



KOREAN SOCIETY FOR INTERNET INFORMATION

The 15th Asia Pacific International Conference on Information Science and Technology (APIC-IST 2020)

July 5-7, 2020, Sheraton Seoul Palace Gangnam Hotel, Korea

<http://www.apicist.org/2020>

Proceedings of APIC-IST 2020

| Organized by |

Korean Society for Internet Information (KSII)

The University of Queensland (Australia)

| Sponsored by |



Contents



10-8	Research on Response to Malicious Insider Threats using Feature Importance Analysis Dong-Wook Kim, Gun-Yoon Shin, Sung-Sam Hong (Gachon Univ., ROK)	263-264
10-9	Evaluation of Weather data, Power Demand and SMP Prediction Model by using LightGBM Yun-Jae Shin, KangYoon Lee (Gachon Univ., ROK)	265-267
11-1	An Efficient Resource Provisioning Scheme based on Machine Learning for Heterogeneous Internet of Things Sa Math, Prohim Tam, Chaebeen Nam, Seokhoon Kim (Soonchunhyang Univ., ROK)	268-270
11-2	Financial Fraud Detection adopting Distributed Deep Learning in Big Data Priyanka Purushu (AT&T, USA), Jongwook Woo (California State Univ., Los Angeles, USA)	271-273
11-3	BP Neural Network based RFID Indoor Location Method Chengbo Zhang (Nanjing Univ. of Information Science and Technology, China), Haoran Pei (Anqing Normal Univ., China), Zilong Jin (Nanjing Univ. of Information Science and Technology, China), Lejun Zhang (Yangzhou Univ., China)	274-277
11-4	Gaussian Process for Uncertainty in Predictive Analysis Oakyoungh Han, Jaehyoun Kim (Sungkyunkwan Univ., ROK)	278-280
11-5	Cross-Architecture Linux Malware Classification Method based on Binary Lifting Hae-Seon Jeong, Jin Kwak (Ajou Univ., ROK)	281-284
11-6	Detecting Technique of Rowhammer Malware Based on Static Analysis Minkyung Lee, Deuk-Hun Kim, Jin Kwak (Ajou Univ., ROK)	285-289

Detecting Technique of Rowhammer Malware Based on Static Analysis

Minkyung Lee¹, Deuk-Hun Kim² and Jin Kwak³

^{1,3}Department of Cyber Security, Ajou University
Suwon, Republic of Korea

²ISAA Lab., Department of Computer Engineering, Ajou University
Suwon, Republic of Korea

[e-mail: leemkl@ajou.ac.kr, dhkim.isaa@gmail.com, security@ajou.ac.kr]

*Corresponding author: Jin Kwak

Abstract

Recently, the rowhammer attack which is one of the side-channel attacks was announced by google project. It can be exploited such as privilege escalation by using the characteristics of the DRAM and can be attacked remotely using JavaScript. Thus, a side-channel attack is being advanced from using physically generated information through physical access to using the information through remote access. Preventing this, various studies have been being conducted on detecting rowhammer attacks. Most of them proposed dynamic detection techniques that monitor DRAMs. However, it is required that a study of static detection techniques without executing malicious code since dynamic detection techniques can generate overhead in the detection process. In this paper, we analyze rowhammer attacks and propose a static detection technique to counter it.

Keywords: Side-Channel Attack, Rowhammer Attack, Malware Detection

1. Introduction

The rowhammer attack exploits the physical architecture of modern Dynamic Random Access Memory (DRAM), which is composed of numerous rows of densely capacitive bit cells [1]. The rowhammer attack was first known by a study of Carnegie Mellon University. V. Van Der Veen et al. had acquired root authority on android through the rowhammer attack and discovered an attack with full control [2,3]. The attack also used to JavaScript in the Internet environment [4]. In addition, the rowhammer attack enabled encryption key tampering [5]. As such, various attacks are emerging using rowhammer attack which can operate in various environments.

Various studies are being conducted to detect and mitigate the rowhammer to counter these rowhammer attacks. However, the detection techniques should directly execute the

rowhammer attack and also continuously monitor an attack. Mitigation methods also require structural changes. Therefore, research on static detection methods that can be detected before executing the attack code are required but insufficient.

