KpqC 2라운드 후보 TiMER의 메시지 인코딩에 대한 단일파형공격

국민대학교 무기체계 기술보호 연구실

2024년 11월 29일 (금)

유성환, 한재승, 한동국

2024년 한국정보보호학회 동계학술대회

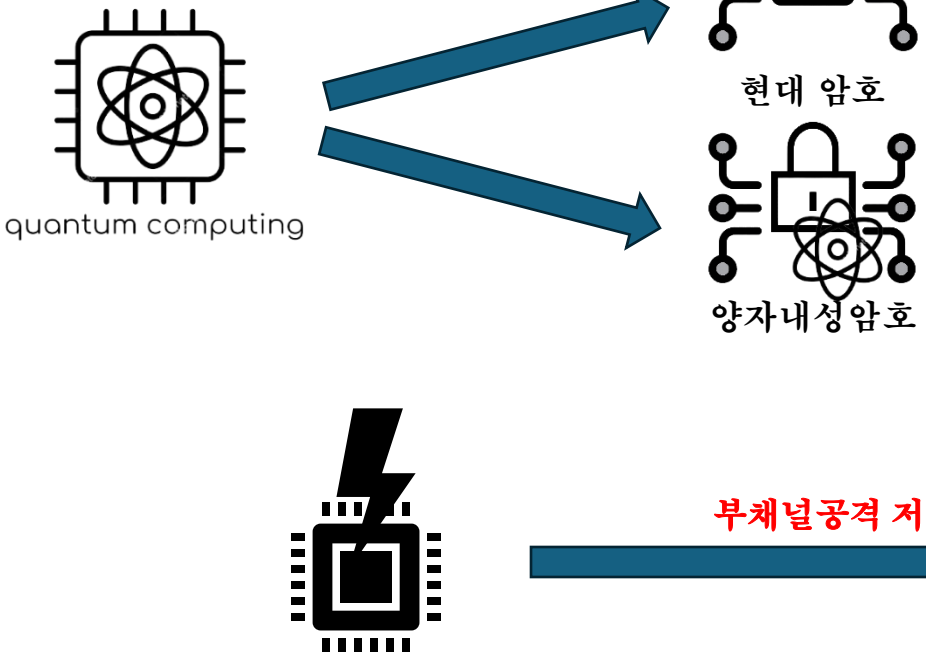
2024년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구결과임. (IITP-2024-00164800,)

Contents

Contents

Motivation

❖ 양자내성암호 등장 및 부채널공격 저항성 연구 필요



Post-Quantum Cryptography PQC

KpqC Competition Round 2

(April 2024 ~ November 2024)

국내·외 공모 중

구분	알고리즘
DSA	AIMer
	HAETAE
	MQ-Sign
	NCC-Sign
PKE/KEM	SMAUG-T
	NTRU+
	PALOMA
	REDOG

