INVITED: Who Is the Major Threat to Tomorrow's Security? You, the Hardware Designer

Wayne Burleson
University of Massachusetts
Amherst

Onur Mutlu
ETH Zürich
Carnegie Mellon University

Mohit Tiwari
University of Texas Austin

ABSTRACT

More and more security attacks today are perpetrated by exploiting the hardware: memory errors can be exploited to take over systems, side-channel attacks leak secrets to the outside worlds, weak random number generators render cryptography ineffective, etc. At the same time, many of the tenets of efficient design are in tension with guaranteeing security. For instance, classic secure hardware does not allow to optimize common execution patterns, share resources, or provide deep introspection.

We provide brief descriptions of three recently-exposed hardware vulnerabilities along with extensive references for background and to learn more about these areas. Specifically, we first discuss the Rowhammer problem in modern DRAM chips, which enables attackers to circumvent memory isolation, and other potential vulnerabilities due to aggressive memory technology scaling. We next describe hardware Trojans implemented below the gate level, which can resist most detection techniques, and other manufacturing vulnerabilities in security primitives. Finally, we explain side channels that can achieve very large information leakage capacities, and various potential defenses against them. We conclude by noting that the intersection of hardware design and security attacks and countermeasures will continue to present a rich area for research and development for many years to come.

1. THE ROWHAMMER PROBLEM AND OTHER ISSUES WE MAY FACE AS MEMORY BECOMES DENSER

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 process technology scales down to smaller dimensions, memory chips become more vulnerable to *disturbance*, a phenomenon in which different memory cells interfere with each others' operation. We have shown, in our ISCA 2014 paper [41], the existence of *read disturbance errors* in commodity DRAM chips that are sold and used in the field today. Repeatedly reading from the same address in DRAM could corrupt data in nearby addresses. Specifically, when a DRAM row is opened (i.e., activated) and closed (i.e., precharged) repeatedly (i.e., *hammered*), enough times within a DRAM refresh interval, one or more bits in physically-adjacent DRAM rows can be flipped to the wrong value. This DRAM failure mode is popularly called *RowHammer* [40, 86, 1, 2, 79, 45, 6]. We tested 129 DRAM modules manufactured in seven recent years (2008–2014) and found that 110 of them exhibit RowHammer errors, the earliest of which dates back to 2010. In particular, *all* modules from 2012–2013 were vulnerable to RowHammer, implying that RowHammer is a recent phenomenon affecting more advanced process technology generations.

RowHammer exposes a *security threat* since it leads to a breach of memory protection, wherein accesses to one row (e.g., an OS page) modifies the data stored in another memory row (e.g., another OS page). Malicious software can be written to take advantage of these disturbance errors. We call these *disturbance attacks* [41], or *RowHammer attacks*. Such attacks can be used to corrupt system memory, crash a system, or take over the entire system. Confirming the predictions of our ISCA paper [41], researchers from Google Project Zero recently developed a user-level attack that exploits RowHammer to gain kernel privileges and thus take over an entire system [79]. More recently, researchers showed that RowHammer can be exploited remotely via the use of JavaScript [29]. As such, the RowHammer problem has widespread and profound real implications on system security, threatening the foundations of memory isolation on top of which modern system security principles are built.

RowHammer has recently been the subject of many popular analyses and discussions on hardware-induced security problems [40, 86, 1, 2, 30, 45, 6]. Several major system manufacturers increased DRAM refresh rates to reduce its probability of occurrence [5, 31, 49, 27]. Multiple memory test programs are now designed to test for it [68, 3]. Some recent reports suggest that even state-of-the-art DDR4 DRAM chips are vulnerable to RowHammer [45]. Our ISCA 2014 paper [41] discusses and analyzes many countermeasures to the RowHammer problem. We show that simple modifications to the memory controller, enabled by physical address mapping information provided by DRAM chips, can prevent the problem at low cost and low performance overhead. We believe such cooperation

© 2016 ACM. ISBN 123-4567-24-567/08/06.
DOI: 10.475/123_4

between and co-design of system and memory, i.e., *system-memory co-design*, as suggested and explored by recent works [65, 36, 67, 56, 39, 70, 38], can lead to more robust and secure memory systems.

