SICADA
SIde Channel Analysis Design Academy

PQC 부채널 분석

2019년 07월 17일 (금)
국민대학교
발표자 : 심보연

PKC (Public Key Cryptosystem)

RSA, ECC

Factoring and Discrete Logarithms

인증서 선택

Building e-world that you can trust INITECH

저장매체 선택

하드디스크 이동식(G:) 보안토큰 저장토큰 휴대폰

발급대상	발급자	구분	만료일자
[redacted]	금융결제원	은행/신용...	2015-03-23

인증서 보기 인증서 찾기 인증서 삭제

인증서 암호는 대소문자를 구분합니다.

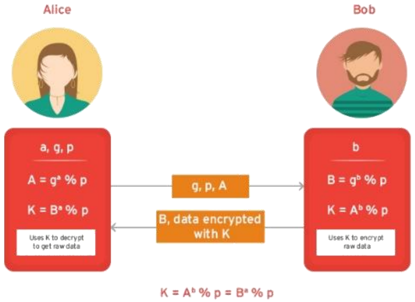
인증서 암호

확인 취소

Certificate

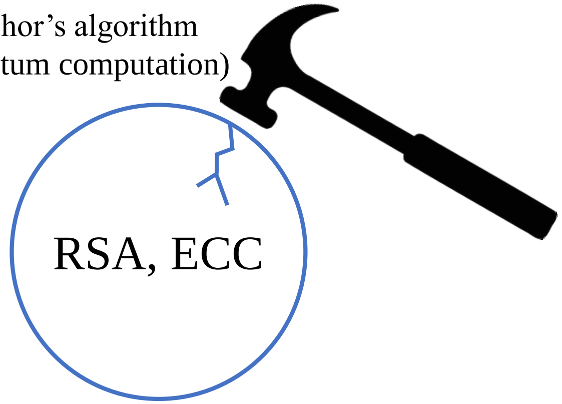
Smart Car

e-Passport



Key Exchange

PKC (Public Key Cryptosystem)



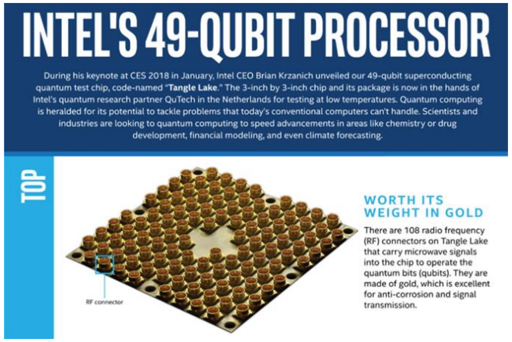
1994 Shor's algorithm
(for quantum computation)

RSA, ECC

Quantum
Computer

estimated to arrive in the next 10 to 15 years

Factoring and Discrete Logarithms



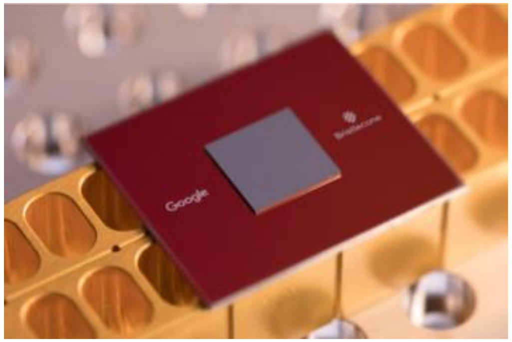
INTEL'S 49-QUBIT PROCESSOR

During his keynote at CES 2018 in January, Intel CEO Brian Krzanich unveiled our 49-qubit superconducting quantum test chip, code-named "**Tangle Lake**." The 3-inch by 3-inch chip and its package is now in the hands of Intel's quantum research partner QuTech in the Netherlands for testing at low temperatures. Quantum computing is heralded for its potential to tackle problems that today's conventional computers can't handle. Scientists and industries are looking to quantum computing to speed advancements in areas like chemistry or drug development, financial modeling, and even climate forecasting.

