ARM TrustZone 기반 경량 비밀 분리형 양자내성 암호 아키텍처
A Lightweight Secret-Isolated Post-Quantum Cryptographic Architecture for ARM TrustZone

노경민 권나희 김보겸* 전수빈**
Kyoungmin Roh Nahee Kwon Bokgyeom Kim* Subin Jeon**

단국대학교 사이버보안학과
***단국대학교 소프트웨어학과**
****단국대학교 물리학과 및 사이버보안학과**
{imsie1, alsrl030902, bogamie, 32203966}@dankook.ac.kr

요 약

임베디드 및 모바일 장치는 장기간 운영되는 특성상 양자컴퓨터 시대에 대비한 양자내성 암호의 적용이 점점 필수화되고 있다. ARM TrustZone-A 는 이러한 환경에서 민감 연산을 보호하기 위한 하드웨어 기반 격리를 제공하지만, Kyber 와 같은 PQC 알고리즘을 Secure World 에 전면 배치하는 기존 방식은 고비용 Polynomial 연산까지 모두 포함하므로 TCB 가 크게 증가하고 검증 부담이 확장되는 문제가 있다. 본 연구는 Kyber 의 연산 구조가 비밀 의존 연산과 공개 연산으로 명확히 구분된다는 점에 기반하여, 비밀 연산을 Secure World 에 고립시키는 Secure-Isolated PQC 아키텍처를 제안한다. 이를 위해, 비밀키와 관련된 비밀 연산은 모두 Secure World 에서 수행하고, 나머지 계산량이 큰 공개 Polynomial 연산들은 모두 Normal World 로 이전한다. 이 구조는 기존 방식과 동일한 SMC 호출과 entrypoint 개수를 유지하면서도 Secure World 의 LoC 를 약 90% 감소시키며, Normal World 가 완전히 손상된 공격자 모델에서도 비밀 데이터의 노출을 방지한다. 또한 임베디드 환경에서의 제약된 메모리 환경에서도 무거운 PQC 가 실행될 수 있도록 구성되었다.

1. 서 론

양자컴퓨터의 현실적 위협이 가시화되면서, 임베디드 및 모바일 장치에서도 양자내성 암호 (PQC)를 도입해야 할 필요성이 빠르게 증가하고 있다[1]. 이러한 장치들은 장기간 현장에서 운영되며 OTA 업데이트가 제한되는 경우가 많기 때문에, 장기 보안성을 보장하기 위한 PQC 의 조기 도입이 필수적이다[2].

이러한 환경에서 PQC 를 안전하게 배치하기 위한 대표적 방법은 ARM TrustZone-A (TZ-A) 기반 Trusted Execution Environment (TEE)를 활용하는 것이다[3]. TZ-A 는 단일 프로세서에서 Secure World (SW)와 Normal World (NW)를 하드웨어 수준에서 분리하여, 비밀 연산과 상태 정보를 SW 내부에 고립시키는 하드웨어 격리 기술이다. SW 는 신뢰 기반 코드가 실행되는 보호 영역이며, NW 는 일반 OS 및 애플리케이션이 위치한다. NW 는 Secure Monitor Call (SMC)을 통해서만 SW 기능에 접근할 수 있으며, 이는 임베디드 환경에서 비밀 연산으로 보호하기 위한 사실상의 표준 메커니즘으로 활용되고 있다.

기존 연구들은 이러한 TZ-A 의 격리 특성을 활용하여 PQC 전체 스택을 SW 에 배치하는 Full-TEE 방식을 주로 채택해 왔다[4][5]. 이 방식은 비밀 정보가 SW 외부로 유출되지 않는다는 직관적 장점이 있으나, Kyber 와 같은 PQC 는 고비용 polynomial 연산과 대규모 연산을 포함하므로, 전체 연산을 SW 에 배치하면 Trusted Computing Base (TCB)가 지나치게 커지고 SW 메모리·검증 부담이 기하급수적으로 증가한다.

본 연구는 Kyber 알고리즘의 세부 연산 구조를 분석한 결과, 비밀 의존 연산과 공개 polynomial 연산이 명확히 분리되어 있음을 관찰하였다. 우리는 이 구조적 특성을 활용하여, 비밀 의존 연산만 SW 에 고립시키고 공개 polynomial 연산은 NW 에서 수행하는 Split 기반 PQC 아키텍처를 설계한다. 이 설계는 Secret-Isolation 을 유지하면서도 TEE 의 부담을 최소화한다. 본 연구의 기여점은 다음과 같다.

