P&S SoftMC

Understanding and Improving Modern DRAM Performance, Reliability, and Security with Hands-On Experiments

Hasan Hassan

Prof. Onur Mutlu

ETH Zürich

Fall 2021

13 October 2021

TRRespass

RowHammer in 2020

- Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi, **"TRRespass: Exploiting the Many Sides of Target Row Refresh"** *Proceedings of the 41st IEEE Symposium on Security and Privacy (**S&P**)*, San Francisco, CA, USA, May 2020.
  [Slides (pptx) (pdf)]
  [Talk Video (17 minutes)]
  [Source Code]
  [Web Article]
  Best paper award.

TRRespass: Exploiting the Many Sides of Target Row Refresh

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

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

TRRespass

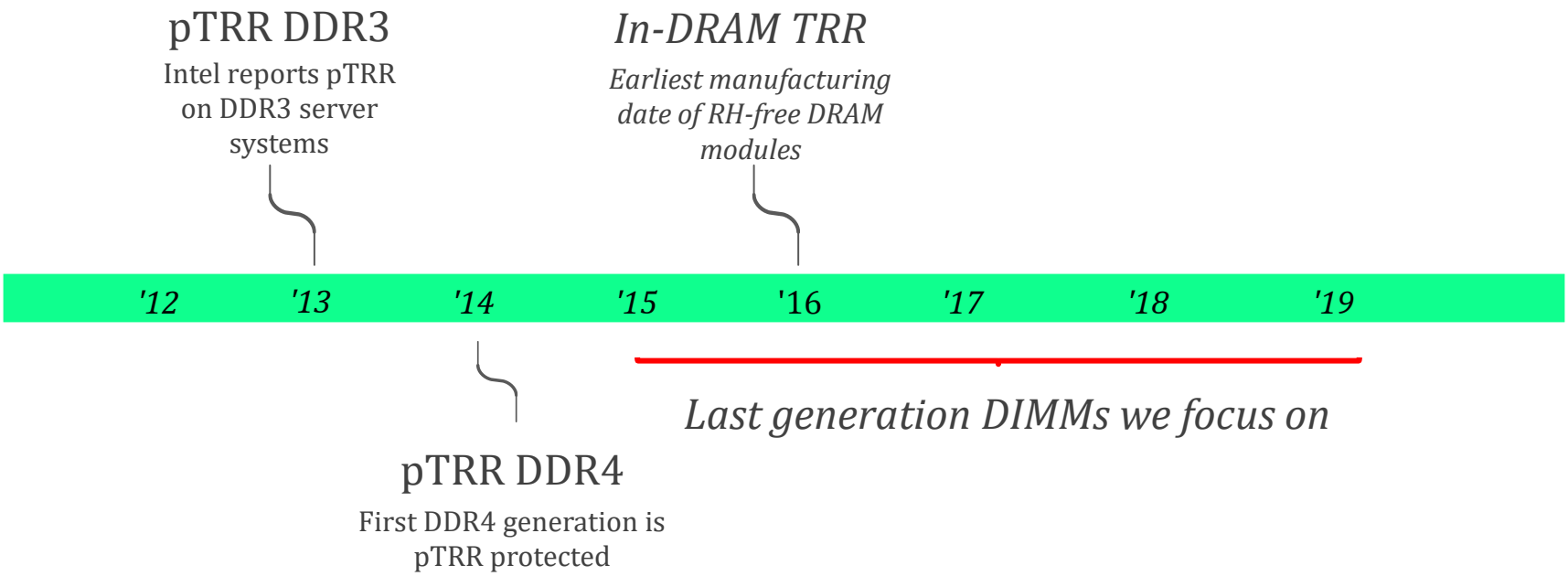
- First work to show that TRR-protected DRAM chips are vulnerable to RowHammer in the field
  - Mitigations advertised as secure are not secure
- Introduces the Many-sided RowHammer attack
  - Idea: Hammer many rows to bypass TRR mitigations (e.g., by overflowing proprietary TRR tables that detect aggressor rows)
- (Partially) reverse-engineers the TRR and pTRR mitigation mechanisms implemented in DRAM chips and memory controllers
- Provides an automatic tool that can effectively create many-sided RowHammer attacks in DDR4 and LPDDR4(X) chips

Target Row Refresh (TRR)

- How does it work?
  1. *Track activation count of each DRAM row*
  2. *Refresh neighbor rows if row activation count exceeds a threshold*
  - Many possible implementations in practice
  - Security through obscurity

- In-DRAM TRR
  - Embedded in the DRAM circuitry, i.e., not exposed to the memory controller

