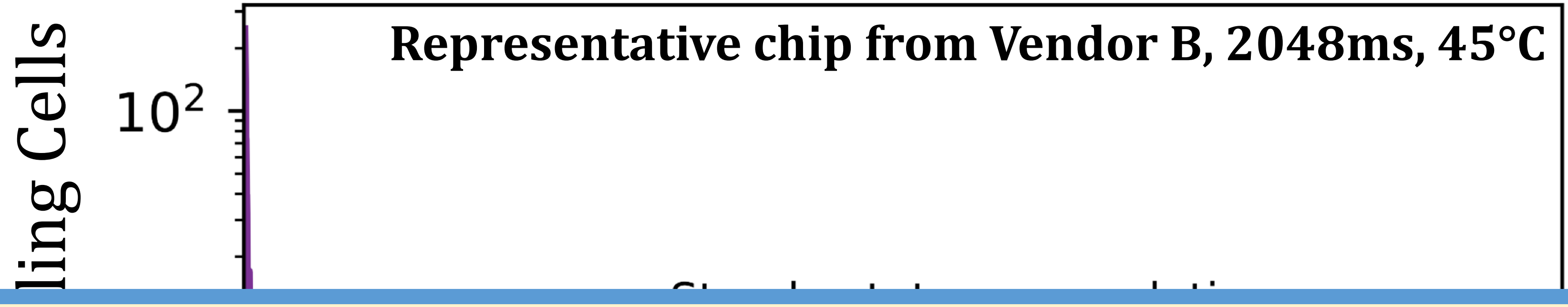
1. Temperature
2. Data Pattern Dependence
3. Retention Time Distributions
4. **Variable Retention Time**
5. **Individual Cell Characterization**

Long-term Continuous Profiling



- New failing cells continue to appear over time
  - Attributed to **variable retention time (VRT)**
- The set of failing cells changes over time

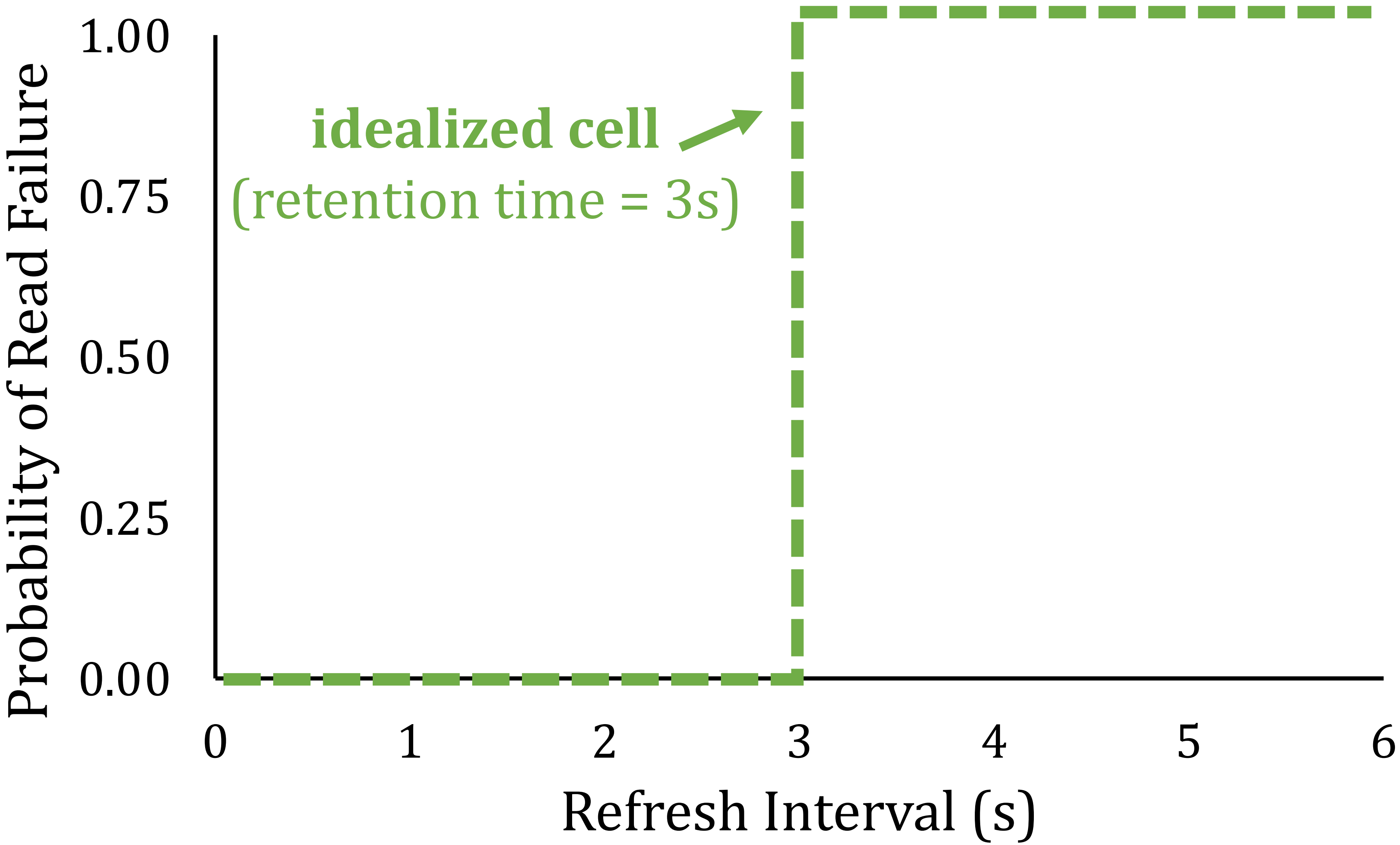
Long-term Continuous Profiling



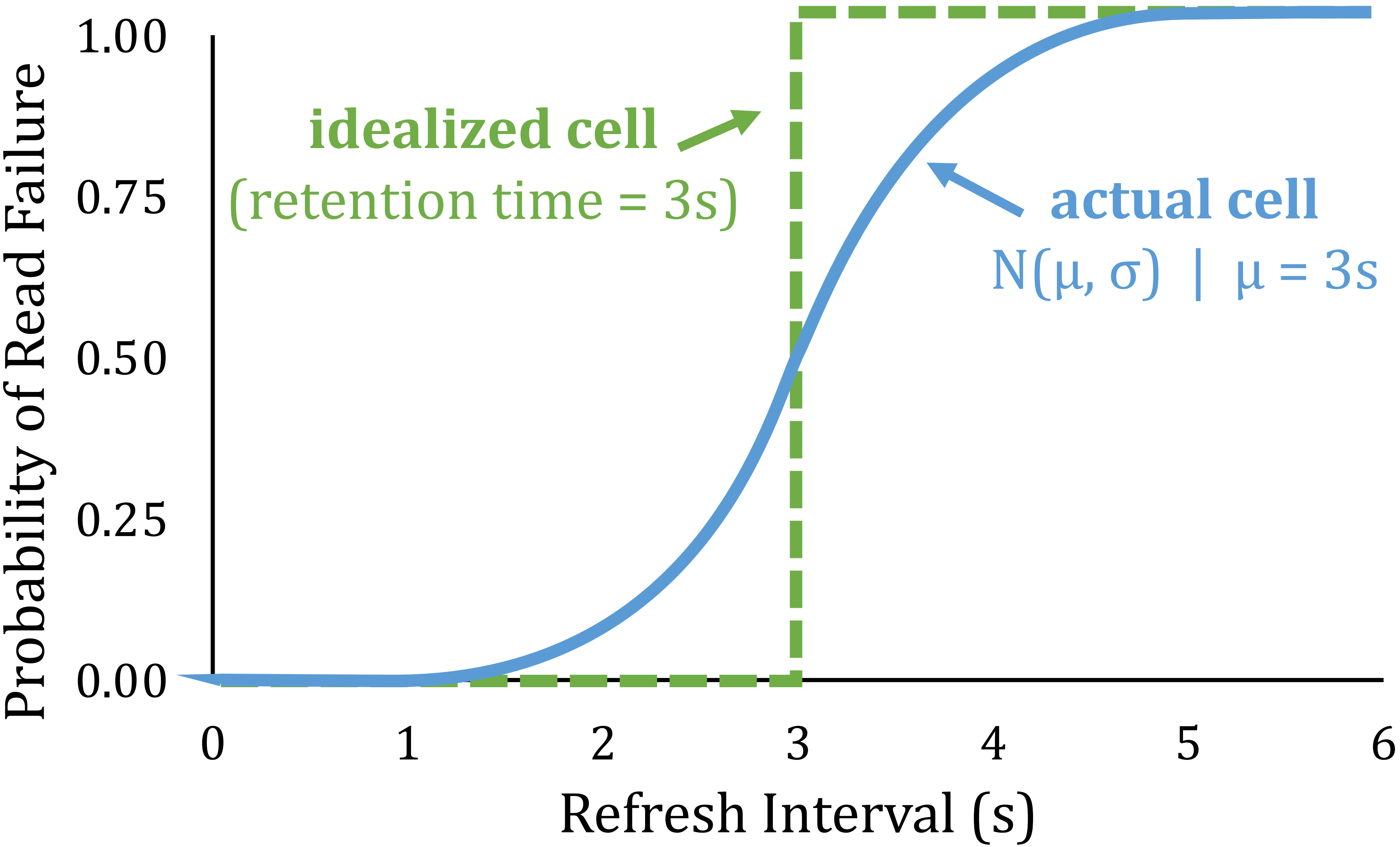
Error correction codes (ECC)
and online profiling are *necessary*
to manage new failing cells

- New failing cells continue to appear over time
  - Attributed to **variable retention time (VRT)**
- The set of failing cells changes over time

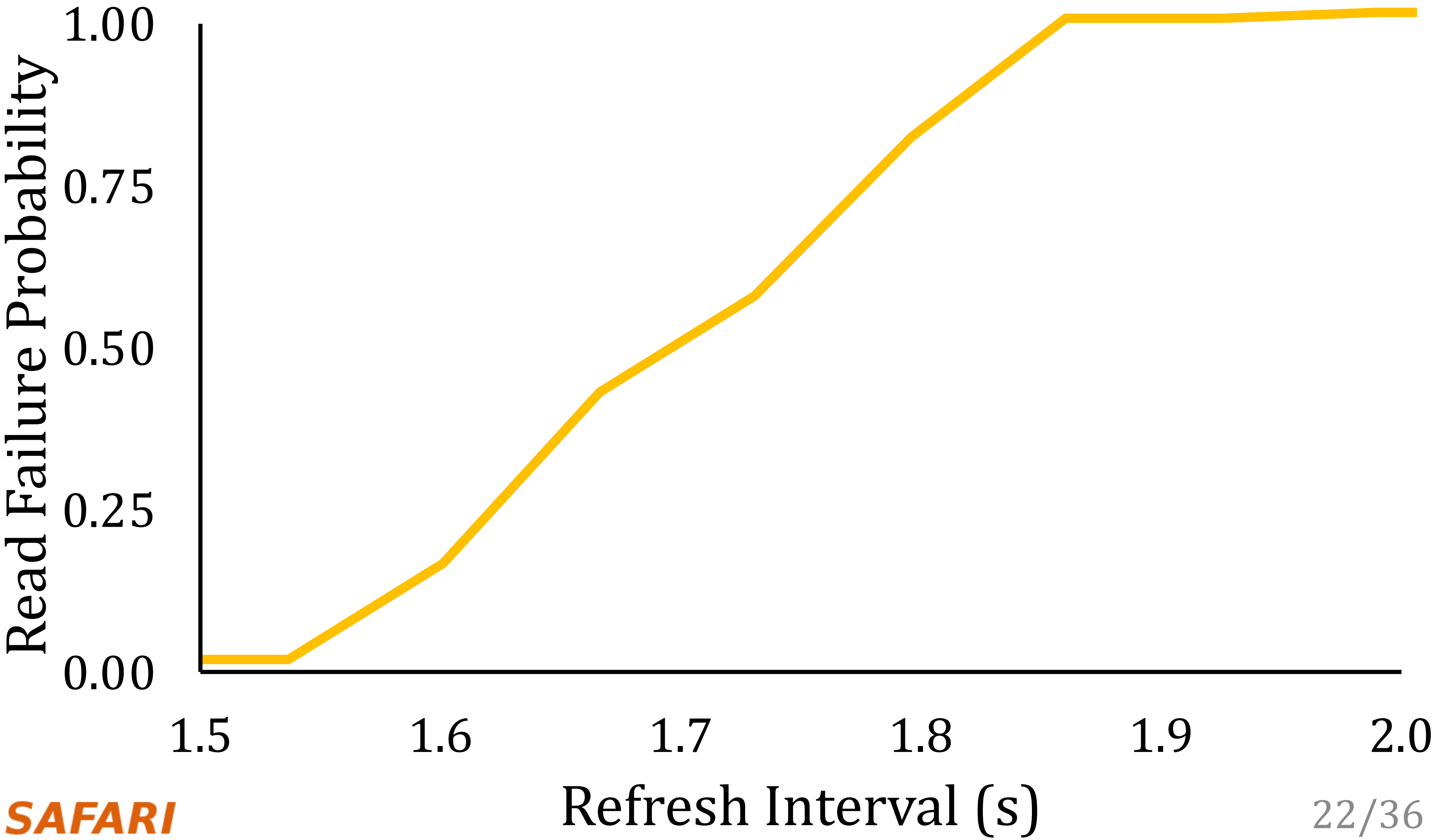
Single-cell Failure Probability (Cartoon)



