Exploring the Correlation Between DRAM Latencies and Rowhammer Attacks

Md Sadik Awal and Md Tauhidur Rahman
Security, Reliability, Low-power, and Privacy (SeRLoP) Research Lab
ECE Department, Florida International University, Miami, Florida, USA
E-mail: {mawal003 and mdtrahma}@fiu.edu

***Abstract*—The rowhammer vulnerability primarily emanates from the physical characteristics of dynamic random access memory (DRAM). Various factors influence the timing of bit flips in DRAM, including cell density, memory access patterns, memory architecture, and bit flip probability. While prior research has focused on these factors, this article explores the correlation between DRAM latencies and rowhammer attacks. Latency variations inherently exist among the DRAM cells within a chip, and manufacturers usually provide recommended latency settings to ensure reliable operation and maximize manufacturing yield. However, reducing DRAM latency for improved memory access may increase the probability of unreliable or faulty memory operations. In this article, we demonstrate that cells with lower latencies, particularly those related to row activation, are more susceptible to rowhammer attacks. This vulnerability arises from the increased access frequency and smaller size of rows with shorter latencies. Furthermore, we discover that the highly coupled cells are more susceptible to rowhammer, as they can influence and impact one another, and these vulnerable cells can be identified from the erroneous characteristics observed in DRAM cells at reduced latency.**

***Index Terms*—Hardware security, switching activity, DRAM, rowhammer, memory access latency**

I. Introduction

Injecting faults into the memory is one of the potential means of defeating existing security, safety, and defense mechanisms [1]–[5]. These faults can result from various techniques, such as physical tampering, electrical manipulation, or inherent vulnerabilities in the memory technology [5]. Furthermore, Moore's law, driving the miniaturization of memory cells and transistors, has led to a higher density of memory components, resulting in increased electrical coupling and interference between adjacent rows in the memory array. Exploiting this electrical crosstalk, rowhammer attacks rapidly activate or "hammer" the same row repeatedly to induce bit flips in nearby rows. The hammered row serves as an aggressor that attack victim rows with unintended electrical interference, causing stored bits to flip and corrupting data [1]–[5]. These induced disturbances or flips allow attackers to alter contents in unauthorized memory regions, gain extra privileges, and bypass memory isolation boundaries through controlled memory corruption. For example, attackers can use rowhammer for various attacks such as crashing or corrupting a system, breaking memory-protection schemes, hijacking system control, taking over a remote servers, gaining arbitrary browser access, compromising co-hosted virtual machines [5], [6]. If successfully exploited, it can also compromise the safety of physical systems relying on accurate and dependable data. In safety-critical environments like automotive applications, memory corruption resulting from rowhammer can lead to catastrophic consequences [7]. The situation could deteriorate with the aging of devices, as a DRAM may become increasingly susceptible to rowhammer attacks over time (see Appendix).

There are various countermeasures against rowhammer attacks [8]–[12]. Memory access mitigation primarily focuses on reducing access frequency, disrupting reliance on specific access patterns that attackers may rely on, implementing randomized access to make unauthorized writing harder and developing hardware-based memory randomization to hide physical memory location from attackers [8]–[12]. Isolating kernel memory from user space reduces the risk of memory-based attacks and protects kernel data structures [8]–[12]. Some modern DRAMs use target row refresh to automatically refresh adjacent rows when rowhammer activity is detected [8]–[12]. However, some of the countermeasures provide incomplete protection. For example, memory access mitigation techniques do not provide comprehensive protection against all rowhammer variants [8]–[12]. Some countermeasures, such as target row refresh, rely on specific hardware features that may not be present on the target system [8]–[12]. Many other hardware/software-based mitigation techniques either need hardware changes, follow heuristic approaches, impact performance, are complex, or are incompatible with legacy systems [8]–[12]. Furthermore, new attack methodologies might make the old solutions ineffective [8]–[12].

Identifying rowhammer-susceptible cells can be an effective solution to address the shortcomings. However, identifying such DRAM cells is challenging, primarily because of unknown logical-to-physical mapping, memory configurations, random physical factors, and complexity. Prior work used statistics to find rowhammer-vulnerable cells by studying memory cell features and testing a smaller sample of representative memory [13]. However, this method relies on performing rowhammer attacks on the DRAM under test, which is time-consuming and expensive since they also need to reverse-engineer DRAM to locate the cells physically. Furthermore, it can age the devices and impact the reliability. In this article, we make the following contributions.

