비밀 의존성 기반 분할을 통한 임베디드 TrustZone-A 환경의 경량 ML-KEM 아키텍처*

노경민†, 권나희†, 박수현‡, 조성제¶
단국대학교 사이버보안학과, 단국대학교 컴퓨터학과‡, 단국대학교 소프트웨어학과¶
{imsie1, alsrl030902, parksh, sjcho}@dankook.ac.kr

A Lightweight ML-KEM Architecture via Secret-Dependency-Based Partitioning for Embedded TrustZone-A Systems

Kyoungmin Roh†, Nahee Kwon†, Suhyeon Park‡, Seong-je Cho¶
Deptartment of Cybersecurity, Department of Computer Science and Engineering,
Department of Software Science, Dankook University
{imsie1, alsrl030902, parksh, sjcho}@dankook.ac.kr

요 약

임베디드 시스템에서 ML-KEM을 안전하게 배포하기 위해서는 비밀 정보를 보호하면서도 제한된 계산·메모리 자원을 고려해야 한다. ARM TrustZone-A 기반 TEE는 비밀 연산을 Secure World에 격리할 수 있지만, ML-KEM 전체를 Secure World에서 수행하는 Full-TEE 방식은 공개 연산까지 TCB에 포함하여 코드 규모와 실행 부담을 증가시킨다. 본 논문은 ML-KEM 연산을 비밀키 의존성과 비밀 출력 발생 여부에 따라 분할하는 Split 기반 아키텍처를 제안한다. 제안 구조는 비밀키 의존 연산과 shared secret 생성을 Secure World 내부에 유지하고, 비밀키에 의존하지 않는 공개 연산은 Normal World에서 수행한다. 실험 결과, 제안 구조는 Full-TEE 대비 Secure World의 TCB를 약 90.8%, 전체 Latency를 약 25.5% 감소시켰다. 이는 임베디드 TrustZone-A 환경에서 ML-KEM의 비밀 보호 경계를 유지하면서도 시스템 가용성을 개선할 수 있음을 보여준다.

1. 서 론

임베디드 시스템은 제한된 계산 성능과 메모리 자원 하에서 키 교환, 인증, 암호화와 같은 보안 기능을 수행한다. 특히 IoT 기기와 같이 장기간 운용되는 플랫폼에서는 비밀키와 민감한 암호 연산을 일반 실행 환경으로부터 보호해야 한다. 이를 위해 ARM TrustZone-A 기반 Trusted Execution Environment (TEE)는 실행 환경을 Normal World (이하 NW)와 Secure World (이하 SW)로 분리하고, 비밀키 저장과 민감 연산을 SW 내부에 격리한다. 기존 RSA/ECC 기반 암호 시스템에서는 이러한 구조를 활용하여 주요 암호 연산을 SW에 배치함으로써, NW가 손상되더라도 핵심 비밀 정보가 직접 노출되지 않도록 한다[1, 2]. 양자컴퓨터의 발전에 따라 임베디드 시스템에서도 ML-KEM (Module-Lattice-Based Key-Encapsulation Mechanism)과 같은 양자내성암호 (Post-Quantum Cryptography, PQC)의 도입이 요구되고 있다[3]. 이에 따라 초기 임베디드 PQC 연구는 주로 일반 실행 환경에서 Kyber/ML-KEM 구현을 최적화하고 ARM Cortex-A 계열에서의 실행 가능성과 성능을 평가하였다[4]. 이러한 연구들은 임베디드 플랫폼에서도 PQC 적용이 가능함을 보였지만, NW가 손상될 경우 비밀 정보가 노출될 수 있다는 문제를 충분히 다루지는 못한다.

최근에는 이러한 한계를 보완하기 위해 PQC 연산을 ARM TEE 내부에 배치하여 비밀키를 보호하려는 연구가 진행되고 있다[5]. 그러나 PQC 연산은 계산량과 메모리 사용량이 기존 RSA/ECC 기반 암호보다 많다[3, 6]. 따라서 모든 PQC 연산을 SW에서 수행하는 Full-TEE 방식은 비밀 정보 보호에는 유리하지만, 공개 연산까지 TCB(Trusted Computing Base)에 포함하여 SW의 코드 규모, 검증 부담, 실행 지연을 증가시킨다[5, 7]. 즉, Full-REE 방식은 성능 측면에서는 유리하지만 비밀 정보 보호가 취약하고, Full-TEE 방식은 보호 경계는 명확하지만 임베디드 환경에서 실행 부담이 크다.