- ARM TZ-A 의 분리 실행 모델에 최적화된 Secret-Isolated PQC 아키텍처를 제안한다.
- 기존 Full-TEE 방식 대비 SW TCB 를 약 90% 축소하는 구조적 이점을 제시한다.
- 임베디드·모바일 환경에서 PQC 배치를 위한 경량·구조적 설계 방향을 제안하고, 향후 TrustZone 기반 PQC 연구의 기반을 마련한다.

2. 관련 연구

2.1 TrustZone 기반 경량 TEE 연구

기존 TZ 연구는 SW 코드량과 권한을 최소화하여 TCB 를 줄이는 것이 보안성 확보의 핵심이라는 점을 반복적으로 강조해 왔다[6][7]. 이러한 연구들은 OS 드라이버·GPU 런타임 등 시스템 계층에서 필수 기능만 SW 에 남기는 구조를 제안해 TCB 축소에 실질적 기여를 제공하였다.

그러나 암호 알고리즘 내부의 secret-dependent vs. non-secret 연산 구조는 고려하지 않아, PQC 와 같은 복잡한 연산을 어떤 기준으로 분리해야 하는지 제공하지 못한다. 본 연구는 이 한계를 해결하기 위해 Kyber 연산의 비밀 의존선을 분석하고, secret-dependent 연산만 SW 에 고립시키는 명확한 분리 기준을 제안한다는 점에서 기존 TEE 경량화 연구와 차별화 된다.

2.2 PQC 전체를 Secure World 에 배치하는 Full-TEE 방식

최근 PQC 연구에서는 Kyber 전체 스택을 SW 에 배치하는 Full-TEE 방식이 주로 사용되었다[4][5]. 이 방식은 구현이 단순하고 보안 경계가 명확하다는 장점이 있다.

문제는 Kyber 의 NTT 연산, matrix-A 확장 등 고비용 polynomial 연산까지 모두 SW 에 포함되면서 TCB 가 수천~만 LoC 로 비대화 되고 SW 메모리 부담과 검증 비용이 급증한다는 점이다. 임베디드 환경에서는 이러한 구조가 실질적으로 비현실적이다.

본 연구는 Full-TEE 의 장점인 Secret-Isolation 은 유지하면서도, 대부분의 공개 연산을 NW 로 분리하여 SW LoC 를 약 90% 감소시키는 Split 구조를 제안하여 이러한 한계를 해결한다.

2.3 Classical PKC 의 TEE Offloading 연구

RSA/ECC 를 대상으로 한 기존 연구는 핵심 연산만 SW 에 남기고 나머지 연산을 NW 에 오프로딩하는 방식을 제안해 TEE 성능 최적화에 기여하였다[8][9].

그러나 classical PKC 는 구조가 단순하여 PQC 처럼 대규모 polynomial 연산과 다단계 파이프라인을 포함하지 않는다. 따라서 이러한 기준을 Kyber 에 그대로 적용하기 어렵다.

본 연구는 기존 PKC 오프로딩의 개념을 계승하되, Kyber 의 모듈러 LWE 기반 구조를 세밀히 분석하여 PQC 에 적용 가능

한 분리 기준을 재정의한다는 점에서 기존 연구의 한계를 확장한다.

3. 위협 모델 및 보안 목표

3.1 위협 모델

본 연구는 ARM TZ-A 환경에서 발생할 수 있는 현실적 공격 시나리오를 가정한다. 공격자는 NW 전역을 완전히 제어할 수 있으며, 이는 임베디드 장치에서 흔히 발생하는 권한 상승, 악성 애플리케이션 주입, OS 수준 취약점 악용 등을 포함한다.

이러한 능력을 가진 공격자는 NW 메모리 및 코드에 대한 임의적 읽기·쓰기, 암호 연산 중 생성되는 공개 중간값의 변조, 시스템 콜 흐름 교란, SW 에 전달되는 입력값·메시지 포맷·호출 순서의 조작 등을 수행할 수 있다. 또한 공격자는 과거 입력값이나 내부 상태를 재주입하여 SW 동작을 오도시키는 rollback 기반 공격을 시도할 수 있다.

그러나 SW 는 TZ 하드웨어에 의해 물리적으로 격리된 신뢰 영역으로 간주하며, Secure Monitor·Trusted OS·SW 내부 암호 모듈은 TCB 로 신뢰한다. 공격자는 SW 의 메모리 및 코드, 비밀 파라미터 또는 secret-dependent 연산의 중간 결과에 직접 접근할 수 없다. 즉, 공격자는 TEE 경계 밖(NW)에서는 강력한 능력을 가지지만, SW 내부에는 침투할 수 없는 비대칭적 위협 모델을 따른다.

