# Cortex-M 프로세서에서 MPU 를 사용한 보안 연구 조사

구윤주 [1], 강하영 [2], 권동현 [3]
[1] 부산대학교 정보컴퓨터공학부 학부생
[2] 부산대학교 정보융합공학과 석사과정
[3] 부산대학교 정보컴퓨터공학부 교수*

ventus07@pusan.ac.kr, rkdgkdud12345@pusan.ac.kr, kwondh@pusan.ac.kr

# Investigating Security Studies Using MPU on Cortex-M Processors

Yun-Ju Gu[1], Ha-Young Kang[2], Dong-Hyun Kwon[3]
[1]Dept. of Computer Science and Engineering, Pusan National University
[2]Dept. of Information Convergence Engineering, Pusan National University
[3]Dept. of Computer Science and Engineering, Pusan National University

**요 약**

Cortex-M 프로세서는 가상 메모리를 지원하는 MMU 를 제공하지 않는 대신 메모리 보호 옵션으로 MPU 를 제공한다. MPU 를 사용해 특정 메모리 영역에 대한 접근 권한과 속성을 정의할 수 있고, 이를 이용해 중요한 데이터를 보호하거나 특정 동작을 수행하기 위해 고의로 예외를 발생시킬 수 있다. 본 논문에서는 메모리 보호에 MPU 를 사용한 연구에 대해 조사하였다.

## 1. 서론

임베디드 시스템 시장이 성장함에 따라 가격과 효율성을 중시한 Cortex-M 프로세서의 사용이 증가하고 있다. X86 과 Cortex-A 와 달리 가상 메모리를 지원하는 Memory Management Unit(MMU)을 제공하지 않아 애플리케이션 태스크와 커널이 같은 물리적인 주소 공간을 공유하게 된다. Cortex-M 프로세서는 이에 대한 대책으로 기본적인 메모리 보호 메커니즘만을 제공하고 있다[1]. Cortex-M 프로세서에서는 메모리 보호 옵션으로 Memory Protection Unit(MPU)을 지원하는데, 이를 사용하면 특정 메모리 영역에 대해 읽기, 쓰기, 실행 등 접근 권한을 제한하고 캐시 사용 여부나 공유 가능 여부와 같은 메모리 속성을 정의할 수 있다. 본 논문은 Cortex-M 프로세서에서 MPU 를 사용하여 메모리 보호에 적용한 연구를 조사하였다.

## 2. Kage

마이크로컨트롤러 기반의 임베디드 시스템에서 어플리케이션 코드와 커널 코드의 제어 데이터를 모두 보호하는 시스템이다[2]. Kage 는 코드를 신뢰할 수 있는 코드와 신뢰할 수 없는 코드로, 메모리 영역을 권한 있는 영역과 권한 없는 영역으로 구분하고 보호하고자 하는 제어 데이터를 권한 있는 메모리 영역에 저장한다. 신뢰할 수 있는 코드는 제어 데이터에 직접적인 접근이 허용되며, 복귀 주소 저장과 같이 신뢰할 수 없는 코드가 접근을 시도하는 경우 예외를 발생시켜 처리하도록 하였다. 이때 MPU 를 사용해 권한이 없어도 접근가능한 메모리 영역과 보호를 위해 권한이 있는 경우에만 접근이 가능한 메모리 영역을 구분하였다.

## 3. SuM

Cortex-M 프로세서를 위한 섀도우 스택과 이의 무결성을 보장하기 위해 선택적 마스킹이라는 내부 주소 공간 격리를 구현하였다[3]. 섀도우 스택이란 함수 호출 시 복귀 주소를 별도의 메모리 공간에 저장해둔 뒤 호출한 함수로 돌아가기 전 현재 설정된 복귀 주소와 저장된 주소를 비교해 제어 흐름 무결성을 보장하는 보호 기법이다. SuM 은 MPU 를 사용해 허용되지 않은 코드가 섀도우 스택에 접근하는 것을 막는다.

* Corresponding Author

반면 허가된 안전한 코드가 섀도우 스택을 수정할 수 있도록 하기 위해 예외 우선순위를 설정할 수 있는 FaultMask 레지스터를 사용하여 일시적으로 MPU 를 해제한다. 한편, MPU 설정을 위한 레지스터 등과 같은 시스템 컨트롤 레지스터를 보호하기 위해 Data Watchpoint Trace(DWT)를 사용하여 시스템 메모리 영역의 모든 접근을 모니터링한다.

