2025년도 한국통신학회 추계종합학술발표회

DRAM 메모리의 로우 해머와 오류정정부호에 대해

이대희, 하태욱, 김규리, 김상효*
성균관대학교 전자전기컴퓨터공학과

*iamshkim@skku.edu

On the Rowhammer and ECC of DRAM

Daehee Lee, Taeuk Ha, Gyuri Kim, Sang-Hyo Kim*
Department of Electrical and Computer Engineering, Sungkyunkwan University

요 약

본 논문은 DRAM의 로우 해머(rowhammer) 취약점과 ECC(Error Correction Codes) 기반 방어의 한계를 정리하고, 운영체제 연계 기법인 Copy-on-Flip(CoF)의 효과와 제약을 분석한다. 로우 해머는 특정 행의 고속 반복 활성화를 통해 인접 행의 전하를 교란하여 비트 플립을 유발하며, ECC는 비트 오류를 정정해 위험을 낮추지만 동일 부호어 내에서의 누적 플립에는 취약하다. 본 논문은 로우 해머 관련 ECC 선행 연구를 정리하고 ECC 강화와 운영체제·메모리 컨트롤러·펌웨어를 결합한 다층적 방어의 필요성과, 그 오버헤드 최소화 방안을 향후 연구 과제로 제시한다.

Ⅰ. 서 론

DRAM(dynamic random access memory)이 1967년 Robert Dennard에 의해 제안된 이후[1], 지속적인 공정 기술 스케일링을 통한 셀 축소화 및 셀 집적화가 계속되었다[2]. 이는 비트 당 비용을 줄이는 등 여러 이점을 가져왔지만, 작아진 커패시터 용량에 따른 마진의 감소, 가까워진 셀 사이 상호 간섭의 증가와 같은 문제점 또한 동반하였다[3].

이로 인해 증가한 오류에 대응하기 위해 오류정정부호(Error Correction Codes, ECC)가 DRAM에 도입되기 시작하였는데, 이를 통해 데이터가 프로세서에 전달되기 전 자체적인 오류 정정이 가능했으나 추가 저장 공간과 시간 지연이라는 오버 헤드가 존재하며 정정 범위를 넘어서는 오류는 정정하지 못하기 때문에 적은 패리티와 시간 지연을 가지면서 더 많은 오류를 정정하는 DRAM ECC 기법에 대해 많은 연구가 진행되고 있다[4], [5].

2014년 공개된 로우 해머(rowhammer)는 이러한 상호 간섭의 영향을 의도적으로 증가시켜 데이터 손실을 유발하는 S/W 기반 공격으로[3], 로우 해머를 통한 공격 방법과 이를 방지하는 기법들에 대한 연구가 지속적으로 이루어지고 있다[6]-[12]. 로우 해머의 첫 발표 이후 초기 사람들은 ECC가 장착된 DRAM은 로우 해머의 공격 대상이 될 수 없다는 인식을 갖고 있었는데, 로우 해머로 인한 비트 오류를 ECC를 통해 정정할 수 있기 때문이다.

본 논문은 DRAM 로우 해머와 그 원리를 소개하고 ECC를 통한 로우 해머 방어 관련 연구의 진행과 한계, 그리고 향후 연구 방향에 대해 이야기한다.

Ⅱ. 본 론

A. DRAM 동작 방식

로우 해머를 이해하기 위해선 먼저 DRAM의 동작 구조에 대해 알아야 한다. 그림1과 같이 DRAM 칩은 다수의 셀(cell)이 2차원 격자 형태로 배열된 구조로, 각 셀은 1개의 트랜지스터와 커패시터 쌍으로 구성된 1T1C 구조를 갖는다. 각 셀은 커패시터의 충전 여부로 0 또는 1을 읽고 쓸 수 있다. 셀에 대한 접근은 행 단위로 이루어지는데, 접근하고자 하는 행의 wordline이 on되면 해당 행의 데이터가 로우 버퍼(row-buffer)에 전달되고, 그 중 bitline을 통해 접근하고자 하는 셀을 선택해 데이터를 읽거나 쓰게 된다.

DRAM이 동적(dynamic) RAM이라고 불리는 이유는 커패시터에 저장된 전하가 유지되지 않기 때문이다. 이 전하는 여러 이유로 점차 감소하며, 따라서 시간이 지남에 따라 셀은