본 논문은 이러한 두 배치 방식의 한계를 완화하기 위해 PQC 알고리즘 중에서도 NIST에서 선정된 표준 양자 후 키 캡슐화 메커니즘인 ML-KEM[3]을 선정하여 비밀키 의존성에 따라 분리하는 Split 기반 TrustZone-A 아키텍처를 제안한다. 제안 구조는 비밀키 의존 연산과 shared secret 생성을 SW 내부에 유지하고, 공개 데이터만을 사용하는 계산 집약적 연산은 NW에서 수행한다. 이를 통해 Full-REE의 비밀 노출 문제와 Full-TEE의 TCB 및 지연 시간 증가 문제를 동시에 완화한다.

2. 관련 연구

RSA/ECC 기반 암호 시스템에서는 비밀키 보호를 위해 주요 암호 연산을 ARM TrustZone-A 기반 TEE의 SW에 배치하는

* 본 연구는 2026년 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학사업 지원을 받아 수행되었음(2024-0-00035) 또한 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원-학석사연계ICT핵심인재양성 지원을 받아 수행된 연구임(IITP-2026-RS-2023-00259867).
† 공동 1저자임.
¶ 교신 저자임.

방식이 사용되어 왔다[1, 2]. 또한 TrustZone 기반 TEE 연구들은 SW의 권한과 코드 범위를 줄여 TCB를 최소화하는 것이 보안성 확보에 중요함을 보였다[7].

이러한 연구들은 TEE 기반 암호 연산 분리가 임베디드 환경에서 유효한 설계 방향임을 보였지만, RSA/ECC는 연산 구조가 비교적 단순하여 ML-KEM과 같은 PQC의 대규모 다항식 연산과 비밀키 의존성을 그대로 반영하기 어렵다.

양자내성암호 도입이 요구되면서 초기 임베디드 PQC 연구는 주로 PQC 알고리즘을 NW 또는 일반 TLS 라이브러리에 통합하여 실행 가능성과 성능 오버헤드를 평가하였다[4]. 이후 최근 연구들은 비밀키 보호를 위해 ML-KEM과 같은 PQC 연산을 SW에 배치하는 Full-TEE 방식을 다루고 있다[5]. 그러나 Full-TEE는 공개 연산까지 TCB에 포함하여 SW 코드 규모와 지연 시간을 증가시킨다.

3. 위협 모델

본 논문은 NW가 손상될 수 있는 TrustZone-A 기반 임베디드 시스템을 가정한다. 공격자는 NW의 운영체제, 애플리케이션, 드라이버를 제어할 수 있으며, NW 메모리 읽기·쓰기, 공개 입력 변조, SW 호출 반복, shared memory 조작을 수행할 수 있다. 따라서 NW에서 처리되거나 저장되는 데이터는 공격자에게 노출될 수 있다고 본다.

반면 Secure Monitor, SW, OP-TEE 커널 및 TrustZone-A의 하드웨어 격리는 신뢰한다. 공격자는 SW 내부 코드와 데이터에 직접 접근할 수 없으며, SW에 저장된 decapsulation key, 비밀 난수, 비밀 중간값 및 shared secret을 읽거나 수정할 수 없다. 본 논문의 목표는 NW가 손상되더라도 ML-KEM의 장기 비밀키와 shared secret이 SW 외부로 노출되지 않도록 하는 것이다.

추가적으로, 본 논문은 물리적 공격, 결함 주입, 캐시·전력·타이밍 기반 부채널 공격 및 TEE 자체 취약점은 범위에 포함하지 않는다.

4. Split 기반 PQC 아키텍처

제안 구조는 ML-KEM의 비밀 정보를 SW 내부에 유지하면서, SW에 포함되는 코드와 실행 부담을 줄이는 것을 목표로 한다. 이를 위해 ML-KEM 연산을 비밀키 의존성 및 비밀 출력 발생 여부에 따라 분할한다. 비밀키를 생성·저장·사용하거나 shared secret을 생성하는 연산은 SW에서 수행하고, 비밀키에 의존하지 않는 공개 연산은 NW에서 수행한다. 그림 1은 Full-TEE와 제안 Split 구조의 차이를 나타낸다.