We also believe that, as memory technologies scale to higher densities, other problems may start appearing (or may already be going unnoticed) that can potentially threaten the foundations of secure systems. There have been recent large-scale field studies of memory errors showing that both DRAM and NAND flash memory technologies are becoming less reliable [63, 80, 81, 82, 62, 78]. As detailed experimental analyses of real DRAM and NAND flash chips show, both technologies are becoming much more vulnerable to cell-to-cell interference effects [41, 13, 18, 15, 11, 12, 66], data retention is becoming significantly more difficult in both technologies [54, 53, 36, 39, 38, 70, 21, 58, 14, 17, 55, 11, 12, 16, 66], and error variation within and across chips is increasingly prominent [53, 48, 22, 20, 11, 12]. Emerging memory technologies [65, 61], such as Phase-Change Memory [46, 91, 72, 71, 87, 73, 47], STT-MRAM [23, 44], and RRAM/ReRAM/memristors [88] are likely to exhibit similar and perhaps even more exacerbated reliability issues. We believe these reliability problems may surface as security problems as well, as in the case of RowHammer. Principled electronic design, automation, and testing as well as principled co-architecting of the system and memory, therefore, have a critically important role to play in ensuring the security of future memory systems.

2. MANUFACTURING VULNERABILITIES IN SECURITY PRIMITIVES

Secure systems are built on a foundation of security primitives implemented in hardware and low-level software. These primitives increasingly rely on physical properties of the system to introduce entropy and avoid physical attacks [42]. However the manufacturing process enables hypothetical threats where very subtle modifications can be introduced that statistically weaken security primitives. Quantifying these vulnerabilities and developing countermeasures is an ongoing topic of research [51].

Recently, hardware Trojans have drawn the attention of governments and industry as well as the scientific community [37]. More generally, one of the main concerns is that integrated circuits, e.g., for military or critical infrastructure applications, could be maliciously manipulated during the manufacturing process, which often takes place abroad. There has been much speculation about types of Trojans and their implementation. In CHES 2014, we proposed a new approach for implementing hardware Trojans below the gate level [8], and we evaluated their impact on the security of the target device. Instead of adding additional circuitry to the target design, we insert our hardware Trojans by changing the dopant polarity of existing transistors. Since the modified circuit appears legitimate on all wiring layers (including all metal and polysilicon), our family of Trojans is resistant to most detection techniques, including fine-grain optical inspection and checking against reference "golden chips".

We demonstrate the effectiveness of our approach by inserting Trojans into two designs: 1) a digital post-processing derived from Intel's cryptographically secure RNG design used in the Ivy Bridge processors; 2) and a side-channel resistant SBox implementation; and by exploring their detectability and their effects on security. Countermeasures for these attacks and their observability have been discussed in follow up papers and on-line discussions.

An RNG or entropy harvesting device can also be biased directly through similar techniques to the stealthy hardware Trojans [7] [83]. In this more subtle case, the statistics of a metastable or delay-based arbiter can be modified through low-level doping or transistor sizing, or even parasitic capacitances. Physical unclonable functions are also vulnerable to this type of subtle statistical attack [75] [74]. Finally, countermeasures that have been introduced to mitigate physical side channel vulnerability can also be manipulated [52] [89] [76]. In all of these cases, detection of the direct manipulation is difficult and even detection of its ultimate impact on security is challenging [84].

Together these vulnerabilities motivate the need for methodologies and tools which carefully characterize these security properties and then validate both pre- and post-silicon that they are satisfied [90].

3. SIDE-CHANNEL ATTACKS AND SIDE-CHANNEL FREE ARCHITECTURES

Encryption and isolation are critical security primitives – yet, attackers can subvert both by inferring secret keys and values using unintentional side-effects of computation. Such *side-channels* include cache [69, 9] and memory [35, 85] usage, instruction latency [4, 28], program execution time [77, 10], memory address bus [57, 26], and even physical side-effects such as power draw [59, 43], electro-magnetic radiation [50, 19], or thermal hotspots [60]. Systems are complex and hence finding and quantifying of information leaks through side-channels is an extremely hard problem.