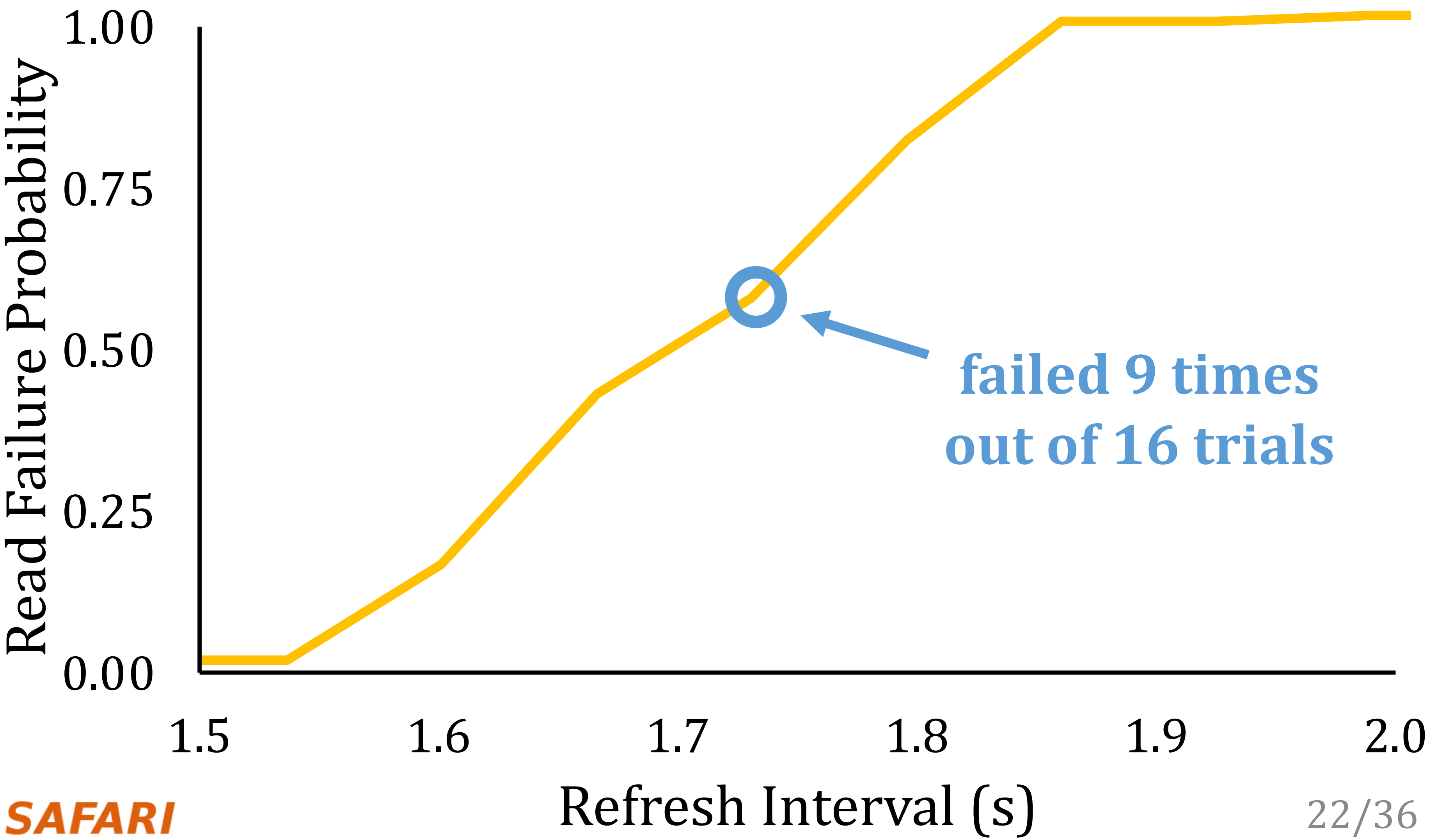
Single-cell Failure Probability (Cartoon)



Single-cell Failure Probability (Real)

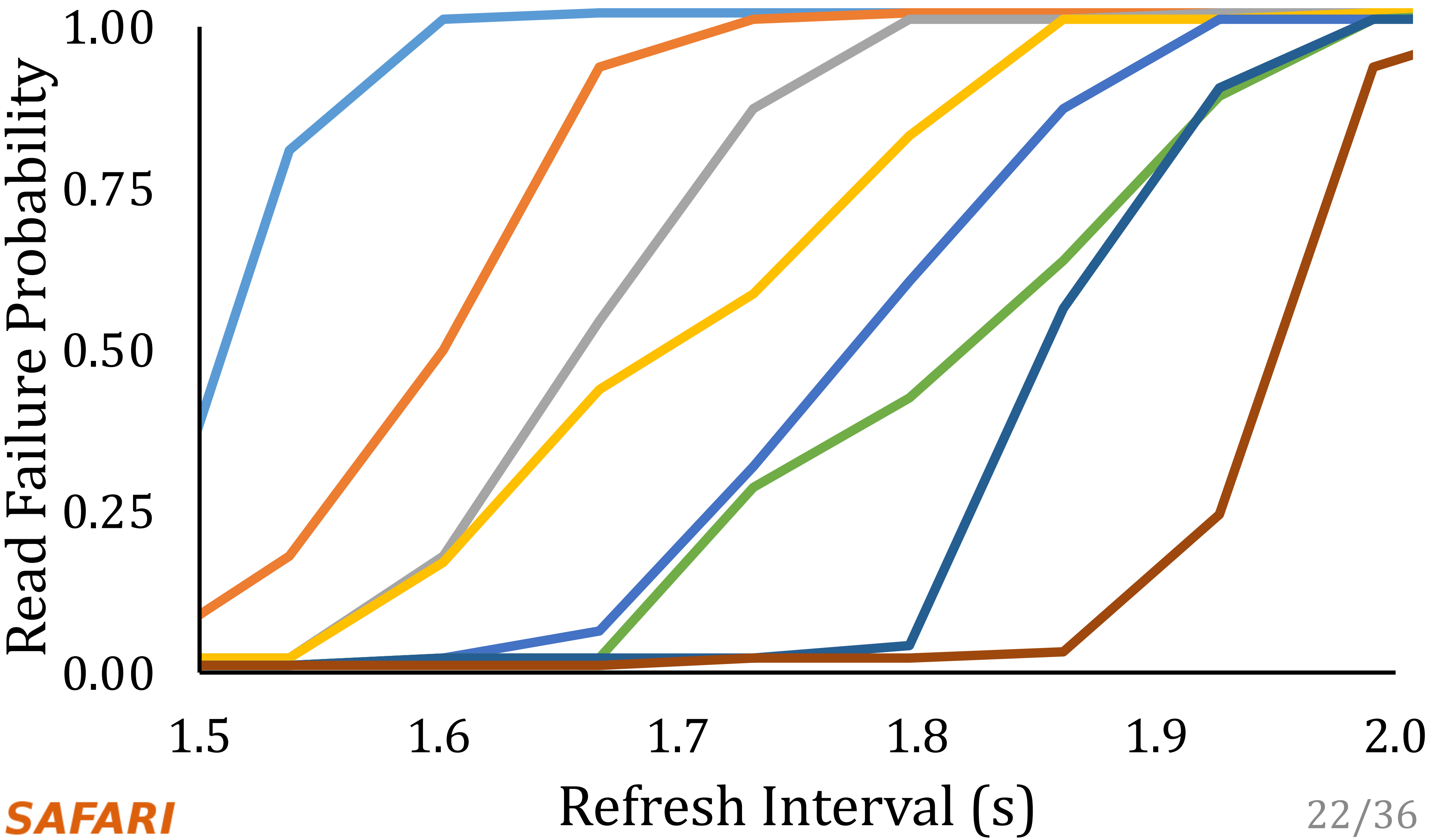


Single-cell Failure Probability (Real)

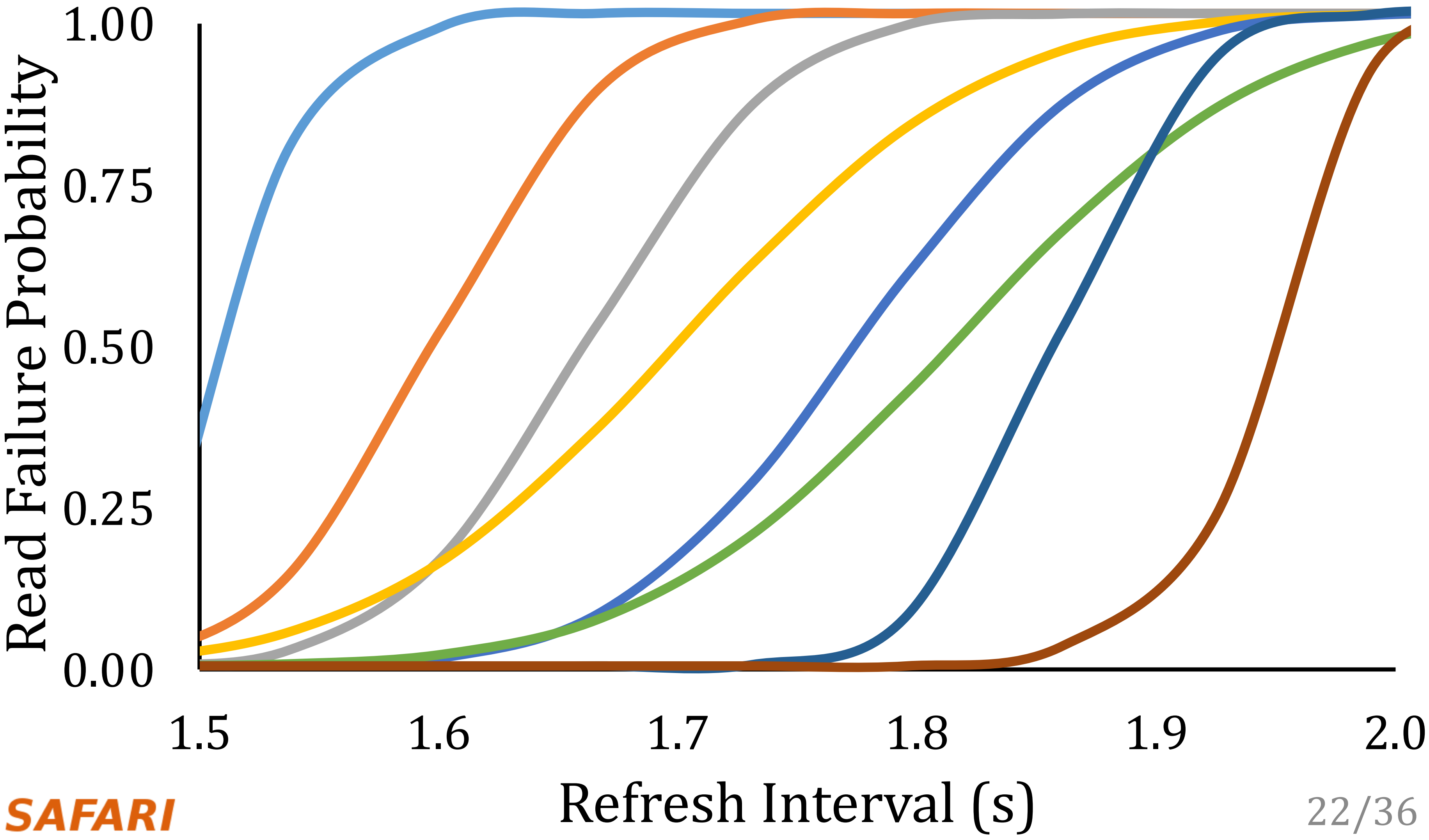


failed 9 times out of 16 trials

Single-cell Failure Probability (Real)