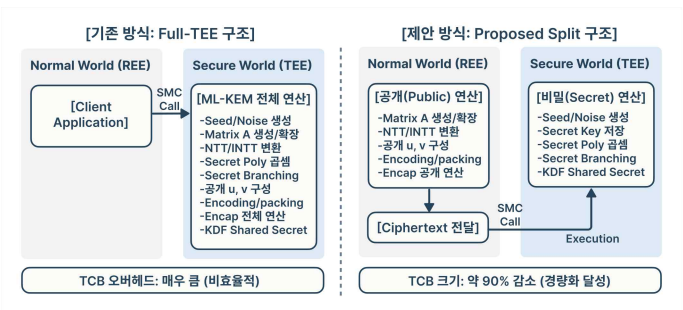

그림 1. Full-TEE와 Split 기반 ML-KEM 구조 비교

표 1은 KeyGen, Encaps, Decaps에서 수행되는 주요 연산의 배치를 나타낸다. KeyGen과 Decaps의 비밀키 의존 연산은 SW에 유지되며, Encaps의 공개 계산과 공개키 생성에 필요한 반복 연산은 NW에서 수행된다. 단, Encaps와 Decaps에서 생성되는 shared secret은 세션 보호에 사용되는 비밀 출력이므로 NW로 반환하지 않고 SW 내부에 저장한다.

표 1. ML-KEM 연산의 비밀키 의존성 기반 분할

단계	Secure World (SW)	Normal World (NW)
KeyGen	비밀 seed/noise 생성, secret vector 생성 및 저장, 공개키 구성에 필요한 비밀 의존 값 생성	public seed 기반 matrix expansion, 공개 데이터 encoding/packing, 공개키 전달
Encaps	shared secret 도출 및 저장, session key handle 생성, shared secret 기반 crypto service 제공	공개키 파싱, 공개 입력 기반 ciphertext 생성, ciphertext packing, Encaps 결과 전달
Decaps	decapsulation key 기반 복호화, ciphertext 검증, implicit rejection, shared secret 도출 및 저장	ciphertext 전달, Decaps 요청, session key handle 사용

제안 구조는 세 가지 불변식을 따른다. 첫째, 비밀키와 비밀키 의존 중간값, shared secret은 SW 밖으로 노출되지 않는다. 둘째, NW에 위임되는 연산은 공개 입력만으로 결정되며, 공격자가 연산 결과를 변조하더라도 SW 내부의 비밀 값은 직접 노출되지 않는다. 셋째, NW와 SW 사이의 인터페이스는 공개 파라미터, ciphertext, 공개 연산 결과, SW 내부 crypto service 호출로 제한된다. 제안 구조는 Full-TEE와 동일하게 장기 비밀키와 shared secret을 SW 외부로 노출하지 않는다. 다만 공개 연산은 NW에서 수행되므로, SW는 입력 형식과 ciphertext 검증을 통해 변조된 공개 입력이 비밀 상태에 영향을 주지 않도록 제한한다.

5. 구현 및 평가

실험은 Raspberry Pi 3 Model B 기반 ARM Cortex-A53 환경에서 OP-TEE를 사용하여 수행하였다. ML-KEM 파라미터 세트는 ML-KEM-512를 사용하였으며, 비교 대상은 Full-REE, Full-TEE, 그리고 제안 Split 구조이다. 각 구조는 동일한 ML-KEM 구현을 기반으로 구성하였으며, Latency, TCB 크기, SMC Call 수를 기준으로 비교하였다.

5.1 Latency 비교

Latency는 ML-KEM의 KeyGen, Encaps, Decaps 연산을 각각 2,000회 반복 실행한 평균 수행 시간으로 측정하였다. 표 2는 각 구조의 연산별 Latency를 나타낸다.

표 2. ML-KEM 연산별 Latency (ms, $N=2000$)

연산	Full-REE	Full-TEE	Split
KeyGen	0.919	1.561	1.181
Encaps	1.072	1.659	1.104
Decaps	1.336	1.881	1.512

세 연산의 평균 Latency는 Full-REE, Full-TEE, Split에서 각각 1.109ms, 1.700ms, 1.266ms로 측정되었다. Split 구조는 Full-TEE 대비 평균 Latency를 25.5% 감소시켰으며, Full-REE 대비 추가 지연은 0.157ms에 그쳤다. 특히 Encaps에서는 Fu

ll-TEE의 1.659ms가 Split에서 1.104ms로 감소하여 33.5% 개선되었다. 이는 Encaps가 비밀키를 사용하지 않는 공개 데이터 기반 연산이므로, 대부분의 계산을 NW에서 수행할 수 있기 때문이다.

KeyGen과 Decaps에서도 Split은 Full-TEE 대비 각각 24.3%, 19.6%의 Latency 감소를 보였다. 다만 두 연산은 비밀키 생성 또는 비밀키 기반 복원 과정을 포함하므로 Encaps보다 개선 폭이 작다. 결과적으로 Split 구조는 Full-TEE의 과도한 SW 실행 비용을 줄이면서 Full-REE에 가까운 Latency를 달성한다. 동시에 Full-REE와 달리 비밀키 의존 연산을 SW에 유지하므로, NW가 손상된 환경에서도 핵심 비밀 정보의 직접적 노출을 제한할 수 있다.

