마이크로아키텍처 기반 부채널 공격 연구 조사

정세현 [1], 오현영 [2*]
[1] 가천대학교 AI · 소프트웨어학부 학부생
[2] 가천대학교 AI · 소프트웨어학부 교수
jjsshh3116@ gachon.ac.kr, hyoh@ gachon.ac.kr

A Survey of Microarchitectural Side-channel Attacks

Se-Hyeon Jeong, Hyunyoung Oh
Dept. of AI·Software, Gachon University

요 약

본 논문은 최신 마이크로아키텍처 부채널 공격을 분석한다. 이 공격들은 각각 투기적 실행, 데이터 메모리 종속 프리페처(DMP), 페이지 워크와 같은 CPU 의 핵심 성능 최적화 기능을 악용하여, 포인터 인증이나 상수 시간 구현과 같은 방어 기술을 우회한다. 공격 원리를 분석하고 그 위협성과 발전 방향을 조망하고자 한다.

1. 서론

부채널 공격(Side-Channel Attack)은 암호 알고리즘의 논리적 결함을 직접 공격하는 대신, 하드웨어의 물리적 또는 비기능적 특성을 관찰하여 암호 키와 같은 비밀 정보를 유출하는 심각한 보안 위협이다. 다양한 부채널 공격 중에서 마이크로아키텍처 부채널 공격은 현대 CPU 의 성능 최적화를 위해 필수적으로 도입된 캐시 메모리, 투기적 실행(Speculative Execution), 프리페처(Prefetcher) 등의 기능을 악용하므로 특히 더 위협적이다[1]. 이러한 공격들은 단순히 비밀 정보를 유출하는 것을 넘어, 포인터 인증(Pointer Authentication, PA)과 같은 방어 기술까지 우회할 수 있음을 증명하기 때문에 그 위협성을 가중한다. 이에 본 논문에서는 마이크로아키텍처 기반 부채널 공격의 최신 연구 동향을 공격 대상, 악용 메커니즘, 전제 조건, 유출 정보 유형의 네 가지 기준으로 분류하고 비교 분석하여 그 위험성과 발전 방향을 조망하고자 한다.

2. 배경지식

2.1 부채널 공격(Side-Channel Attack)

부채널 공격은 연산 과정에서 발생하는 시간 차이, 캐시 변화, 소비 전력, 전자파 방출 등의 물리적 특성을 관찰해 암호 키와 같은 비밀 정보를 유추하는 기법이다. 그중에서도 마이크로아키텍처 부채널 공격은 CPU 의 성능 최적화를 위해 도입된 내부 기능들을 악용한다[1].

2.2 투기적 실행(Speculative Execution)

CPU 가 분기문의 결과를 미리 예측하고, 예측된 경로의 명령어들을 앞당겨 실행하는 최적화 기술이다. 만약 예측이 틀렸다면, CPU 는 실행했던 모든 작업을 폐기하고 올바른 경로에서 실행을 재개하므로 프로그램의 최종 결과에는 영향을 미치지 않는다[2].

2.3 프리페처(Prefetcher)

CPU가 다음에 필요로 할 데이터를 미리 예측하여 캐시에 가져다 놓음으로써 메모리 접근 시간을 줄이는 하드웨어 유닛이다. 일반적인 프리페처는 메모리 접근 주소의 패턴을 분석하지만 데이터 메모리 종속 프리페처(DMP, Data Memory-dependent Prefetcher)는 '데이터 값' 자체를 주소로 해석하여 다음 데이터를 가져온다[3].

3. 연구사례

3.1 PACMAN

PACMAN 연구는 메모리 변조 취약점과 투기적 실행 취약점을 결합하여 ARM 의 메모리 방어 기술인 포인터 인증을 우회하는 공격 기법을 제시한다[2]. 이 공격은 CPU 가 분기 예측 실패 후 잘못 예측한 경로의 명령어들을 잠시 실행했다가 폐기하는 투기적 실행 특성을 악용한다.