This paper is organized as follows: section 2 of this paper discusses studies related to rowhammer attack and mitigation methods of rowhammer. In section 3, we proposed how to detect rowhammer attacks based on static analysis of opcodes and APIs. Finally, section 4 presents the conclusion of this paper.

2. Related Works

2.1 Rowhammer Attack

The rowhammer attack exploits the physical structure of modern DRAM, which consists of densely bit cells. It is consists of N-type Metal

Oxide Semiconductor(NMOS) and capacitor which have the characteristic of electrical charge being short-circulated over time. Therefore, a continuous re-read operation should generate refreshing to prevent electric leakage. However, if refreshing occurs frequently, the contents of the surrounding cells can be changed due to the influence of the electric field. The change in cell storage value is called bit flip phenomenon. The phenomenon poses a security threat in modern DRAM chips as attackers can repeatedly open or close DRAM rows in the same memory bank to induce bit flips in the adjacent rows.

Table 1. Assembly Code to Exploiting the Attack

Code :		
1	mov(X), %eax	// read values of address X and Y
2	mov(Y), %ebx	
3	clflush(X)	// evict data in the cache
4	clflush(Y)	
5	mfence	// prevent parallelism
6	jmp code	// repeat above stage

The rowhammer attack operates as Table 1, and the operating stage is as follows [1].

Step 1. Two mov Instruction

It reads the values of address X and Y.

Step 2. Two clflush Instruction

It evicts the data stored in the cache.

Step 3. mfence Instruction

It ensures the data is fully flushed when continuous memory commands are executed.

Step 4. Repeat Step 1. ~ Step 3.

Finally, it returns to the first command and repeats the read operation.

2.2 Mitigating Rowhammer Attacks

Studies to mitigate rowhammer attacks are divided into hardware and software methods, such as Table 2. The hardware approach was designed to additional hardware resources to monitor the status of the victim row, and the software approach was designed to prevent data loss due to activities in adjacent rows by applying a refresh command when approaching the victim row, regardless of whether or not a memory error occurred.

□ Mitigating Rowhammer based on Hardware

D.H. Kim et al. proposed two techniques: Counter-Based Row Activation(CRA) and Probabilistic Row Activation(PRA). CRA is located in reserved areas of DRAM and tracks activation for each row through Row Activation Counter(RAC). However, this measure shows performance degradation on execution time by the counter's memory access. Instead of preventing CRA's storage overhead, the PRA performs row activation of adjacent rows due to a constant probability each time a given row is accessed [6]. While PRAs can reduce storage overhead than CRAs, the more rows reach a certain probability, the more overhead they occur. V. Van der Veen et al have proposed Guradion to prevent DMA-based attacks in the ARM environment. By adding two empty rows called guard row, the buffer is physically isolated so that it can fall more than one row away from the attack row. As a result, it mitigated rowhammer attacks by eliminating the possibility that an attacker could cause a bit flip. However, the Guardion can affect device performance because the process of creating guard rows consumes the DRAM memory of the user's device [3]. In addition, DRAM manufacturers have recently applied Target Row Refresh (TRR) in DDR4 DRAM, which refreshes adjacent rows if the target row approaches above the threshold to mitigate rowhammer attack. However, even with the TRR approach, an attacker can reactivate a bit flip attack on the latest hardware through a new hammering pattern [7].

□ Mitigating Rowhammer based on Software

Z.B. Aweke et al. have proposed a software-based two-step protocol mechanism called ANVIL. ANVIL constantly monitors the number of LCC cache misses and examines whether the number of cache misses exceeds a predetermined threshold. If the cache is missing for some time, the software module triggers the DRAM access sampling. ANVIL optionally performs a row refresh when it detects that the software module repeatedly accesses certain rows in the same bank [8]. ANVIL requires continuous monitoring of specific rows in the same bank. S. Vig proposed a bit flip detection method using a dynamic tree structure.