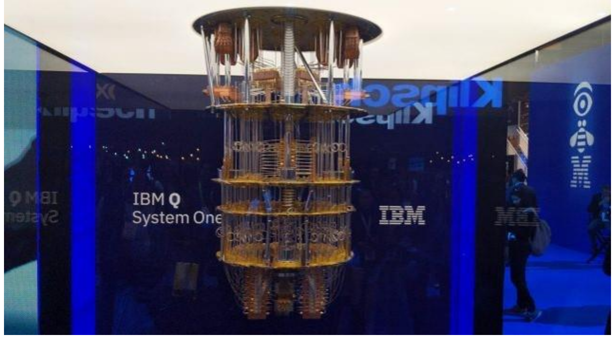
WORTH ITS WEIGHT IN GOLD

There are 108 radio frequency (RF) connectors on Tangle Lake that carry microwave signals into the chip to operate the quantum bits (qubits). They are made of gold, which is excellent for anti-corrosion and signal transmission.

49-qubit chip
"Tangle-Lake"
January 2018



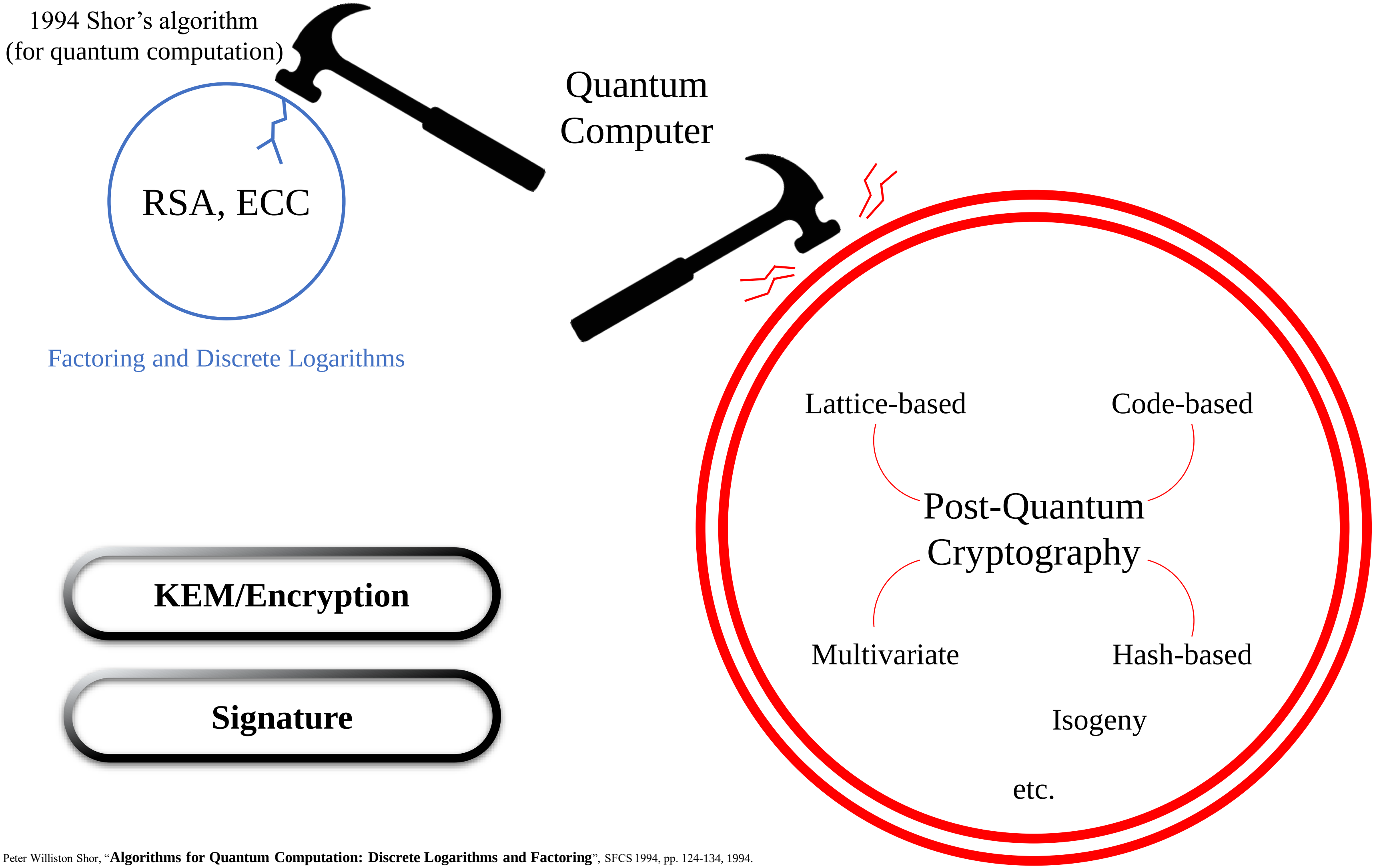
72-qubit chip
"Bristlecone"
March 2018



53-qubit quantum computer
"Q System One"
January 2019

[1] Peter Williston Shor, "**Algorithms for Quantum Computation: Discrete Logarithms and Factoring**", SFCS 1994, pp. 124-134, 1994.

PKC (Public Key Cryptosystem)



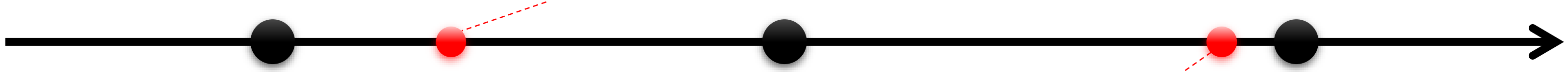
KEM/Encryption

Signature

[1] Peter Williston Shor, "**Algorithms for Quantum Computation: Discrete Logarithms and Factoring**", SFCS 1994, pp. 124-134, 1994.

PKC (Public Key Cryptosystem)

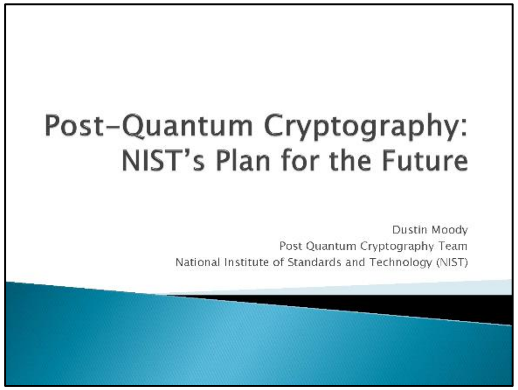
- 2020/2021 : Select algorithms or start a 3rd Round
- 2022-2024 : Draft standards available



Dec 20, 2016
Formal Call for Proposals

February 24-26, 2016

PQCrypto 2016