- We reduce the DRAM latency and correlate resulting latency errors with rowhammer errors.
- We find *wordlines* generating more errors at reduced latency are more rowhammer-susceptible due to frequent access. Furthermore, DRAM cells flipping at reduced latency are smaller and leak faster.
- We experimentally support our claim on commercially available DRAM modules. We investigate if rowhammer-

vulnerable cells or *wordlines* can be identified by lowering latency and analyzing the resulting latency errors.

The rest of the paper is organized as follows. In Section II, we briefly present the DRAM architecture, operation, and timing. This section also covers rowhammer attacks and countermeasures. Section III experimentally demonstrates the correlation between rowhammer errors and reduced latency errors. We conclude our work in Section IV.

II. Background

A. DRAM Architecture, Operation, and Latency

The organization of a modern DRAM system is illustrated in Fig. 1a. Modern DRAM systems include channels (data transfer pathways), ranks (sets of memory modules treated as a unit), banks (logical divisions within DRAM modules for concurrent access), DRAM chips (individual circuits storing data in cells), DRAM cells (the smallest units of data storage using capacitors and transistors), and a memory controller (manages data flow between the computer's main memory and the processor, facilitating efficient access). Memory commands, address space, and data flow between the memory controller and DRAM modules occur via the memory channel, which is usually 64-bit wide. Each DRAM chip includes multiple banks to support parallelism.

Within a memory bank, DRAM cells are arranged in a two-dimensional array, with rows referred to as *wordlines* (or *pages*) and columns as *bitlines*, which are connected to the row-buffer. The chip density and size determine the total number of rows. A DRAM cell consists of an access transistor for reading or writing data and a capacitor to retain charge. The charging state of the capacitor determines the stored value: a fully charged capacitor signifies a logic '1', while an empty capacitor signifies a logic '0'. The access transistor connects the capacitor to a *bitline* and is controlled by the *wordline*.

Initially, all *bitlines* are precharged to $V_{dd}/2$ during the precharge state. Subsequently, the *wordline* is activated in the following state by elevating its value to V_{dd}. Upon activating the pass-transistor in the DRAM cell, charge flows between the capacitor and the respective *bitline*. If the stored value is '0', the charge moves from the *bitline* to the capacitor, and vice versa. In the final stage, the sense amplifier connected to the *bitline* amplifies the *bitline* voltage, resulting in a robust logic '1' (or '0').

Timing is crucial for reliable DRAM operation; it ensures synchronization with the memory controller and other system components [14], [15]. Different types of DRAM have specific timing requirements that must be adhered to for compatibility and optimal performance [14], [15]. Fig. 1c, represents an overview of all the significant timing parameters associated with a DRAM module. Initially, all *bitlines* are precharged to $V_{dd}/2$. To access data from a specific *wordline*, an *ACTIVATE (ACT)* command is applied to the corresponding *wordline*. Subsequently, the memory controller transmits a *READ/WRITE* command to either sense the voltage perturbation on the *bitlines* or write data to the memory cells. The minimum time interval between the *ACT* command and the *READ/WRITE* command is defined as the activation time latency $tRCD$. The *Column Access Strobe (CAS)* latency, tCL, represents the minimum waiting time to retrieve the first data bit on the data bus after issuing a *READ* command. Following a successful *READ/WRITE* operation, a *precharge command (PRE)* is applied to deactivate any previously activated *wordlines*, restoring the *bitlines* to their initial precharged state at $V_{dd}/2$. If a *WRITE* command is used, the *PRE* command should be further delayed by the tWR period (write recovery time) at the end of the write data burst. The *PRE* command must be maintained for at least the tRP (precharge time) duration before sending the next *ACT* command. The period from the activation state to the commencement of the precharge state is referred to as the row active time or restoration latency $tRAS$. The sum of $tRAS$ and t_{RP} represents the total time necessary to access a single row in a bank, known as the row cycle time (tRC).

Reduced DRAM latency [16]–[19], while beneficial for improving memory access speed [15], can also potentially lead to an increased risk of unreliable or faulty memory operations. This can sometimes result in data corruption or system instability if not properly managed and tested [15]–[20]. For example, at the reduced t_{RP}, some of the *bitlines* will not be fully precharged to $V_{dd}/2$ (i.e., they show deviation from $V_{dd}/2$) and might produce the wrong output [15]. Similarly, an incorrect operation is observed if the $tRCD$ is lowered below the recommended latency [15]–[19], [21].