본 연구에서 고려하는 핵심 공격 표면은 entry-point 기반 공격이다. TZ-A 는 SW 가 독립적으로 실행되지 않고 NW 의 호출에 의해 트리거되는 call-driven execution model 을 따른다. 공격자는 다음과 같은 방식으로 SW 침입을 시도할 수 있다.

- **입력 교란(Input Manipulation)**
  공격자는 SW 호출 시 전달되는 ciphertext, seed, command ID 등을 임의로 변조하여 SW 내부에서 오작동을 유도하려 할 수 있다.
  대응: 제안 구조는 SW 가 처리하는 모든 입력에 대해 형식·범위·길이·상태 검증을 강제하고, secret-dependent 경로에 영향을 주는 값은 SW 내부에서만 재생성한다. 그 결과 외부 입력이 secret 연산 경로에 직접 영향을 주지 않도록 설계하였다.
- **호출 순서 및 패턴 조작 공격**
  공격자는 SMC 호출 순서를 뒤바꾸거나 반복 호출하여, SW 가 비정상 상태에서 실행되도록 시도할 수 있다.
  대응: 제안된 SW 모듈은 각 entry-point 마다 상태 기반 제어를 적용하며, secret-dependent 연산을 수행하는 모듈은 독립적 내부 상태를 유지하며, 유효한 흐름 외의 호출은 즉시 거부한다. 또한 secret-dependent branching 이 발생하는 구간은 SW 내부에서만 수행되므로, NW 의 호출 패턴 조작이 secret-dependent control flow 에 영향을 줄 수 없다.
- **Rollback 및 재주입 기반 공격**
  공격자는 과거의 ciphertext 나 이전 상태를 재사용하여 SW 가 동일한 연산을 반복 수행하게 만들고, 이를 통해 secret-dependent 결과의 일관성을 검사하는 형태의 간접 공격을 시도할 수 있다.
  대응: 비밀 키를 만드는 모듈은 외부 입력이 없는 내부 난수원(TRNG)으로 seed 와 noise 를 생성하며, SW 는 secret 상태를 외부로 노출하지 않는다. 따라서 동일 입력 재주입이 secret-dependent 중간값에 대한 oracle 역할을 수행할 수 없으며, rollback 공격은 구조적으로 무력화된다.

3.2 보안 목표

본 연구는 TZ-A 환경에서 PQC 연산의 안전한 수행을 위해 비밀 의존 연산을 SW 내부에 고립시키는 아키텍처를 설계하는 것을 목표로 한다. 이를 위해 다음과 같은 보안 목표를 정의한다.

첫째, **비밀 정보의 기밀성 보장**이다. 비밀키, 난수 시드, secret-dependent 중간값 등 모든 민감 데이터는 SW 내부에서만 처리되며 NW 로는 노출되지 않는다. NW 는 공개 정보에 기반한 비밀 비의존 연산(선형 연산 · polynomial 등)만 수행하며, SW 는 비밀 연산을 독립적으로 처리함으로써 NW 가 완전히 탈취된 상황에서도 비밀이 보호되도록 한다.

둘째, **TCB 최소화**를 목표로 한다. 제안 아키텍처는 전체 암호 스택을 SW 에 배치하는 기존 방식과 달리, 비밀 의존 연산만을 TEE 내부에 배치함으로써 필요한 신뢰 기반을 최소화한다. 이는 SW 코드의 크기와 복잡도를 줄여 검증 부담을 낮추고, 잠재적 취약점 및 코드 재사용 가능성을 구조적으로 축소한다.

한편, ARM TZ-A 하드웨어 자체의 결함, Secure Monitor 및 운영체제 수준의 취약점, 마이크로아키텍처 기반 부채널 공격 등 본 설계가 통제할 수 없는 계층의 문제는 본 연구의 보호 범위에 포함되지 않는다. 제안 아키텍처는 정의된 TEE 신뢰 가정 위에서만 보안성을 논의한다.