In recent work [33], we showed that side-channels can achieve capacities of hundreds of kilo-bytes per second – orders of magnitude higher than previously observed. Digging deeper, we observe that attackers rely on one of two mechanisms to create side-channels – (1) *contention* for a shared resource such as space in a last-level cache, or bandwidth for memory, I/O, or OS-level resources; and (2) *observation* channels where an attacker observes signals emanating from the victim/source program. Observing the power draw, EM, thermal hotspots, and even the address bus are examples of such observation channels.

We show that contention channels have an interesting property – *an attacker can only read a bit from the channel by overwriting the current bit.* Intuitively, this property arises because by contending for the shared resource, an attacker causes the source program's state to be perturbed (e.g., a cache line to be evicted or the memory bandwidth lowered). Interestingly, this destructive read property applies to both stateful (e.g. cache) as well as stateless bandwidth-based channels, and has a major implication for designing secure hardware and systems – *we can detect and prevent side-channel attacks using introspection hardware to detect anomalous contention.* Instead of using security-specific hardware to strictly partition all resources to isolate two security domains, a defense can simply monitor more general purpose event-counters to detect an attack as anomalous hardware activity. We demonstrate that introspection-based defenses work even against an intelligent

attacker who adapts to our proposed defense, and introduce new resource *contention counters* to improve detection rates beyond using only the standard performance counter interface.

External observation channels do not have the destructive read property and are considerably harder to defend against. We show initial results in bringing cryptographic techniques to seal leaks through the memory address bus – such Oblivious memory techniques incur large slowdowns and are best suited to extremely sensitive data [57, 26]. We also demonstrate analog channel attacks such as power against complex mobile software and draw common threads between the analog and digital observation channels to identify a potential defense – statistical and software-level obfuscation techniques that enable a quantifiable performance-security trade-off.

The take-away for a system designer is that side-channel attacks are extremely dangerous, but general purpose mechanisms that allow software to explicitly measure these channels are a low-cost, low-slowdown alternative to dedicated security-logic to partition or cryptographically seal these channels.

4. CONCLUSIONS

The three examples discussed above are just a sampling of the numerous hardware level vulnerabilities that are currently being explored. We refer the reader to leading hardware security conferences [24, 32] and security tracks at leading hardware conferences (e.g., [25, 64, 34]) for the latest research in these areas.

Although the three examples are quite different, they do exhibit some common themes. Shared resources, whether memory, power supply, or physical substrate, all provide subtle ways for untrusted processes to interact with the system and potentially gain control. Complete isolation of resources seems like the obvious solution, but is usually prohibitive in terms of cost, power, or performance. Low-level design details can sometimes be hidden from final system validation steps. These can provide a point of entry for a sophisticated attacker at the manufacturing level.

Fortunately, some common countermeasures, or at least common approaches, seem to be able thwart many of these problems. They require designer awareness, tools, and methodologies that ensure that security properties are guaranteed at various levels of design and manufacturing and across various system components, such as memory and the processor. Unfortunately, the list of new vulnerabilities seems to grow as fast as or faster than appropriate countermeasures can be devised. Skills in both system security as well as hardware design, design automation, and computer architecture are needed. Therefore, we believe the intersection of hardware design and security attacks and countermeasures will continue to present a rich area for research and development for many years to come.

5. REFERENCES

[1] RowHammer Discussion Group. https://groups.google.com/forum/#!forum/rowhammer-discuss.

[2] RowHammer on Twitter. https://twitter.com/search?q=rowhammer&src=typd.

[3] B. Aichinger. The Known Failure Mechanism in DDR3 Memory referred to as Row Hammer. http://ddrdetective.com/files/6414/1036/5710/The_Known_Failure_Mechanism_in_DDR3_memory_referred_to_as_Row_Hammer.pdf, September 2014.