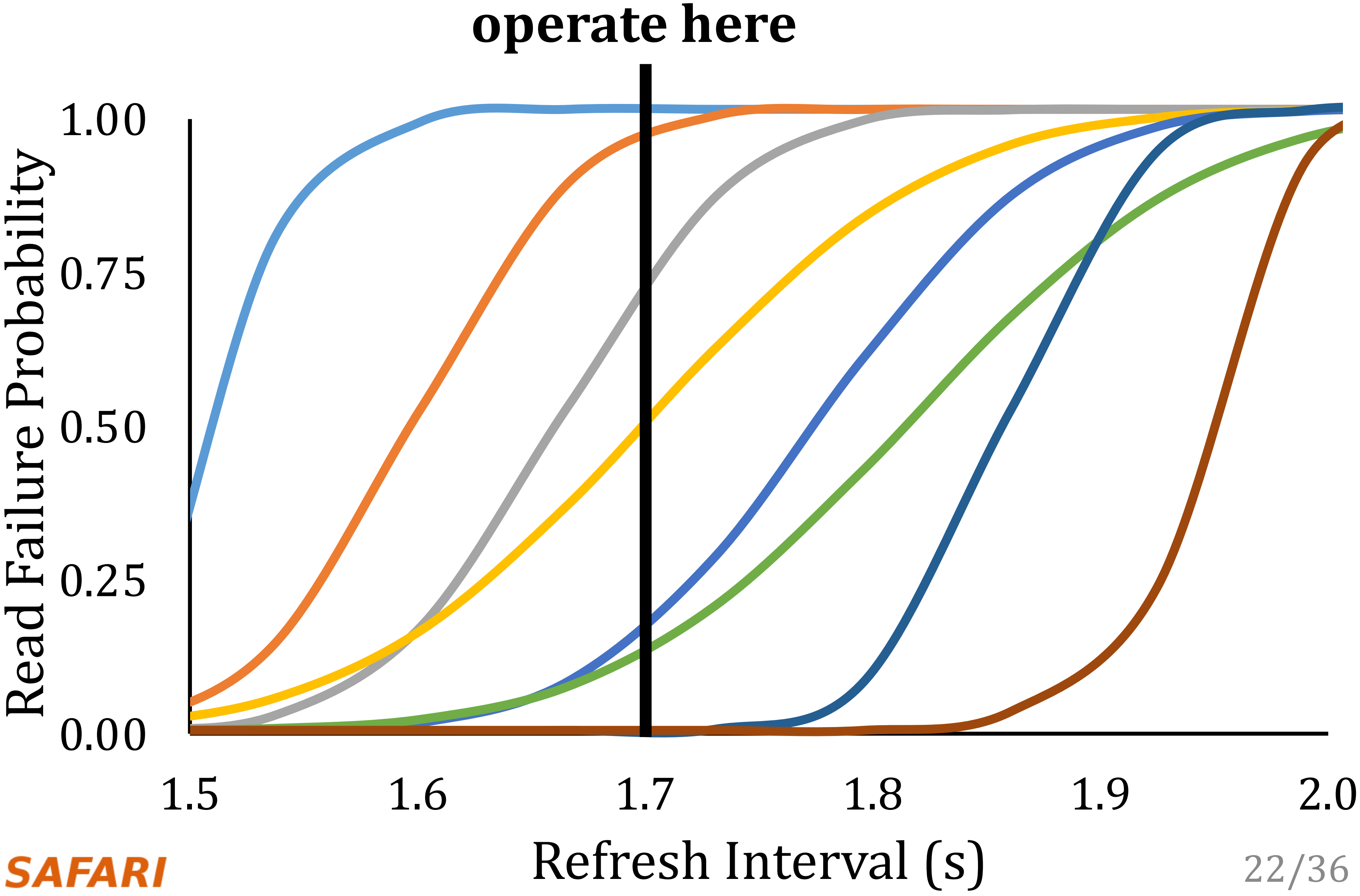


Single-cell Failure Probability (Real)

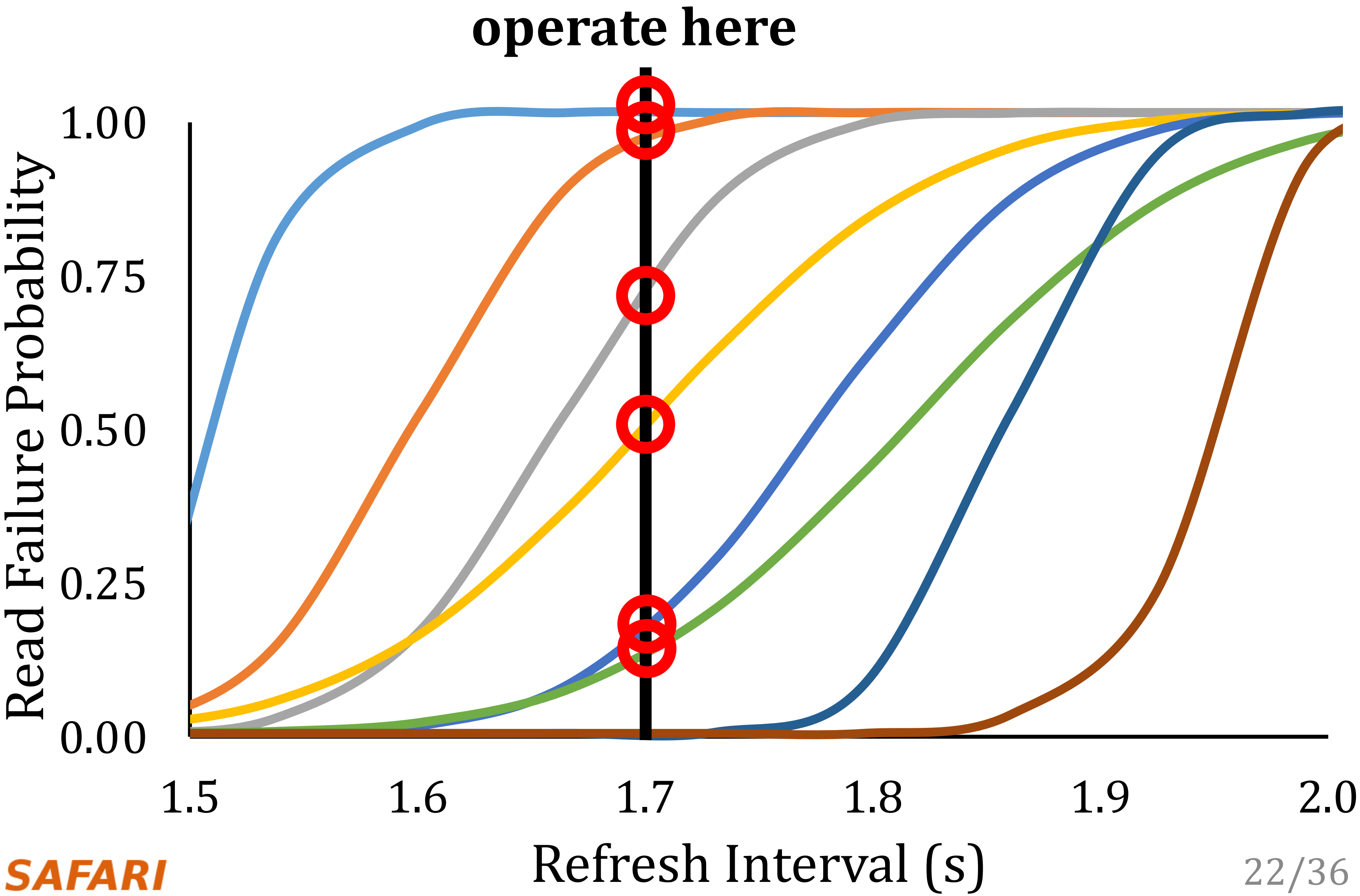


SAFARI

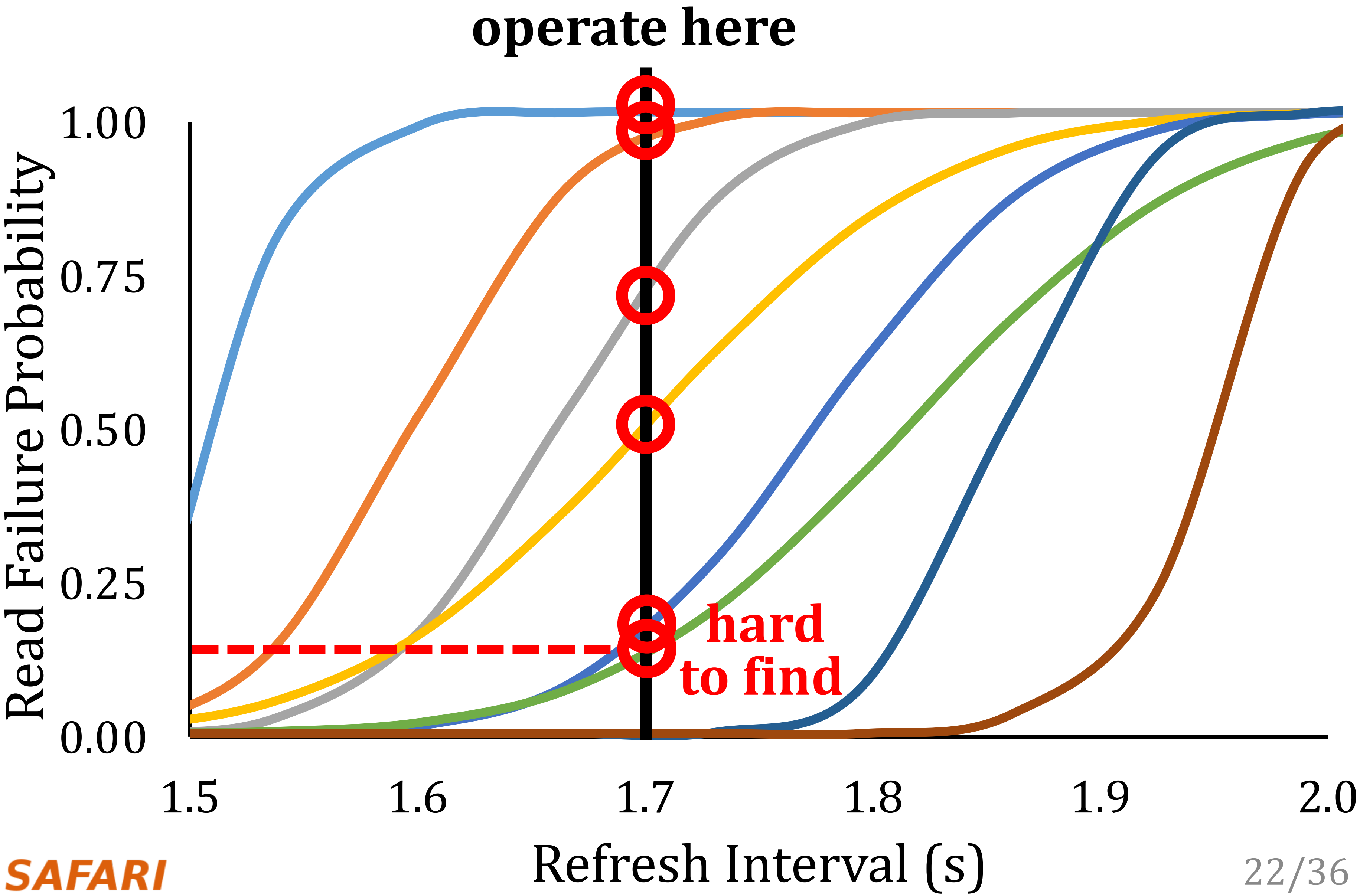
Single-cell Failure Probability (Real)



