ARM CCA 환경에서의 안전하고 유연한 VM-가속기 설계 스펙트럼에 관한 조사

김은민[1], 백윤흥[2]
[1]서울대학교 전기정보공학부 석박통합과정
[2]서울대학교 전기정보공학부 교수

anniekim@sor.snu.ac.kr, ypaeck@snu.ac.kr

A Survey of the Secure and Flexible VM–Accelerator Design Spectrum under Arm CCA

Eunmin Kim[1], Yunheung Paek[1]
[1]Dept. of Electrical and Computer Engineering (ECE) and Inter-University Semiconductor Research Center (ISRC), Seoul National University

요 약

본 논문은 VM 기반 TEE 환경에서 가속기(GPU, FPGA, TPU 등)를 안전하게 활용하기 위한 설계들을 ARM CCA 문맥에서 체계적으로 비교·분석한다. 기존 TEE가 CPU 중심 보호에 머무르며 가속기 경로를 동등 수준으로 보호하지 못한다는 한계를 출발점으로, VM-가속기 할당 모델을 핵심 축으로 삼아 최근 제안된 방법론을 검토한다. 나아가 VM lifecycle 관점에서 가속기가 VM 생성부터 소멸까지 정적으로 할당이 유지되느냐, 실행 중 동적 할당이 가능하느냐를 토대로 그에 따른 보안 함의를 정리한다. 이 분석을 기반으로 ARM CCA 환경에서 안전한 가속기 실행 환경을 구축하는데 있어서 가속기 활용 유연성을 어떻게 더 안전하게 적용할지에 대한 연구 방향성을 제시한다.

1. 서론

최근 GPU, FPGA, TPU 등 가속기의 보편화로 대규모 인공지능 모델의 추론 및 학습 효율이 크게 향상되었고, 이에 힘입어 기업이나 산업 현장뿐 아니라 일반 사용자도 일상적으로 AI 서비스를 활용하고 있다[1]. 특히 PC에 국한되지 않고 모바일 단말에도 ChatGPT, Gemini 와 같은 기능이 내장되면서, 개인 사용자 접근성이 비약적으로 높아졌다[2]. 이와 같은 보급은 모델 추론이 데이터 센터 중심에서 엣지, 모바일 중심으로 확장되고 있음을 의미하며, 가속기 사용이 특정 고성능 서버에 한정되지 않는다는 점에서 보안 경계의 재정의를 요구한다.

사용자 기반의 급속한 확장은 곧 개인 정보가 포함될 수 있는 데이터가 모델로 광범위하게 유입됨을 의미하며, 이에 따른 프라이버시 침해 위험이 대두되었다[3]. 이 문제를 완화하기 위해 전통적으로 ARM TrustZone, Intel SGX 와 같은 신뢰 실행 환경(Trusted Execution Environment)이 CPU 상의 안전한 실행 컨텍스트를 제공해 왔으나, 기존 신뢰 실행 환경(TEE)는 가속기 및 그 I/O 경로를 동일 수준으로 보호하지 못한다는 근본적 한계를 가진다. 예컨대, 모델의 입력과 출력, 중간 값이 Direct Memory Access (DMA) 경로를 통해 장치로 이동하고, 장치가 MMIO를 통해 제어되는 오늘날의 구조에서 CPU 측 격리만으로는 데이터 경로 전체의 기밀성과 무결성을 확보하기 어렵다.

이에 따라 CPU 쪽 보호만으로는 충분하지 않으며, 가속기까지 포함하는 confidential computing 보안 경계가 요구된다. 이를 위한 한 축으로, 가상머신 (VM) 기반 TEE (e.g., Intel TDX, ARM CCA, AMD SEV-SNP)가 확산되면서, VM에 가속기를 안전하게 할당 (static, dynamic)하는 시스템이 주목받고 있다[4-6].