공격의 주된 대상은 ARM 포인터 인증 기능이며, 최종 목표는 PA 의 방어 원리인 '검증 실패 시 프로그램 종료'을 회피하여 올바른 PAC(Pointer Authentication Code) 값을 알아내는 것이다. 공격자는 PA 검증 명령어가 포함된 'PACMAN 가젯'을 투기적 실행 경로상에서 실행시킨다. PAC 추측이 틀리면 원래는 프로그램이 종료되어야 하지만 투기적 실행 중 발생한 예외는 실행 결과가 폐기되면서 함께 사라진다. 공격자는 이때 PAC 검증 성공 여부에 따라 달라지는 TLB 의 상태 변화를 부채널로 관찰하여 충돌 없이 올바른 PAC 값을 유추해 낸다. 올바른 PAC 을 공격자가 알게 되면 공격자는 악의적인 코드가 위치한 메모리 주소로 프로그램 실행 흐

* 교신저자

름을 제어할 수 있게 된다.

저자들은 Apple M1 SoC 환경에서 공격을 시연했으며, PAC 값을 90%의 성공률로 찾아내고 제어 흐름 탈취에 성공함을 보였다.

3.2 GoFetch

GoFetch 는 상수 시간(Constant-Time)으로 구현된 암호 알고리즘을 데이터 메모리 종속 프리페처(DMP)를 이용해 무력화한 연구이다. 이 공격은 메모리의 내용을 직접 읽고 다음 접근할 주소를 예측하는 DMP 의 하드웨어 특성을 악용한다[4].

이 공격의 대상은 시간이나 메모리 접근 패턴이 비밀 값에 의존하지 않도록 설계된 상수 시간 암호 알고리즘이며, 최종 목표는 각 암호 알고리즘의 비밀 키를 추출하는 것이다. 공격자는 암호 연산 과정에서 비밀 키의 영향을 받는 중간 결과값이 공격자가 제어하는 특정 주소 값이 되도록 입력 값을 조작한다. CPU 코어와 무관하게 동작하는 DMP 는 중간 값을 실제 포인터로 착각하여 역참조(dereference)한다. 그 결과로 특정 데이터가 캐시에 로드된다. 공격자는 Prime+Probe 방식을 이용해 캐시의 흔적을 관찰하여 비밀 키를 유추한다.

저자들은 Apple M 시리즈(M1, M2, M3) 환경에서 공격을 시연했으며, RSA, Diffie-Hellman, Kyber, Dilithium 의 상수 시간 구현에서 성공적으로 키를 추출해냈다.

3.3 Peek-a-Walk

Peek-a-Walk 는 CPU 의 주소 변환 과정인 페이지 워크(Page Walk)에서 발생하는 부채널(PWSC)을 규명한 연구이다[5]. 여기서 페이지 워크란, CPU 의 MMU(메모리 관리 장치)가 프로그램이 사용하는 가상 주소를 실제 물리 메모리 주소로 변환하기 위해 여러 단계의 페이지 테이블을 탐색하는 과정을 말한다. 이 탐색 과정에서 MMU 가 각 단계의 페이지 테이블 항목(PTE)을 읽을 때마다 L1 캐시에 흔적을 남긴다. 이 연구는 GoFetch 와 같이 비밀 값이 '포인터처럼 보이는' 유효한 주소여야 한다는 제약을 넘어, 완전히 임의의 값이라도 비밀 값을 유출할 수 있음을 증명하여 한 단계 더 발전한 공격 기법을 제시한다.

공격의 목표는 투기적 실행이나 DMP 에 의해 역참조되는 임의의 64 비트 비밀 값에서 42 비트를 복원하는 것이다. 공격 과정은 두 가지 과정으로 구성된다. 첫째, 공격을 수행했을 때와 안 했을 때의 캐시 상태를 비교하고 그 차이만을 추출하여, 정상 페이지 워크 잡음 속에서 비밀 값에 의한 순수한 페이지 워크 흔적만을 분리해낸다. 둘째, 메모리 매핑을 제어하여 페이지 워크를 의도적으로 한 단계씩만 진행시킨 뒤, 새로 나타나는 흔적을 관찰하여 유출된 비밀 비트 조각들의 순서를 정확히 재구성한다.