4. 제안 방법

4.1 아키텍처 구조

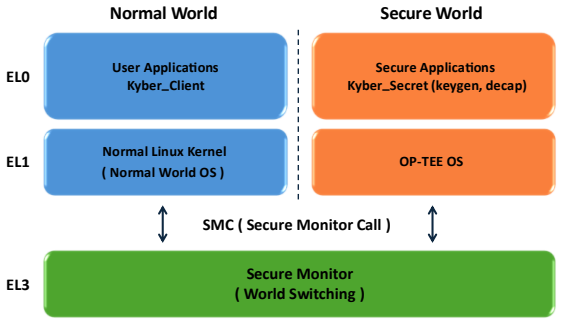

그림 1. 제안 아키텍처의 구조도

제안하는 시스템 아키텍처는 ARM Cortex-A 의 TZ 기반 isolation 실행 환경을 바탕으로 설계되었다. 전체 구조는 그림 1 과 같이 Kyber 알고리즘 연산을 Secret 과 Non-Secret 연산으로 구분하고 각각을 SW 와 NW 에 배치하는 방식으로 구성된다.

NW 에서 실행되는 일반 애플리케이션은 SMC 를 통해 Secure Monitor 로 진입한 후 제한된 SW 기능을 호출할 수 있다. Secure Monitor 은 World Switching, 레지스터 컨텍스트 저장 및 복원, 접근 권한 검증을 수행하며 이는 두 World 간 호출의 주요 오버헤드를 구성한다.

암호 연산 중 비밀키 처리, Noise Sampling, Secret-dependent branching 과 같이 기밀성이 요구되는 연산은 모두 SW 에 배치하였다. 반면 NTT 기반 polynomial 연산, matrix-A 생성, hash 기반 확장 함수, ciphertext 조립 등 반복적이며 데이터 패턴이 공개되어 있는 연산은 NW 에서 수행한다. 이러한 분리 전략은 SW 의 **TCB 규모를 최소화**하는 동시에 **무거운 PQC 알고리즘이 임베디드 환경에서도 실행**되게 만든다.

4.2 비밀 연산 고립 PQC 알고리즘

본 연구에서는 모듈러 LWE 기반 KEM 인 Kyber[10]을 split 실행 구조에 가장 적합한 PQC 알고리즘으로 채택하였다. Kyber 는 설계상 다음과 같은 구조적 특성을 가지므로 TZ 기반 분리에 특히 유리하다. Kyber 는 비밀 시드, noise 샘플링, 비밀 벡터, polynomial 계수 연산, NTT 기반 곱셈과 같은 반복적 연산이 명확히 분리되는 특성을 갖는다.

이와 같은 구조적 특성은 TEE 환경에서 secret 연산을 SW 에 고립시키고 대규모의 반복적 공개 연산을 NW 에 분리하는 split 아키텍처를 적용하기에 적합하다. 표 1 은 Kyber 의 비밀 의존 연산과 공개 연산을 어떻게 SW 와 NW 에서 분리했는지를 나타내는 표이다.

표 1. Kyber 의 모듈 분리

Secure World (TEE)	Normal World (REE)
- Seed 및 noise 생성 - Secret key 벡터 저장 - Secret-dependent polynomial 곱셈 - Secret-dependent branching - KDF 기반 최종 shared secret 생성 - 내부 난수원(TRNG) 기반 비밀 파라미터 생성	- 공개 matrix A 생성 및 확장 - Polynomial NTT/INTT 변환 - 공개 u, v 구성 - 공개 encoding/packing - Encapsulation 전체 연산 - Ciphertext 전달 및 SW entry-point 호출

4.3 Secure World Components

Secure World 에는 비밀 정보를 생성, 저장, 사용하는 기능만을 포함하는 최소 신뢰 기반 구성 요소만 포함한다. 본 연구는 다음과 같은 세 가지 secure 모듈을 정의한다. 이 장에서는 각각의 모듈을 설명하고, 그 밑에 모듈의 의사코드(pseudocode)를 보여준다.

첫째, Secure_KeyGen_Secret 은 Kyber 에서 사용하는 시드 및 noise 벡터를 생성하며 생성된 모든 비밀 파라미터는 Secure World 의 전용 메모리 공간에 저장된다. 본 모듈은 외부 입력을 허용하지 않고 내부 난수원을 기반으로 동작함으로써 seed 재사용, bias 유발 등과 같은 Normal World 공격자의 개입을 원천적으로 차단한다. 모듈에서는 seed_sk 의 NW 으로의 유출을 막기 위해서 공개키 연산의 일부를 SW 에서 수행하고, seed_A 를 보내 나머지 공개키 연산을 수행할 수 있도록 한다.