B. Rowhammer Attack and Countermeasures

The continued scaling of DRAM technology provides considerable benefits such as reduced cost-per-bit by enabling the integration of memory cells in a tiny chip/die, but it has also introduced unintended vulnerabilities [5], [6], [12], [22]. The high density and small memory cell sizes of modern DRAM arrays lead to reduced reliability and robustness. This decreased noise immunity enables malicious triggering of disturbance errors to flip bits in neighboring memory cells [5], [6], [12], [22]. The primary reasons for such disturbance errors are [5], [6], [12], [22]–[24]: (i) electromagnetic coupling which can inject noise to adjacent rows during *wordline* voltage changes, (ii) accelerated charge transfer between bridged cells from *wordline* toggling, and (iii) cell leakage from silicon aging damage.

In DRAM, frequently accessing, or "hammering", of a row (i.e., aggressor *wordline*, Fig. 1b) can induce disturbances in nearby rows (i.e., victim *wordlines*, Fig. 1b) and flip the stored bits in the victim rows. This rowhammer phenomenon exploits the physical proximity and electrical coupling of densely packed cells. [5], [6], [12], [22]. By rapidly activating a rowhammer aggressor row before periodic refresh, attackers can deliberately induce bit flips in victim rows based on several factors: (i) aggressor/victim row contents, (ii) hammer access frequency, (iii) architecture, (iv) process variation, (v) operating conditions, and (v) device age. Single-sided rowhammer targets one victim side, while double-sided targets both sides for higher accuracy. However, the double-sided approach is more complex than the single-sided variant, demanding finer control over memory accesses [12], [22].

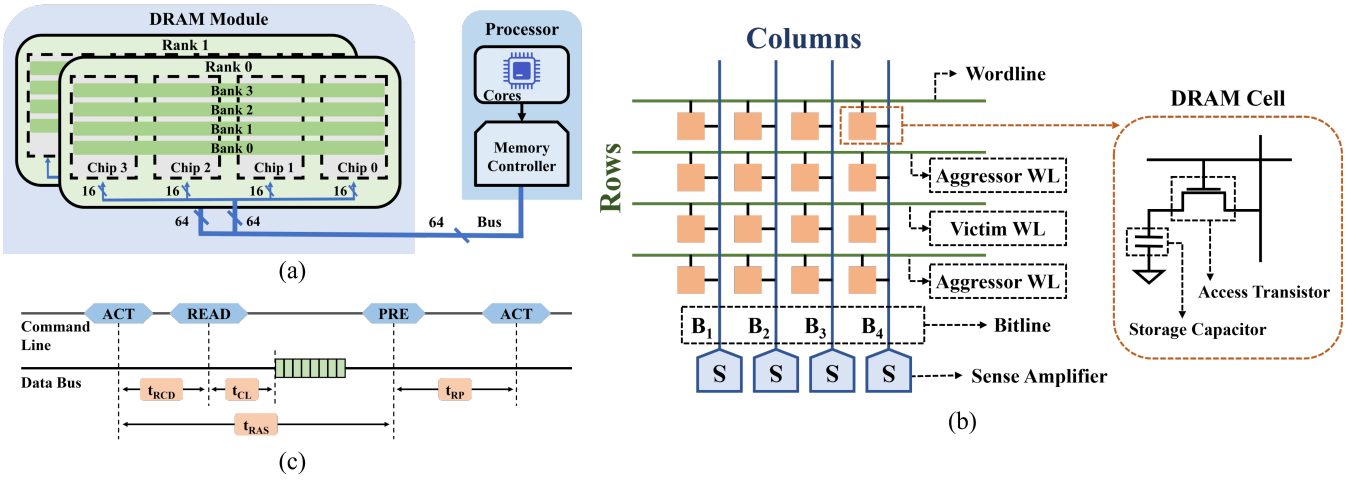

Fig. 1: (a) DRAM Architecture, (b) rowhammer attack: the victim *wordline* is disturbed by continuous writing/reading the adjacent aggressor *wordlines* [6], and (c) DRAM timing [14].