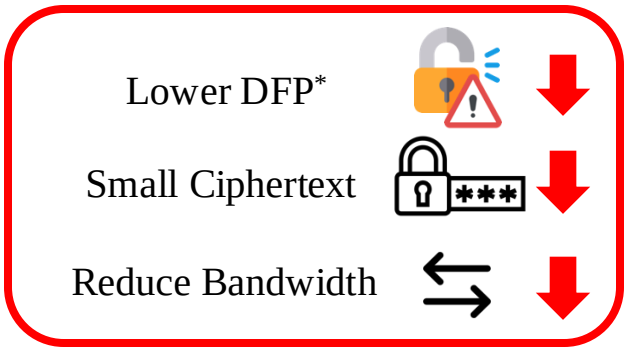
KpqC 2라운드 후보

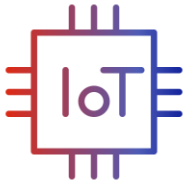
TiMER Algorithm

❖ SMAUG-T 알고리즘 모드

알고리즘	모드
SMAUG-T (PKE/KEM)	TiMER
	SMAUG-T128
	SMAUG-T192
	SMAUG-T256

D2 Error Reconciliation



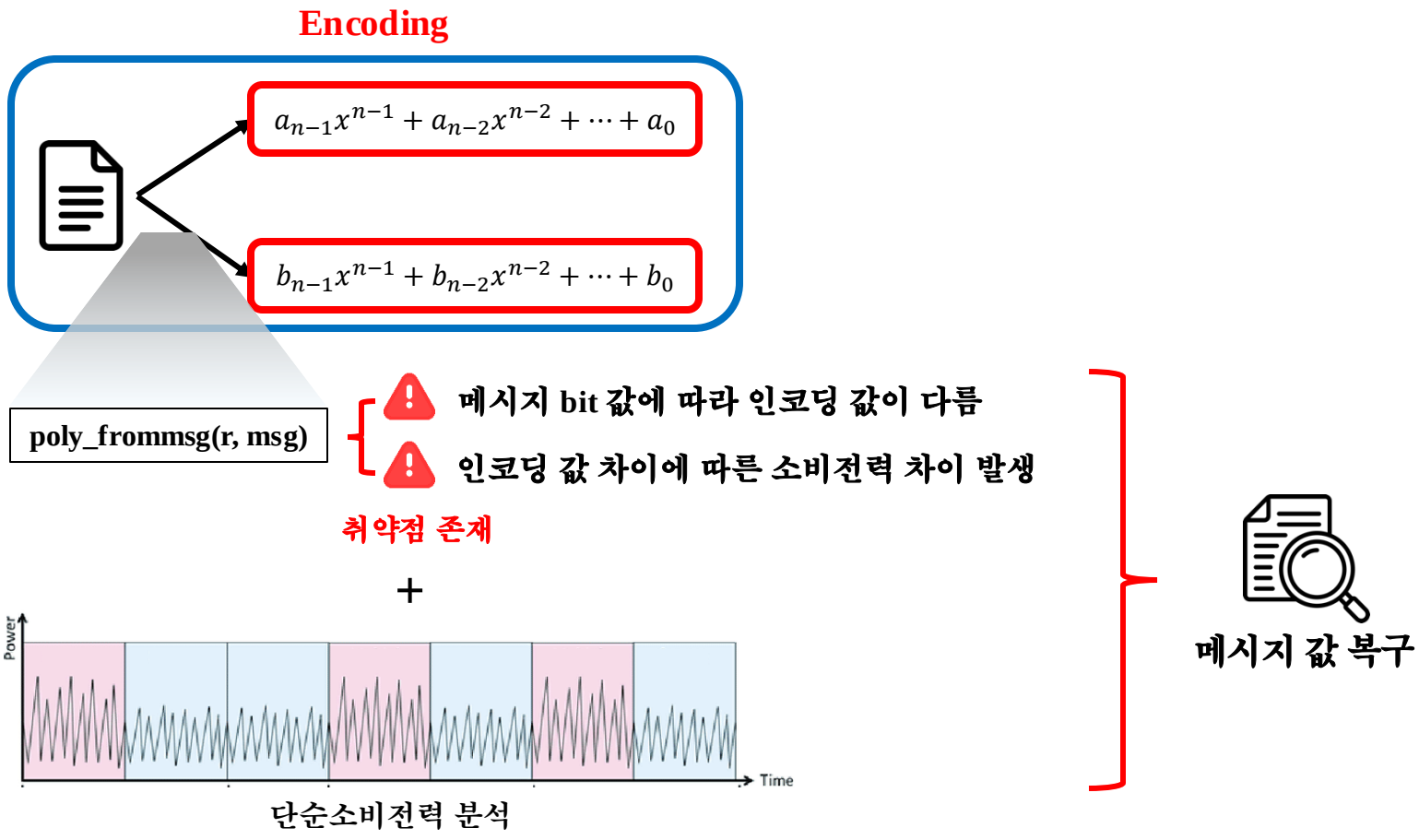


* DFP(Decryption Failure Probability)