5.2 TCB 비교

표 3은 SW에 포함되는 TCB 크기를 비교한 결과이다. TCB는 SW Application 바이너리에 링크된 소스 코드를 기준으로 측정하였으며, 주석과 공백은 제외하였다.

표 3. Full-TEE와 Split 방식 TCB 크기 비교

항목	Full-TEE	Split
LoC	약 12,000 LoC	약 1,100 LoC
모듈 수	15+	3

Full-TEE는 ML-KEM 라이브러리 전체를 SW에 포함하므로 약 12,000 LoC 이상의 코드와 15개 이상의 모듈이 TCB에 포함된다. 반면 Split 구조는 비밀키 의존 연산에 필요한 기능만 SW에 배치하므로, TCB가 약 1,100 LoC와 3개 모듈로 감소한다. 이는 LoC 기준으로 Full-TEE 대비 약 90.8%의 TCB 축소에 해당한다. 이 결과를 통해 Split 구조는 비밀키 의존 연산을 SW에서 보호하면서도, Full-TEE보다 작은 TCB로 ML-KEM을 배치할 수 있다는 것을 알 수 있다.

5.3 SMC Call 비교

표 4는 Full-TEE와 Split 구조의 ML-KEM 연산별 SMC Call 수를 비교한 것이다.

표 4. SMC Call

항목	Full-TEE	Split
KeyGen	1	2
Encaps	1	1
Decaps	1	1
총합	3	4

Split 구조는 KeyGen에서 비밀 상태 생성과 공개키 구성 지원이 분리되어 2회의 SMC Call이 발생한다. Encaps에서는 ciphertext 생성을 위한 공개 계산을 NW에서 수행하지만, shared secret을 SW 내부에 저장하기 위해 1회의 SMC Call이 필요하다. Decaps는 비밀키 기반 복호화와 shared secret 생성을 포함하므로 Full-TEE와 동일하게 1회의 SMC Call이 발생한다. 따라서 Split 구조의 총 SMC Call 수는 Full-TEE보다 1회 증가한다. 그러나 증가한 호출은 shared secret을 SW 내부에 유지하기 위한 보호 비용이며, 성능 평가에서 Split은 여전히 Full-TEE 대비 평균 Latency를 25.5% 감소시킨다.

6. 결론 및 향후 연구

본 논문은 임베디드 TrustZone-A 환경에서 ML-KEM을 경량으로 배치하기 위한 Split 기반 PQC 아키텍처를 제안하였다. 제안 구조는 비밀 상태 의존 연산과 shared secret 생성을 SW에 유지하고, 공개 연산은 NW에서 수행함으로써 비밀 보호 경계를 유지하면서도 SW의 TCB와 실행 부담을 줄인다. 실험 결과, Full-TEE 대비 TCB를 약 90.8%, 평균 Latency를 약 25.5% 감소시켰다.

본 연구는 ML-KEM을 대상으로 하므로 다른 PQC 알고리즘에는 별도 분할 분석이 필요하며, side-channel 및 fault injection 공격은 범위에 포함하지 않았다. 향후 연구에서는 다양한 PQC 알고리즘으로의 확장성과 실제 하드웨어 기반 부채널·결함 공격 대응을 분석할 계획이다.

참고 문헌

[1] Arm, “TrustZone for Cortex-A,” Arm Developer Documentation.

[2] TrustedFirmware.org, “OP-TEE Documentation.”

[3] National Institute of Standards and Technology (NIST), “Module-Lattice-Based Key-Encapsulation Mechanism Standard,” FIPS 203, Aug., 2024.

[4] P. Sanal, E. Karagoz, H. Seo, F. Karakoyunlu, and A. Aysu, “Kyber on ARM64: Compact Implementations of Kyber on 64-bit ARM Cortex-A Processors,” in *IACR Transactions on Cryptographic Hardware and Embedded Systems*, Vol. 2021, No. 4, pp. 376-407, 2021.

[5] E. Andrade, C. C. Goes, and J. O. de Sousa, “Post-Quantum Algorithms on ARM Trusted Execution Environment (TEE): Findings of this Industrial Challenge,” in *Proceedings of 13th Latin-American Symposium on Dependable and Secure Computing (LADC)*, pp. 192-195, 2024.

[6] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehlé, “CRYSTALS-Kyber: A CCA-secure module-lattice-based KEM,” in *Proceedings of IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 353–367, 2018.

[7] D. Cerdeira, J. Martins, N. Santos, and S. Pinto, “ReZone: Disarming TrustZone with TEE Privilege Reduction,” in *Proceedings of 31st USENIX Security Symposium (USENIX Security)*, pp. 2261–2279, 2022.