Algorithm 1. **Secure Key Generation Module**

```
[1]   # 1. Secure KeyGen
[2]   function Secure_KeyGen_Secret():
[3]       seed_sk ← TRNG()
[4]       (s,e) ← SampleNoise(seed_sk)
[5]       store_secure("seed_sk", seed_sk)
[6]       store_secure("s", s)
[7]       store_secure("e", e)
[8]
[9]       # Part of Public Keygen (seed_A)
[10]      seed_A ← SHAKE128(seed_sk || "A")
[11]
[12]      return seed_A
```

둘째, Secure_ScalarMul 은 ECC 기반 키 교환 또는 인증 프로토콜과의 연동을 위해 비밀 스칼라 곱 연산을 제공한다. ECC 스칼라 값은 유출 시 전체 프로토콜 보안성이 붕괴되므로 본 모듈은 모든 스칼라 및 연산 중간값을 SW 내부에서 처리하여 multi-protocol 환경에서의 key isolation 을 보장한다.

Algorithm 2. **Secure Secret-dependent Multiplication**

```
[1]   # 2. Secure_ScalarMul
[2]   function Secure_ScalarMul(PoP, secretID):
[3]       secret ← load_secure(secretID)
[4]       result ← Multiply(PoP, secret)
[5]       return result
```

셋째, Secure_Decap 은 Kyber decapsulation 과정 중 비밀 벡터를 참조하거나 secret dependent branching 이 발생하는 구간만을 수행한다. 전체 decap 중 공개 연산은 NW 에서 동작 시키되 correctness-check, re-encryption 비교 등 secret 기반 control flow 는 SW 에서 독립적으로 처리하여 side-channel 및 rollback 기반 조작을 차단한다.

Algorithm 3. **Secure Kyber Decapsulation Module**

```
[1]   # 3. Secure Decap
[2]   function Secure_Decap(ciphertext c):
[3]       (u, v) ← Unpack(c)
[4]       s ← load_secure("s")
[5]
[6]       # compute u*s (secret-dependent)
[7]       us ← Multiply(polynomialNTT(u), s)
[8]
[9]       # recover message
[10]      m’ ← Decode(v – us)
[11]
[12]      # generate shared secret
[13]      K ← KDF(m’, c)
[14]
[15]      return K
```

여기에서 decapsulation 결과로 생성되는 공유 비밀 K 는 Kyber 표준 사양에서도 호출자에게 반환되는 세션 키이므로, SW 에서 계산된 후 NW 로 전달되는 것이 정상 동작이다. 이는 위에서 언급했던 long-term security 를 직접 결정하는 값인 “민감 데이터”가 아닌 세션 키이므로 본 구조의 위협 모델과 일관된다.

4.4 Normal World Components

NW 는 Secret 기반 요소를 포함하지 않는 공개적이고 deterministic 연산만 수행하도록 제한한다. 구체적으로는 공개 파라미터 생성, polynomial NTT/INTT 연산, key formatting 및 패딩 처리가 포함된다. Public-Side NW 에서는 다음과 같은 연산이 주로 이루어진다. 먼저, 공개키의 Generation 이 수행된다. 공개키의 생성은 다음과 같은 의사 코드로 진행된다.

Algorithm 4. **Public-Side Key Generation**

```
[1]   # 4. Public KeyGen
[2]   function Public_KeyGen(seed_A):
[3]       A ← ExpandMatrix(seed_A)
[4]
[5]       # request secret multiplication
[6]       As ← SW_Multiply_Secret(A, "s")
[7]
[8]       # request full t computation
[9]       t ← SW_Multiply_Secret(A, "full_t")
[10]
[11]      pk ← Pack(seed_A, t)
[12]      return pk
```

이후, Encapsulation 연산이 수행된다. Encapsulation 은 Kyber

구조상 secret-dependent 연산을 포함하지 않으므로 전체가 NW 에서 수행된다. 다음은 Encapsulation 의 의사 코드를 나타낸 것이다.

Algorithm 5. **Public-Side Encapsulation**

```
[1]   # 5. Public Encap
[2]   function Public_Encap(pk):
[3]       (seed_A, t) ← Unpack(pk)
[4]
[5]       m ← RandomBytes()
[6]       r, e1, e2 ← SamplePublicNoise(m)
[7]
[8]       A ← ExpandMatrix(seed_A)
[9]       u ← PolynomialNTT(A) * r + e1
[10]      v ← t*r + e2 + Encode(m)
[11]
[12]      c ← Pack(u, v)
[13]      K_nw ← KDF_NW(m, c)
[14]      return (c, K_nw)
```

마지막으로, NW 는 ciphertext 를 Secure World 에 전달하는 역할만 수행한다. 그 의사 코드는 다음과 같다.