Contribution

❖ TiMER에 대한 단일파형 공격기법 제안

- D2 인코딩의 취약점과 단순소비전력을 활용해 단일파형으로 메시지 값 복구

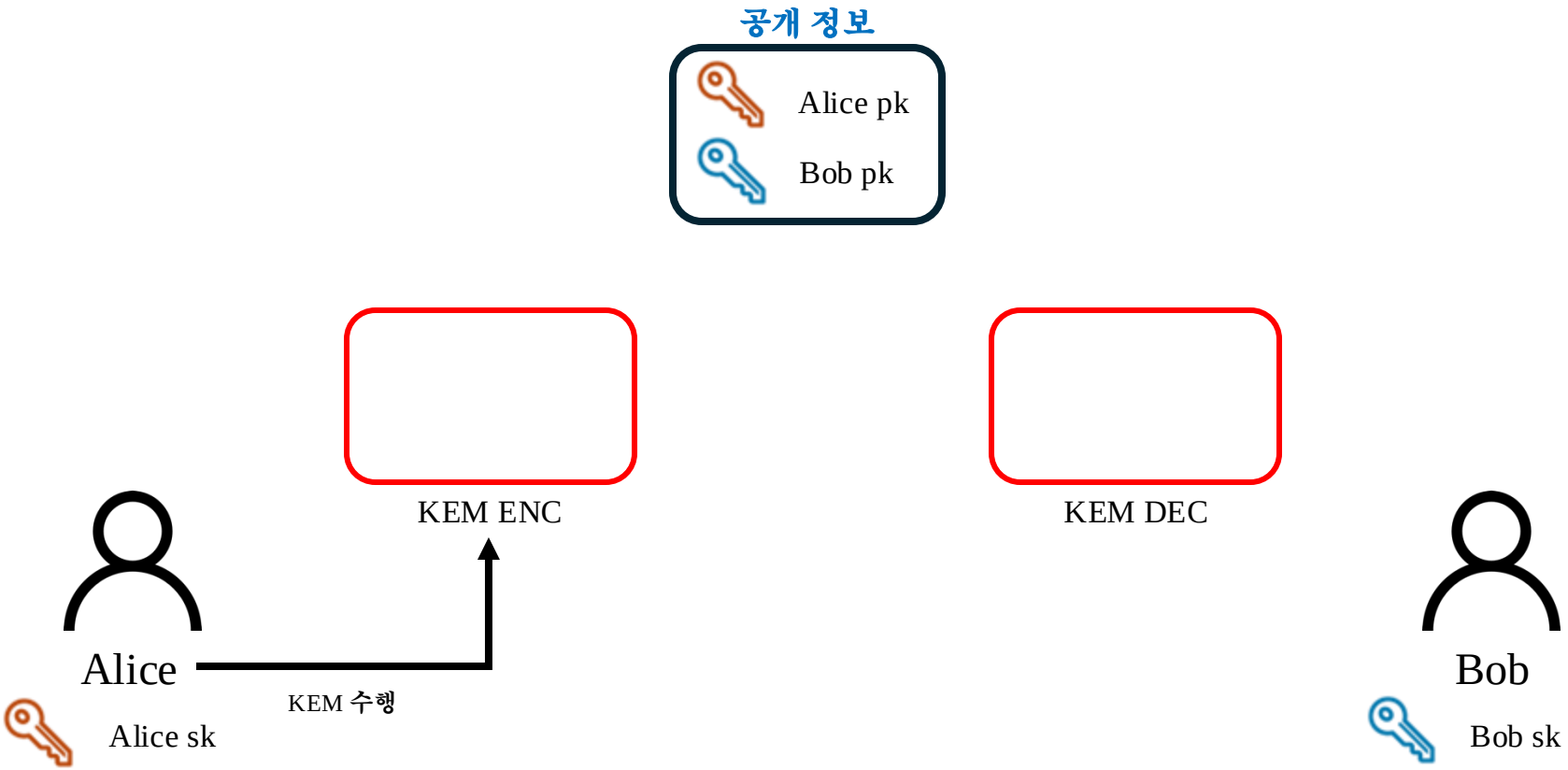


Contents

Background

❖ KEM 구조에서 단일파형 공격의 중요성

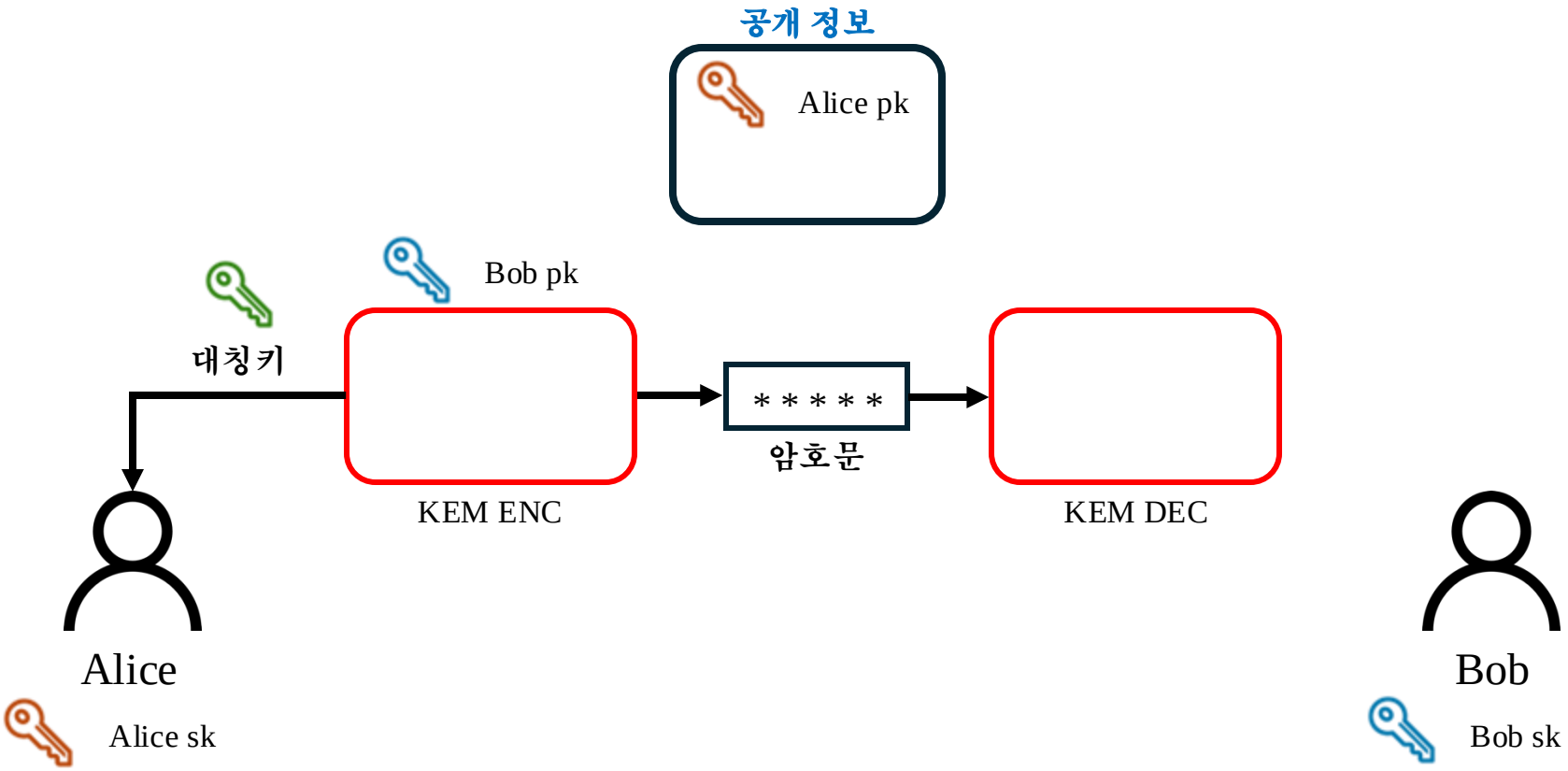
- KEM (Key Encapsulation Mechanism) 이란?
  - ✓ KEM 구조는 PKE(Public Key Encryption, 공개키 암호) 알고리즘을 사용한다.



Background

❖ KEM 구조에서 단일파형 공격의 중요성

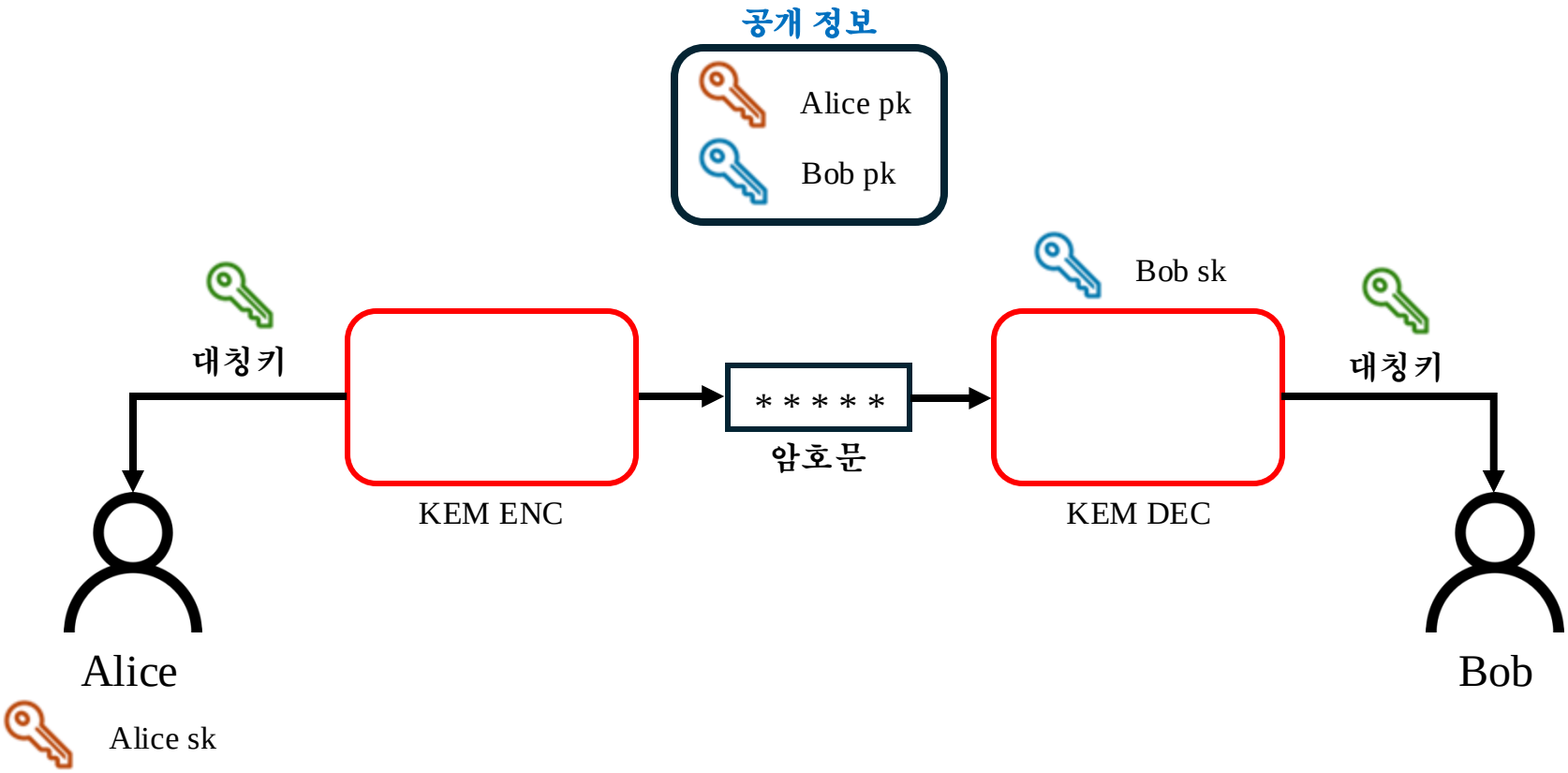
- KEM (Key Encapsulation Mechanism) 이란?
  - ✓ KEM 구조는 PKE(Public Key Encryption, 공개키 암호) 알고리즘을 사용한다.