특히 ARM은 모바일 시장에서 99%라는 매우 높은 CPU 공급 점유율을 보유하고 있어[7], ARM이 제시한 VM 기반 TEE인 Confidential Compute Architecture (CCA)의 파급력은 크다[8]. CCA는 기존 TrustZone의 world를 확장한 개념으로 주소 격리를 담당하는 테이블을

추가함으로써 주소 공간 단위의 격리를 강제한다. 또한 CPU 측 Stage-2 translation 과 System MMU(SMMU; IOMMU) 기반의 I/O 격리를 결합하여, VM-장치 경계에서의 자원 소유권과 접근 제어를 정교하게 다룰 수 있는 토대를 제공한다. 이러한 배경 위에서, 가속기 보안은 더 이상 CPU 내부의 문제가 아니라 플랫폼 end-to-end 차원의 문제로 자리 잡고 있다.

본 논문은 위 문제의식을 바탕으로, ARM CCA 문맥에서 CPU 중심의 TEE 를 가속기까지 확장하려는 최근 접근을 조망하고, ACAI[4], CAGE[5], PORTAL[6] 세 연구를 사례로 삼아 VM 으로의 가속기 할당 관점에서 비교 분석한다. 각 시스템의 핵심 메커니즘을 중심으로 비교하고, 가속기 할당에서의 차이점을 분석하여 CCA 기반 시스템에서 가속기를 안전하고 효율적으로 운용하기 위한 암호화와 dynamic binding 의 하이브리드 형식의 연구 방향을 제시한다.

2. 배경지식

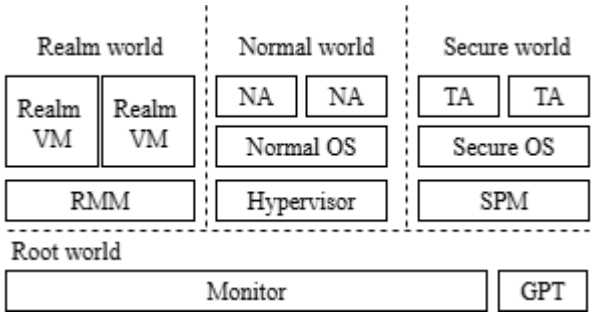
○ ARM TrustZone

ARM 은 안전한 컴퓨팅 환경을 제공하기 위해 하드웨어 기반 기술인 TrustZone 을 제시했다. TrustZone 는 기존 CPU state 를 secure mode 와 normal mode 로 나누어 실행 환경을 격리하는 기술을 제공한다. 각 mode 에 따라 world 가 존재하고 운영체제 또한 각각 존재한다. 각 normal OS 와 secure OS 위에서 normal app 과 trusted app 이 각각 실행된다. Secure world 에서 실행되는 앱에 대한 정보는 normal world 에서 볼 수 없고, secure world 에서는 자기 world 뿐만 아니라 normal world 도 볼 수 있다.

○ ARM CCA

CCA 는 이러한 TrustZone 을 확장한 구조로 Armv9 [9]에서 confidential computing 을 지원하기 위해 처음 등장한 기술이다. 그림 1 에서도 확인할 수 있듯이, 기존의 TrustZone 에 realm world 와 root world 라는 2 개의 world 를 추가한 구조이다.

각 world 가 가지고 있는 주소 공간을 Physical Address Spaces (PAS) 라고 칭하며, CCA 는 총 4 개의 PAS 를 가지고 있게 된다. 이 때 realm world 는 다수의 confidential VM 을 실행 할 수 있고, 이를 realm VM 이라고 부른다. 이러한 realm 들은 normal world 의 hypervisor 역할에 해당되는 Realm Management Monitor (RMM)에 의해 관리된다.



(그림 1) ARM CCA 구조.

RMM 은 root world 에 있는 Granule Protection Table (GPT) 라는 전역 테이블을 참고하여 해당 주소가 어느 PAS 에 속한 주소인지 확인하는 과정인 Granule Protection Check (GPC)를 거쳐서 격리를 달성한다. Realm world 는 normal world 는 볼 수 있지만 secure world 를 볼 수는 없다.

Root world 는 가장 높은 권한의 world 로 나머지 모든 world 들을 관장하며, CCA 에서 보장하는 보안 정책을 담당하는 개체인 monitor 가 존재한다. RMM 의 관리도 물론 monitor 가 담당하며, GPT 도 root world 에서만 접근이 가능하고 업데이트도 오로지 monitor 를 통해서만 가능하다.