Detection, neutralization, and elimination are the three primary solutions for building a rowhammer-resistant electronic system [8]–[12]. Common software-based solutions against rowhammer attacks include static analysis, monitoring performance counters or access patterns, preventing exhaustion-based page placement or physical proximity to the kernel pages, increasing the refresh frequency to recharge the leaky cells more often, restricting CLFLUSH command to avoid disabling cache locality, using cell isolation techniques to reduce coupling/crosstalk effects, and implementing error-correcting codes [8]–[12]. Hardware-based countermeasures involve target row refreshing, memory isolation, error correction, advanced memory controller, rowhammer-resistant memory architectures, etc. Some of these methods relay on counting all potentially vulnerable aggressor row activations or using a probabilistic model to refresh neighbor rows [25]–[28]. However, most of the existing countermeasures require hardware changes, leading to performance degradation, increased power overhead, or potential weakening as the device ages [8]–[12]. Moreover, as advanced attack methods consistently bypass previously deemed secure countermeasure techniques [11], the need for more robust solutions becomes increasingly evident.

TABLE I: Number of cells/wordlines affected by the rowhammer attack.

	Rowhammer Errors					
DRAM	**#1**	**#2**	**#3**	**#4**	**#5**	**#6**
Affected cells	10342	9874	7854	0	5	0
Affected *wordlines*	4712	5487	6041	0	3	0

III. Rowhammer and Latency Correlation

Capacitive crosstalk and electron injection/diffusion/drift are the two primary causes of rowhammer errors [29]. Although there is no straightforward correlation between the rowhammer vulnerabilities and DRAM latency, this article aims to discover the correlation between them. In rowhammer attacks, the attacker changes the data frequently within an aggressor row to induce changes in the contents of the victim row, and more frequent memory access typically results in a higher occurrence of rowhammer errors. On the other hand, different cells within the same DRAM chip have different reliable operation latencies primarily due to the design (i.e., architectural), process, and manufacturing variations [15], [30], [31]. Reduction of DRAM latency from the standard values may result in unreliable operations, such as changes in memory content or insufficient time for accurate data writing [15], [30], [31]. The strategy of lowering DRAM latency to boost DRAM performance and generating random numbers or device signatures has been utilized in the past [32], [33].

We execute rowhammer attacks on DRAM to explore potential correlations between latency and rowhammer effects. We experimented with six DDR3 modules from two different manufacturers[1] using SoftMC on Xilinx ML605 FPGA as shown in Fig. 2, [5], [21]. In most cases, we observed that each *wordline* typically suffers one, two, or three rowhammer errors. Table I shows the error profile of three DRAMs that experience rowhammer errors. We did not observe any rowhammer errors in other DRAMs.

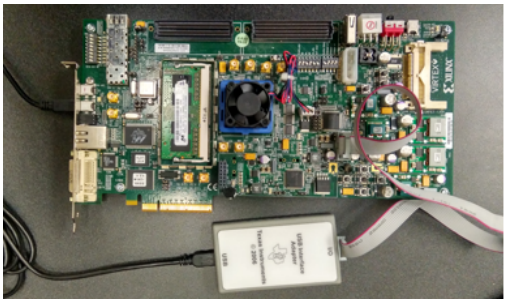

Fig. 2: Experimental setup for rowhammer attack and latency reduction [21].

To obtain the latency distributions, we used the same SoftMC to profile the error distribution at reduced latency [21]. We observed the outputs at the reduced latency with different input patterns (e.g., all 1s, all 0s, or checkerboard patterns). The standard latencies for these DRAM modules

[1]Manufacturer 1: Micron, Manufacturer 2: Samsung

TABLE II: Error counts from the reduced latency.

Latency (L) Reduction	Latency Errors (LE)					
	Number of *wordlines* (LEW)			Number of cells (LEC)		
	DRAM 1	DRAM 2	DRAM 3	DRAM 1	DRAM 2	DRAM 3
20% of standard tRCD	0	0	512	0	0	1,387
40% of standard tRCD	513,299	604,419	577,812	10,310,000	11,010,000	10,251,000
60% of standard tRCD	13,887,912	15,722,144	12,112,334	389,803,324	512,322,113	591,225,781
20% of standard tRP	5,837	63,264	69,822	12,130	131,400	112,200
40% of standard tRP	549,673	652,383	532,527	10,312,465	11,011,123	10,251,337
60% of standard tRP	15,321,126	14,852,132	14,735,221	427,801,223	517,833,312	612,017,884
20% of standard tRAS	0	0	0	0	0	0
60 % of standard tRAS	0	0	0	0	0	0

TABLE III: Latency reduction errors vs. rowhammer errors.