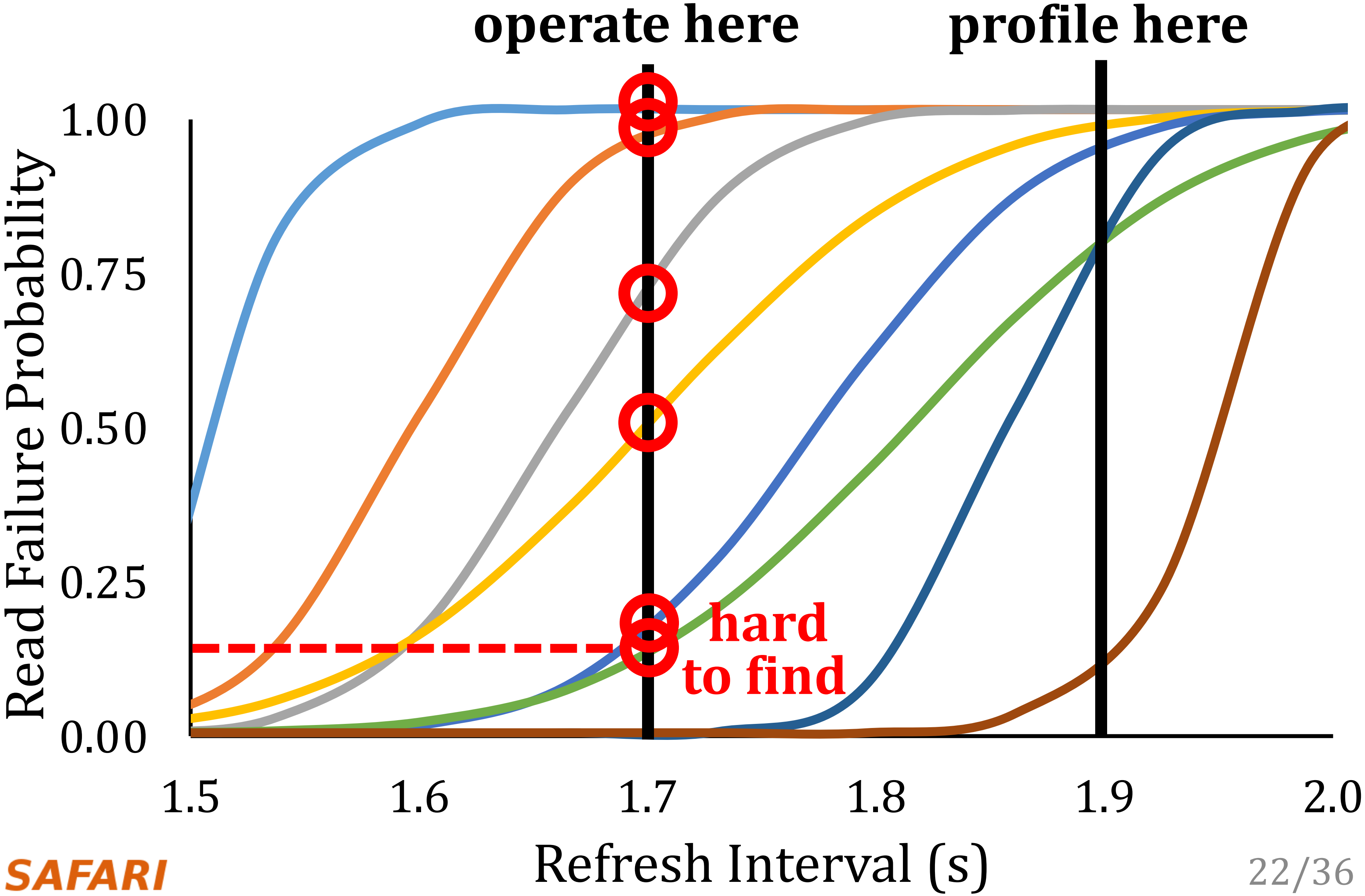
Single-cell Failure Probability (Real)



Single-cell Failure Probability (Real)

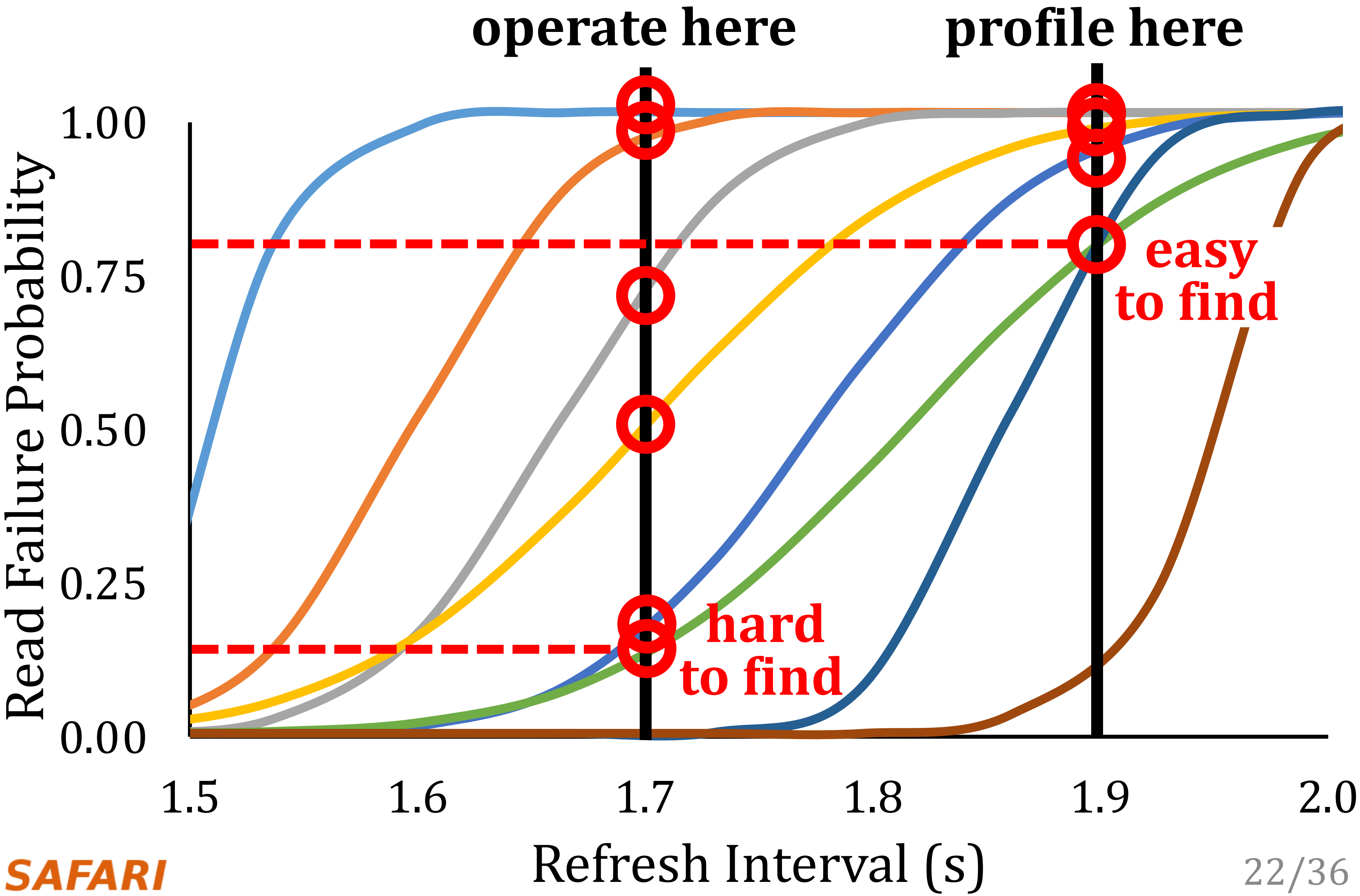


Single-cell Failure Probability (Real)



SAFARI

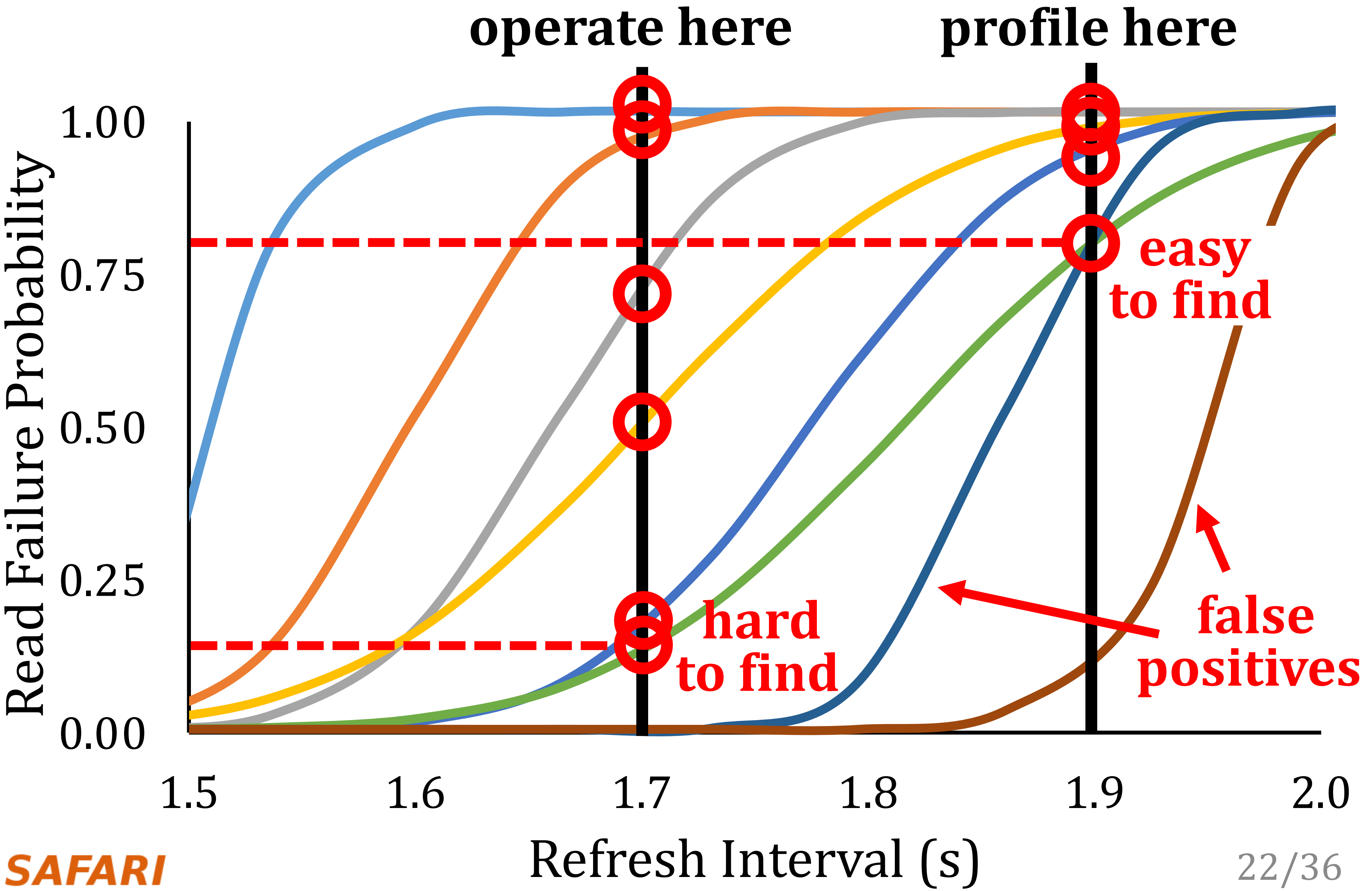
Single-cell Failure Probability (Real)