Post-Quantum Cryptography:
NIST's Plan for the Future

Dustin Moody
Post Quantum Cryptography Team
National Institute of Standards and Technology (NIST)

April 11-13, 2018

NIST First PQC Standardization Conference

co-located with

PQCrypto 2018

August 22-24, 2019

NIST Second PQC Standardization Conference

co-located with

January 30, 2019
Second Round Candidates announced
(26 algorithms)

NIST Round 2 Candidates

KEM/Encryption

(9) Lattice-based	
FrodoKEM	LWE
LAC	RLWE
NewHope	RLWE
Round5	LWR/RLWR
Crystals-Kyber	MLWE
Saber	MLWR
Three Bears	IMLWE
NTRU	NTRU
NTRU Prime	NTRU

(7) Code-based	
Classic McEliece	Goppa
NTS-KEM	Goppa
BIKE	Short Hamming
HQC	Short Hamming
LEDAcrypt	Short Hamming
RQC	Low rank
ROLLO	Low rank

(1) Isogeny	
SIKE	Isogeny

Signature

(3) Lattice-based	
qTESLA	Fiat-Shamir
Crystals-Dilithium	Fiat-Shamir
FALCON	Hash then sign

(4) Multivariate	
GeMMS	HFE
LUOV	UOV
Rainbow	UOV
MQDSS	Fiat-Shamir

(2) Symmetric-based	
SPHINCS+	Hash
Picnic	ZKP

https://csrc.nist.gov/CSRC/media/Presentations/Round-2-of-the-NIST-PQC-Competition-What-was-NIST/images-media/pqcrypto-may2019-moody.pdf