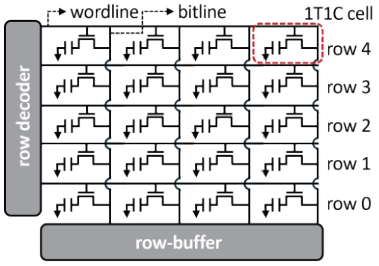


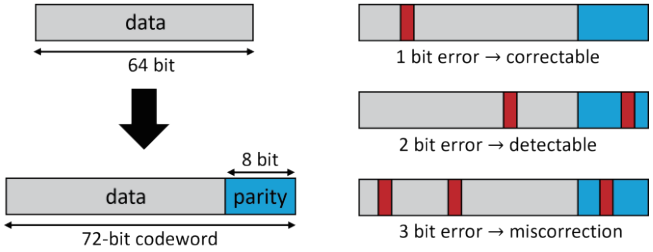
그림 1. DRAM 을 구성하는 1T1C cell 격자 배열 구조

저장된 데이터를 잃을 수 있다. 이를 방지하기 위해 데이터를 유지할 수 있는 유지시간(retention time) 이내에 적어도 한 번씩 모든 셀을 재충전해줘야 하며, 이를 리프레시(refresh)라고 부른다. 한 행이 로우 버퍼로 불려올 때 축전기에 저장된 전하가 손실되기 때문에 로우 버퍼에 불러온 후 해당 행을 재충전 해주는데, 리프레시는 이 점을 이용해 유지시간 내에 모든 row를 한 번씩 접근하는 방식으로 행해진다.

B. 로우 해머

로우 해머란 DRAM의 특정 행을 매우 빠르게 반복 접근함으로써 인접 행의 전하를 교란하고 이를 통해 비트 플립(bit flip)을 유도하는 하드웨어 취약점 및 공격을 말한다. DRAM의 특정 행을 반복적으로 접근하면 그 행의 wordline에서 on/off가 반복되고, 이로 인한 전계 변화가 누적되어 이웃 행에 전기적 간섭을 일으켜 커패시터의 전하 손실 속도의 상승을 유도한다. 그 결과 커패시터 전압이 리프레시 전에 임계치 이하로 감소하여 비트 값이 뒤집히게 된다[3]. 이후 자바 스크립트와 같은 원격 트리거 가능성이 제시되고[6] 로우 해머를 통해 결정론적 루트 권한 획득이 가능함이 발표되면서[7] 로우 해머는 보안 상의 주요 문제점으로 떠올랐다.

로우 해머에 대한 대응책으로는 크게 7가지 방식이 소개되었는데, 칩 자체를 더 좋게 만드는 것, 리프레시 주기를 강화하는 것, 테스트 기반 취약 셀 퇴역을 각각 제조 단계와 사용자 단계에서 수행하는 것, 자주 접근되는 행의 인접 행을 추가 리프레시하는 것, 행에 접근할 때 낮은 확률로 인접 행을 리프레시 하는 것, 그리고 ECC를 통해 오류를 정정하는 것이다.



(a) SECDED 부호어 구성 (b) 비트 오류에 따른 ECC 결과

그림 2. SECDED ECC 동작 구조

C. ECC 기반 로우 해머 대응

DRAM에서는 로우 해머 외에도 셀의 노화, 전송 타이밍 문제 등의 원인으로 오류가 발생할 수 있다. 이러한 오류를 정정하기 위해 DRAM ECC가 도입되었으며, ECC는 패리티라 불리는 추가 비트를 필요로 하는 대신 정정 능력 내의 오류를 탐지 및 정정할 수 있다. 예를 들어 DDR4 ECC DIMM의 경우 SECDED (Single Error Correction-Double Error Detection) 방식이 주로 사용되었는데, 이는 그림 2-(a)와 같이 원래 64비트 크기인 데이터에 8비트 패리티를 추가로 사용해 72비트로 늘려 저장 및 전송하는 대신 이 72비트 부호어(codeword) 내에서 발생하는 단일 비트 오류를 정정하거나 이중 비트 오류를 탐지할 수 있다. 다만 이를 벗어나 3개 이상의 비트에서 오류가 발생하는 경우에는 오정정(miscorrection)이 발생하여 오류가 포함된 데이터를 전송하게 된다.

로우 해머의 공개 이후 초기에는 ECC 메모리는 로우 해머에 대해 완전하진 않지만 비교적 안전하다는 인식이 일반적이었는데, 이는 로우 해머로 인한 무작위 오류 대부분을 ECC를 통해 정정할 수 있으며, ECC 과정이 숨겨지도록 설계되었기 때문에 의도적으로 정정 능력을 벗어나는 오류를 발생시키는 것이 어려웠기 때문이다.

그러나 2019년 ECCploit은 ECC 메모리라 할지라도 로우 해머 공격에서 안전하지 않음을 제시하였다[8]. 로우 해머를 통해 비트 오류를 누적해 단일 부호어 내에서 3 비트 이상 오류를 발생시켜 오정정을 유도해 공격하는 방법을 사용하는데, ECC로 인해 가려졌던 비트 오류 정보를 얻기 위해 오류 정정이 진행됨으로써 발생하는 미세한 시간 지연을 포착해 숨겨진 비트 오류를 관찰하였다. SECDED 보다 강력한 ECC를 적용한다고 하더라도 그 정정 범위가 있는 한 이를 초과하는 오류를 유도할 수 있으며, 따라서 ECC만으로는 로우 해머에 안전할 수 없음을 실증하였다.