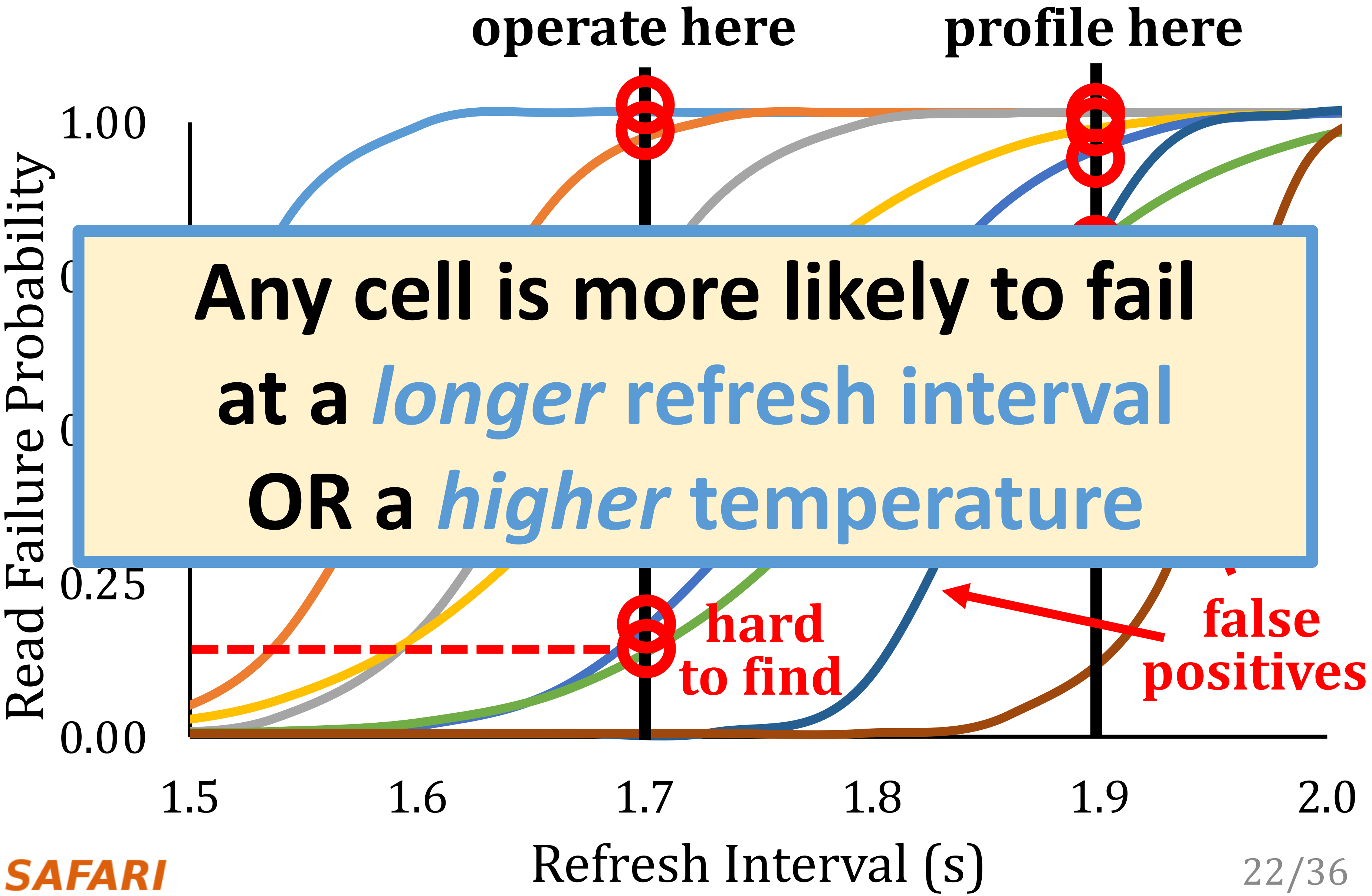
hard to find

easy to find

Single-cell Failure Probability (Real)



Single-cell Failure Probability (Real)



Any cell is more likely to fail at a *longer* refresh interval OR a *higher* temperature

Reach Profiling

Key idea: profile at a *longer refresh interval* and/or a *higher temperature*

- **Pros**
  - **Fast + Reliable:** reach profiling searches for cells *where they are most likely to fail*
- **Cons**
  - **False Positives:** profiler may identify cells that fail under profiling conditions, but not under operating conditions

Towards an Implementation

Reach profiling is a **general methodology**

3 key questions for an implementation:

What are desirable profiling conditions?

How often should the system profile?

What information does the profiler need?

Three Key Profiling Metrics

1. **Runtime:** how long profiling takes
2. **Coverage:** portion of all possible failures discovered by profiling
3. **False positives:** number of cells observed to fail during profiling but never during actual operation

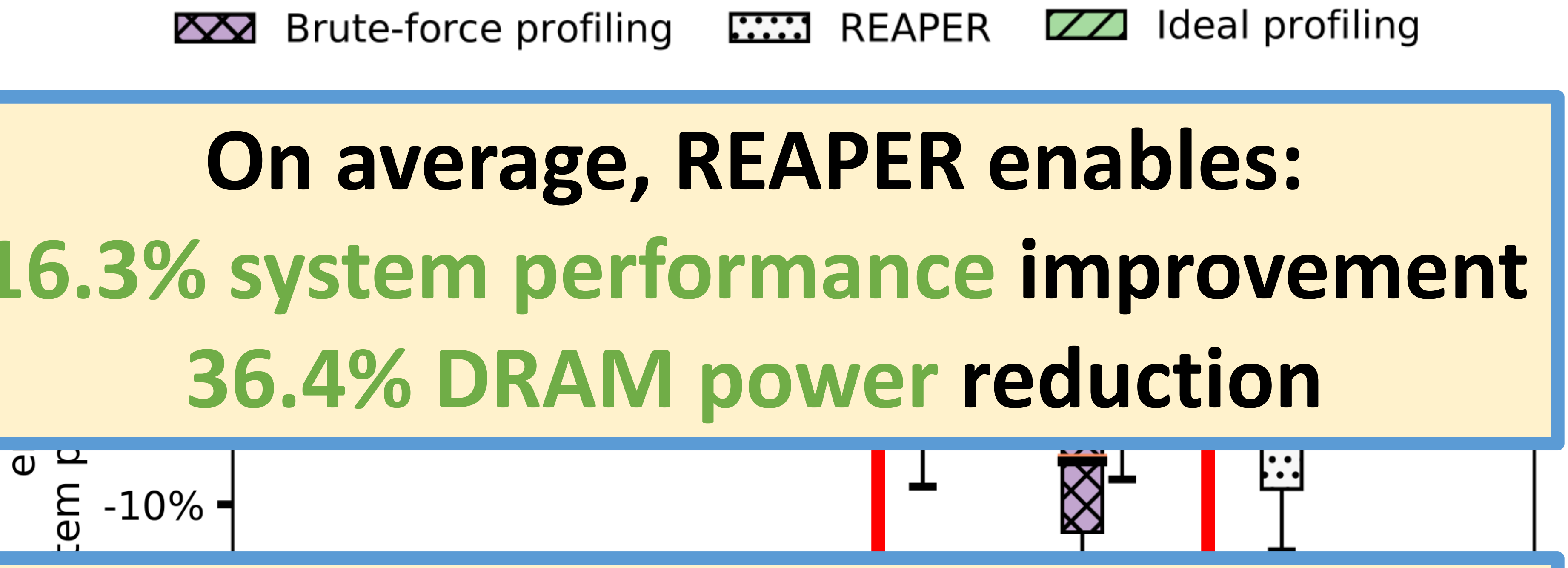
Three Key Profiling Metrics

1. **Runtime:** how long profiling takes

2. **Coverage:** portion of all possible failures discovered by profiling

We explore how these three metrics change under **many** different profiling conditions

Simulated End-to-end Performance



On average, REAPER enables:
16.3% system performance improvement
36.4% DRAM power reduction

REAPER enables longer refresh intervals,
which are unreasonable
using brute-force profiling

Reprofile rarely

Reprofile often

REAPER Summary

Problem:

- DRAM refresh performance and energy overhead is high
- Current approaches to retention failure profiling are slow or unreliable

Goals:

1. Thoroughly analyze profiling tradeoffs
2. Develop a **fast** and **reliable** profiling mechanism

Key Contributions:

1. **First** detailed characterization of 368 LPDDR4 DRAM chips
2. **Reach profiling:** Profile at a **longer refresh interval** or **higher temperature** than target conditions, where cells are more likely to fail

Evaluation:

- **2.5x** faster profiling with **99%** coverage and **50%** false positives
- REAPER enables **16.3% system performance improvement** and **36.4% DRAM power reduction**
- Enables longer refresh intervals that were previously unreasonable

The Takeaway, Reinforced

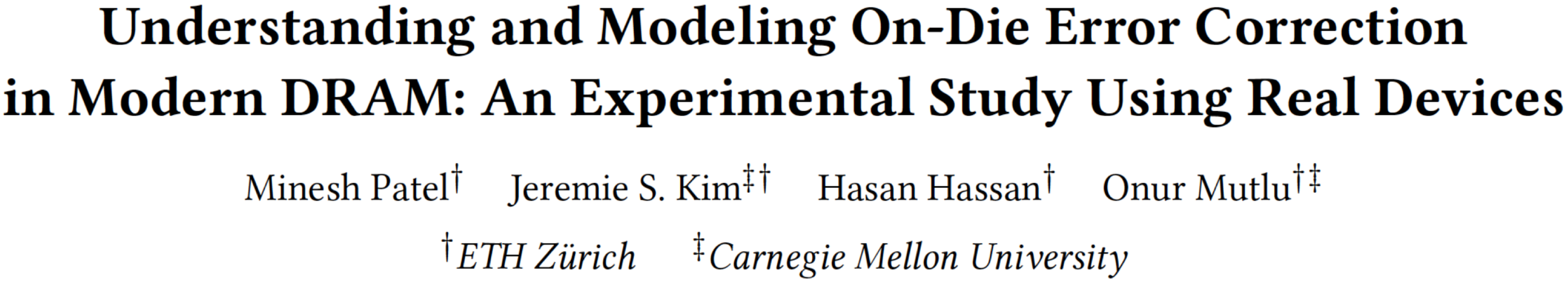
Main Memory Needs

Intelligent Controllers

for Reliability & Security

Understanding In-DRAM ECC

- Minesh Patel, Jeremie S. Kim, Hasan Hassan, and Onur Mutlu, **"Understanding and Modeling On-Die Error Correction in Modern DRAM: An Experimental Study Using Real Devices"** *Proceedings of the 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Portland, OR, USA, June 2019.
  [Source Code for EINSim, the Error Inference Simulator]
  Best paper session.



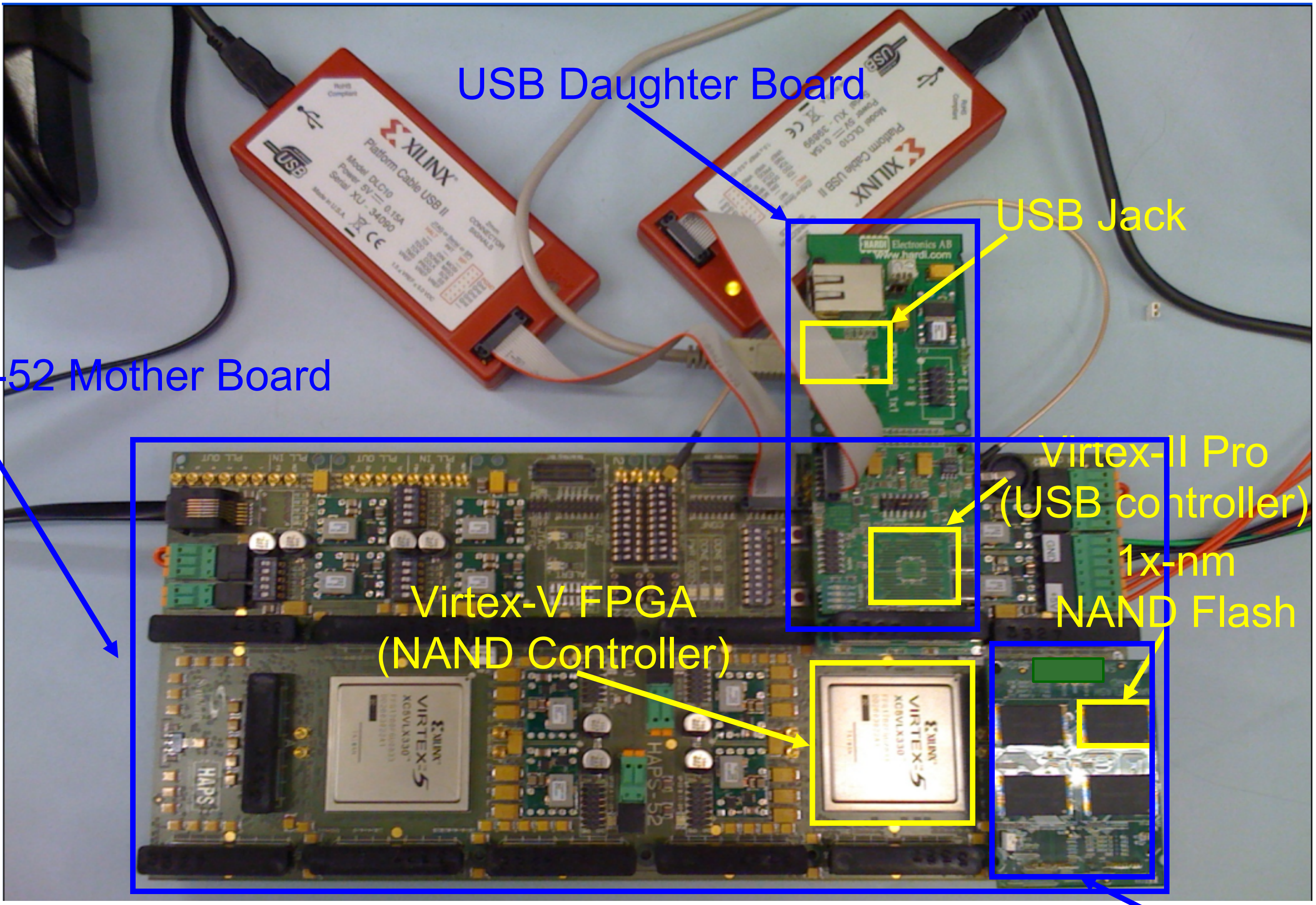
Understanding and Modeling On-Die Error Correction in Modern DRAM: An Experimental Study Using Real Devices