Background

❖ KEM 구조에서 단일파형 공격의 중요성

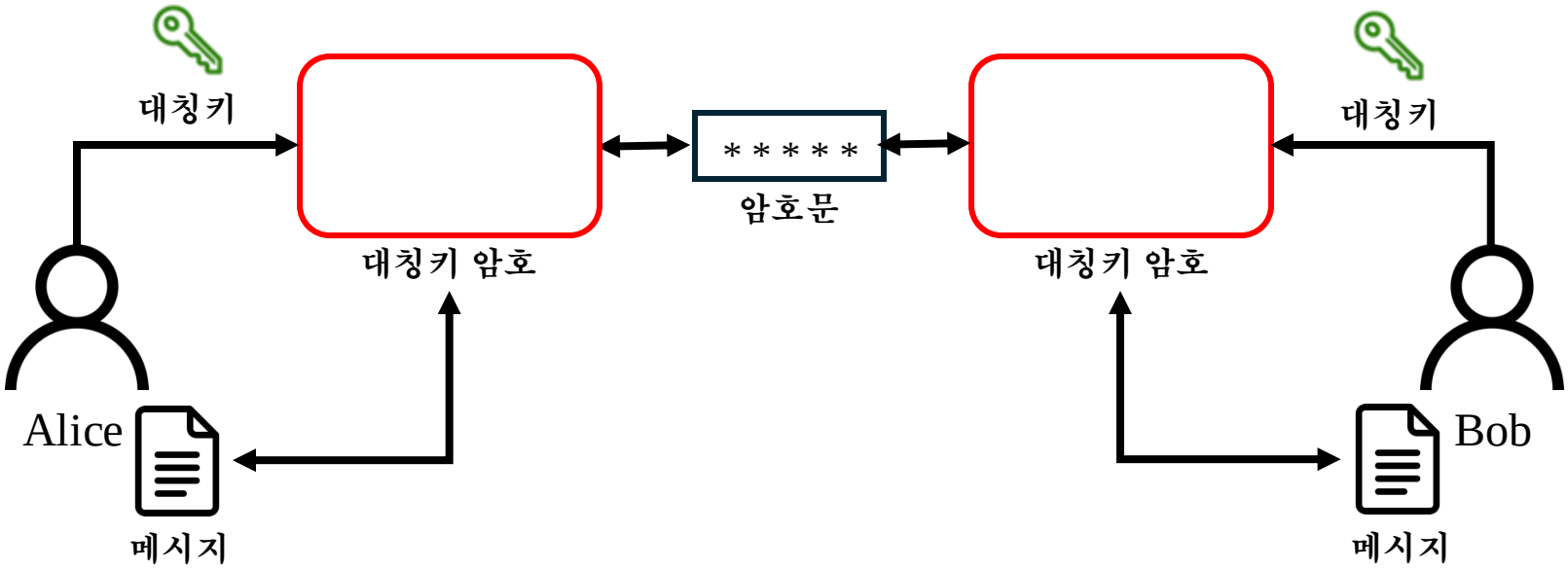
- KEM (Key Encapsulation Mechanism) 이란?
  - ✓ KEM 구조는 PKE(Public Key Encryption, 공개키 암호) 알고리즘을 사용한다.



Background

❖ KEM 구조에서 단일파형 공격의 중요성

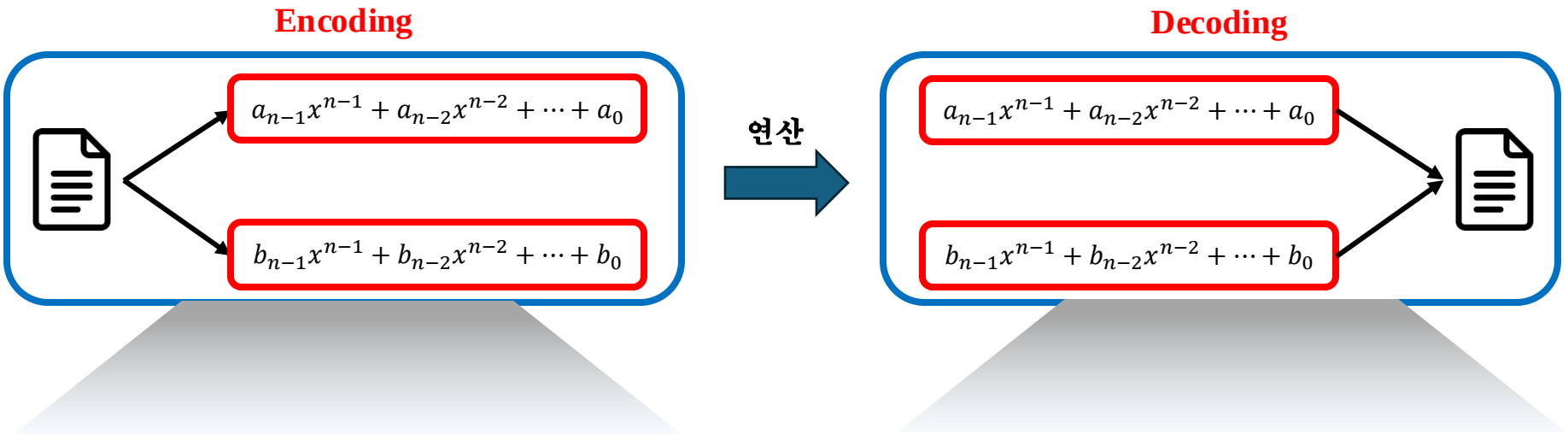
- KEM (Key Encapsulation Mechanism) 이란?
  - ✓ KEM 구조는 PKE(Public Key Encryption, 공개키 암호) 알고리즘을 사용한다.