Algorithm 5. **Public-Side Interface for Decapsulation**

```
[1]   # 5. Decap Request
[2]   function NW_Request_Decapsulation(c):
[3]       K ← SW_Decapsulate(c)
[4]       return K
```

이러한 연산들은 데이터 접근 패턴과 제어 흐름이 secret 과 무관하므로 NW 에서 수행되더라도 정보 누출 위험이 없다. 또한 Kyber 알고리즘 중 대부분의 고비용 polynomial 연산이 NW 에서 처리되므로 전체 시스템 성능을 개선할 수 있다. 또한 임베디드 환경에서 SW 의 메모리 부담을 줄여주는 역할도 수행한다. 따라서 Secret 이 필요한 시점에만 SMC 를 통해 SW 의 모듈을 호출함으로써 NW 는 공개 연산 중심의 경량 역할을 유지하면서도 전체 파이프라인의 secret isolation 을 보장한다.

5. 평가

본 장에서는 제안하는 Secret-Isolated PQC 아키텍처가 TZ-A 기반 환경에서 제공하는 구조적 이점과 보안적 효과를 평가한다. 기존 방식인 Full-TEE Kyber 과 비교하여, **TCB 규모, cross-world 호출 패턴** 측면에서 차이를 분석한다. 모든 분석은 Kyber 의 ARM64 버전을 OP-TEE 환경에 맞게 수정한 것을 바탕으로 분석하였다.

5.1 TCB 감소 분석

Full-TEE 방식은 Kyber 의 모든 연산을 SW 에서 수행하므로 TCB 가 수천~만 LoC 규모로 확대된다. 이는 SW 의 메모리 제약을 심화시키고, 코드 감사 및 보안 검증 난이도를 증가시키는 주요 원인이 된다.

반면, 제안 아키텍처는 비밀(seed, s, e) 및 Secret-dependent 연산만 SW 에 남기고, 반복적이고 공개적인 polynomial 연산은 NW 에 이전한다. 다음 표 2 는 실제 구현 코드를 기준으로 구성 요소를 분류한 결과를 나타낸 것이다.

표 2. Full-TEE 와 Split 구조의 TCB 비교

항목	Full-TEE	Split	감소율
SW LoC	12,000+ LoC	약 1,100 LoC	≈ 90% 감소
SW 모듈 수	15+	3	≈ 80% 감소
SW 연산	100%	5-10%	대폭 감소
SW memory	높음	낮음	구조적 감소

표를 보면, SW 에 상주하는 코드량은 약 1,100 LoC 수준으로 축소되며, 이러한 축소는 side-channel 공격 가능성과 memory safety 취약점의 실질적 노출 범위를 줄이고, 검증 가능한 신뢰 경계를 형성한다.

5.2 Cross-World Call 분석

SW 는 독립적으로 실행되는 프로세스가 아니며, Normal World (REE)가 SMC 를 통해 진입을 요청해야만 코드가 실행되는 call-driven execution model 을 따른다. 따라서 SW 내부에서 Kyber 의 모든 연산을 수행하는 Full-TEE 구조라 하더라도, Normal World 는 각 기능(KeyGen, Encap, Decap)을 호출하기 위해 반드시 한 번의 NW→SW 진입을 수행해야 한다. 즉, SW 내부 연산량과 관계없이, 기능 단위로 최소 1 회의 cross-world call 이 발생한다.

Full-TEE Kyber 구성에서는 세 기능(KeyGen, Encapsulation, Decapsulation)이 모두 단일 Trusted Application(TA)이 제공하는 각각의 command handler 로 구현되며, NW 는 각 기능을 수행할 때 TEEC_InvokeCommand()를 통해 정확히 1 회 SMC 호출을 발생시킨다. Secure World 내부에서는 수천 회의 polynomial 연산·noise 샘플링·NTT 변환 등이 수행되지만, NW 와 SW 간의 경계 전환은 기능 단위 호출 1 회로 제한된다. 이러한 구조는 entry-point 개수를 최소화하지만, SW 내부의 코드 크기와 책임 범위가 지나치게 확장되는 문제가 발생한다.

반면, 제안하는 Split 구조에서는 Kyber 의 secret-dependent 연산만 SW 에서 수행하며, NTT/INTT, polynomial multiplication, matrix A 생성 등 공개적이고 deterministic 한 연산은 모두 Normal World 로 이전된다. 그 결과, SW 가 수행하는 연산은 더 작은 기능 단위로 재구성되며, 특히 KeyGen 에서는 "비밀 생성 기능"과 "비밀 기반 곱셈 기능"이 구분되어 두 개의 entry handler 로 분리된다. 그러나 Encapsulation 기능은 secret-dependent 요소를 포함하지 않으므로 SW 에서 제거되어, 전체 entry-point 개수는 Full-TEE 와 동일하게 유지된다. 즉, entry-point 수는 증가하지 않지만, entry-point 의 역할이 기능 중심에서 secret-centric 구조로 재편되는 구조적 차이가 존재한다. 다음 표 2 와 그림 2 는 Full-TEE 와 제안 Split 구조의 SMC 호출 횟수와 SW entry-point 구성을 비교한 것이다.