Table 2. Mitigating Rowhammer Attacks

Type	Method of Mitigation	Description	Disadvantages
Hardware	CRA & PRA	Perform row activation whenever a given row is accessed	Overhead occurrences
	Guardion	Add guard row for buffer isolation	Device performance degradation due to memory consumption
	TRR	The technique which refreshes the target row if it approaches above the threshold value applied to DDR4 DRAM	If TRR is applied, bit flip attack is possible with new hammering pattern
Software	ANVIL	Monitor the number of missing LLC cache and perform a row refresh on missing.	Require of continuous monitoring
	Skewed Hash Tree	Identify frequently accessed victim rows within a specific some time through sliding window mechanism and verify integrity with a hash tree	Memory overhead according to Leaf node size

The detection method uses Memory Controller(MC) consisting of checker and on-chip memory. The method uses a sliding window mechanism by monitoring the DRAM rows to identify the victim row that is frequently accessed within the time. The victim row identified in the sliding window is dynamically placed in the integration tree and used to verify its integrity by using the skewed hash tree. However, the corresponding detection method has memory overhead depending on the size of the leaf node [9].

Hardware-based mitigation methods can affect device performance because they require additional hardware resources. In addition, various mitigation techniques can be re-activated to enable rowhammer attacks as attacks become more intelligent. Software-based mitigation methods require continuous monitoring and unnecessary row refresh causes unnecessary power and performance overhead. Therefore, research is needed to apply a rowhammer detection technique through static analysis to prevent rowhammer attacks in advance.

3. Proposed Technique

3.1 Analysis of Malware

Opcodes and APIs of three types of attack files released on Github were analyzed. The opcodes frequency used in Table 1 was checked to enable bit flip. After that, opcodes and APIs used the same in all attack

files were analyzed. As a result, the signature is as follows.

❑ **mov(a), %b**

It used to copy data. It copies the value located in the parameter entered address(a) to the register(b). This is the command used to perform repeated read operations. The opcode is repeated to perform the hammering operation [1].

❑ **cflush**

This command used to remove a cache line that contains an address that has been passed over to a parameter [10]. By removing the cache line, it prevents the memory from being read from the cache when performing repetitive read operations. Therefore, the memory can be read from the DRAM [1].

❑ **mfence**

This command used to prevent parallel execution of memory load and storage commands [10]. It prevents parallelism when successive memory commands are executed. Therefore, it ensures that the data is completely flushed [1].

❑ **cpuid**

It is a command that informs the model, manufacturer, etc. of processor [10]. It is used to different attack patterns, such as identifying system information through cpuid and reordering some commands according to the information.

❑ **rdtscp**

Rdtscp is a serialization command that returns the TSC value of the process [10]. It measures

memory access time in a rowhammer attack.

□ open("/proc/self/pagemap")

As a command to open pagemap, pagemap contains information about page tables and /proc [11]. This allows physical addresses to be obtained for the mapped page, which was used to identify the victim row in the attack.

□ mmap

It is an API that makes respond to a given length of memory at the starting position of a given object. It was used to set the page non-empty in a rowhammer attack.

Based on the above opcodes and APIs analysis, the frequency used for the three types of attack files and other attack files frequently using clflush were as shown in Table 3. At this time, the move operation measured the number of times it had repeated access to the same memory row. In addition, open operation measured the number of commands to open /proc/self/pagemap.

Table 3. Frequency of Opcodes and APIs using the Rowhammer Attack

(unit : count)

	Row hammer 1	Row hammer 2	Row hammer 3	Other Attack
mov	10,000,000	1,024,000	1,200,000	0
clflush	1,000,000	1,024,000	1,200,000	7,424
cpuid	3	0	1	0
rdtscp	3	0	1	65,536
mfence	0	10	2	0
mmap	1	1	1	0
open	1	1	1	0

The rowhammer attack showed that mov, clflush, which enables bit flip, was used more than 1 million times in all attack files, and it was confirmed that cpuid, rdtscp, and mfence were used selectively. Also, mmap and open("/proc/self/pagemap") were used once in all attack files. For other attack files, opcode commonly used with rowhammer was identified as clflush and rdtscp, but clflush was found to be less frequent than rowhammer.

3.2 Detecting Technique based on Static Analysis

The following detection techniques are proposed that can detect rowhammer attack files based on

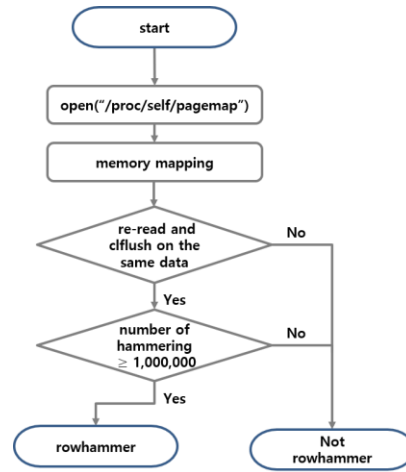


Fig. 1. Flow Chart for Detecting Rowhammer Malware

Table 3 such as **Fig. 1**. Step 1~2 use the instruction for setting, which is commonly used for rowhammer operation. Also, Step 3~4 use commonly used opcode to cause bit flip.