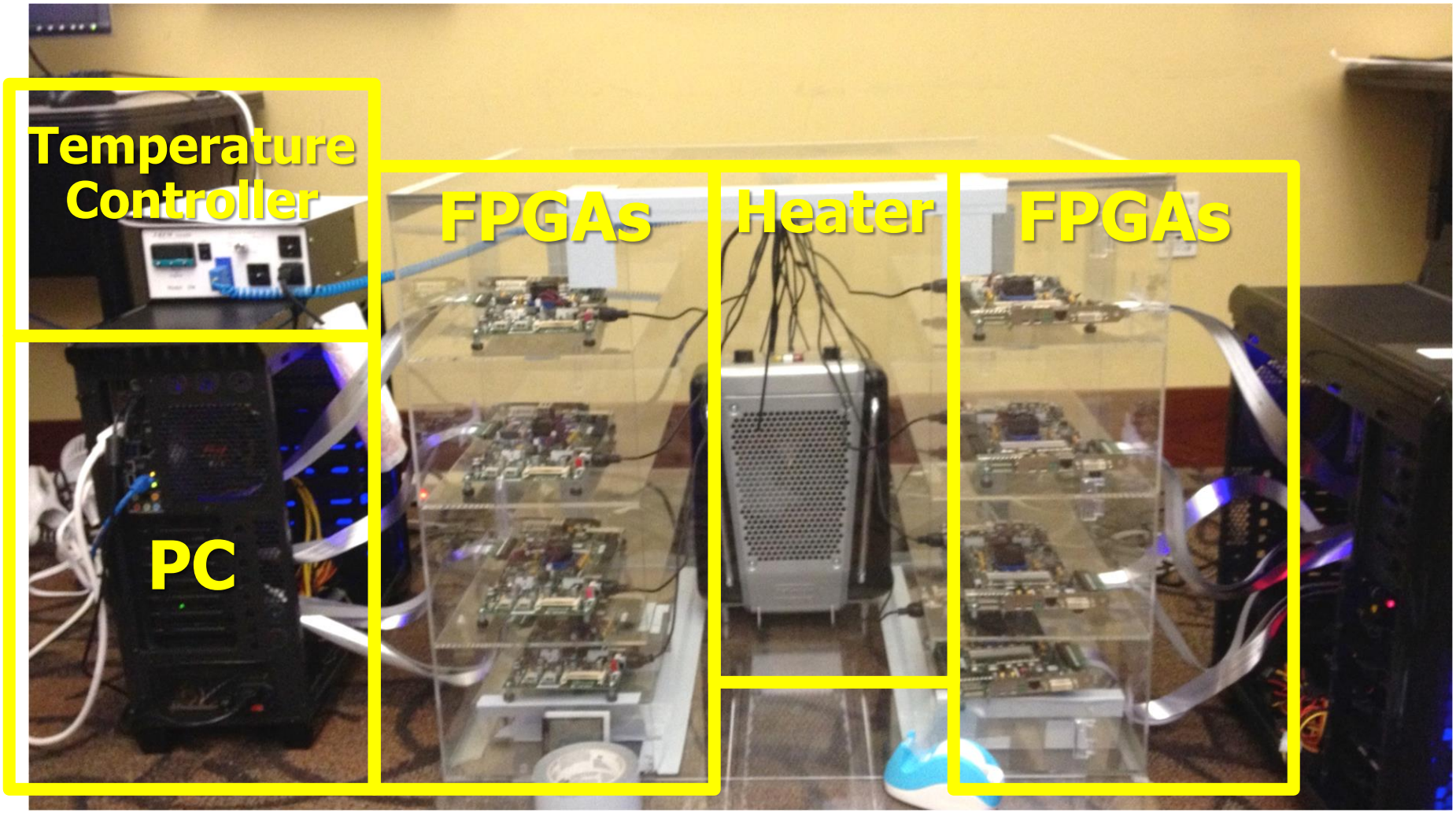
Timeline of TRR Implementations



Our Goals

- Reverse engineer in-DRAM TRR to demystify how it works
- Bypass TRR protection
  - A Novel hammering pattern: **The Many-sided RowHammer**
  - Hammering up to **20 aggressor rows** allows bypassing TRR
- Automatically test memory devices: **TRRespass**
  - Automate hammering pattern generation

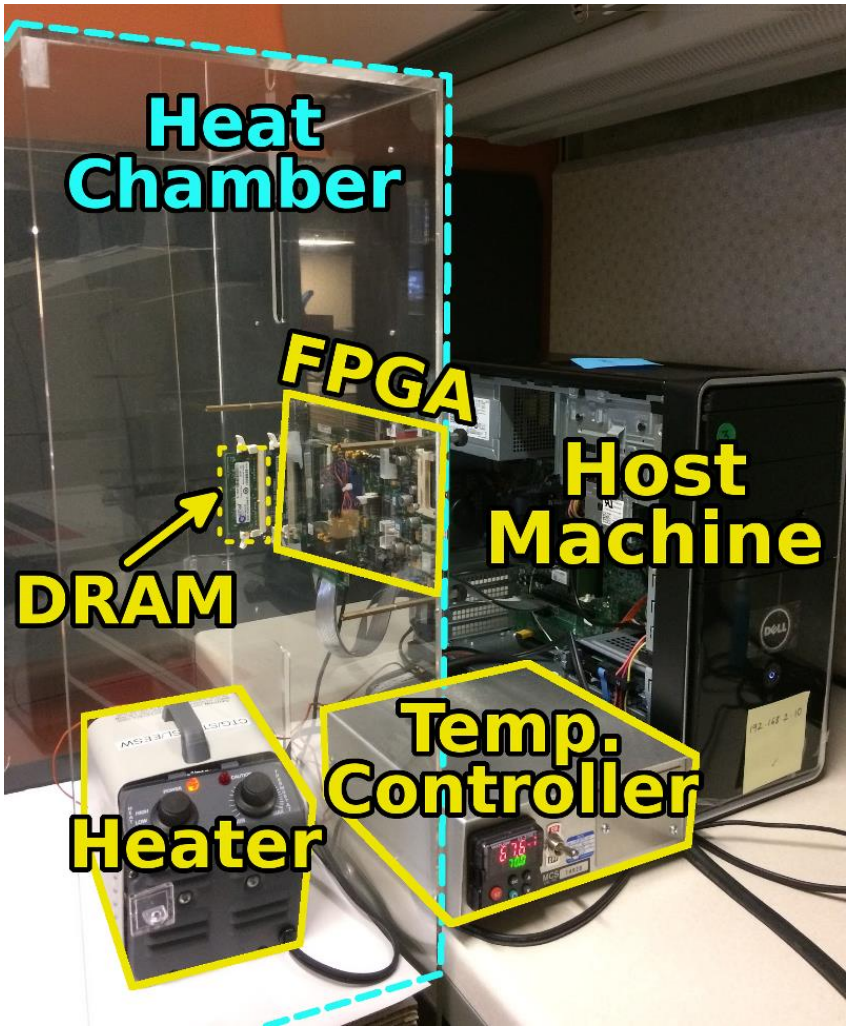
Infrastructures to Understand Such Issues