Intel Core i9-13900k CPU 환경에서 수행된 실험에서 스펙터 공격(Specrue Attack)과 결합하여 물리 메모리의 98% 이상을 유출하고 Dilithium 암호키를 추출했다.

4. 공격 기법 비교 분석

3 장에서 살펴본 세 가지 공격을 비교하고 그 위협성을 다각적으로 평가한다. 각 공격의 특성은 아래 표와 같이 요약할 수 있다.

<표 1> 마이크로아키텍처 부채널 공격 비교 분석

구분	PACMAN	GoFetch	Peek-a-Walk
공격 대상	ARM 포인터 인증	상수 시간 암호 알고리즘	투기적 실행 또는 DMP 에 의해 접근되는 임의의 메모리 값
악용 메커니즘	투기적 실행	DMP	페이지 워크
공격 전제 조건	메모리 변조 취약점	비밀 값에 영향을 받는 중간 값을 포인터로 사용	투기적 실행 또는 DMP
유출 정보	포인터 인증 코드	암호 알고리즘 비밀 키	임의의 64 비트 값

위 표의 분석을 통해 최신 공격들이 특정 방어 기술(PACMAN)이나 암호 구현(GoFetch)을 넘어, 메모리상의 임의의 데이터를 유출하는 방향(Peek-a-Walk)으로 발전하고 있음을 알 수 있다. 즉, 공격의 대상이 점점 더 보편화되고 있으며, 유출 가능한 정보의 범위 또한 특정 코드나 키 값에서 일반적인 메모리 데이터로 확장되는 추세이다. 이러한 발전 방향은 마이크로아키텍처 수준의 취약점이 시스템 전반에 미치는 영향이 더욱 심각해질 수 있음을 시사한다.

5. 결론

본 논문에서는 세 가지 최신 마이크로아키텍처 부채널 공격을 공격 대상, 악용 메커니즘 등의 비교 기준을 통해 심층적으로 분석하였다. 각 공격은 투기적 실행, DMP, 페이지 워크 등 CPU 성능 최적화를 위해 도입된 기능이 어떻게 기존의 방어 기법을 무력화하는 새로운 공격 경로로 악용될 수 있는지 보여주었다.

이 연구들은 CPU 의 성능 최적화가 예기치 않은 보안 위협으로 직결될 수 있음을 명확히 보여준다. 따라서 하드웨어 설계 단계에서부터 마이크로아키텍처 수준의 보안을 고려하는 접근이 필수적이며, 소프트웨어 방어 기술 또한 하드웨어의 동작 방식에 대한 깊은 이해를 바탕으로 설계되어야 할 것이다.

사사문구

이 논문은 2025 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원(No. RS-2024-00337414, SW 공급망 운영환경에서 역공학 한계를 넘어서는 자동화된 마이크로 보안 패치 기술 개발)과 한국산업기술기획평가원의 지원(No. RS-2024-00406121, 자동차보안취약점기반위협분석시스템개발(R&D))을 받아 수행된 연구 결과임.

참고문헌

[1] Lou, Xiaoxuan, et al. "A survey of microarchitectural side-channel vulnerabilities, attacks, and defenses in cryptography." ACM Computing Surveys (CSUR) 54.6 (2021): 1-37.

[2] Ravichandran, Joseph, et al. "PACMAN: attacking ARM pointer authentication with speculative execution." Proceedings of the 49th Annual International Symposium on Computer Architecture. 2022.

[3] Vicarte, Jose Rodrigo Sanchez, et al. "Augury: Using data memory-dependent prefetchers to leak data at rest." 2022 IEEE Symposium on Security and Privacy (SP). IEEE, 2022.

[4] Chen, Boru, et al. "{GoFetch}: Breaking {Constant-Time} Cryptographic Implementations Using Data {Memory-Dependent} Prefetchers." 33rd USENIX Security Symposium (USENIX Security 24). 2024.

[5] Wang, Alan, et al. "Peek-a-Walk: Leaking Secrets via Page Walk Side Channels." 2025 IEEE Symposium on Security and Privacy (SP). IEEE, 2025.