Background

D2 Error Reconciliation

- D2 Encoding은 오류 조정(Error Reconciliation) 기법 중 하나로 주로 격자 기반에서 사용



✓ 메시지 bit 값에 맞춰 계수 할당

$c_{n-1}(= 1010 \ldots 110)$

a_{n-1} =0x7fff b_{n-1} = 0x7fff

연산

✓ 계수를 통해 메시지 bit 도출

$$a_{n-1} \longrightarrow a'_{n-1} = a_{n-1} - \frac{q}{2}$$

$$b_{n-1} \longrightarrow b'_{n-1} = b_{n-1} - \frac{q}{2}$$

$$msg\ bit = \begin{cases} 1 & if |a'_{n-1}| + |b'_{n-1}| \geq \frac{q}{2} \\ 0 & if |a'_{n-1}| + |b'_{n-1}| < \frac{q}{2} \end{cases}$$

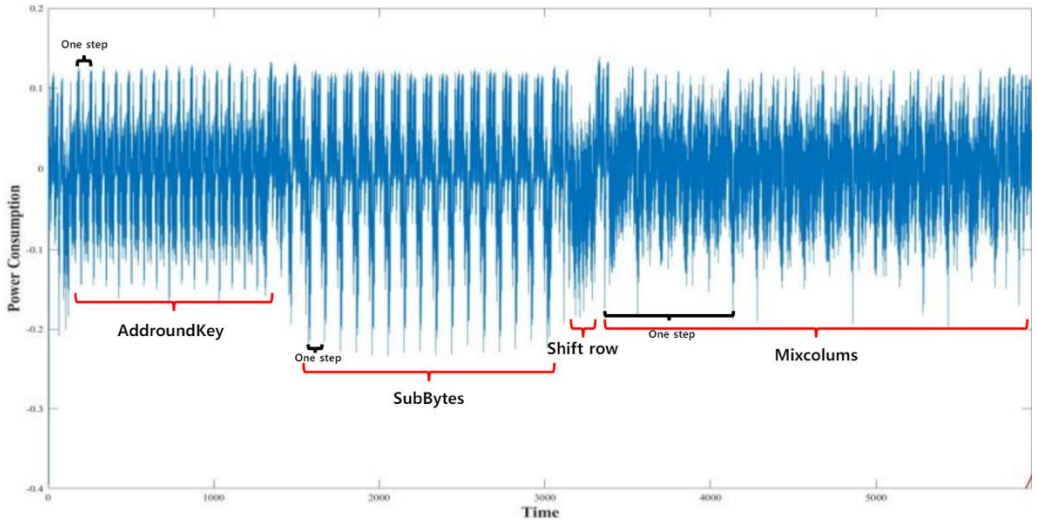
Background

SPA(단순전력분석) 개념

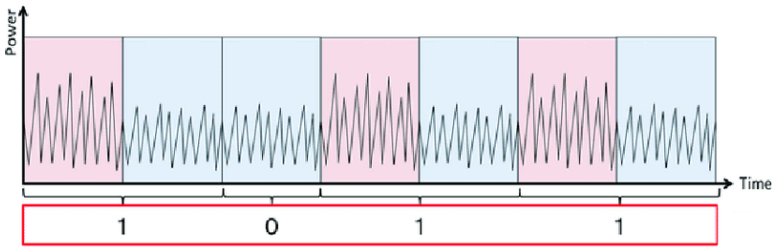
- SPA는 부채널분석 방법 중 소비전력 파형 기반으로 동작 과정을 분석하는 기법
- SPA 분석은 대개 코드 실행 개요를 파악할 때 사용
- AES 암호 알고리즘의 함수 구조, RSA 곱셈 연산 과정을 분석할 때 사용

HW*와 소비전력 개념

- HW는 주어진 값을 2진수로 표현했을 때의 1의 개수로 HW와 소비전력은 비례관계를 가짐



SPA를 통해 AES 구조 분석



SPA를 통해 RSA 곱셈 연산 분석

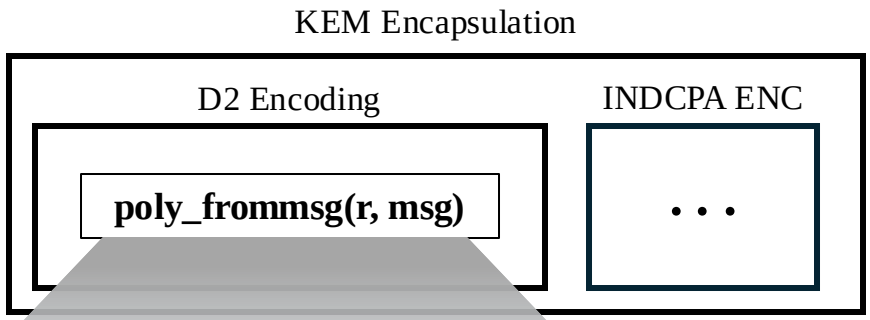
* HW(Hamming Weight)

Contents

Related Work

❖ TiMER 알고리즘에 대한 부채널 분석 선행 연구

- KpqC 1라운드(2022.11) 후보 TiGER에 대한 부채널 분석 선행 연구 수행
  - ✓ D2 Encoding의 부채널 분석




```
for (size_t j = 0; j < 8; j++) {
    mask = -((msg[i] >> j)) & 1;
    r->coeffs[8 * i + j] = mask & Modulus_Q/2;
    r->coeffs[8 * i + j + 128] = mask & Modulus_Q/2;
}
```

TiGER D2 Encoding Code

* mask 자료형은 unsigned int, 32bit

메시지 bit	mask 값	Hex 값
0	0	0x00000000
1	-1	0xffffffff

mask의 HW차이 = 소비전력 차이 = 32

Related Work

❖ TiMER 알고리즘에 대한 부채널 분석 선행 연구

- KpqC 1라운드(2022.11) 후보 TiGER에 대한 부채널 분석 선행 연구 수행
  - ✓ D2 Encoding의 부채널 분석

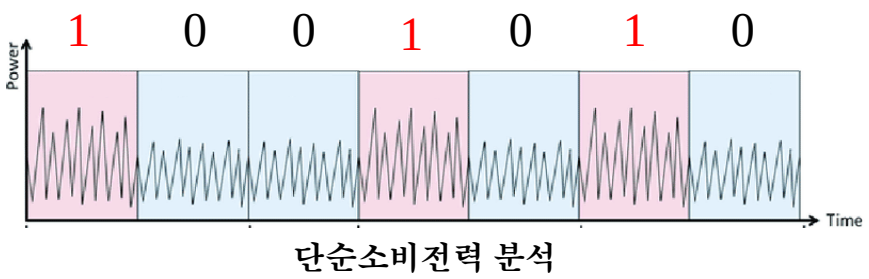
```
for (size_t j = 0; j < 8; j++) {
    mask = -((msg[i] >> j)) & 1;
                                Modulus_Q/2;
    r->coeffs[8 * i + j + 128] = mask & Modulus_Q/2;
}
```