○ ARM stage-2 translation

Stage-2 translation[10]은 hypervisor 가 VM 의 메모리를 제어할 수 있게 해주는 기능이다. 기존의 virtual address (VA)에서 physical address (PA)로 전환되었던 메모리 translation 과 비슷하게, VA 에서 PA 로 가는 과정에서 intermediate PA (IPA)가 추가된 형태이다. Stage-1 에서는 VA 를 IPA 로 변환한다. 이 과정에서 OS 는 IPA 를 실제 PA 라고 착각한다. 이 후, stage-2 에서는 IPA 를 진정한 PA 로 변환시키는 과정이 발생한다. 이러한 과정을 통해서 realm world 의 각 realm 을 격리할 수 있을 뿐만 아니라 시스템에 존재하는 모든 장치들을 관리하고 있는 SMMU[11]도 stage-2 translation 을 사용하여 장치들간의 주소 공간을 격리할 수 있게 된다.

3. VM 가속기 할당

본 연구는 VM 과 가속기 간 매핑[4-6]을 VM 의 전체 lifecycle 동안 매핑되어 있어야 하는 static binding 과 동적으로 장치 매핑을 바꿀 수 있는 dynamic binding 으로 구분한다.

Static binding 은 VM 생성 시점에 매핑하고 싶은 가속기를 함께 attestation 하여 매핑하고, VM 이 소멸될 때까지 소유권을 유지한다. 생성할 때 attestation 을 같이 하여 해당 가속기 고유 키를 VM 이 접근하는

형식으로 관리 경로가 단순하고 키 동기화가 용이하나, 자원 활용률과 스케줄링 유연성이 제한된다.

Dynamic binding 은 이미 실행 중인 VM 에 대해서도 가속기를 attach, detach 하여 가속기를 동적으로 할당할 수 있다. 이를 통해 장치 활용률이 유연해지지만, 해당 VM 에서 할당된 가속기에 대한 정보를 계속 유지하고 있어야 한다는 점이 존재한다.

4. 비교 분석

Work	Binding	Approach	Encryption
ACAI[4]	static	GPC + PCIe	o
CAGE[5]	static	multi GPTs	o
PORTAL[6]	dynamic	multi GPTs	x

<표 1> VM-가속기 매핑 기존 연구 비교.

표 1 은 ARM CCA 환경에서 안전한 가속기 사용을 위해 VM 과 가속기를 매핑시킨 기존 연구들을 binding lifecycle 과 격리와 메모리 보호 관점에서 요약했다. 요지는 다음과 같다.

○ ACAI[4] Peripheral Component Interconnect Express(PCIe)를 통해 가속기가 연결된 환경에서 VM 과 가속기 간의 attestation 을 통해서 생성과 동시에 binding 을 한다. 이 때, 가속기의 attestation report 를 검증한 뒤 VM 과의 매핑을 생성하기 때문에 binding 된 VM 이 소멸될 때까지 해당 가속기에 대한 소유권을 일관되게 유지한다. 또한 CPU 와 공유하는 메모리는 PCIe 에서 제공하는 암호화 기능을 사용하여 암호화하여 보호한다. 개별 장치에 대한 메모리 격리는 똑같이 GPC 를 지원하는 SMMU 를 통하였고, SMMU 를 보호하기 위해 root world 를 통해 관리되도록 하였다.

○ CAGE[5]는 ACAI 와 비슷하게 static binding 을 통한 안전한 가속기 실행 환경을 제안했다. 그 과정 속에서 context switching 으로 인해 발생하는 속도 지연을 해결하기 위해 사용자의 민감 데이터와 상관없는 작업과 관련있는 작업을 구분지어 성능을 향상시켰다. 다만, 이 연구는 가속기 전체를 다룬 것이 아니라 GPU 에 국한된 기술을 제안했다. 또한 ACAI 와는 다르게 SMMU 의 stage-2 translation 을 사용하지 않고, 동일한 주소공간이지만 각 권한을 다르게 명시해놓은 다수의 GPT 를 사용함으로써 각 GPU 가 사용할 공간을 격리했다.