Minesh Patel[†] Jeremie S. Kim[‡†] Hasan Hassan[†] Onur Mutlu[†‡]

[†]*ETH Zürich* [‡]*Carnegie Mellon University*

Understanding Flash Memory Vulnerabilities

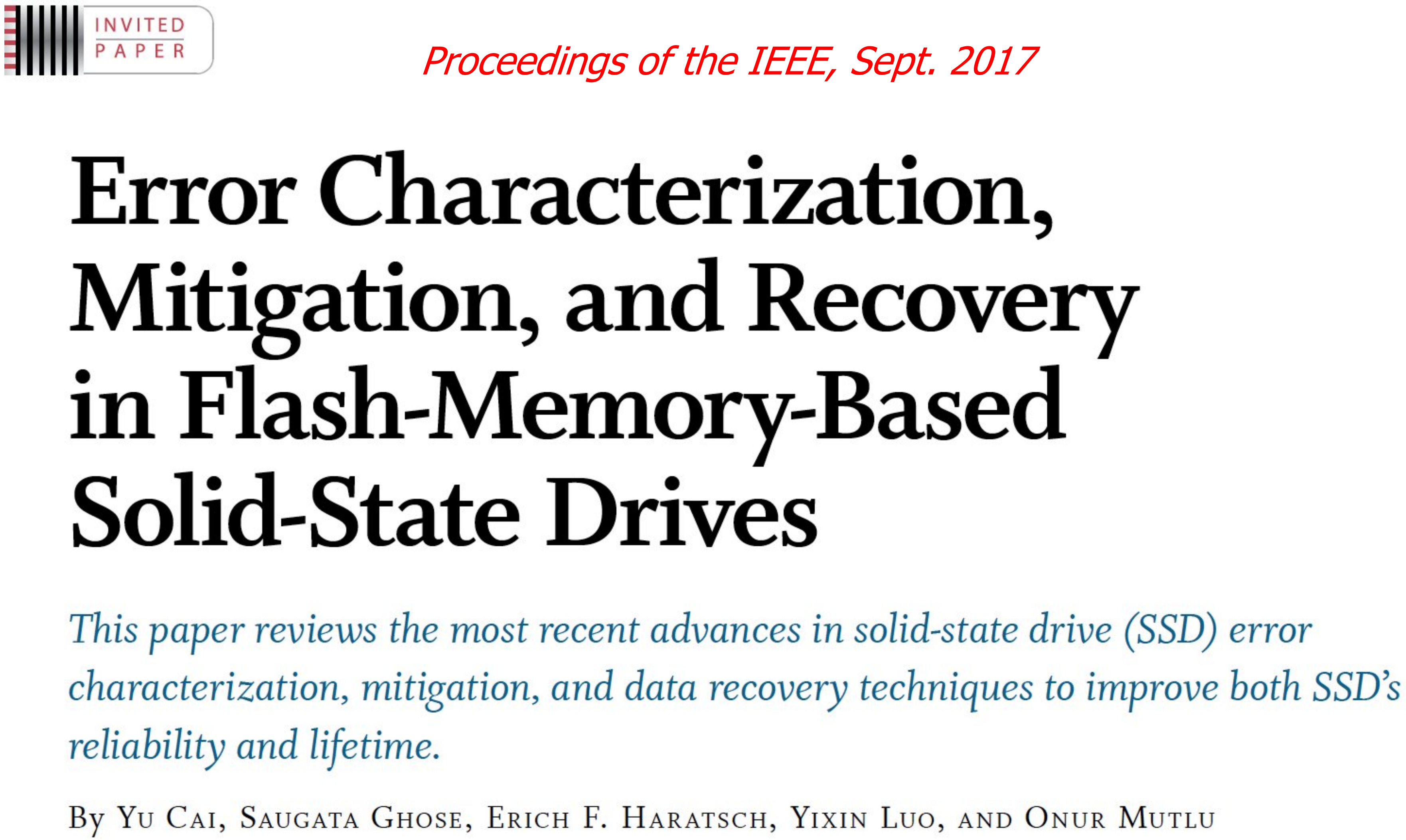
Understand and Model with Experiments (Flash)



[DATE 2012, ICCD 2012, DATE 2013, ITJ 2013, ICCD 2013, SIGMETRICS 2014, HPCA 2015, DSN 2015, MSST 2015, JSAC 2016, HPCA 2017, DFRWS 2017, PIEEE 2017, HPCA 2018, SIGMETRICS 2018]

Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017.

Understanding Flash Memory Reliability



INVITED PAPER

Proceedings of the IEEE, Sept. 2017

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

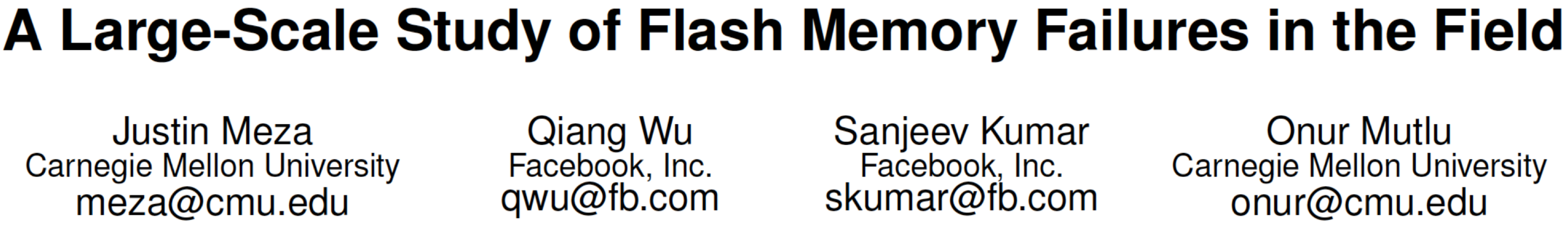
This paper reviews the most recent advances in solid-state drive (SSD) error characterization, mitigation, and data recovery techniques to improve both SSD's reliability and lifetime.

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

https://arxiv.org/pdf/1706.08642

Understanding Flash Memory Reliability

- Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu, **"A Large-Scale Study of Flash Memory Errors in the Field"** *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems* (***SIGMETRICS***), Portland, OR, June 2015.
[Slides (pptx) (pdf)] [Coverage at ZDNet] [Coverage on The Register] [Coverage on TechSpot] [Coverage on The Tech Report]



A Large-Scale Study of Flash Memory Failures in the Field

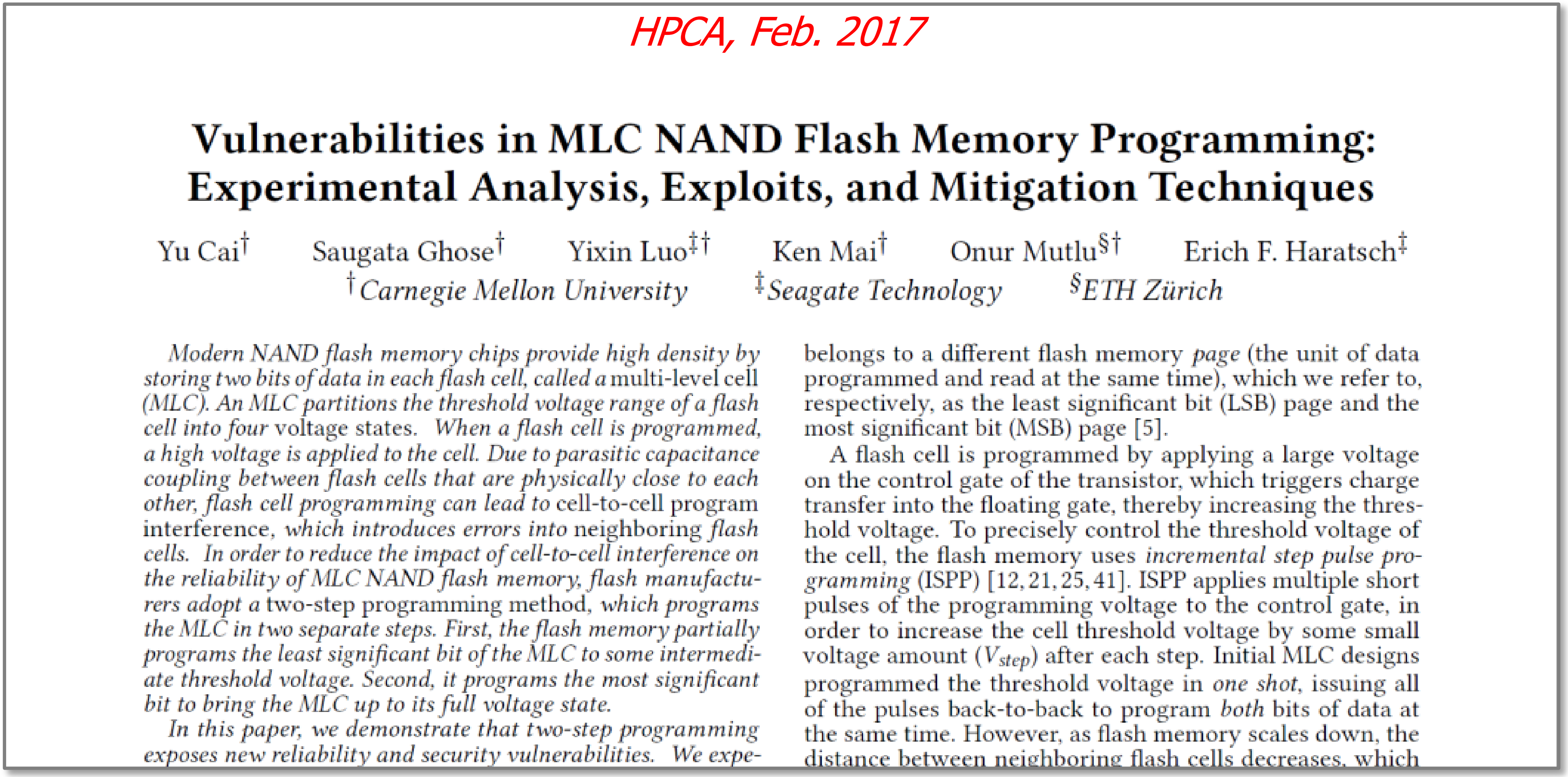
Justin Meza
Carnegie Mellon University
meza@cmu.edu

Qiang Wu
Facebook, Inc.
qwu@fb.com

Sanjeev Kumar
Facebook, Inc.
skumar@fb.com

Onur Mutlu
Carnegie Mellon University
onur@cmu.edu

NAND Flash Vulnerabilities **[HPCA'17]**



HPCA, Feb. 2017

Vulnerabilities in MLC NAND Flash Memory Programming: Experimental Analysis, Exploits, and Mitigation Techniques