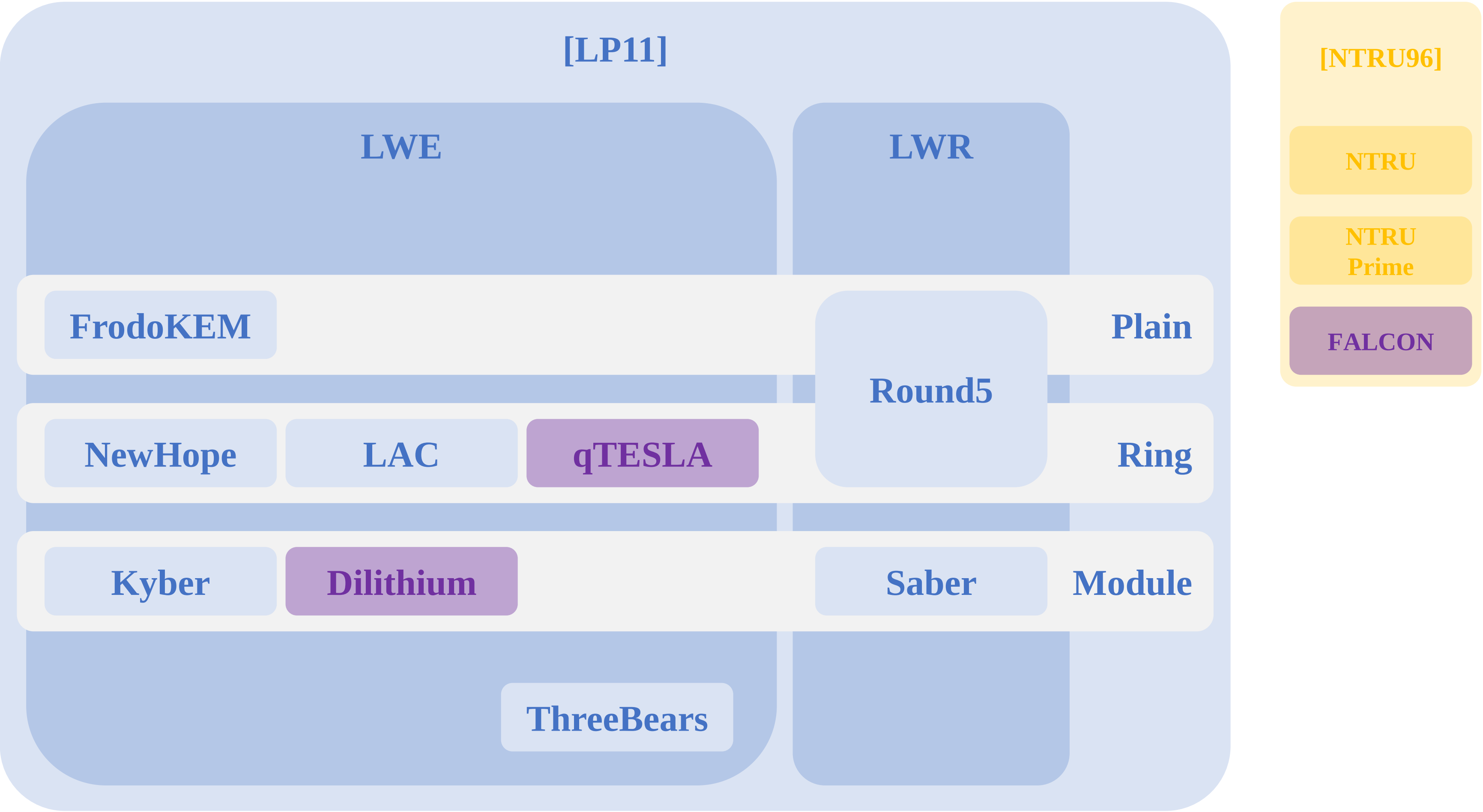
NIST PQC Standardization

The selection criteria

- **Security**
  - ✓ against both classical and quantum attacks
- **Performance**
  - ✓ measured on various “classical” platforms
- **Other properties**
  - ✓ Drop-in replacements – Compatibility with existing protocols and networks
  - ✓ Perfect forward secrecy
  - ✓ Resistance to side-channel attacks
  - ✓ Simplicity and flexibility
  - ✓ Misuse resistance

https://csrc.nist.gov/CSRC/media/Presentations/Round-2-of-the-NIST-PQC-Competition-What-was-NIST/images-media/pqcrypto-may2019-moody.pdf