○ PORTAL[6]은 앞선 두 연구의 static binding 의 한계를 보완한 연구로 VM 생성 후 실행 단계에서도 매핑된 가속기에 대한 상태 정보를 계속 유지하고 있음을 통해서 가속기 변경 요청을 통해서 해당 정보를 활용해서 다른 VM 으로 할당할 수 있게 해주는 방식이다. 이 과정에서 CAGE[5]와 비슷하게 GPT 를 여러 개 사용함으로써 격리를 달성했다. 또 다른 차이점으로는 GPT 를 나누어놓은 객체가 다르고, GPC 를 담당하는 SMMU 를 보호하기 위해 별도의 realm 을 만들어 보호했다는 점이다.

또한 메모리 암호화 관점에서 다른 두 연구[4-5]와는 다르게 PORTAL[6]은 ARM 환경에서 CPU 와 GPU 가 unified memory 라는 형태로 메모리를 구성하고 있기 때문에 bus probing 공격의 가능성이 매우 낮기 때문에 암호화를 할 필요가 없다고 주장한다. 매번 메모리에 접근을 할 때 복호화 과정을 할 필요도 없어졌기 때문에 성능 면에서 이점을 보인다.

5. 논의

앞선 4 장과 표 1 를 통한 비교를 종합하면, 세 연구[4-6]은 동일한 CCA 기반 격리를 지향하지만 연결 모델(PCIe 외장 혹은 SoC 통합), 암호화 여부, 가속기 binding 종류에서 서로 다른 디자인을 택하고 있다.

○ 연결 모델의 차이는 어느 환경에서 사용할지에 대한 대답도 되겠지만, 결국 메모리 암호화의 유무에 영향을 끼친다. PORTAL 에서 주장하는 암호화가 필요없다는 것은 unified memory 구조를 고려했을 때의 주장이므로 모든 환경에서 암호화를 사용하지 않아도 된다는 것은 아니게 된다.

○ 장치 유연성 관리 면에서는, PORTAL[6]과 같이 dynamic binding 을 채택할 때 multi-tenancy 환경에서의 자원 집적률과 할당 탄력성을 크게 개선할 수 있다. 첫째, 가속기 활용에 따라 가속기를 on-demand 로 회수하거나 재배치할 수 있어 과소나 과대 배정을 줄이고, 동일한 가속기 pool 로 더 많은 VM 을 수용할 수 있는 효과를 얻는다.

다만 위에서 논의한대로 ARM unified memory 에서 암호화를 적용하지 않더라도 기대하는 보안 효과를 달성할 수 있을 지에 대한 추후 논의가 더 필요하다.

따라서 본 연구가 제시하는 방향은 dynamic binding 을 유지하되 선택적 암호화를 결합하는 하이브리드다.

구체적으로는 (1) portal region 과 같은 가속기와 VM 이 공유하는 일부 메모리 구역에 대한 부분 암호화 (page 또는 버퍼 단위), (2) 가속기 할당을 동적으로 변경하는 구간에 한정한 단기 암호화 (transition-only encryption)로 경계 시점 노출을 축소하는 식이다. 이때 세션 키 (VM-가속기)와 버퍼 키를 분리해 운영하고, 이와 관련된 부분에 대해서는 monitor 에서 관리하도록 하거나 system realm 을 따로 만든 것처럼 key realm 을 생성하여 관리하는 방식도 채택할 수 있다.

6. 결론

본 논문은 ARM CCA 문맥에서 VM-가속기 매핑을 binding lifecycle (static vs. dynamic)과 격리, 보호 (암호화 여부, GPC 와 SMMU 활용)의 축으로 정리하고 기존 연구들의 설계 차이가 보안에 미치는 함의를 비교 분석했다. 요약하면, static binding 은 초기 VM 생성과 동시에 발생하는 가속기 attestation 을 통해 VM 이 소멸할 때까지 할당된 가속기만 사용해야 하고, dynamic binding 은 multi-tenancy 에서 자원 유연성을 극대화하되 transition 구간의 관리 비용과 경계 노출을 최소화할 장치가 필요하다.