	0.6tRCD			0.6tRP			Union (0.6tRCD, 0.6tRP)		
	DRAM 1	DRAM 2	DRAM 3	DRAM 1	DRAM 2	DRAM 3	DRAM 1	DRAM 2	DRAM 3
Coverage of cell (in %)	64.16	69.13	64.42	71.15	73.36	68.33	88.15	83.12	78.33
Coverage of *wordline* (in %)	73.54	72.03	78.17	73.53	72.12	77.24	84.51	92.12	92.24

are 13.75ns for tRCD, 35ns for tRAS, and 13.75ns for tRP. The output of a cell (i.e., the read bit at the reduced latency) is called the "same output" or "errorless output" if the output of the cell is the same as the written bit. If the output of a cell is different than the written one, it is called "flipped" or "faulty" output. We observed a positive correlation between the reduction of latencies and an increase in total error (i.e., the number of flipped outputs). Table II shows the error profiles of the DRAMs at the reduced latency. **The findings are also illustrated in and Fig. 3 where X-axis represents the percent reduction of latency and Y-axis represents the LE in log-scale.**

Based on our experimental data, we have observed the following.

- **tRCD (row to column delay)** determines the time it takes to open a memory row and access any column within that row. Reducing tRCD can lead to quicker access to data within a row, improving overall memory access performance for all subsequent accesses to that row. A reduced tRCD may enable the attacker to activate and access rows more quickly, potentially increasing the frequency of memory accesses to the aggressor row. In our experiment, we observed that a substantial reduction in tRCD is required for error detection. However, pushing tRCD to lower levels further led to significantly increasing errors. Furthermore, increasing the refresh interval resulted in a significant overall rise in error count.
- **tRP (row precharge time)** is the time it takes to close a row after it has been activated. A reduced tRP can potentially affect almost all cells of a row. When tRP is reduced, the rows can be closed more quickly. Consequently, the attacker can access it more often, increasing the chances of triggering the rowhammer effect and potentially causing bit flips in nearby rows. The error patterns when reducing tRP and tRCD are quite similar.
- **tRAS (row active time)** specifies the minimum time a row must remain active before it can be precharged or deactivated. It defines the duration of time that a row is open for data access. Nevertheless, no errors were detected at the lowered tRAS setting when adhering to the recommended refresh interval. However, as we extended the refresh interval, errors emerged in specific DRAM modules operating with the reduced tRAS. This increase in the refresh interval corresponded with a noticeable escalation in the overall error count.

In our experiment, we observe that many cells or *wordlines* are affected if we reduce the DRAM latency by 75%. Table III represents the correlation among cells/words that experience errors from rowhammer attack and latency reduction. Our study reveals a significant correlation between individual latency reduction and increases when considering both latency reductions. We introduce the following coverage metrics to quantify the correlation.

- *Cell Coverage*: This metric indicates whether the susceptibility of DRAM cells to rowhammer can be identified by reducing DRAM latency. It is defined as $\frac{Number\ of\ Cells\ with\ Common\ Errors}{Number\ of\ Cells\ with\ Rowhammer\ Errors}$.
- *Wordline Coverage*: In contrast, the *wordline coverage* indicates if a *wordline* is more susceptible to a rowhammer attack if affected at the reduced latency. It is defined as $\frac{Number\ of\ Rows\ with\ Common\ Errors}{Number\ of\ Rows\ with\ Rowhammer\ Errors}$.

A DRAM cell in a victim row is likelier to be flipped when it strongly competes with the DRAM cells in the aggressor rows. Therefore, this study aims to identify those very strongly coupled cells: a cell is more susceptible to rowhammer if it is highly coupled with the cells of aggressor rows. We find some cells produce stable flipped outputs at reduced latencies while others are noisy. These noisy cells switch between right and wrong outputs. The strongly coupled cells produce very stable outputs or very noisy outputs: they are usually affected by the nature of neighbor cells. Authors in [34] observed very similar with SRAM cells. The experiment demonstrated that a cell's start-up values are highly dependent on the start-up nature of its neighbor cells: a stable cell (i.e., the start-up value is always 0 or always 1) is surrounded by stable cells and a noisy cell (i.e., the start-up value always switches between 0 and 1) is surrounded by noisy cells. Our experimental results show that the cells with more stable cells at the reduced latency correlate better with rowhammer errors. We argue that the cells are very noisy or very stable at the reduced latency for the following