NIST Round 2 Candidates : Lattice-based



🟪 : Digital Signature 🟦🟨 : KEM/PKE

[LP11] Richard Lindner and Chris Peikert, “**Better Key Sizes (and Attacks) for LWE-Based Encryption**”, CT-RSA 2011, pp. 319-339.

[NTRU96] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman, “**NTRU: A Ring-Based Public Key Cryptosystem**”, ANTS 1998, pp.267-288.

※ Convolution product = Polynomial multiplication

■ SCA on KEM/PKE

	공격 대상	공격 연산	공격 유형	대응 기법
[CT-RSA 2007]	NTRU	Decryption (Difference in #hash calls)	TA	Fixed hash calls
[RFID Security 2008]	NTRU	Decryption (Polynomial multiplication)	DPA	
[IEICE 2010]	NTRU	Decryption (Polynomial multiplication)	SPA, CPA SOCPA	Masking, Blinding
FA [IEICE 2011]	NTRU	Protected Decryption [IEICE 2011] (Polynomial multiplication)	CA	
FA [Cryptography and Communications 2012]	NTRU	Protected Decryption [IEICE 2011] (Polynomial multiplication)	Chosen Ciphertext DPA	Random delay, Masking, Insert dummy
[Microprocessors and Microsystems 2013]	Knuth-Yao Sampler	Bit scanning	[illegible]	Shuffling
[TIIS 2013]	Ring-LWE scheme	Decryption (Multiplication, addition)	CPA	Masking
[ePrint 2014]	Ring-LWE scheme	Decryption (Addition)	Chosen Ciphertext SPA	
[CHES 2015]	Ring-LWE scheme	Decryption (Multiplication, addition)	CPA	Masking
[AsianHOST 2016]	Ring-LWE scheme	Decryption (Multiplication, addition)	CPA	Additively Homomorphic Masking
[J. Cryptographic Engineering 2016]	Ring-LWE scheme	Protected Decryption (masking) (NTT)	TA^P	Shuffling
[PQCrypto 2016]	NTRU	Decryption (Polynomial multiplication)	SPA, CPA, SOCPA, CA	Random Key Rotation
[CHES 2017]	FrodoKEM, Lizard	Constant-time CDT sampler	SPA	Look-up table based
[Computers and Electrical Engineering 2017]	NTRU (NIST Round 1)	Decryption (Polynomial multiplication)	CA	Constant-time, Initialization with a random
[Applied Science 2018]				
[Applied Science 2018]				

Polynomial multiplication/addition & Sampler

SCA on KEM/PKE

※ Convolution product = Polynomial multiplication

	공격 대상	공격 연산	공격 유형	대응 기법
[DATE 2018]	Binary Ring-LWE scheme	Decryption (Polynomial multiplication)	SPA, DPA	Dummy and memory update, Masking
survey [GLSVLS 2018]				
FA [IEEE Trans. on Computer 2018]				
[HOST 2018]	FrodoKEM, NewHope	Encryption (Multiplication)	Horizontal CPA	Shuffling, Insert dummy
[SAC 2018]	FrodoKEM	Encryption (Multiplication)	TA^P	Shuffling, reduce algorithmic variance
CBA [TCHES 2018]				
FA [TCHES 2018]				
[CT-RSA 2019]	NewHope	Decryption (Key mismatch oracle)	Key reuse based SPA, FA	
[Latincrypt 2019]	Ring-LWE scheme	Protected Decryption (masking) (NTT)	TA^P	Masking
[TIS 2019]	LAC, Ramstake	Decryption (Decoding of error correcting code)	Chosen Ciphertext TA	
[ePrint 2020]	Frodo, LAC, Round5, NTRU-HPS		TA^P(SAC 2018) + Algebraic	
[PQCrypto 2020]	NewHope	Encryption (Message Encoding)	SPA, TA^P	Masking + Shuffling
[TCHES 2020]	NTRU Prime	Decryption (Polynomial Multiplication)	SPA, Chosen-input SPA, Online TA^P, Vertical CPA, Horizontal In-Depth CPA	(Masking, Blinding) + Shuffling
[TCHES 2020]	Round5, LAC, Kyber, NewHope, Saber, FrodoKEM	Decryption (FO transform, decoding ECC)	Chosen Ciphertext SPA	