[4] M. Andrysco, D. Kohlbrenner, K. Mowery, R. Jhala, S. Lerner, and H. Shacham. On subnormal floating point and abnormal timing. In *IEEE S&P*, May 2015.

[5] Apple Inc. About the security content of Mac EFI Security Update 2015-001. https://support.apple.com/en-us/HT204934, June 2015.

[6] 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. In *ASPLOS*, 2016.

[7] G. T. Becker, A. Lakshminarasimhan, L. Lin, S. Srivathsa, V. B. Suresh, and W. Burleson. Implementing hardware trojans: Experiences from a hardware trojan challenge. In *ICCD*, 2011.

[8] G. T. Becker, F. Regazzoni, C. Paar, and W. P. Burleson. Stealthy dopant-level hardware trojans. In *CHES*, 2013.

[9] D. J. Bernstein. Cache-timing Attacks on AES, 2005.

[10] D. Brumley and D. Boneh. Remote timing attacks are practical. In *USENIX Security Symposium*, 2005.

[11] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai. Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis. In *DATE*, 2012.

[12] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai. Threshold voltage distribution in MLC NAND flash memory: characterization, analysis, and modeling. In *DATE*, 2013.

[13] Y. Cai, Y. Luo, S. Ghose, and O. Mutlu. Read disturb errors in MLC NAND flash memory: Characterization, mitigation, and recovery. In *DSN*, 2015.

[14] Y. Cai, Y. Luo, E. F. Haratsch, K. Mai, and O. Mutlu. Data retention in MLC NAND flash memory: Characterization, optimization, and recovery. In *HPCA*, 2015.

[15] Y. Cai, O. Mutlu, E. F. Haratsch, and K. Mai. Program interference in MLC NAND flash memory: Characterization, modeling, and mitigation. In *ICCD*, 2013.

[16] Y. Cai, G. Yalcin, O. Mutlu, E. F. Haratsch, A. Cristal, O. Unsal, and K. Mai. Error Analysis and Retention-Aware Error Management for NAND Flash Memory. *Intel Technology Journal, Special Issue on Memory Resiliency*, May 2013.

[17] Y. Cai, G. Yalcin, O. Mutlu, E. F. Haratsch, A. Cristal, O. S. Ünsal, and K. Mai. Flash correct-and-refresh: Retention-aware error management for increased flash memory lifetime. In *ICCD*, 2012.

[18] Y. Cai, G. Yalcin, O. Mutlu, E. F. Haratsch, O. S. Unsal, A. Cristal, and K. Mai. Neighbor-cell assisted error correction for MLC NAND flash memories. In *SIGMETRICS*, 2014.

[19] R. Callan, A. Zajić, and M. Prvulovic. A practical methodology for measuring the side-channel signal available to the attacker for instruction- level events. In *MICRO*, 2014.

[20] K. Chandrasekar, S. Goossens, C. Weis, M. Koedam, B. Akesson, N. Wehn, and K. Goossens. Exploiting Expendable Process-margins in DRAMs for Run-time Performance Optimization. In *DATE*, 2014.

[21] K. Chang, D. Lee, Z. Chishti, C. Wilkerson, A. Alameldeen, Y. Kim, and O. Mutlu. Improving DRAM Performance by Parallelizing Refreshes with Accesses. In *HPCA*, 2014.

[22] K. K. Chang, A. Kashyap, H. Hassan, S. Ghose, K. Hsieh, D. Lee, T. Li, G. Pekhimenko, S. Khan, and O. Mutlu. Understanding Latency Variation in Modern DRAM Chips: Experimental Characterization, Analysis, and Optimization. In *SIGMETRICS*, 2016.

[23] E. Chen et al. Advances and Future Prospects of Spin-Transfer Torque Random Access Memory. *IEEE Transactions on Magnetics*, 46(6), 2010.

[24] CHES. Conference on Cryptographic Hardware and Embedded Systems. http://www.chesworkshop.org.