Step 1. open("/proc/self/pagemap")

If it uses the API that reads mapped page information, go to Step 2.

Step 2. Memory Mapping

If it uses memory mapping function using mmap, go to Step 3.

Step 3. Re-read and clflush on the Same Data

If it uses repetitive read operations and clflush within the same data, go to Step 4. Otherwise, it detects as a generic file, not a rowhammer attack file.

Step 4. Number of Hammering ≥ 1,000,000

If the hammering operation in Step 3 is repeated more than 1,000,000 times, the file is detected as a rowhammer attack file. Otherwise, it detects as a generic file, not a rowhammer attack file.

4. Conclusions

In this paper, we analyzed the rowhammer attack that modulates data on surrounding victim row due to the influence of the electric field when carrying out repetitive re-read on attack row in modern DRAM with the structure of densely cells. Based on this analysis, we proposed a technique to statically detect rowhammer attack.

In future work, applying the detection method proposed in this paper, we verify the detection method through a confusion matrix of accuracy, precision, and recall, etc.

References

- [1] Y. Kim, R. Daly, J. Kim, C. Fallin, J.H. Lee, D. Lee, C. Wilkerson, K. Lai, and O. Mutlu, "Flipping bits in memory without accessing them: an experimental study of DRAM disturbance errors", *SIGARCH Computer Architecture News*, Vol. 42, No. 3, pp. 361–337, Jun. 2014.
- [2] V. Van Der Veen, Y. Fratantonio, M. Lindorfer, D. Gruss, C. Maurice, G. Vigna, H.Box and K.Razavi, "Drammer: Deterministic rowhammer attacks on mobile platforms", *2016 ACM SIGSAC conference on computer and communications security*, pp. 1675-1689, Oct. 2016.
- [3] V. Van Der Veen, Y. Fratantonio, M. Lindorfer, D. Gruss, C. Maurice, G. Vigna, H.Box and K.Razavi, "GuardION: Practical mitigation of DMA-based rowhammer attacks on ARM", *International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment*, pp. 92-113, Jun. 2018.
- [4] D. Gruss, C. Maurice, and S. Mangard, "Rowhammer. js: A remote software-induced fault attack in javascript", *International conference on detection of intrusions, malware and vulnerability assessment*, pp. 300-321, Jun. 2016.
- [5] S. Bhattacharya and D. Mukhopadhyay, "Curious case of rowhammer: flipping secret exponent bits using timing analysis", *International Conference on Cryptographic Hardware and Embedded Systems*, pp. 602-624, Aug. 2016.
- [6] D.H. Kim, P.J. Nair and M.K. Qureshi, "Architectural support for mitigating row hammering in DRAM memories", *IEEE Computer Architecture Letters*, pp. 9-12, Jun. 2014.
- [7] P. Frigo, E. Vannacci, H. Hassan, V. Van Der Veen, O. Mutlu, C. Giuffrida, H. Bos and K. Razavi, "TRRespass: Exploiting the Many Sides of Target Row Refresh", *Cornell University*, 2020.
- [8] 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(Special Interest Group on Programming Languages) Notices*, Vol. 51, No. 4, pp. 743-755, May. 2016.
- [9] S. Vig, S. Bhattacharya, D. Mukhopadhyay, and S.K. Lam, "Rapid detection of Rowhammer attacks using dynamic skewed hash tree", *Proceedings of the 7th International Workshop on Hardware and Architectural Support for Security and Privacy*, No. 7, pp. 1-8, Jun. 2018.
- [10] M. Kim and M. Lee, "Detecting Meltdown and Spectre Malware through Binary Pattern Analysis", *Journal of the Korea Institute of Information Security & Cryptology*, Vol. 29, No. 6, pp. 1365-1373, Dec. 2019.
- [11] Linux Kernel Organization, "pagemap-The Linux Kernel Archives", <https://www.kernel.org/doc/Documentation/vm/pagemap.txt>