### 4. Randezvous

완전 탐색 공격을 사용하여 ARMv7/8-M 마이크로컨트롤러에 대한 제어 흐름 탈취 공격과 제어 데이터를 유출하는 공격을 완화하는 시스템이다[4]. 임의의 트랩 명령어를 가리키는 코드 포인터인 decoy pointer 라는 개념을 도입하여, 이를 이용해 사용되지 않는 데이터 메모리를 채움으로써 실제 코드 데이터를 위장하고 보호할 수 있다. 공격자가 버퍼 오버플로우 공격으로 데이터 유출을 시도할 때, 실제 데이터와 decoy pointer 를 구분할 수 없기 때문이다. 하지만 여전히 공격자는 메모리 영역에 제어 데이터를 분사하는 spray 공격을 사용하여 영역 내의 모든 제어 데이터 슬롯을 오염시킬 수 있다[4]. 이를 막기 위해 사용되지 않은 데이터 메모리를 랜덤하게 선택해 MPU 로 보호함으로써 임의의 메모리 영역에 쓰기 동작을 막는다. 이때 랜덤하게 선택된 영역의 개수는 MPU 가 지원하는 영역의 개수에 따라 결정된다.

### 5. fASLR

한정된 RAM 과 Flash 를 가지는 IoT 디바이스들을 위해 ARM TrustZone-M 과 MPU 를 사용하여 함수 기반 주소 공간 배열 무작위화를 구현하였다[5]. fASLR 은 함수가 호출될 때 Flash 에서 함수를 가져와 RAM 의 랜덤화 영역에 함수를 위치시킨다. 이렇게 함으로써 함수의 시작 주소가 무작위화되어 함수의 위치가 고정되지 않도록 한다. 이때 TrustZone-M 은 시스템 자원들을 보안 구역과 일반 구역으로 구분하는 데 사용되었으며, fASLR 의 메인 컴포넌트는 보안 구역에, 어플리케이션은 일반 구역에 위치시킨다. MPU 는 두 가지 목적으로 사용되었는데, 먼저 코드 재사용 공격으로부터 일반 구역의 Flash 를 보호하기 위함이고, 실행 불가로 설정된 일반 구역의 코드를 실행시키고자 하는 시도에 대해 예외를 발생시켜 fASLR 이 구현된 예외 처리 핸들러로 분기하기 위함이다.

### 6. 결론

| 연구 | MPU 활용 방법 |
|---|---|
| Kage | 제어 데이터 보호 |
| SuM | 섀도우 스택 보호 |
| Randezvous | 공격자의 메모리 쓰기 방지 |
| fASLR | 예외 처리 핸들러 호출 |

<표 1> 각 연구의 MPU 활용 방법

MPU 는 설정한 메모리 영역에 대한 접근 권한을 설정할 수 있어 제어 데이터와 같은 변경되어서는 안 되는 데이터를 보호하는데 사용하거나, 임의로 예외를 발생시켜 원하는 동작을 수행하도록 하는 데 사용할 수 있다. 하지만 MPU 를 메모리 보호의 주된 방안으로 사용하기에는 한계가 있다.

먼저, MPU 는 그 특성에 따라 사용 상의 한계가 존재한다. 보호하고자 하는 메모리 영역의 주소가 정렬되어 있어야 하며, 설정할 수 있는 영역의 개수가 한정되어 있어 몇 개의 영역을 설정할 수 있는지에 따라 보안 정도에 차이가 생긴다. 또한 MPU 설정을 위한 레지스터들은 시스템 제어 공간에 존재하는데, MPU 는 이 영역에 대해 접근 권한을 설정할 수 없다[3]. 따라서 공격자가 MPU 설정을 변경할 수 없도록 하기 위한 방법을 함께 구현해야 한다.

**Acknowledgement**

이 성과는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. RS-2023-00217689).

### 참고문헌

[1] Xi Tan, et al, Where's the "up"?! A Comprehensive (bottom-up) Study on the Security of ARM Cortex-M Systems, arXiv:2401.15289, 2024

[2] Yufei Du, et al, Holistic Control-Flow Protection on Real-Time Embedded Systems with Kage, 31st USENIX Security Symposium (USENIX Security 22), 2281-2298, 2022

[3] W Choi, et al, SuM: Efficient Shadow Stack Protection on ARM Cortex-M, Computers & Security, 103568, 2023

[4] Z Shen, K Dharsee, J Criswell, Randezvous: Making Randomization Effective on MCUs, Proceedings of the 38th Annual Computer Security Applications Conference, 28-41, 2022

[5] L.Luo, et al, fASLR: Function-Based ASLR via TrustZone-M and MPU for Resource-Constrained IoT Systems, in IEEE Internet of Things Journal, vol.9, no. 18, pp. 17120-17135, 15 Sept.15, 2022