[HOST 2018], [SAC 2018]

❖ Attack Scenario

Generate $seed$

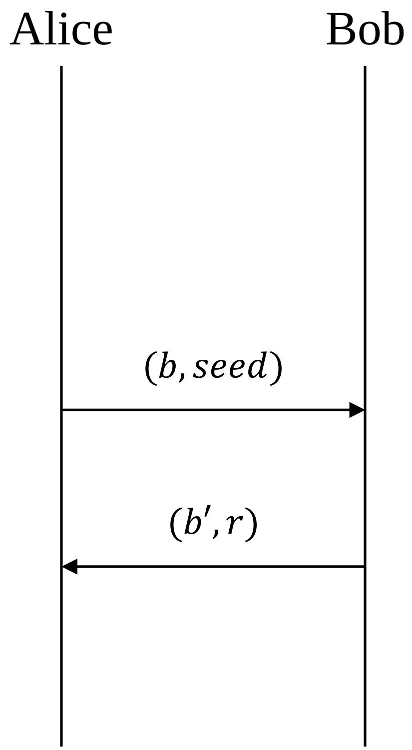
$a \leftarrow GenA(seed)$

$s, e \leftarrow sample()$

$b = as + e$

Public $(b, seed)$, private (s)

Single-Trace Attack



$v' = b's$

$V = Rec(v', r)$

$s', e', e'' \leftarrow sample()$

$a \leftarrow GenA(seed)$

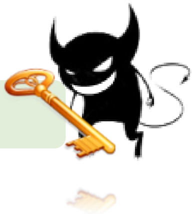
$b' = as' + e'$

$v = bs' + e''$

$r = sample'(v)$

$V = Rec(v, r)$

$ss_{key} = SHA256(V)$ ss_{key} : shared secret key $ss_{key} = SHA256(V)$



---

[1] Aydin Aysu, Youssef Tobah, Mohit Tiwari, Andreas Gerstlauer, and Michael Orshansky, "**Horizontal Side-Channel Vulnerabilities of Post-Quantum Key Exchange Protocols**", HOST 2018, pp. 81-88, April, 2018.

[2] Joppe W. Bos, Simon Friedberger, Marco Martinoli, Elisabeth Oswald, and Martijn Stam, "**Assessing the Feasibility of Single Trace Power Analysis of Frodo**", SAC 2018, pp. 261-234, August, 2018.

■ [HOST 2018], [SAC 2018]

- [HOST 2018] Horizontal Side-Channel Vulnerabilities of Post-Quantum Key Exchange Protocols
- [SAC 2018] Assessing the feasibility of single trace power analysis of Frodo

	[HOST 2018]	[SAC 2018]
Target Scheme	FrodoKEM, NewHope	FrodoKEM
Target Operation	Matrix Multiplication (AS) Schoolbook Multiplication with Barret reduction (as)	Matrix-vector Multiplication (As)
Implementation	Hardware (SAKURA-G)	Software (ELMO simulation – ARM Cortex M0)
Power Model	Hamming Distance	Hamming Weight
Attack Type	Horizontal Correlation Power Analysis	Divide-and-Conquer Template Attack Extend-and-Prune Template Attack
# Traces	Single-Trace	Single-Trace
Countermeasure	Shuffling, Insert dummy	Shuffling / reduce algorithmic variance

- [ePrint 2020] LWE with Side Information; Attacks and Concrete Security Estimation