[25] DAC. Design Automation Conference. https://dac.com/.

[26] C. W. Fletcher, M. v. Dijk, and S. Devadas. A secure processor architecture for encrypted computation on untrusted programs. In *STC*, 2012.

[27] T. Fridley and O. Santos. Mitigations Available for the DRAM Row Hammer Vulnerability. http://blogs.cisco.com/security/mitigations-available-for-the-dram-row-hammer-vulnerability, March 2015.

[28] J. Großschädl, E. Oswald, D. Page, and M. Tunstall. Side-channel analysis of cryptographic software via early-terminating multiplications. In *ICISC*, 2010.

[29] D. Gruss, C. Maurice, and S. Mangard. Rowhammer.js: A remote software-induced fault attack in javascript. *CoRR*, abs/1507.06955, 2015.

[30] R. Harris. Flipping DRAM bits - maliciously. http://www.zdnet.com/article/flipping-dram-bits-maliciously/, December 2014.

[31] Hewlett-Packard Enterprise. HP Moonshot Component Pack Version 2015.05.0. http://h17007.www1.hp.com/us/en/enterprise/servers/products/moonshot/component-pack/index.aspx, 2015.

[32] HOST. IEEE International Symposium on Hardware Oriented Security and Trust. http://www.hostsymposium.org/.

[33] C. Hunger, M. Kazdagli, A. Rawat, A. Dimakis, S. Vishwanath, and M. Tiwari. Understanding contention-based channels and using them for defense. In *HPCA*, Feb 2015.

[34] ISCA. International Symposium on Computer Architecture. http://isca2016.eecs.umich.edu/.

[35] S. Jana and V. Shmatikov. Memento: Learning secrets from process footprints. In *IEEE S&P*, 2012.

[36] U. Kang, H. Yu, C. Park, H. Zheng, J. Halbert, K. Bains, S. Jang, and J. S. Choi. Co-Architecting Controllers and DRAM to Enhance DRAM Process Scaling. In *The Memory Forum (ISCA)*, 2014.

[37] R. Karri, J. Rajendran, K. Rosenfeld, and M. Tehranipoor. Trustworthy hardware: Identifying and classifying hardware trojans. *Computer*, (10):39–46, 2010.

[38] S. Khan, D. Lee, C. Wilkerson, and O. Mutlu. PARBOR: An Efficient System-Level Technique to Detect Data Dependent Failures in DRAM. In *DSN*, 2016.

[39] S. M. Khan, D. Lee, Y. Kim, A. R. Alameldeen, C. Wilkerson, and O. Mutlu. The efficacy of error mitigation techniques for DRAM retention failures: a comparative experimental study. In *SIGMETRICS*, 2014.

[40] Y. Kim, R. Daly, J. Kim, C. Fallin, J. Hye Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu. RowHammer: Reliability Analysis and Security Implications. *CoRR*, abs/1603.00747, Feb. 2016.

[41] Y. Kim, R. Daly, J. Kim, C. Fallin, J. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu. Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors. In *ISCA*, 2014.

[42] P. Kocher, J. Jaffe, and B. Jun. Differential power analysis. In *Advances in Cryptology*, 1999.

[43] P. Kocher, J. Jaffe, and B. Jun. Using unpredictable information to minimize leakage from smartcards and other cryptosystems, Dec. 4 2001. US Patent 6,327,661.

[44] E. Kultursay, M. Kandemir, A. Sivasubramaniam, and O. Mutlu. Evaluating STT-RAM as an Energy-Efficient Main Memory Alternative. In *ISPASS*, 2013.

[45] M. Lanteigne. How Rowhammer Could Be Used to Exploit Weaknesses in Computer Hardware. http://www.thirdio.com/rowhammer.pdf, March 2016.

[46] B. C. Lee, E. Ipek, O. Mutlu, and D. Burger. Architecting Phase Change Memory as a Scalable DRAM Alternative. In *ISCA*, 2009.