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*

Components of In-DRAM TRR

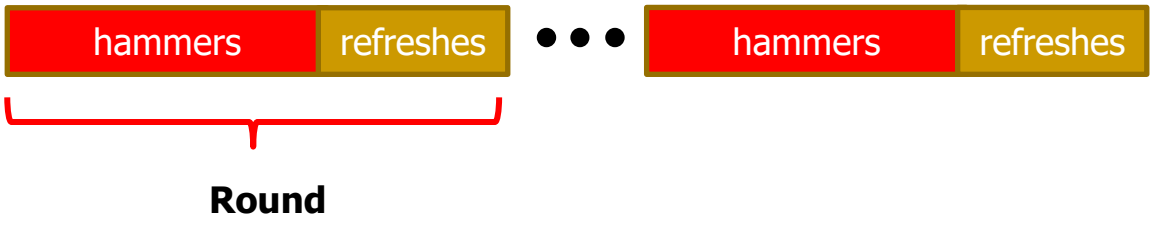
- **Sampler**
  - Tracks aggressor rows activations
  - Design options:
    - Frequency based (record every N^{th} row activation)
    - Time based (record first N row activations)
    - Random seed (record based on a coin flip)
  - Regardless, the sampler has a limited size

- **Inhibitor**
  - Prevents bit flips by refreshing victim rows
    - The latency of performing victim row refreshes is squeezed into slack time available in *tRFC* (i.e., the latency of regular Refresh command)

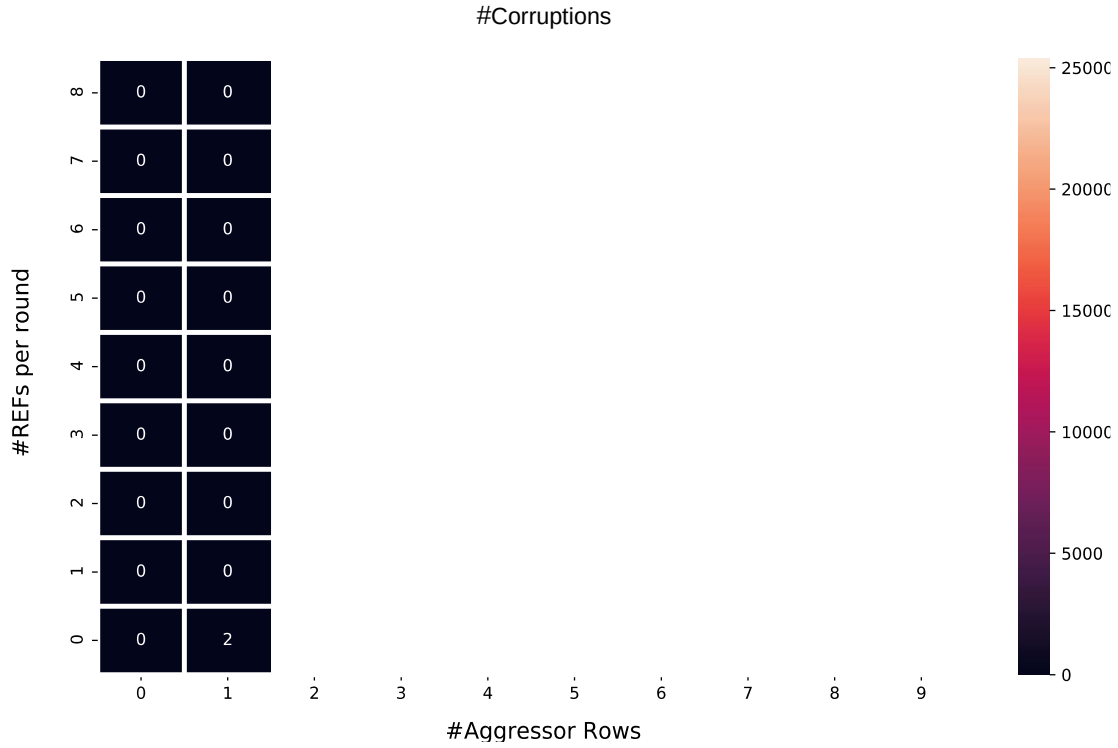
Case Study: Vendor C

How big is the sampler?