이 한계를 보완하기 위해 2023년 CoF(Copy-on-Flip) 기법이 소개되었다[11]. CoF는 ECC가 오류를 고쳤다는 신호가 잡히면, 운영체제가 해당 메모리 영역을 바로 안전한 곳으로 옮기고 기존 영역은 쓰지 않도록 격리하는 이주·격리 방식을 통해 로우 해머에 대응한다. 결과적으로 공격자는 같은 자리에서, 즉 동일 부호어 내에 두 번째, 세 번째 비트 오류를 누적할 수 없기 때문에 ECC 정정 능력을 벗어나는 오류 발생을 막을 수 있다. 하지만 남아있는 문제도 있는데, 먼저 로우 해머 여부와 관계없이 정정이 발생할 때마다 메모리 영역을 이주해야 하기 때문에 성능 저하를 유발할 수 있다. 또한 구조상 고정된 위치에 있는 데이터가 있을 수 있는데, 이는 여전히 로우 해머 공격의 대상이 될 수 있다.

Ⅲ. 결 론

로우 해머는 현재까지 완전히 해결되지 않은 DRAM 하드웨어 취약점으로 보안 상의 중요한 문제점이다. ECC를 통해 로우 해머의 영향을 줄일 수 있으나 정정 능력을 넘어서는 오류는 정정할 수 없는 한계가 존재한다. 따라서 소프트웨어 시스템과 연계한 다층적 방어를 설계하면서도 그 오버헤드를 줄이기 위한 연구가 필요하다.

ACKNOWLEDGMENT

이 논문은 2025년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원(RS-2024-00343913)과 정보통신기획평가원의 지원(RS-2024-00398449)을 받아 수행된 연구임

참 고 문 헌

[1] R. H. Dennard, “Field-effect transistor memory,” July 1967, US Patent 3,387,286.

[2] O. Mutlu, “Memory scaling: A systems architecture perspective,” in Proc. 5th IEEE Int. Memory Workshop (IMW), Monterey, CA, USA, 2013, pp. 21–25.

[3] V. Sridharan and D. Liberty, “A study of DRAM failures in the field,” in Proc. Int’l Conf. High Performance Computing, Networking, Storage and Analysis (SC), Salt Lake City, UT, USA, 2012, Art. no. 76, pp. 1–11.

[4] Y. Kim et al., “Flipping bits in memory without accessing them: An experimental study of DRAM disturbance errors,” in Proc. ACM/IEEE 41st Annu. Int. Symp. Comput. Archit. (ISCA), Minneapolis, MN, USA, 2014, pp. 361–372.

[5] H. Ju, D.-H. Kong, K. Lee, M.-K. Lee, S. Cho, and S.-H. Kim, “How to use redundancy for memory reliability: Replace or code?,” Electronics, vol. 14, no. 9, Art. no. 1812, 2025.

[6] D. Gruss, C. Maurice, and S. Mangard, “Rowhammer.js: A remote software-induced fault attack in JavaScript,” in Proc. Int’l Conf. Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA), San Sebastián, Spain, 2016, pp. 300–321.

[7] V. van der Veen et al., “Drammer: Deterministic Rowhammer attacks on mobile platforms,” in Proc. ACM SIGSAC Conf. Computer and Communications Security (CCS), Vienna, Austria, 2016, pp. 1675–1689.

[8] L. Cojocar, K. Razavi, C. Giuffrida, and H. Bos, “Exploiting correcting codes: On the effectiveness of ECC memory against Rowhammer attacks,” in Proc. IEEE Symp. Security and Privacy (S&P), San Francisco, CA, USA, 2019, pp. 55–71.

[9] P. Frigo et al., “TRRespass: Exploiting the many sides of Target Row Refresh,” in Proc. IEEE Symp. Security and Privacy (S&P), 2020, pp. 747–762.

[10] P. Jattke, V. van der Veen, P. Frigo, S. Gunter, and K. Razavi, “Blacksmith: Scalable rowhammering in the frequency domain,” in Proc. IEEE Symp. Security and Privacy (S&P), San Francisco, CA, USA, 2022, pp. 716–734.

[11] A. Di Dio, K. Koning, H. Bos, and C. Giuffrida, “Copy-on-Flip: Hardening ECC memory against Rowhammer attacks,” in Proc. Network and Distributed System Security (NDSS) Symp., San Diego, CA, USA, 2023.

[12] O. Mutlu, A. Olgun, and A. G. Yağlıkçı, “Fundamentally understanding and solving RowHammer,” in Proc. Asia and South Pacific Design Automation Conf. (ASPDAC), Tokyo, Japan, 2023.