SCA on KEM/PKE

※ Convolution product = Polynomial multiplication

	공격 대상	공격 연산	공격 유형	대응 기법
[DATE 2018]	Binary Ring-LWE scheme	Decryption (Polynomial multiplication)	SPA, DPA	Dummy and memory update, Masking
survey [GLSVLS 2018]				
FA [IEEE Trans. on Computer 2018]				
[HOST 2018]	FrodoKEM, NewHope	Encryption (Multiplication)	Horizontal CPA	Shuffling, Insert dummy
[SAC 2018]	FrodoKEM	Encryption (Multiplication)	TA^P	Shuffling, reduce algorithmic variance
CBA [TCHES 2018]				
FA [TCHES 2018]				
[CT-RSA 2019]	NewHope	Decryption (Key mismatch oracle)	Key reuse based SPA, FA	
[Latincrypt 2019]	Ring-LWE scheme	Protected Decryption (masking) (NTT)	TA^P	Masking
[TIS 2019]	LAC, Ramstake	Decryption (Decoding of error correcting code)	Chosen Ciphertext TA	
[ePrint 2020]	Frodo, LAC, Round5, NTRU-HPS		TA^P(SAC 2018) + Algebraic	
[PQCrypto 2020]	NewHope	Encryption (Message Encoding)	SPA, TA^P	Masking + Shuffling
[TCHES 2020]	NTRU Prime	Decryption (Polynomial Multiplication)	SPA, Chosen-input SPA, Online TA^P, Vertical CPA, Horizontal In-Depth CPA	(Masking, Blinding) + Shuffling
[TCHES 2020]	Round5, LAC, Kyber, NewHope, Saber, FrodoKEM	Decryption (FO transform, decoding ECC)	Chosen Ciphertext SPA	

▣ [PQCrypto 2020] Defeating NewHope with a Single-Trace

❖ Attack Scenario

μ : secret message

$a \leftarrow GenA(seed)$

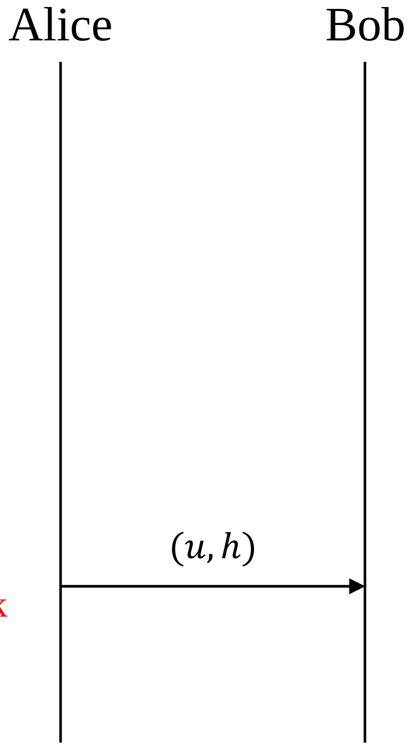
$s', e', e'' \leftarrow sample()$

$u \leftarrow as' + e'$

$v \leftarrow bs' + e'' + Encode(\mu)$

$h \leftarrow Compress(v)$

Single-Trace Attack



NewHope : Key Generation	
Input	
Output	Public key $(b, seed)$, private key (s)
1. Generate $seed$ 2. $a \leftarrow GenA\ (seed)$ 3. $s, e \leftarrow sample()$ 4. $b = as + e$ 5. Return $(b, seed)$, (s)	

$v \leftarrow Decompress(h)$

$\mu \leftarrow Decode(v - us)$

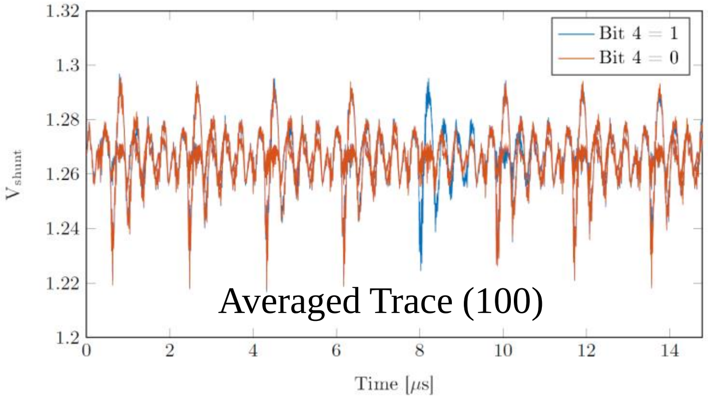
$ss_{key} = SHAKE256(\mu)$ ss_{key} : shared secret key $ss_{key} = SHAKE256(\mu)$

[3] Dorian Amiet, Andreas Curiger, Lukas Leuenberger, and Paul Zbinden, "**Defeating NewHope with a Single Trace**", PQCrypto 2020, pp. 189-205, April, 2020.