TiGER D2 Encoding Code

* mask 자료형은 unsigned int, 32bit

메시지 bit	mask 값	Hex 값
0	0	0x00000000
1	-1	0xffffffff

mask의 HW차이 = 소비전력 차이 = 32



단순소비전력 분석

메시지 값 복구

* HW(Hamming Weight)

Related Work

❖ TiMER 알고리즘에 대한 부채널 분석 선행 연구

- KpqC 2라운드(2024.04) 진출 시 TiGER에 대한 부채널 대응기법 적용
  - ✓ D2 Encoding의 부채널 대응기법 적용

> in the Hamming weight of the *mask* variable in the D2 encoding process. This attack was complemented in TiGER v2.1 by changing the *mask* variable to 1 and 0 and applying a countermeasure to minimize the Hamming weight difference. TiMER also prevents such vulnerability with the same countermeasure.

D2 Encoding 공격에 대한 SMAUG-T 공식 문서

```
for (size_t j = 0; j < 8; j++) {
    mask = -((msg[i] >> j)) & 1;
    r->coeffs[8 * i + j] = mask & Modulus_Q/2;
    r->coeffs[8 * i + j + 128] = mask & Modulus_Q/2;
}
```

TiGER D2 Encoding

```
for (size_t j = 0; j < 8; j++) {
    mask = (msg[i] >> j) & 1;
    mask = (mask * Modulus_Q_2) & Modulus_Q_2;
    r->coeffs[8 * i + j] = mask;
    r->coeffs[8 * i + j + 128] = mask;
}
```

TiMER D2 Encoding

메시지 bit	mask 값	Hex 값
0	0	0x00000000
1	1	0x00000001

mask의 HW차이 = 소비전력 차이 = 1

Contents

Single Trace Analysis To TiMER

TiMER에 대한 단일파형 공격 제안

- 대응기법이 적용된 D2 Encoding에서도 소비전력 차이 발생

* 본 코드는 v4.0.1(2024.10) 버전이다.

```
for (size_t j = 0; j < 8; j++) {
①   mask = (msg[i] >> j) & 1;
②   mask = (mask * Modulus_Q_2) & Modulus_Q_2;
    r->coeffs[8 * i + j] = mask;
    r->coeffs[8 * i + j + 128] = mask;
}
```

→ 이전 부채널 공격에 대한 대응기법

→ D2 Reconciliation을 위한 mask 처리

TiMER D2 Encoding

```
#define Modulus_Q_2 32767 // for D2, Q/2
```

mask 처리에서 사용되는 Q/2

메시지 bit	mask ①	mask ②	Hex 값
0	0	0	0x00000000
1	1	32767	0x00007fff

→ **mask의 HW차이 = 소비전력 차이 = 16**

Single Trace Analysis To TiMER

❖ TiMER에 대한 단일파형 공격 제안

- 대응기법이 적용된 D2 Encoding에서도 소비전력 차이 발생

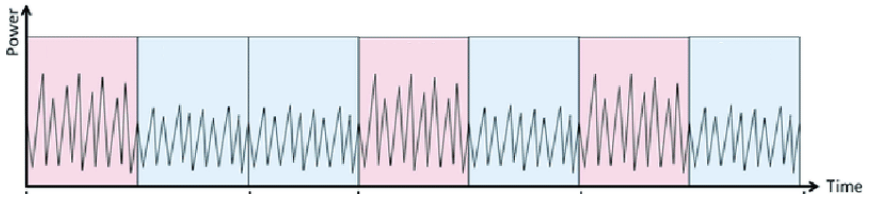
* 본 코드는 v4.0.1(2024.10) 버전이다.

```
for (size_t j = 0; j < 8; j++) {
    mask = (msg[i] >> j) & 1;
    mask = (mask * Modulus_Q_2) & Modulus_Q_2;
    r->coeffs[8 * i + j] = mask;
    r->coeffs[8 * i + j + 128] = mask;
}
```