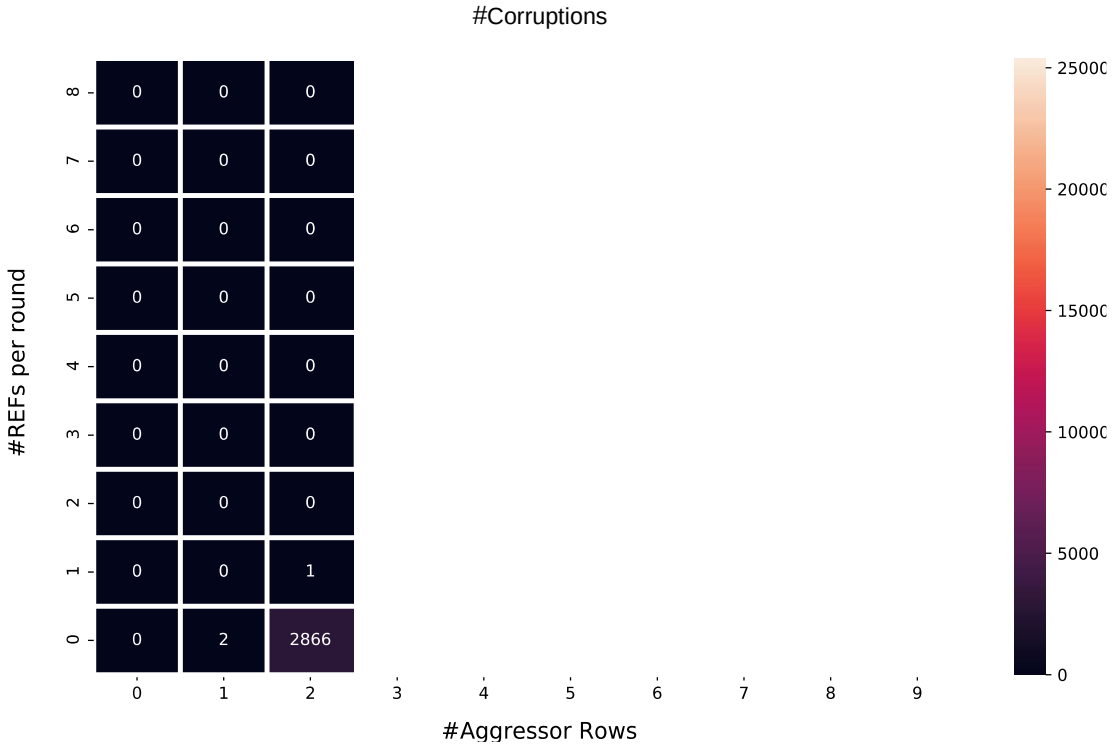
- Pick **N** aggressor rows
- Perform a series of hammers (i.e., activations of aggressors)
  - **8K activations**
- After each series of hammers, issue **R refreshes**
- **10 Rounds**