[47] B. C. Lee, P. Zhou, J. Yang, Y. Zhang, B. Zhao, E. Ipek, O. Mutlu, and D. Burger. Phase change technology and the future of main memory. *IEEE Micro*, 30(1), 2010.

[48] D. Lee, Y. Kim, G. Pekhimenko, S. Khan, V. Seshadri, K. Chang, and O. Mutlu. Adaptive-Latency DRAM: Optimizing DRAM Timing for the Common-Case. In *HPCA*, 2015.

[49] Lenovo. Row Hammer Privilege Escalation. https://support.lenovo.com/us/en/product_security/row_hammer, March 2015.

[50] G. liang Ding, Z. xiang Li, X. long Chang, and Q. Zhao. Differential electromagnetic analysis on aes cryptographic system. In *WMWA*, June 2009.

[51] L. Lin and W. Burleson. Analysis and mitigation of process variation impacts on power-attack tolerance. In *DAC*, 2009.

[52] L. Lin, W. Burleson, and C. Paar. Moles: malicious off-chip leakage enabled by side-channels. In *ICCAD*, 2009.

[53] J. Liu, B. Jaiyen, Y. Kim, C. Wilkerson, and O. Mutlu. An experimental study of data retention behavior in modern DRAM devices: implications for retention time profiling mechanisms. In *ISCA*, 2013.

[54] J. Liu, B. Jaiyen, R. Veras, and O. Mutlu. RAIDR: Retention-Aware Intelligent DRAM Refresh. In *ISCA*, 2012.

[55] Y. Luo, Y. Cai, S. Ghose, J. Choi, and O. Mutlu. WARM: improving NAND flash memory lifetime with write-hotness aware retention management. In *MSST*, 2015.

[56] Y. Luo, S. Govindan, B. Sharma, M. Santaniello, J. Meza, A. Kansal, J. Liu, B. Khessib, K. Vaid, and O. Mutlu. Characterizing Application Memory Error Vulnerability to Optimize Data Center Cost via Heterogeneous-Reliability Memory. In *DSN*, 2014.

[57] M. Maas, E. Love, E. Stefanov, M. Tiwari, E. Shi, K. Asanovic, J. Kubiatowicz, and D. Song. Phantom: Practical oblivious computation in a secure processor. In *CCS*.

[58] J. Mandelman et al. Challenges and future directions for the scaling of dynamic random-access memory (DRAM). *IBM Journal of Research and Development*, 46, 2002.

[59] S. Mangard. A simple power-analysis (spa) attack on implementations of the aes key expansion. In *ICISC*, 2003.

[60] R. J. Masti, D. Rai, A. Ranganathan, C. Müller, L. Thiele, and S. Capkun. Thermal covert channels on multi-core platforms. In *USENIX Security Symposium*, 2015.

[61] J. Meza, Y. Luo, S. Khan, J. Zhao, Y. Xie, and O. Mutlu. A Case for Efficient Hardware-Software Cooperative Management of Storage and Memory. In *WEED*, 2013.

[62] J. Meza, Q. Wu, S. Kumar, and O. Mutlu. A large-scale study of flash memory failures in the field. In *SIGMETRICS*, 2015.

[63] J. Meza, Q. Wu, S. Kumar, and O. Mutlu. Revisiting memory errors in large-scale production data centers: Analysis and modeling of new trends from the field. In *DSN*, 2015.

[64] MICRO. International Symposium on Microarchitecture. http://www.microarch.org/.

[65] O. Mutlu. Memory Scaling: A Systems Architecture Perspective. In *IMW*, 2013.

[66] O. Mutlu. Error Analysis and Management for MLC NAND Flash Memory. In *Flash Memory Summit*, 2014.

[67] O. Mutlu and L. Subramanian. Research problems and opportunities in memory systems. *Supercomputing Frontiers and Innovations*, 2015.

[68] PassMark Software. MemTest86: The original industry standard memory diagnostic utility. http://www.memtest86.com/troubleshooting.htm, 2015.

[69] C. Percival. Cache Missing for Fun and Profit. In *BSDCon*, 2005.