Yu Cai[†] Saugata Ghose[†] Yixin Luo[‡†] Ken Mai[†] Onur Mutlu[§†] Erich F. Haratsch[‡]
[†]*Carnegie Mellon University* [‡]*Seagate Technology* [§]*ETH Zürich*

Modern NAND flash memory chips provide high density by storing two bits of data in each flash cell, called a multi-level cell *(MLC). An MLC partitions the threshold voltage range of a flash cell into four* voltage states. *When a flash cell is programmed, a high voltage is applied to the cell. Due to parasitic capacitance coupling between flash cells that are physically close to each other, flash cell programming can lead to* cell-to-cell program interference, *which introduces errors into* neighboring *flash cells. In order to reduce the impact of cell-to-cell interference on the reliability of MLC NAND flash memory, flash manufacturers adopt a* two-step programming method, *which programs the MLC in two separate steps. First, the flash memory partially programs the least significant bit of the MLC to some intermediate threshold voltage. Second, it programs the most significant bit to bring the MLC up to its full voltage state.*

In this paper, we demonstrate that two-step programming exposes new reliability and security vulnerabilities. We expe-

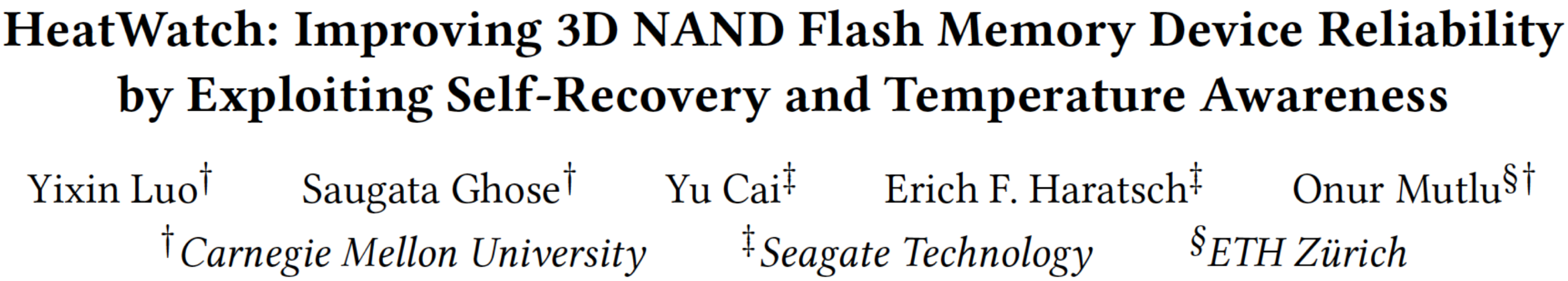
belongs to a different flash memory *page* (the unit of data programmed and read at the same time), which we refer to, respectively, as the least significant bit (LSB) page and the most significant bit (MSB) page [5].

A flash cell is programmed by applying a large voltage on the control gate of the transistor, which triggers charge transfer into the floating gate, thereby increasing the threshold voltage. To precisely control the threshold voltage of the cell, the flash memory uses *incremental step pulse programming* (ISPP) [12, 21, 25, 41]. ISPP applies multiple short pulses of the programming voltage to the control gate, in order to increase the cell threshold voltage by some small voltage amount (V_{step}) after each step. Initial MLC designs programmed the threshold voltage in *one shot*, issuing all of the pulses back-to-back to program *both* bits of data at the same time. However, as flash memory scales down, the distance between neighboring flash cells decreases, which

https://people.inf.ethz.ch/omutlu/pub/flash-memory-programming-vulnerabilities_hpca17.pdf

3D NAND Flash Reliability I **[HPCA'18]**

- Yixin Luo, Saugata Ghose, Yu Cai, Erich F. Haratsch, and Onur Mutlu, **"HeatWatch: Improving 3D NAND Flash Memory Device Reliability by Exploiting Self-Recovery and Temperature-Awareness"**
  *Proceedings of the 24th International Symposium on High-Performance Computer Architecture (**HPCA**)*, Vienna, Austria, February 2018.
  [Lightning Talk Video]
  [Slides (pptx) (pdf)] [Lightning Session Slides (pptx) (pdf)]



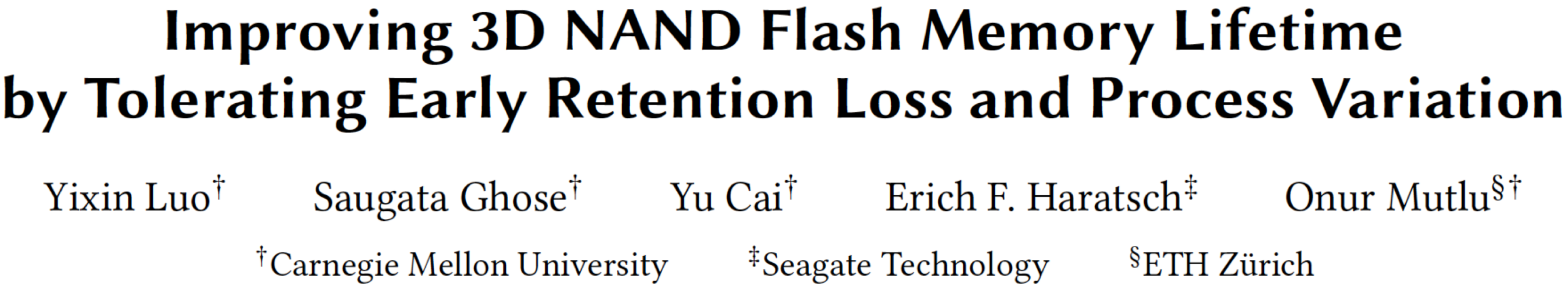
HeatWatch: Improving 3D NAND Flash Memory Device Reliability by Exploiting Self-Recovery and Temperature Awareness

Yixin Luo† Saugata Ghose† Yu Cai‡ Erich F. Haratsch‡ Onur Mutlu§†

†*Carnegie Mellon University* ‡*Seagate Technology* §*ETH Zürich*

3D NAND Flash Reliability II [SIGMETRICS'18]

- Yixin Luo, Saugata Ghose, Yu Cai, Erich F. Haratsch, and Onur Mutlu, **"Improving 3D NAND Flash Memory Lifetime by Tolerating Early Retention Loss and Process Variation"**
Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems (***SIGMETRICS***), Irvine, CA, USA, June 2018.
[Abstract]
[POMACS Journal Version (same content, different format)]
[Slides (pptx) (pdf)]



Improving 3D NAND Flash Memory Lifetime by Tolerating Early Retention Loss and Process Variation

Yixin Luo[†] Saugata Ghose[†] Yu Cai[†] Erich F. Haratsch[‡] Onur Mutlu[§†]

[†]Carnegie Mellon University [‡]Seagate Technology [§]ETH Zürich

Another Talk: NAND Flash Memory Robustness

- Yu Cai, Saugata Ghose, Erich F. Haratsch, Yixin Luo, and Onur Mutlu, **"Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives"** *to appear in Proceedings of the IEEE*, 2017.

Cai+, "Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis," DATE 2012.

Cai+, "Flash Correct-and-Refresh: Retention-Aware Error Management for Increased Flash Memory Lifetime," ICCD 2012.

Cai+, "Threshold Voltage Distribution in MLC NAND Flash Memory: Characterization, Analysis and Modeling," DATE 2013.

Cai+, "Error Analysis and Retention-Aware Error Management for NAND Flash Memory," Intel Technology Journal 2013.

Cai+, "Program Interference in MLC NAND Flash Memory: Characterization, Modeling, and Mitigation," ICCD 2013.

Cai+, "Neighbor-Cell Assisted Error Correction for MLC NAND Flash Memories," SIGMETRICS 2014.

Cai+,"Data Retention in MLC NAND Flash Memory: Characterization, Optimization and Recovery," HPCA 2015.

Cai+, "Read Disturb Errors in MLC NAND Flash Memory: Characterization and Mitigation," DSN 2015.

Luo+, "WARM: Improving NAND Flash Memory Lifetime with Write-hotness Aware Retention Management," MSST 2015.

Meza+, "A Large-Scale Study of Flash Memory Errors in the Field," SIGMETRICS 2015.

Luo+, "Enabling Accurate and Practical Online Flash Channel Modeling for Modern MLC NAND Flash Memory," IEEE JSAC 2016.

Cai+, "Vulnerabilities in MLC NAND Flash Memory Programming: Experimental Analysis, Exploits, and Mitigation Techniques," HPCA 2017.

Fukami+, "Improving the Reliability of Chip-Off Forensic Analysis of NAND Flash Memory Devices," DFRWS EU 2017.

Luo+, "HeatWatch: Improving 3D NAND Flash Memory Device Reliability by Exploiting Self-Recovery and Temperature-Awareness," HPCA 2018.

Luo+, "Improving 3D NAND Flash Memory Lifetime by Tolerating Early Retention Loss and Process Variation," SIGMETRICS 2018.

Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017.

Two Other Solution Directions

There are Two Other Solution Directions

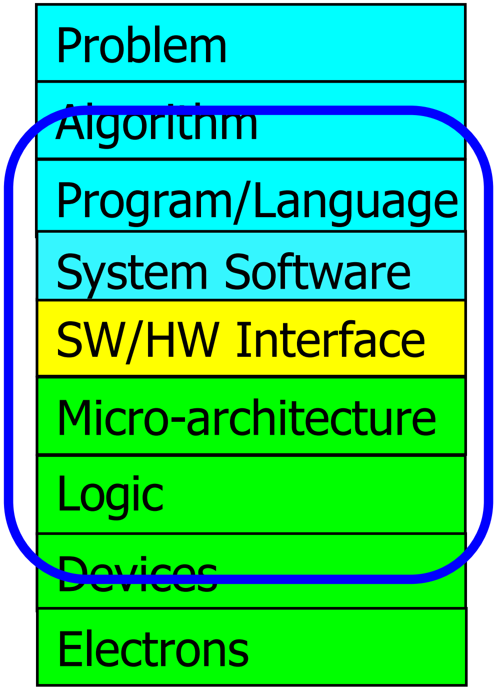
- **New Technologies:** Replace or (more likely) augment DRAM with a different technology
  - Non-volatile memories

- **Embracing Un-reliability:**

  Design memories with different reliability and store data intelligently across them

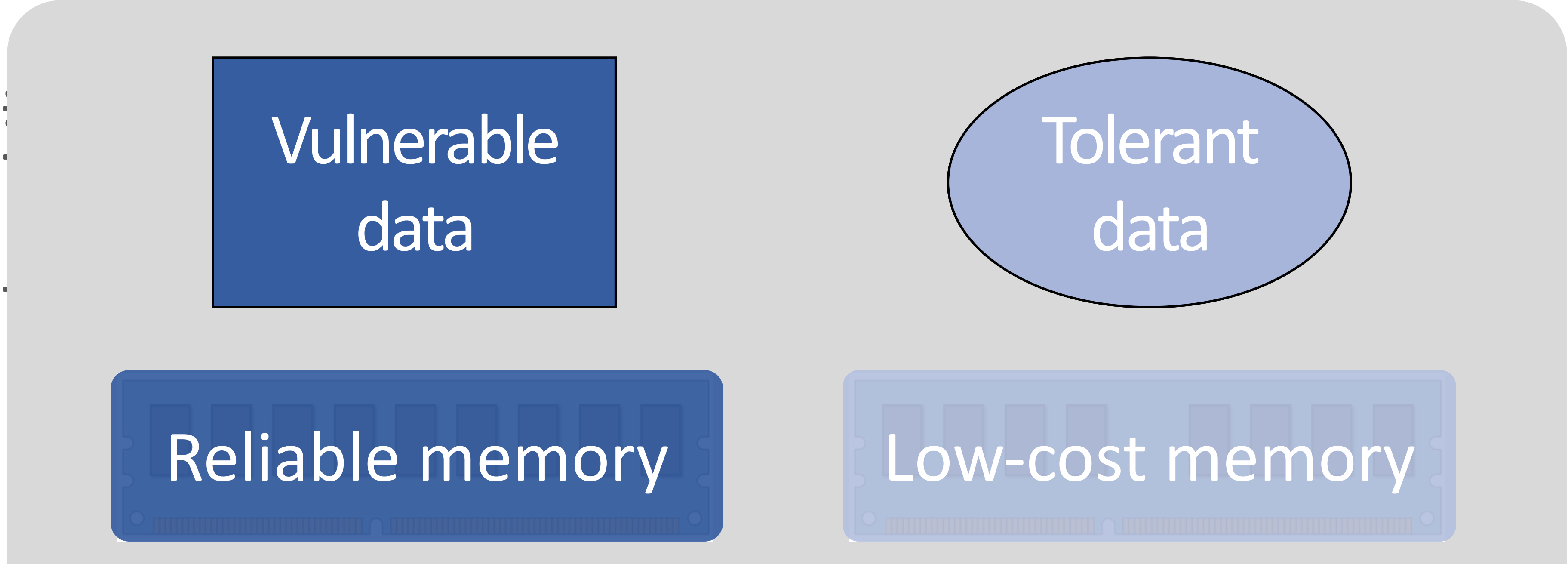
  [Luo+ DSN 2014]