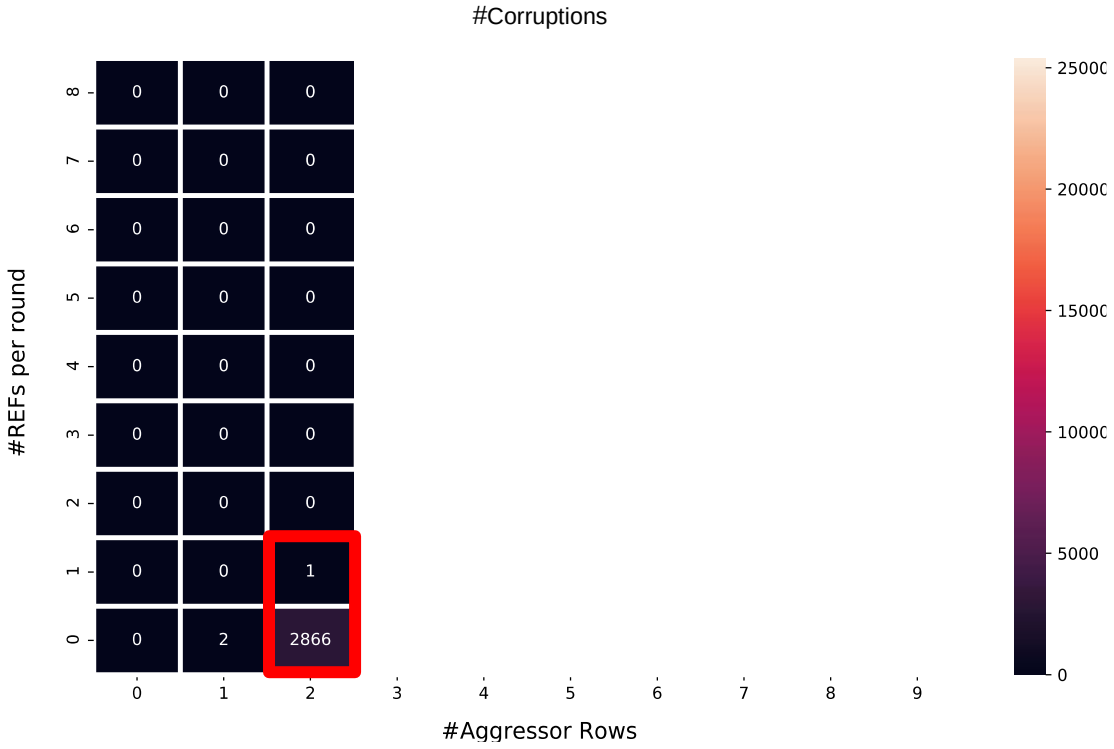
Case Study: Vendor C



Case Study: Vendor C

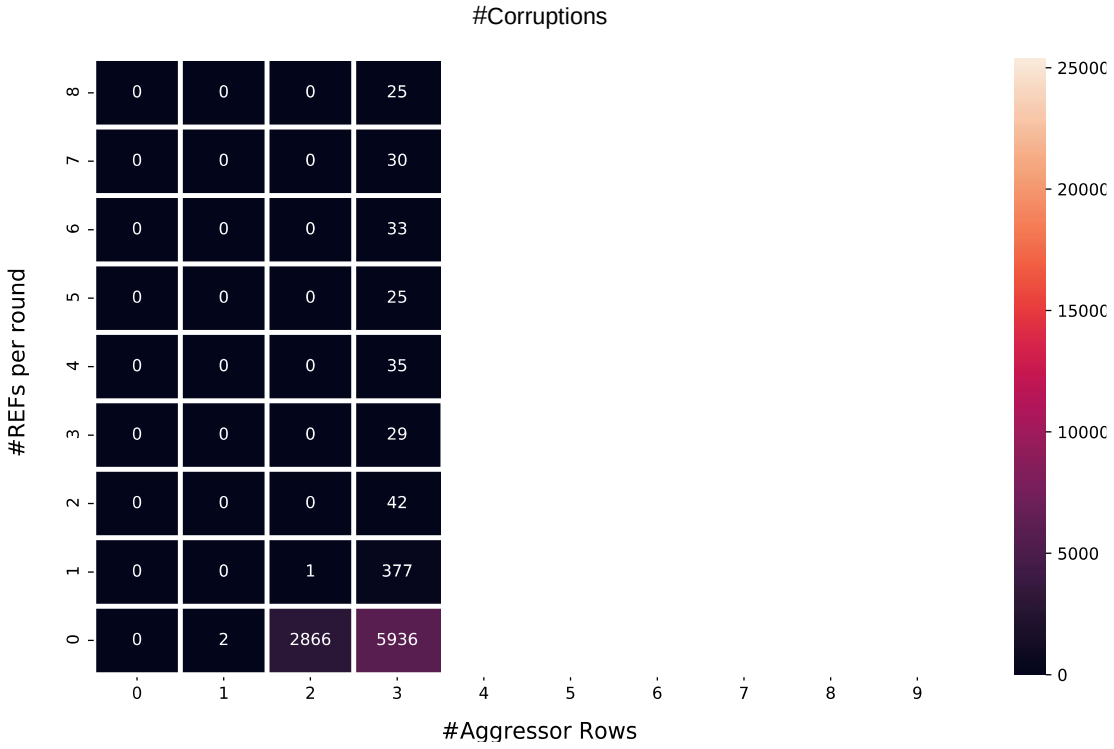


Case Study: Vendor C

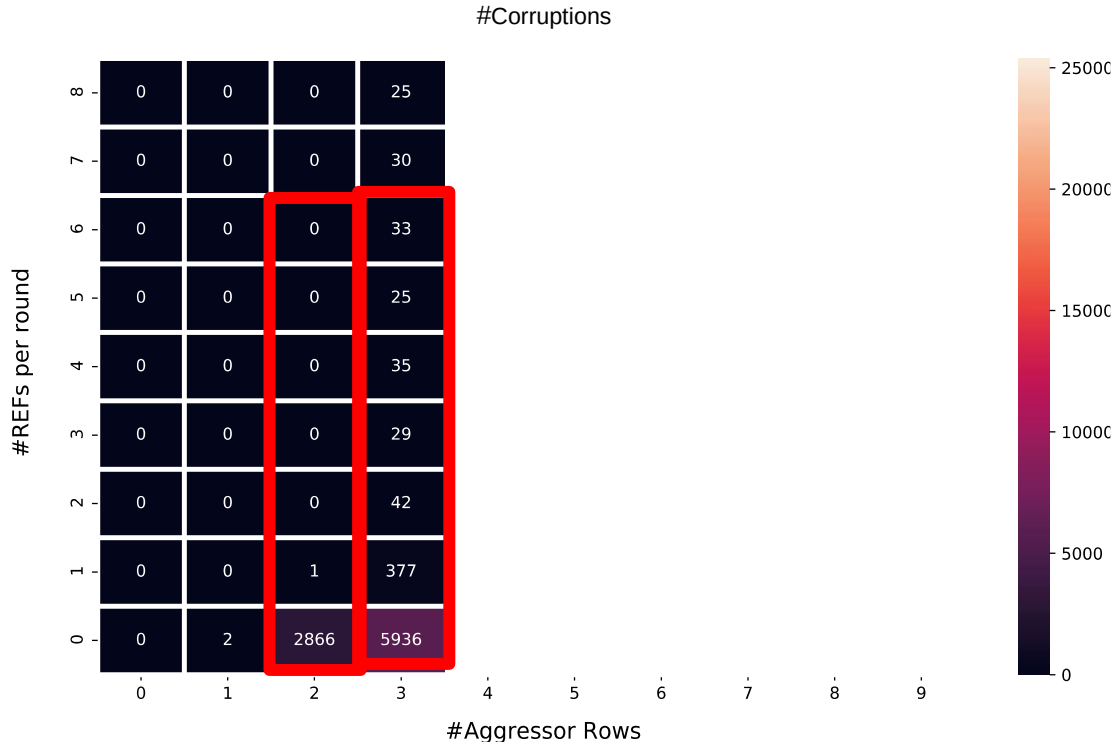


1. The TRR mitigation **acts on a refresh command**

Case Study: Vendor C