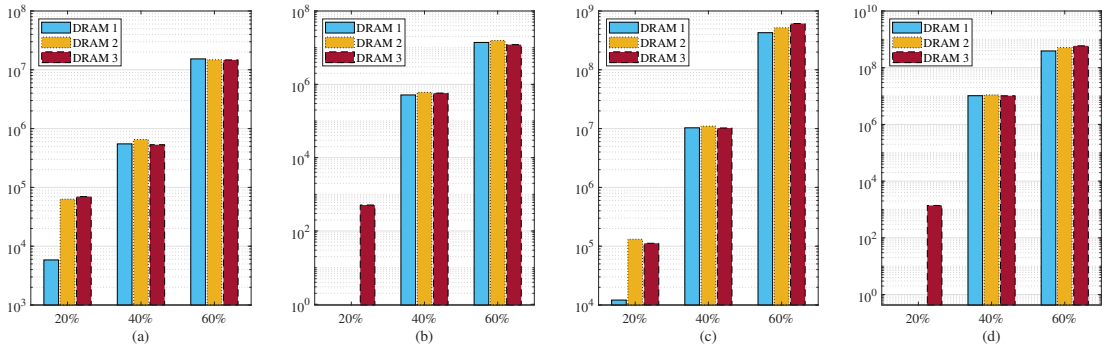

Fig. 3: Errors at the reduced latency: LEW for (a) tRP, (b) tRCD; LEC for (c) tRP, (d) tRCD.

three reasons: (i) they are strongly coupled with the neighbor cells, (ii) power supply or other environmental noise, and (ii) the size of the capacitors. However, the sample size is too small to make such a claim.

From the above results, we have the following conclusion.

- The smaller cells are more prone to leakage and, therefore, more susceptible to rowhammer attacks. If two DRAM cells are equally distant from the buffer row, the low latency cells usually have low capacitors and, therefore, are more susceptible to rowhammer attack. This is because it can result in reduced charge retention capability.
- Smaller cells and reduced latencies can make cells more susceptible to disturbance, increasing the likelihood of errors. Therefore, a cell that creates flipped output at the reduced latency has more chance to flip from rowhammer attacks.
- It is possible to access rows more frequently if they have lower DRAM latencies related to row activation, which makes the attack easier. Low latency *wordlines* can be accessed more frequently.
- More noisy or more stable outputs at the reduced latencies may be tied with the coupling. These cells are more susceptible to rowhammer attack. However, we need more data and samples to make such a conclusion.
- DRAM latency reduction can be used to identify the susceptible rows and can be used along with other mitigation techniques, such as a more frequent refresh of vulnerable words or rows.

IV. Conclusion

This article presented the correlation between errors resulting from rowhammer attacks and latency reduction. Our experimental results on DDR3 DRAM show that: (i) *wordlines* with lesser latency can be accessed more frequently, making them prime candidates for aggressor attacks; (ii) *wordlines* experiencing increased errors due to latency reduction typically have cells with smaller capacitance, resulting in faster discharge compared to cells with larger capacitance; (iii) cells exhibiting more stable (i.e., producing consistent "wrong" or "right" output) or noisier with the latency reduction may have a strong coupling effect, rendering them more susceptible to rowhammer attacks. In the future, we plan to collect data from additional DRAM samples and formulate an algorithm capable of testing DRAM for rowhammer vulnerabilities by analyzing latency errors.

Acknowledgement

This work is supported partly by the National Science Foundation Awards # 2214108 and # 2114200.

References

[1] S. Ravi, A. Raghunathan, P. Kocher, and S. Hattangady, "Security in embedded systems: Design challenges," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 3, no. 3, pp. 461–491, 2004.

[2] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in internet of things: The road ahead," *Computer networks*, vol. 76, pp. 146–164, 2015.

[3] A. Barenghi *et al.*, "Fault injection attacks on cryptographic devices: Theory, practice, and countermeasures," *Proceedings of the IEEE*, vol. 100, no. 11, pp. 3056–3076, 2012.

[4] B. Yuce *et al.*, "Analyzing the fault injection sensitivity of secure embedded software," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 16, no. 4, pp. 1–25, 2017.

[5] O. Mutlu and J. S. Kim, "Rowhammer: A retrospective," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 8, pp. 1555–1571, 2019.