표 3. SMC Call & Entry point count 비교

항목	Full-TEE	Split
KeyGen	1	2
Encap	1	0
Decap	1	1
총합	3	3

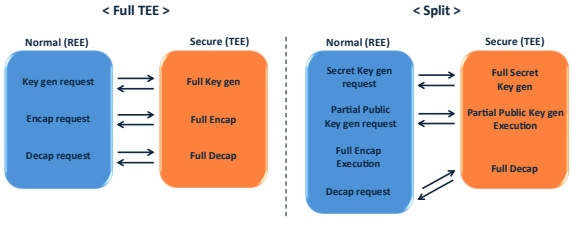

그림 2. Full TEE 와 Split 구조의 SMC Call 비교

표 3 과 그림 2 에서 보면, KeyGen 은 Split 에서 2 번이다. 이 이유는 제안 Split 구조에서는 public key 와 secret key 의 연산이 분리되어 있어 entry-point 와 SMC call 이 기존에 비해 증가하였다. 하지만, Split 구조에서는 Encapsulation 이 NW 에서만 이루어지기 때문에 entry-point 와 SMC Call 이 없어진다. 이때, Decapsulation 은 동일하기 때문에 따라서 전체적으로 봤을 때 두 구조의 entry-point 와 SMC Call 의 개수는 동일하지만, 제안 구조는 TCB 의 크기와 구조적 분리로 인한 보안성과 가용성이 늘어난다.

6. Discussion

6.1 제안 구조의 단점

제안한 Split 아키텍처는 TZ-A 환경에서 PQC 연산을 효율적으로 분리한다는 장점을 가지지만, 구조적 특성으로 인해 다음과 같은 한계와 제약을 가진다.

- **Entry-point 를 지날 때의 latency 가능성:** 본 연구에서는 NW 에서 공격자가 입력을 조작할 수 있으므로 그를 막기 위해 다양한 검증을 수행한다. 그에 따라 NW 에서 SW 로 넘어가는 entry-point 에서 latency 가 발생할 수 있다.
- **Kyber 이외의 PQC 알고리즘에 대한 일반성 부족:** Kyber 은 비밀 벡터 기반 곱셈과 공개 polynomial 연산이 명확히 구분되는 구조적 특성을 가지므로 본 Split 구조에 적합하지만, 다른 PQC 알고리즘은 연산 패턴이 더 복잡하고 secret-dependent 경로가 더 넓게 분포한다. 따라서 본 논문의 Split 전략이 모든 PQC 전체에 일반적 솔루션으로 간주되기 어렵고, 알고리즘별 맞춤형 분석이 필요하다.
- **실제 하드웨어 기반 평가의 부재:** 본 연구는 OP-TEE 소프트웨어를 중심으로 평가되었기 때문에 실제 Cortex-A 하드웨어 특성이 충분히 반영되지 않았다. 따라서 보드 기반의 검증이 반드시 필요하다.

6.2 향후 연구

향후 연구에서는 다음과 같은 확장 방향을 고려한다. 우선, 제안한 Split 아키텍처를 다양한 PQC 알고리즘에 적용하여 알고리즘별 메모리 사용량 및 연산 성능을 정량적으로 비교할 계획이다. 이를 통해 Split 구조의 일반성과 적용 가능 범위를 보다 명확히 평가하고자 한다. 또한 본 연구는 소프트웨어 기반 환경에서 검증되었으므로 향후 연구에서는 실제 ARM Cortex-A 보드에서의 하드웨어 동작 특성을 반영한 평가를 수행하여 구조적 장단점을 실제 기기 관점에서 검증하고자 한다. 마지막으로 Split 구조의 경량성을 활용하여 Cortex-M 기반 저전력, resource 제약 환경에서의 적용 가능성을 탐색함으로써 임베디드 환경에서의 보안 아키텍처로 확장할 계획이다.