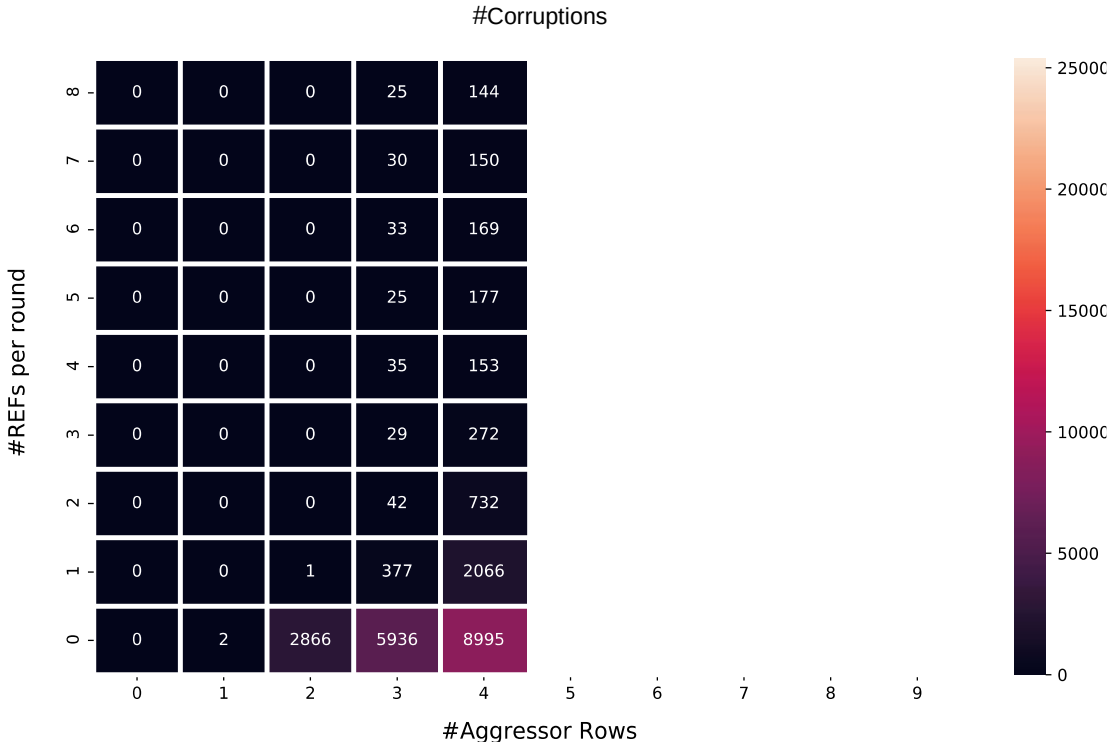


Case Study: Vendor C

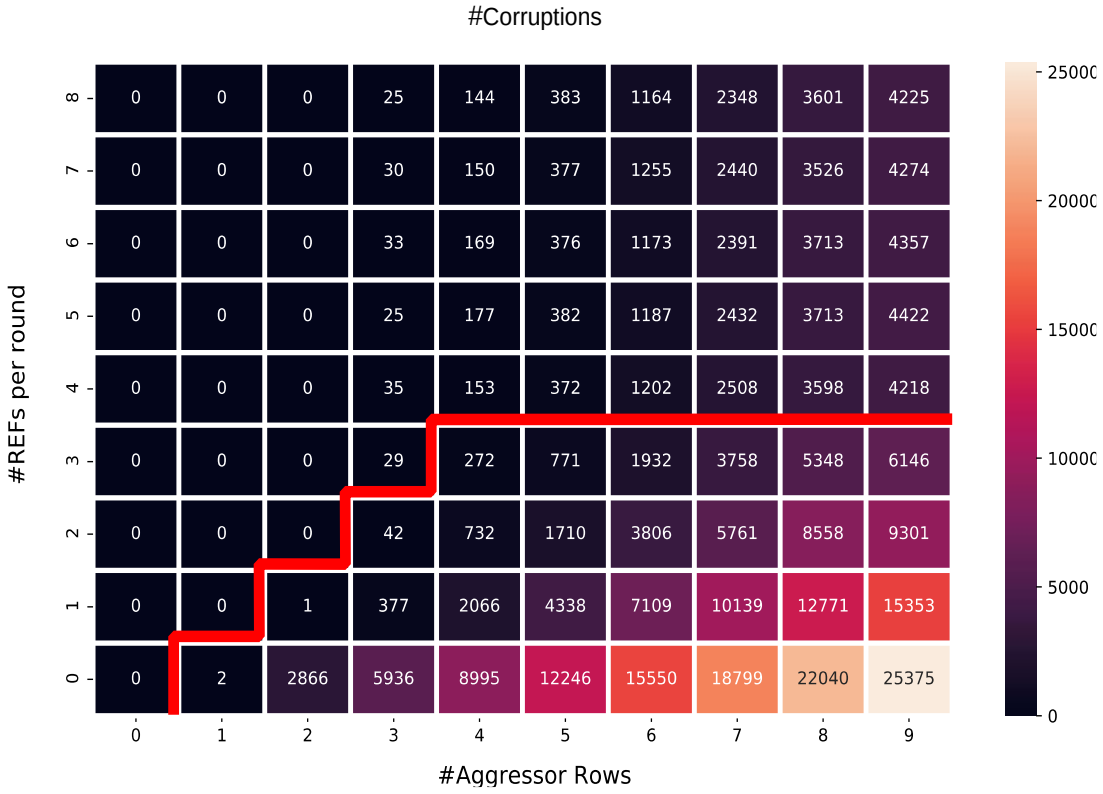


2. The mitigation **can sample more than one aggressor** per refresh interval
3. The mitigation **can refresh only a single victim** within a refresh operation