특히 PORTAL[6]이 전제하는 비암호화 기반의 메모리 격리는 ARM 의 unified memory 환경에서는 실용적이지만, PCIe 를 통해서 장치가 연결된 환경에서는 곧바로 똑같이 적용할 수 없는 문제가 있다. 또한 bus probing 이 불가능하다는 전제 뿐만이 아니라 ARM unified memory 환경에서 평문으로 인한 다른 보안 문제가 나타날 수 있는지에 대한 것과 ARM 환경이 아닌 다른 환경에서도 이러한 설계를 채택할 수 있는지에 대한 연구가 필요하다.

사사문구

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2023-00277326), supported by the BK21 FOUR program of the Education and Research Program for Future ICT Pioneers, Seoul National University in 2025, supported by the Institute of Information & Communications Technology Planning & Evaluation(IITP) grant funded by the Korea government(MSIT) (No. 2021-0-00528, Development of Hardware-centric Trusted Computing Base and Standard Protocol for Distributed Secure Data Box), supported by the Culture, Sports and Tourism R&D Program through the Korea Creative Content Agency grant funded by the Ministry of Culture, Sports and Tourism in 2025 (Project Name: Training Global Talent for Copyright Protection and Management of On-Device AI Models, Project Number: RS-2025-02221620, Contribution Rate: 100%, supported by Inter-University Semiconductor Research Center (ISRC), and supported by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (No.RS-2024-00438729, Development of Full Lifecycle Privacy-Preserving Techniques using Anonymized Confidential Computing).

참고문헌

[1] Maslej, Nestor, et al., "Artificial intelligence index report 2025.", *arXiv preprint arXiv:2504.07139* (2025).

[2] Samsung Electronics., "Samsung Galaxy S25 Series Sets the Standard of AI Phone as a True AI Companion." Sam sung Newsroom, 22 Jan. 2025, https://news.samsung.com/global/samsung-galaxy-s25-series-sets-the-standard-of-ai-phone-as-a-true-ai-companion. Accessed 29 Sept. 2025.

[3] Rigaki, Maria, and Sebastian Garcia, "A survey of privacy attacks in machine learning." ACM Computing Surveys 56, 4, pp.1-34, 2023.

[4] Sridhara, Supraja, et al. "{ACAI}: Protecting Accelerator Execution with Arm Confidential Computing Architecture." 33rd USENIX Security Symposium (USENIX Security 24), Philadelphia, USA, 2024, pp.3423-3440.

[5] Wang, Chenxu, et al. "Cage: Complementing arm cca with gpu extensions." Network and Distributed System Security (NDSS) Symposium, San Diego, California, 2024.

[6] Sang, Fan, et al. "PORTAL: Fast and Secure Device Access with Arm CCA for Modern Arm Mobile System-on-Chips (SoCs).", IEEE Symposium on Security and Privacy (SP), San Francisco, California, 2025, pp.4099-4116.

[7] Forbes, “Arm Stock: AI Chip Favorite Is Overpriced”, 21 Mar. 2024, https://www.forbes.com/sites/bethkindig/2024/03/21/arm-stock-ai-chip-favorite-is-overpriced/ Accessed 29 Sept. 2025.

[8] ARM, “Introducing arm confidential compute architecture,”, 2024, https://developer.arm.com/documentation/den0125/0300/

[9] ARM, “Arm Architecture Reference Manual Supplement Armv9, for Armv9-A architecture profile,”, 2022, https://developer.arm.com/documentation/ddi0608/latest, Accessed 29 Sept. 2025.

[10] ARM, “Stage 2 Translation”, https://developer.arm.com/documentation/102142/0100/Stage-2-translation, Accessed 29 Sept. 2025.

[11] ARM, “ARM System Memory Management Unit Architecture Specification”, https://developer.arm.com/documentation/ihi0062/b, Accessed 29 Sept. 2025.