[70] M. Qureshi, D. H. Kim, S. Khan, P. Nair, and O. Mutlu. AVATAR: A Variable-Retention-Time (VRT) Aware Refresh for DRAM Systems. In *DSN*, 2015.

[71] M. K. Qureshi, J. Karidis, M. Franceschini, V. Srinivasan, L. Lastras, and B. Abali. Enhancing Lifetime and Security

of Phase Change Memories via Start-Gap Wear Leveling. In *MICRO*, 2009.

[72] M. K. Qureshi, V. Srinivasan, and J. A. Rivers. Scalable High Performance Main Memory System using Phase-Change Memory Technology. In *ISCA*, 2009.

[73] S. Raoux et al. Phase-change random access memory: A scalable technology. *IBM Journal of Research and Development*, 52, Jul/Sep 2008.

[74] U. Ruhrmair, J. Solter, F. Sehnke, X. Xu, A. Mahmoud, V. Stoyanova, G. Dror, J. Schmidhuber, W. Burleson, and S. Devadas. PUF modeling attacks on simulated and silicon data. *Information Forensics and Security, IEEE Transactions on*, 8(11), 2013.

[75] U. Rührmair, X. Xu, J. Sölter, A. Mahmoud, F. Koushanfar, and W. Burleson. Power and Timing Side Channels for PUFs and their Efficient Exploitation. *IACR Cryptology ePrint Archive*, 2013.

[76] U. Rührmair, X. Xu, J. Sölter, A. Mahmoud, M. Majzoobi, F. Koushanfar, and W. Burleson. Efficient power and timing side channels for physical unclonable functions. In *CHES*, 2014.

[77] W. Schindler. A timing attack against RSA with the chinese remainder theorem. In *CHES*, 2000.

[78] B. Schroeder, R. Lagisetty, and A. Merchant. Flash Reliability in Production: The Expected and the Unexpected. In *USENIX FAST*, 2016.

[79] M. Seaborn. Exploiting the DRAM rowhammer bug to gain kernel privileges. http://googleprojectzero.blogspot.com.tr/2015/03/exploiting-dram-rowhammer-bug-to-gain.html.

[80] V. Sridharan, N. DeBardeleben, S. Blanchard, K. B. Ferreira, J. Stearley, J. Shalf, and S. Gurumurthi. Memory errors in modern systems: The good, the bad, and the ugly. In *ASPLOS*, 2015.

[81] V. Sridharan and D. Liberty. A study of DRAM failures in the field. In *SC*, 2012.

[82] V. Sridharan, J. Stearley, N. DeBardeleben, S. Blanchard, and S. Gurumurthi. Feng shui of supercomputer memory: positional effects in DRAM and SRAM faults. In *SC*, 2013.

[83] V. B. Suresh and W. P. Burleson. Entropy extraction in metastability-based trng. In *HOST*, 2010.

[84] K. Tiri, M. Akmal, and I. Verbauwhede. A dynamic and differential cmos logic with signal independent power consumption to withstand differential power analysis on smart cards. In *ESSCIRC*, 2002.

[85] Y. Wang, A. Ferraiuolo, and G. E. Suh. Timing Channel Protection for a Shared Memory Controller. In *HPCA*, 2014.

[86] Wikipedia. Row hammer. https://en.wikipedia.org/wiki/Row_hammer.

[87] H.-S. P. Wong et al. Phase Change Memory. *Proceedings of the IEEE*, 2010.

[88] H.-S. P. Wong et al. Metal-oxide RRAM. *Proceedings of the IEEE*, 2012.

[89] X. Xu and W. Burleson. Hybrid side-channel/machine-learning attacks on PUFs: a new threat? In *DATE*, 2014.

[90] X. Xu, V. Suresh, R. Kumar, and W. Burleson. Post-silicon validation and calibration of hardware security primitives. In *ISVLSI*, 2014.

[91] P. Zhou, B. Zhao, J. Yang, and Y. Zhang. A durable and energy efficient main memory using phase change memory technology. In *ISCA*, 2009.