Case Study: Vendor C



Case Study: Vendor C



4. Sweeping the number of refresh operations and aggressor rows while hammering reveals the sampler size

Many-Sided Hammering

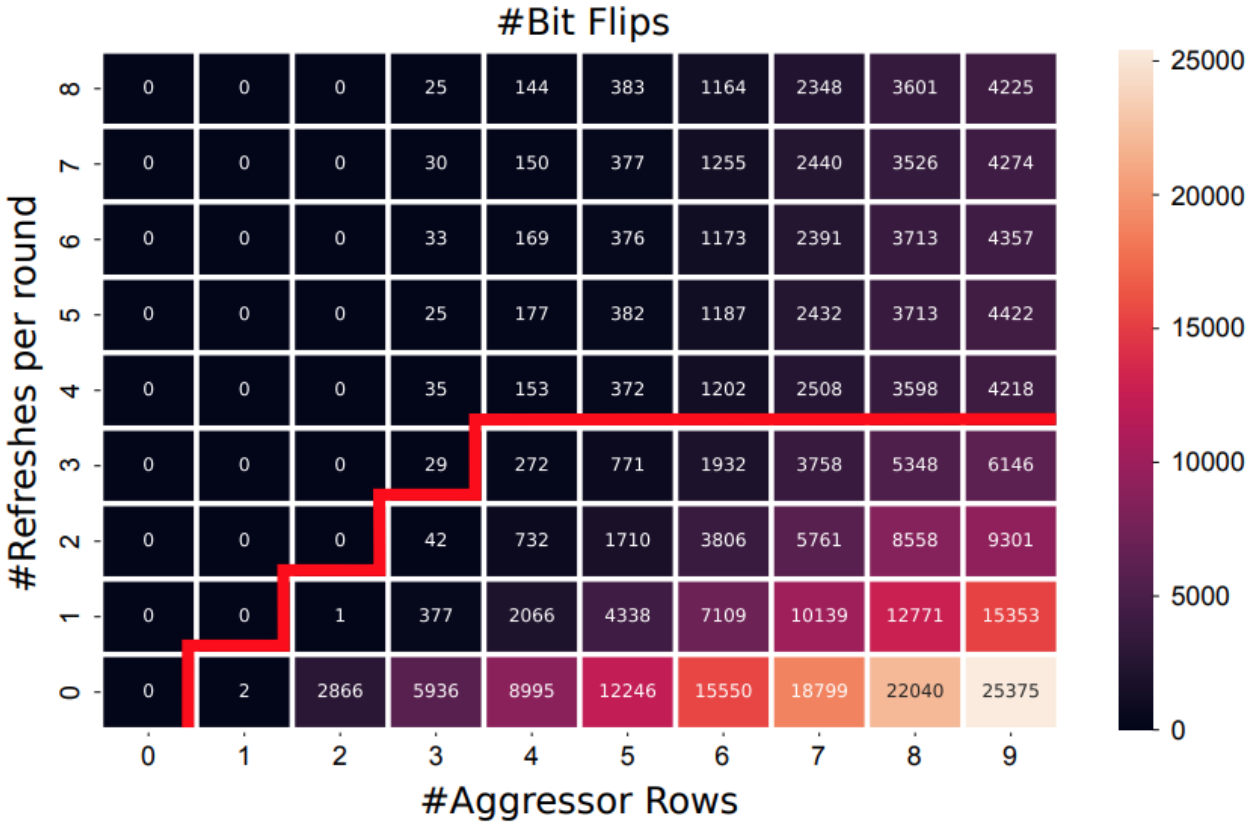
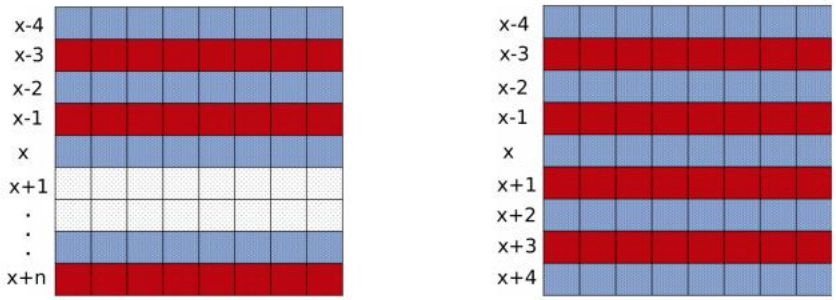


Fig. 9: Refreshes vs. Bit Flips. Module C_{12}: Number of bit flips detected when sending r refresh commands to the module. We report this for different number of aggressor rows (n). For example, when hammering 5 rows, followed by sending 2 refreshes, we find 1,710 bit flips. This figure shows that the number of bit flips stabilizes for $r \geq 4$, implying that the size of the sampler may be 4.

Some Observations

***Observation* 1:** The TRR mitigation acts (i.e., carries out a targeted refresh) on **every** refresh command.

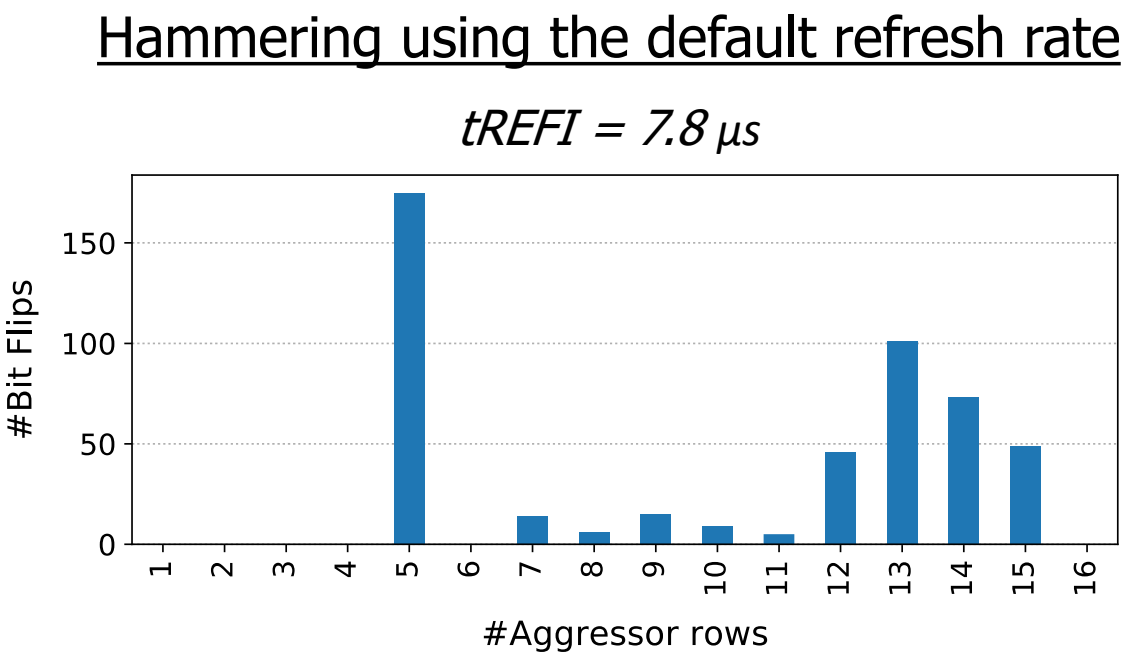
***Observation* 2:** The mitigation can sample **more than one** aggressor per refresh interval.
***Observation* 3:** The mitigation can refresh only a **single** victim within a refresh operation (i.e., time `tRFC`).
***Observation* 4:** Sweeping the number of refresh operations and aggressor rows while hammering reveals the sampler size.