TiMER D2 Encoding

Hex 값
0x00000000
0x00007fff

소비전력 차이 = 16



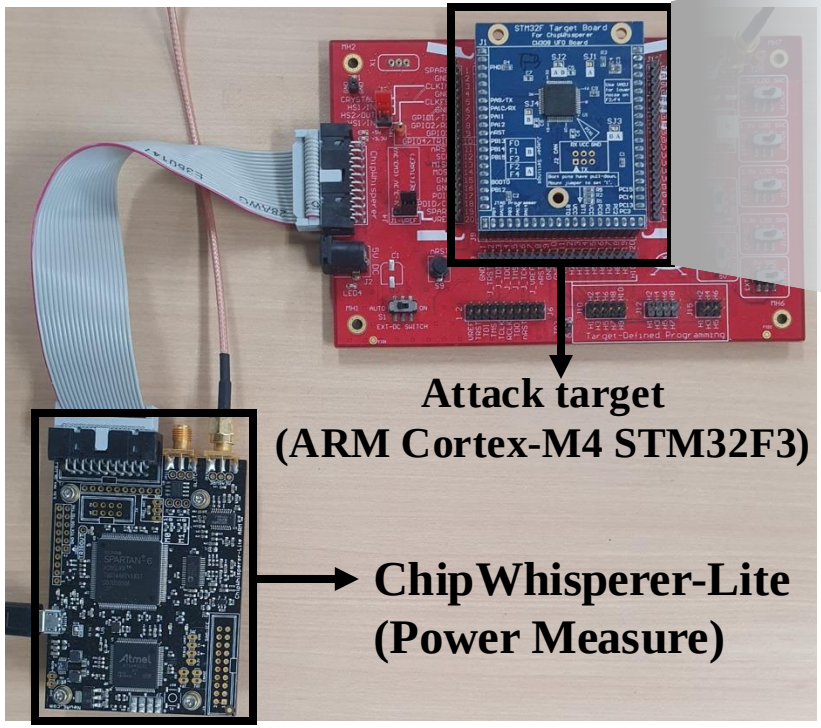
단순소비전력 분석

메시지 값 복구

Single Trace Analysis To TiMER

Experimental Environment

- KEM 구조에 대한 공격이기에 파형 1개만 수집

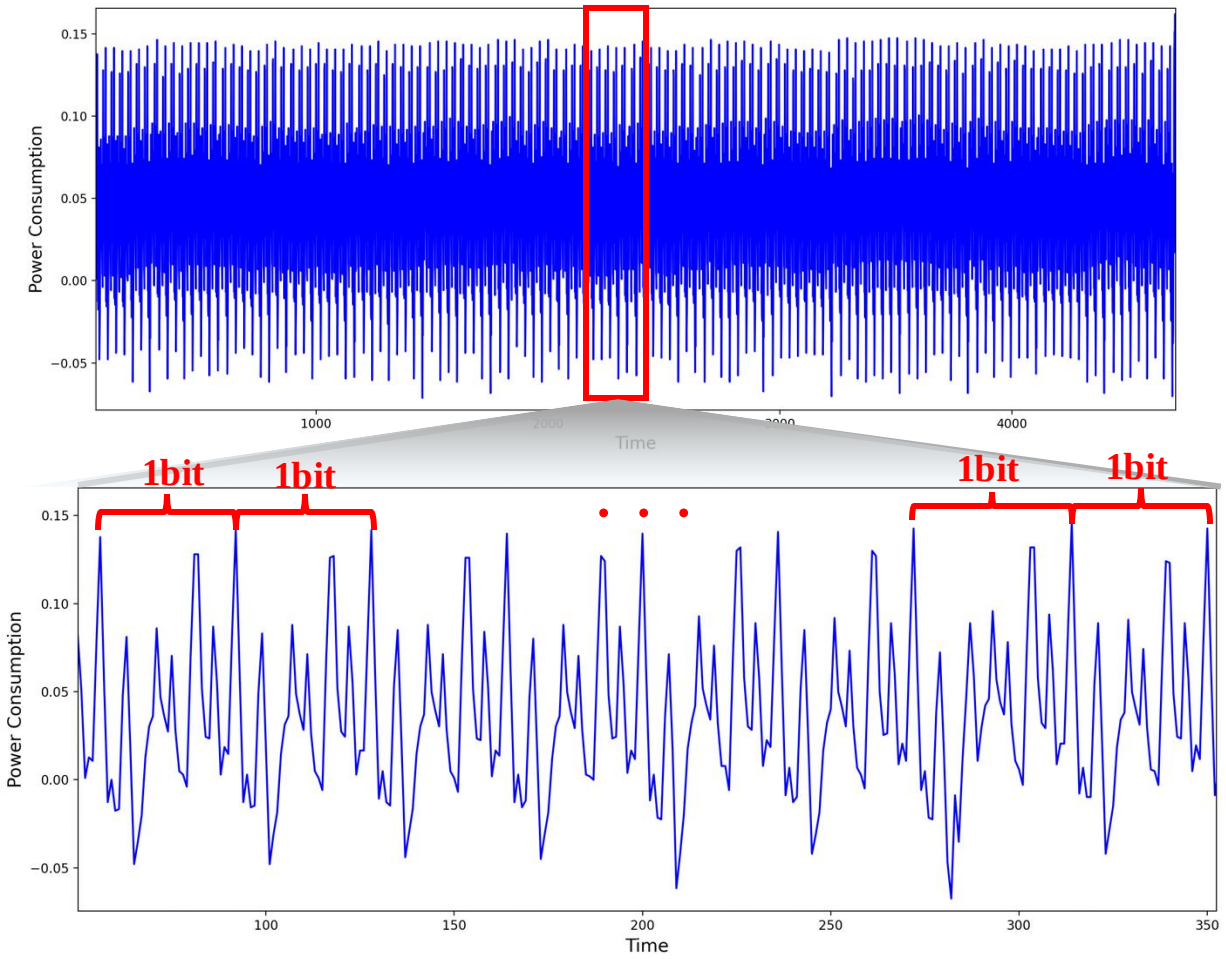



```
for (size_t j = 0; j < 8; j++) {
    mask = (msg[i] >> j) & 1;
    mask = (mask * Modulus_Q_2) & Modulus_Q_2;
    r->coeffs[8 * i + j] = mask;
    r->coeffs[8 * i + j + 128] = mask;
}
```

Single Trace Analysis To TiMER

16byte(128bit) 메시지 값에 대한 D2 Encoding 단일 파형 분석

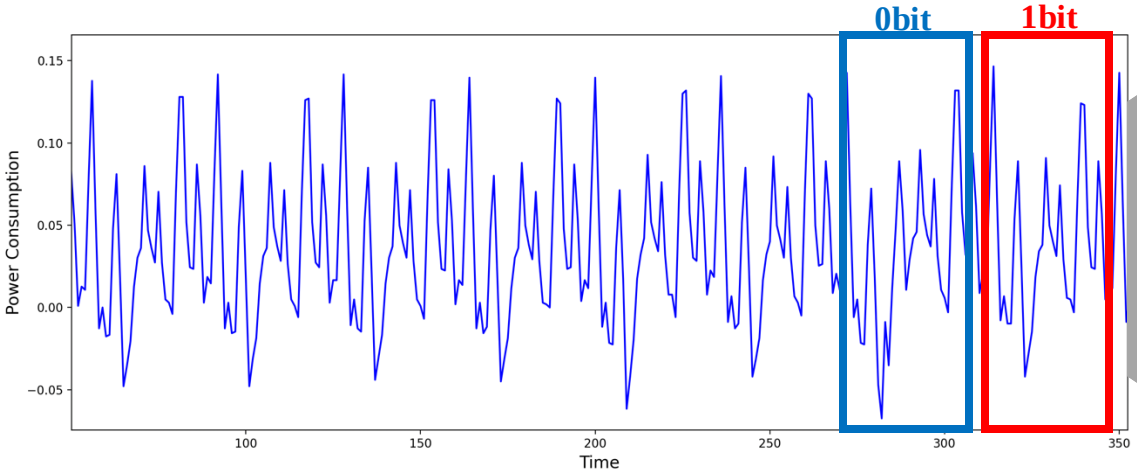
- 메시지 : 0xdf309b6905e8b9306aa43016aca4c54b