- …



Fundamental solutions to security require co-design across the hierarchy

Exploiting Memory Error Tolerance with Hybrid Memory Systems



Vulnerable data

Tolerant data

Reliable memory

Low-cost memory

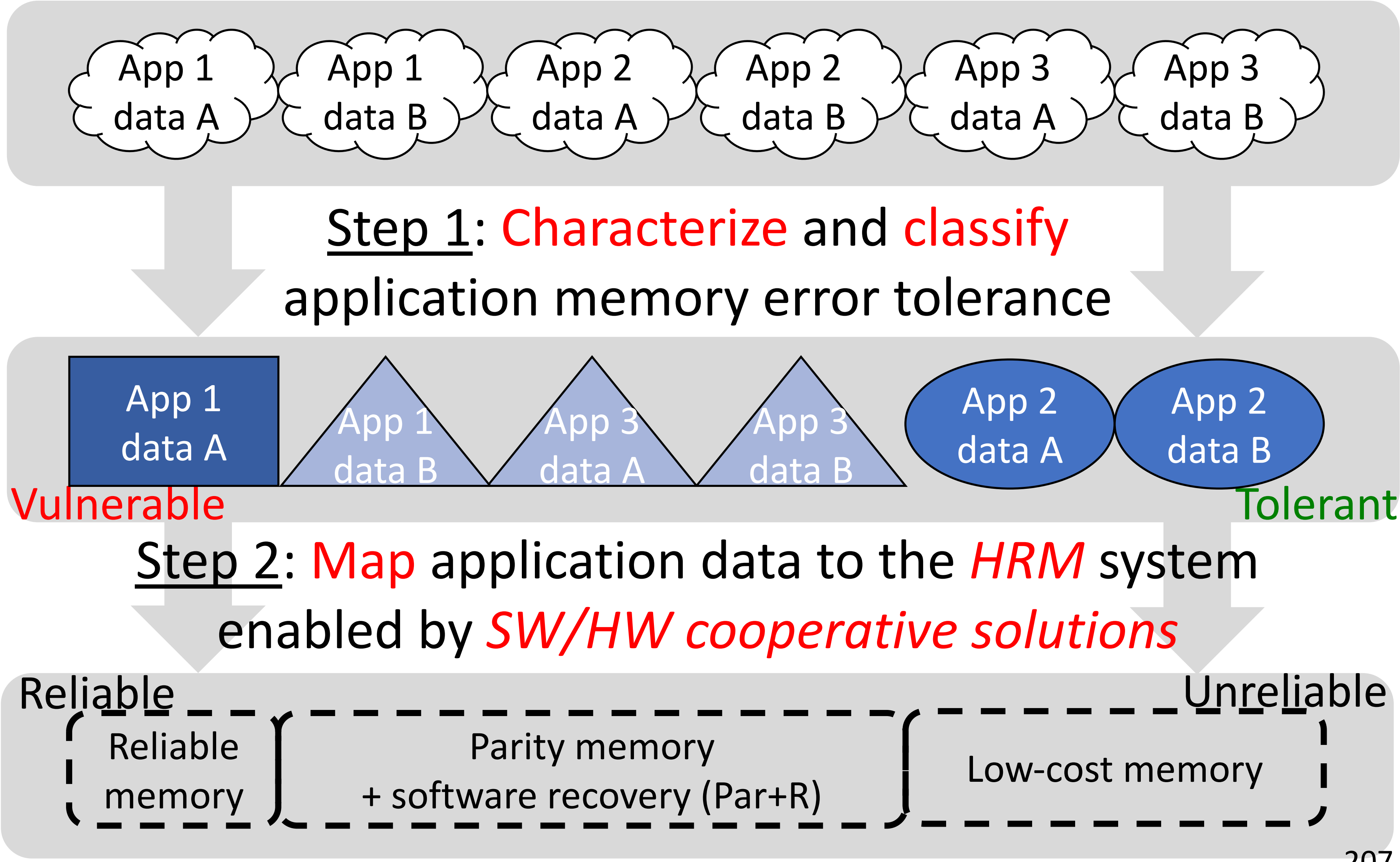
On Microsoft’s Web Search workload

Reduces server hardware cost by 4.7 %

Achieves single server availability target of 99.90 %

Heterogeneous-**R**eliability **M**emory [DSN 2014]

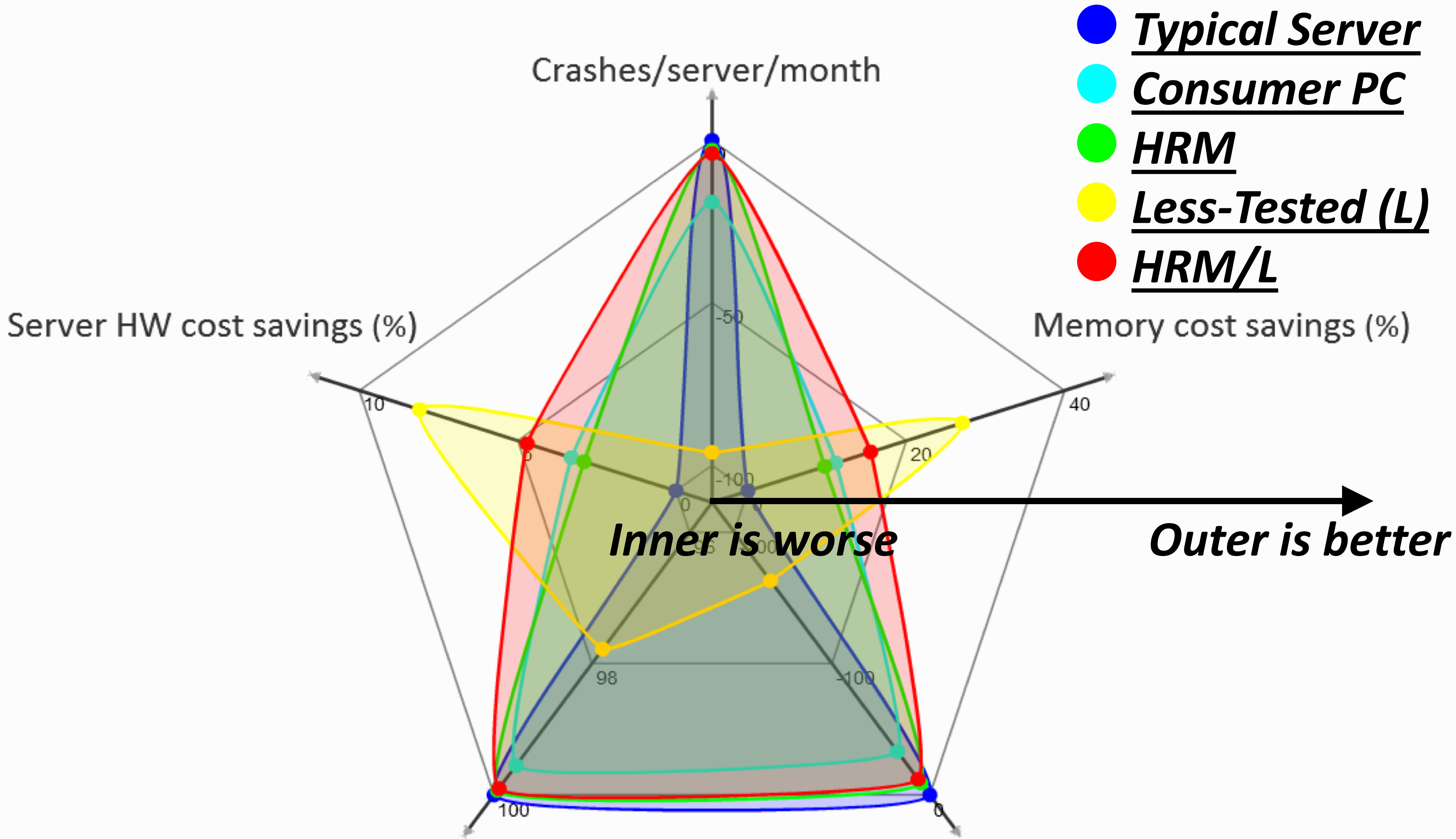
Heterogeneous-Reliability Memory



Step 1: Characterize and classify application memory error tolerance

Step 2: Map application data to the *HRM* system enabled by *SW/HW cooperative solutions*

Evaluation Results



● ***Typical Server***

● ***Consumer PC***

● ***HRM***

● ***Less-Tested (L)***

● ***HRM/L***

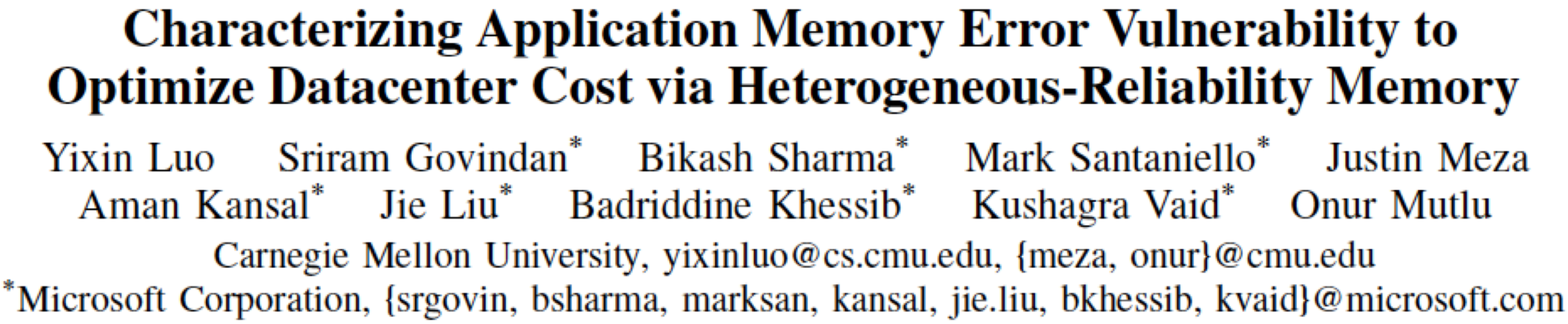
Inner is worse

Outer is better

● ● Bigger area means better tradeoff

More on Heterogeneous-Reliability Memory

- Yixin Luo, Sriram Govindan, Bikash Sharma, Mark Santaniello, Justin Meza, Aman Kansal, Jie Liu, Badriddine Khessib, Kushagra Vaid, and Onur Mutlu, **"Characterizing Application Memory Error Vulnerability to Optimize Data Center Cost via Heterogeneous-Reliability Memory"** *Proceedings of the 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (**DSN**)*, Atlanta, GA, June 2014. [Summary] [Slides (pptx) (pdf)] [Coverage on ZDNet]



Characterizing Application Memory Error Vulnerability to Optimize Datacenter Cost via Heterogeneous-Reliability Memory

Yixin Luo Sriram Govindan* Bikash Sharma* Mark Santaniello* Justin Meza
Aman Kansal* Jie Liu* Badriddine Khessib* Kushagra Vaid* Onur Mutlu
Carnegie Mellon University, yixinluo@cs.cmu.edu, {meza, onur}@cmu.edu
*Microsoft Corporation, {srgovin, bsharma, marksan, kansal, jie.liu, bkhessib, kvaid}@microsoft.com