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

Fig. 12: Hammering patterns discovered by *TRRespass*. Aggressor rows are in red (■) and victim rows are in blue (■).

Case Study: Vendor C



BitFlips vs. Number of Aggressor Rows

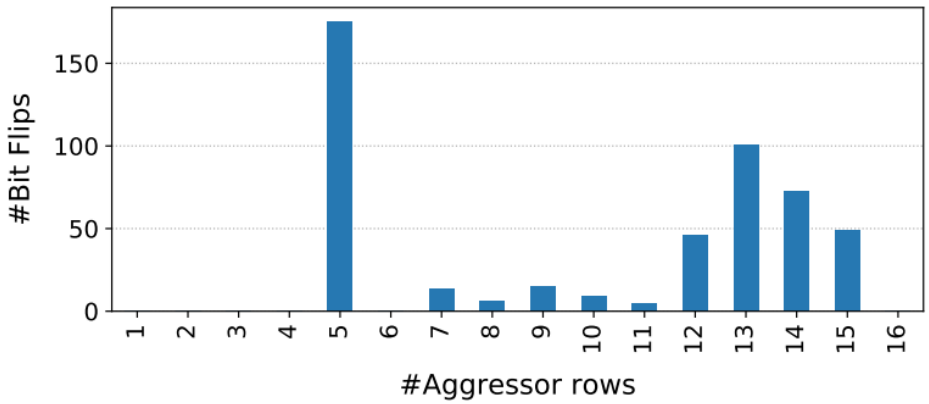


Fig. 10: Bit flips vs. number of aggressor rows. Module $\mathcal{C}_{12}$: Number of bit flips in bank 0 as we vary the number of aggressor rows. Using SoftMC, we refresh DRAM with standard `tREFI` and run the tests until each aggressor rows is hammered 500K times.

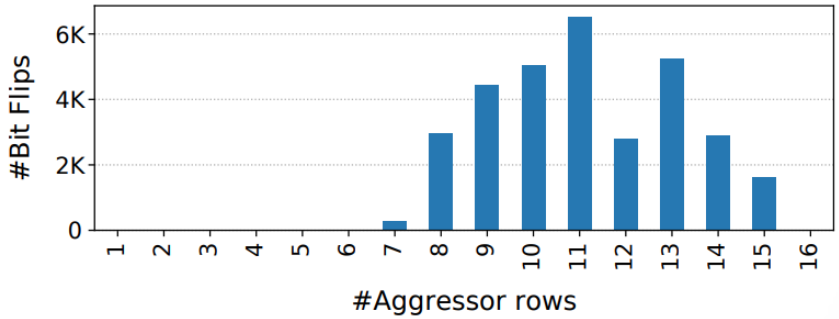


Fig. 11: Bit flips vs. number of aggressor rows. Module $\mathcal{A}_{15}$: Number of bit flips in bank 0 as we vary the number of aggressor rows. Using SoftMC, we refresh DRAM with standard `tREFI` and run the tests until each aggressor rows is hammered 500K times.

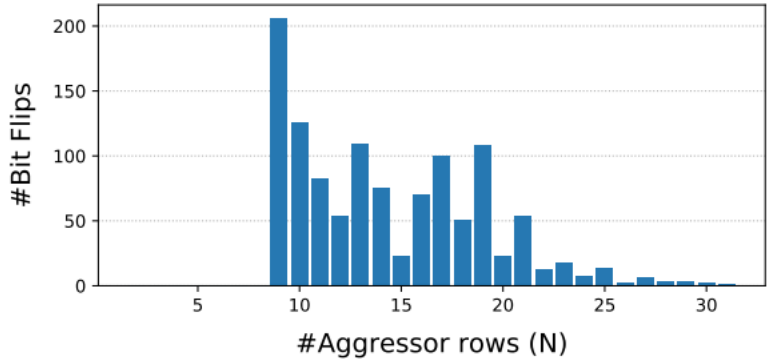


Fig. 13: Bit flips vs. number of aggressor rows. Module $\mathcal{A}_{10}$: Number of bit flips triggered with *N-sided* RowHammer for varying number of *N* on Intel Core i7-7700K. Each aggressor row is one row away from the closest aggressor row (i.e., `VAVAVA...` configuration) and aggressor rows are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

TRRespass Key Results

- 13 out of 42 tested DDR4 DRAM modules are vulnerable
  - From all 3 major manufacturers
  - 3-, 9-, 10-, 14-, 19-sided attacks needed
- 5 out of 13 mobile phones tested vulnerable
  - From 4 major manufacturers
  - With LPDDR4(X) DRAM chips
- These results are scratching the surface
  - TRRespass tool is not exhaustive
  - There is a lot of room for uncovering more vulnerable chips and phones

TRRespass Key Takeaways

RowHammer is still
an open problem

Security by obscurity
is likely not a good solution

More on TRRespass

- Pietro Frigo, Emanuele Vannacci, Hasan Hassan, Victor van der Veen, Onur Mutlu, Cristiano Giuffrida, Herbert Bos, and Kaveh Razavi, **"TRRespass: Exploiting the Many Sides of Target Row Refresh"** *Proceedings of the 41st IEEE Symposium on Security and Privacy (**S&P**)*, San Francisco, CA, USA, May 2020.
  [Slides (pptx) (pdf)]
  [Talk Video (17 minutes)]
  [Source Code]
  [Web Article]
  Best paper award.

TRRespass: Exploiting the Many Sides of Target Row Refresh

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

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

P&S SoftMC

Understanding and Improving Modern DRAM Performance, Reliability, and Security with Hands-On Experiments

Hasan Hassan

Prof. Onur Mutlu

ETH Zürich

Fall 2021

13 October 2021