[6] O. Mutlu, "The rowhammer problem and other issues we may face as memory becomes denser," in *Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017*. IEEE, 2017, pp. 1116–1121.

[7] H. Aydin and A. Sertbaş, "Cyber security in industrial control systems (ics): a survey of rowhammer vulnerability," *Applied Computer Science*, vol. 18, no. 2, 2022.

[8] A. G. Yağlıkçı, J. S. Kim, F. Devaux, and O. Mutlu, "Security analysis of the silver bullet technique for rowhammer prevention," *arXiv preprint arXiv:2106.07084*, 2021.

[9] A. G. Yağlikçi, M. Patel, J. S. Kim, R. Azizi, A. Olgun, L. Orosa, H. Hassan, J. Park, K. Kanellopoulos, T. Shahroodi *et al.*, "Blockhammer: Preventing rowhammer at low cost by blacklisting rapidly-accessed dram rows," in *IEEE International Symposium on High-Performance Computer Architecture*. IEEE, 2021, pp. 345–358.

[10] C. Bock, F. Brasser, D. Gens, C. Liebchen, and A.-R. Sadeghi, "Rip-rh: Preventing rowhammer-based inter-process attacks," in *Proceedings of the 2019 ACM Asia Conference on Computer and Communications Security*, 2019, pp. 561–572.

[11] J. S. Kim, M. Patel, A. G. Yağlıkçı, H. Hassan, R. Azizi, L. Orosa, and O. Mutlu, "Revisiting rowhammer: An experimental analysis of modern dram devices and mitigation techniques," in *2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2020, pp. 638–651.

[12] Z. B. Aweke, S. F. Yitbarek, R. Qiao, R. Das, M. Hicks, Y. Oren, and T. Austin, "Anvil: Software-based protection against next-generation rowhammer attacks," *ACM SIGPLAN Notices*, vol. 51, no. 4, pp. 743–755, 2016.

[13] M. Farmani, M. Tehranipoor, and F. Rahman, "Rhat: Efficient rowhammer-aware test for modern dram modules," in *2021 IEEE European Test Symposium (ETS)*. IEEE, 2021, pp. 1–6.

[14] D. S. Specification, "Jedec standard," 2009.

[15] D. Lee *et al.*, "Adaptive-latency dram: Optimizing dram timing for the common-case," in *IEEE 21st International Symposium on High Performance Computer Architecture*. IEEE, 2015, pp. 489–501.

[16] B. B. Talukder, J. Kerns, B. Ray, T. Morris, and M. T. Rahman, "Exploiting dram latency variations for generating true random numbers," in *2019 IEEE International Conference on Consumer Electronics (ICCE)*. IEEE, 2019, pp. 1–6.

[17] B. B. Talukder, B. Ray, D. Forte, and M. T. Rahman, "Prelatpuf: Exploiting dram latency variations for generating robust device signatures," *IEEE Access*, vol. 7, pp. 81 106–81 120, 2019.

[18] M. T. Rahman and B. M. S. B. Talukder, "Systems and methods for identifying counterfeit memory," Oct. 5 2021, u

[19] B. Ray and M. T. Rahman, "Systems and counterfeit memory," Sep. 7 2021, uS Patent 1

[20] B. B. Talukder, V. Menon, B. Ray, T. Neal, and M the avoidance of counterfeit memory: Identifyi *2020 IEEE International Symposium on Hard and Trust (HOST)*. IEEE, 2020, pp. 111–121.

[21] H. Hassan *et al.*, "Softmc: A flexible and pr frastructure for enabling experimental dram s *International Symposium on High Performance (HPCA)*. IEEE, 2017, pp. 241–252.

[22] M. T. Aga, Z. B. Aweke, and T. Austin, "Wh bad: Exploiting anti-dos measures to accelera in *2017 IEEE International Symposium on Har and Trust (HOST)*. IEEE, 2017, pp. 8–13.

[23] M. Fieback, "Dram reliability: Aging analysis a model," 2017.

[24] B. Schroeder, E. Pinheiro, and W.-D. Weber, "I a large-scale field study," *Communications of t* pp. 100–107, 2011.

[25] K. Loughlin, S. Saroiu, A. Wolman, and B. I time: rethinking our approach to rowhammer n *ings of the Workshop on Hot Topics in Opera* 88–95.

[26] K. Kim, J. Woo, J. Kim, and K.-S. Chung, protection and low hardware overhead meth *2021 IEEE 39th International Conference on Co* IEEE, 2021, pp. 212–219.