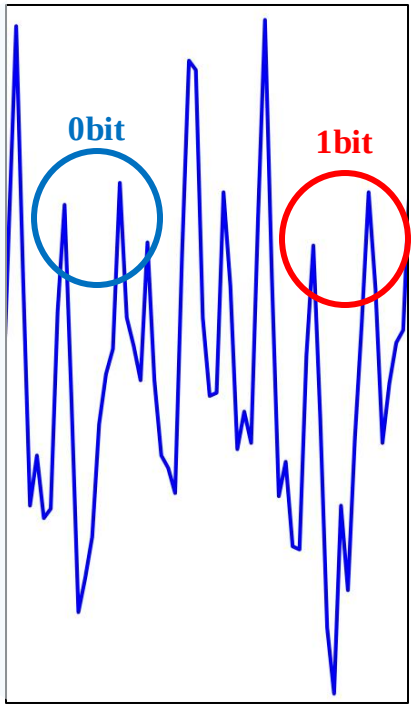


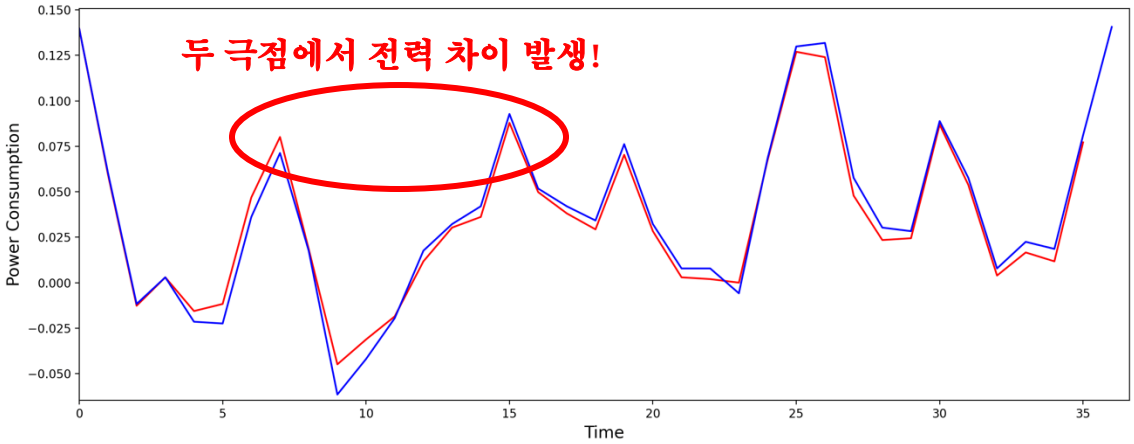
Single Trace Analysis To TiMER

16byte(128bit) 메시지 값에 대한 D2 Encoding 단일 파형 분석

- 메시지 0과 1bit의 파형(소비전력) 비교



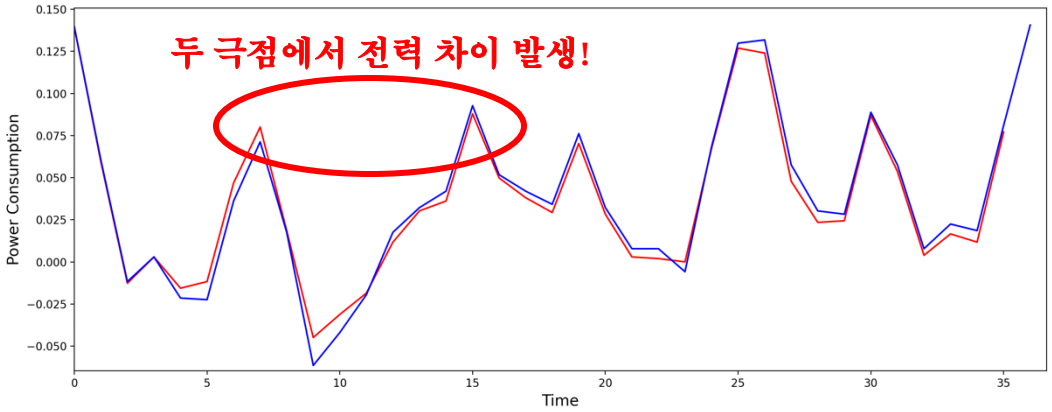




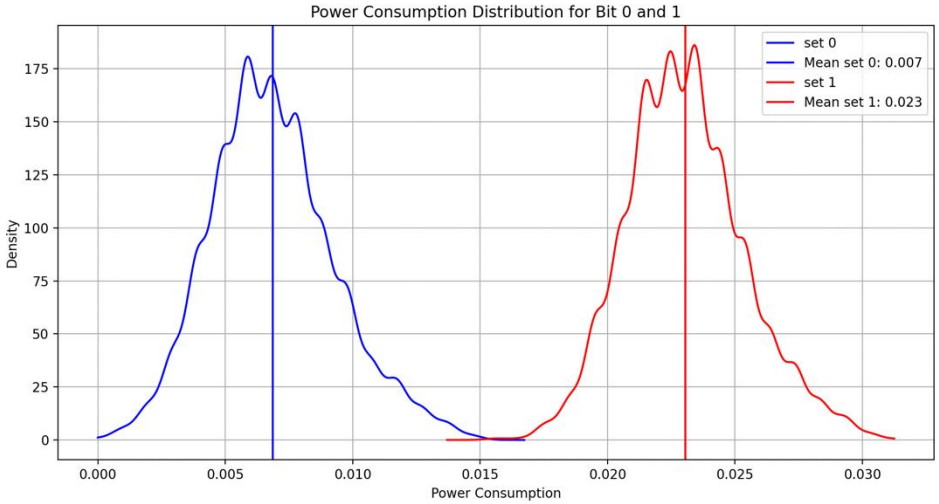
Single Trace Analysis To TiMER

16byte(128bit) 메시지 값에 대한 D2 Encoding 단일 파형 분석

- 0과 1의 대한 두 극점의 전력 차



bit	평균 전력 차
0	0.009765625
1	0.023590625



Single Trace Analysis To TiMER

16byte(128bit) 메시지 값에 대한 D2 Encoding 단일 파형 분석

- 극점의 평균 전력 차를 토대로 단일 파형 분석 결과 메시지 복구
  - ✓ 메시지 : 0xdf309b6905e8b9306aa43016aca4c54b

```
→ python3 exploit.py
1 byte : 0xdf
2 byte : 0x30
3 byte : 0x9b
4 byte : 0x69
5 byte : 0x5
6 byte : 0xe8
7 byte : 0xb9
8 byte : 0x30
9 byte : 0x6a
10 byte : 0xa4
11 byte : 0x30
12 byte : 0x16
13 byte : 0xac
14 byte : 0xa4
15 byte : 0xc5
16 byte : 0x4b
```

분석 결과

메시지 복구 성공!!

0xdf309b6905e8b9306aa43016aca4c54b

Contents

Conclusion

- KpqC Round 2 TiMER의 D2 Encoding에서 차분 식별을 통해 메시지 값을 복구하는 단일파형 공격 제안

- ARM Cortex-M4 MCU 환경에서 단일파형 공격 검증 실험 수행


Q & A