7. 결론

본 연구는 ARM TZ-A 환경에서 Kyber 기반 양자 내성 암호를 효율적이고 안전하게 수행하기 위한 Secret-Isolated Split 아키텍처를 제안하였다. 기존 Full TEE 방식이 Kyber 전체 스택을 SW 에 배치함으로써 TCB 확장과 검증 부담을 초래한 것과 달리, 본 연구는 비밀 연산만 SW 에 남기고, 반복적이고 공개적인 연산은 NW 로 분리하는 효율적 구조를 제시하였다. 제안 방식은 entry point 및 SMC 호출 횟수를 변경하지 않으면서도 SW 에 비밀 연산을 고립시켜 TCB 규모를 90% 이상 감소시켰으며, 이에 따라 잠재적 공격 표면을 크게 축소하였다. 또한 SW 의 메모리 및 처리 자원이 제한적인 임베디드 환경에서 공개 연산을 NW 로 이전함으로써 전체 실행 부담을 완화할 수 있게 되었다. 본 연구는 Kyber 기반 PQC 를 자원 제약이 있는 시스템에서 보다 실용적으로 안전하게 배치하기 위한 경량 아키텍처의 설계 가능성을 제시한 첫 시도로서 중요한 의미를 가지며, 향후 TrustZone 기반 PQC 최적화 연구의 기반이 될 수 있다.

참고문헌

[1] Soundes Marzougui and Juliane Krämer, “Post-Quantum Cryptography in Embedded Systems,” in *ARES '19: Proceedings of the 14th International Conference on Availability, Reliability, and Securtiy*, no. 48, pp. 1-7, August 2019. doi: doi.org/10.1145/3339252.3341475

[2] Conner Bradley and David Barrera, “Escaping Vendor Mortality: A New Paradigm for Extending IoT Device Longevity,” in *NSPW '23: Proceedings of the 2023 New Security Paradigms Workshop*, pp. 1-16, December 2023. doi: doi.org/10.1145/3633500.363350

[3] Wenhao Li, Yubin Xia, and Haibo Chen, “Research on ARM TrustZone,” in *GETMobile: Mobile Computing and Communications*, vol. 22, pp. 17-22, January 2019. doi: doi.org/10.1145/3308755.3308761

[4] Ewerton Andrade, Cristiano Coimbra Goes, and Janislley Oliveira de Sousa, “Post-Quantum Algorithms on ARM Trusted Execution Environment (TEE): findings of this industrial challenge,” in *LADC '24: Proceedings of the 13th Latin-American Symposium on Dependable and Secure Computing*, pp. 192-195, December 2024. doi: doi.org/10.1145/3697090.3699864

[5] Cristiano Goes, Janislley Sousa, João Bezerra Neto, and Ewerton Andrade, “Key-Encapsulation Mechanisms Embedded in Trusted Execution Environment: An Evaluation,” in *2025 IEEE International Conference on Consumer Electronics (ICCE)*, January 2025. doi: doi.org/10.1109/ICCE63647.2025.10929991

[6] David Cerderia, José Martins, Nuno Santos, and Sandro Pinto, “REZONE: Disarming TrustZone with TEE Privilege Reduction,” in *31st USENIX Security Symposium (USENIX Security 22)*, pp. 2261-2279, August 2022.

[7] Chenxu Wang, Yunjie Deng, Zhenyu Ning, Kevin Leach, Jin Li, Shoumeng Yan, Zhengyu He, Jiannong Cao, and Fengwei Zhang, “Building a Lightweight Trust Execution Environment for ARM GPUs,” in *IEEE Transactions on Dependable and Secure Computing*, vol. 21, pp. 3801-3816, August 2024. doi: doi.org/10.1109/TDSC.2023.3334277

[8] Vipul Gupta, Douglas Stebila, Stephen Fung, Sheueling Chang, Nils Gura, and Hans Eberle, “Speeding up Secure Web Transactions Using Elliptic Curve Cryptography,” in *Proceedings of Network and Distributed System Security Symposium (NDSS) 2004*, 2004.

[9] Manuel Suárez-Albela, Paula Fragna-Lamas, and Tiago M. Frenández-Caramés, “A Practical Evaluation on RSA and ECC-Based Cipher Suites for IoT High-Security Energy-Efficient Fog and Mist Computing Devices,” in *Sensors 2018*, no. 11, 2018. doi: doi.org/10.3390/s18113868

[10] Joppe Bos, Leo Ducas, Eike Kiltz, T Lepoint, Vadim Lyubashevsky, and John M. Schanck, “CRYSTALS-Kyber: A CCA-Secure Module-Lattice-Based KEM,” in *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pp. 353-367, 2018. doi: doi.org/10.1109/EuroSP.2018.00032