[27] E. Lee, I. Kang, S. Lee, G. E. Suh, and J. H. Ahn, "Twice: Preventing row-hammering by exploiting time window counters," in *Proceedings of the 46th International Symposium on Computer Architecture*, 2019, pp. 385–396.

[28] S. M. Seyedzadeh, A. K. Jones, and R. Melhem, "Mitigating wordline crosstalk using adaptive trees of counters," in *2018 ACM/IEEE 45th Annual International Symposium on Computer Architecture (ISCA)*. IEEE, 2018, pp. 612–623.

[29] Y. others, "Understanding rowhammer under reduced wordline voltage: An experimental study using real dram devices," in *52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, 2022, pp. 475–487.

[30] K. K. Chang *et al.*, "Understanding latency variation in modern dram chips: Experimental characterization, analysis, and optimization," in *Proceedings of the 2016 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Science*, pp. 323–336.

[31] D. Lee *et al.*, "Design-induced latency variation in modern dram chips: Characterization, analysis, and latency reduction mechanisms," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 1, no. 1, pp. 1–36, 2017.

[32] J. S. Kim, M. Patel, H. Hassan, and O. Mutlu, "The dram latency puf: Quickly evaluating physical unclonable functions by exploiting the latency-reliability tradeoff in modern commodity dram devices," in *IEEE International Symposium on High Performance Computer Architecture*. IEEE, 2018, pp. 194–207.

[33] J. S. Kim, M. Patel, H. Hassan, L. Orosa, and O. Mutlu, "D-range: Using commodity dram devices to generate true random numbers with low latency and high throughput," in *IEEE International Symposium on High Performance Computer Architecture*. IEEE, 2019, pp. 582–595.

[34] A. Hosey *et al.*, "Advanced analysis of cell stability for reliable sram pufs," in *IEEE 23rd Asian Test Symposium*. IEEE, 2014, pp. 348–353.

[35] M. K. Bepary, B. M. S. B. Talukder, and M. T. Rahman, "Dram retention behavior with accelerated aging in commercial chips," *Applied Sciences*, vol. 12, no. 9, p. 4332, 2022.

[36] Z. Guo, M. T. Rahman, M. M. Tehranipoor, and D. Forte, "A zero-cost approach to detect recycled soc chips using embedded sram," in *2016 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, 2016, pp. 191–196.

APPENDIX

Device Aging vs. Disturbance Errors: With transistor aging, a DRAM might become leakier, or the cell isolation techniques to improve the robustness of the DRAM might become weaker [35]. Consequently, a DRAM might become more vulnerable to rowhammer attack. To demonstrate how rowhammer vulnerability changes as DRAM ages, we applied high temperature and high voltage for the 3, 5, and 8 hours

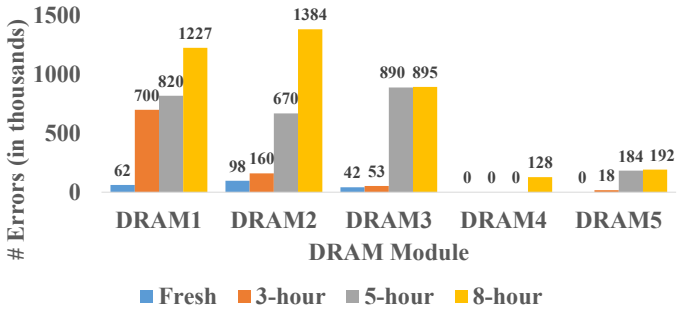

Fig. 4: A recycled DRAM might become more vulnerable to rowhammer as device ages.

- **Observation 1**: The number of erroneous cells (disturbance errors) increases as the device ages (Fig. 4).
- **Observation 2**: Some modules, such as DRAM4 and DRAM5, may develop vulnerability to rowhammer as the device ages, even if they were initially resistant. For DRAM4, no disturbance errors are observed at 0 hours (fresh), 3 hours, or 5 hours of aging, but a few errors are noticed after 8 hours of aging. Regarding DRAM5, disturbance errors are detected after aging the device for only 3 hours, and their frequency increases with further aging.
- **Observation 3**: The minimum number of required row accesses to flip the first bit reduces as the device ages. It requires $\sim 13\%$ fewer attempts to flip the first bit for DRAM1 with 8 hours of accelerated aging.