■ [PQCrypto 2020] Defeating NewHope with a Single-Trace

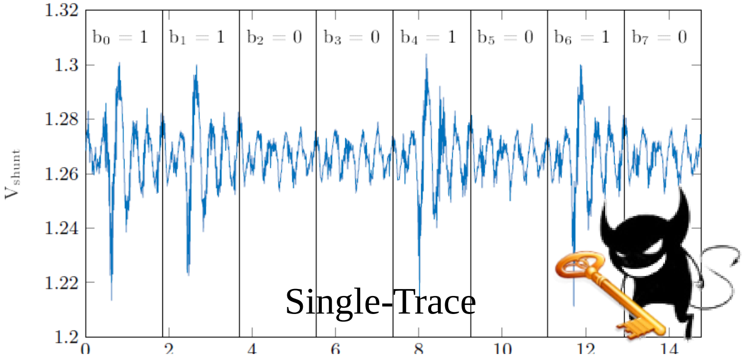
펄스 없으면
공격 성공률 100%

Optimization Level 0



Averaged Trace (100)

Fig. 2. Measurement traces on top of each other. Every trace is 100 times averaged. Code compiled with optimization disabled.

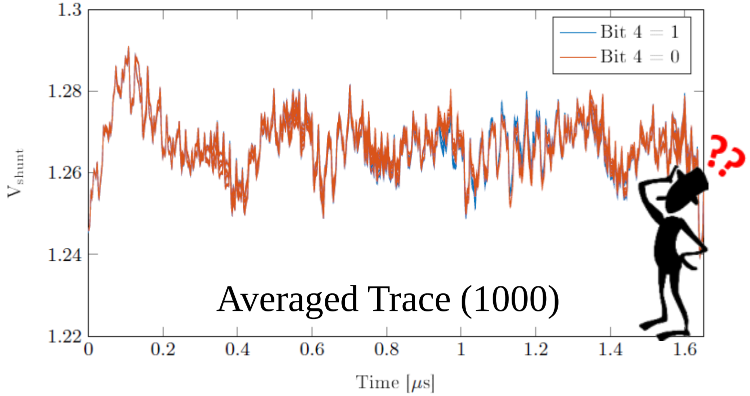


Single-Trace

47% 펄스 → 후처리 없어도 99.5%

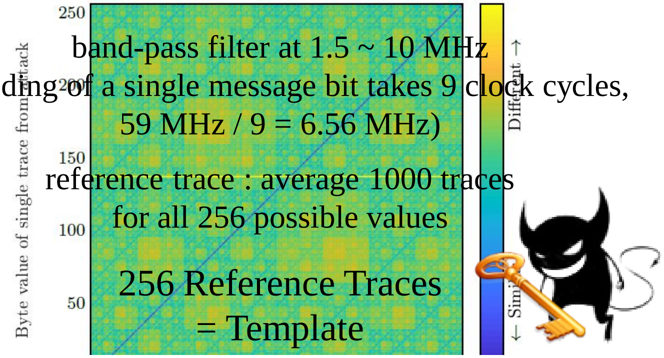
Fig. 3. A single trace measurement where message byte 1 is set to the value 83 (binary 0101 0011). Code compiled with optimization disabled.

Optimization Level 3



Averaged Trace (1000)

Fig. 4. All measurement traces on top of each other. Every trace is 1000 times averaged. Code compiled with optimization enabled (O3).



band-pass filter at 1.5 ~ 10 MHz
(encoding of a single message bit takes 9 clock cycles,
59 MHz / 9 = 6.56 MHz)

reference trace : average 1000 traces
for all 256 possible values

256 Reference Traces
= Template

4% 펄스 → 2-byte 전수조사 → > 99%

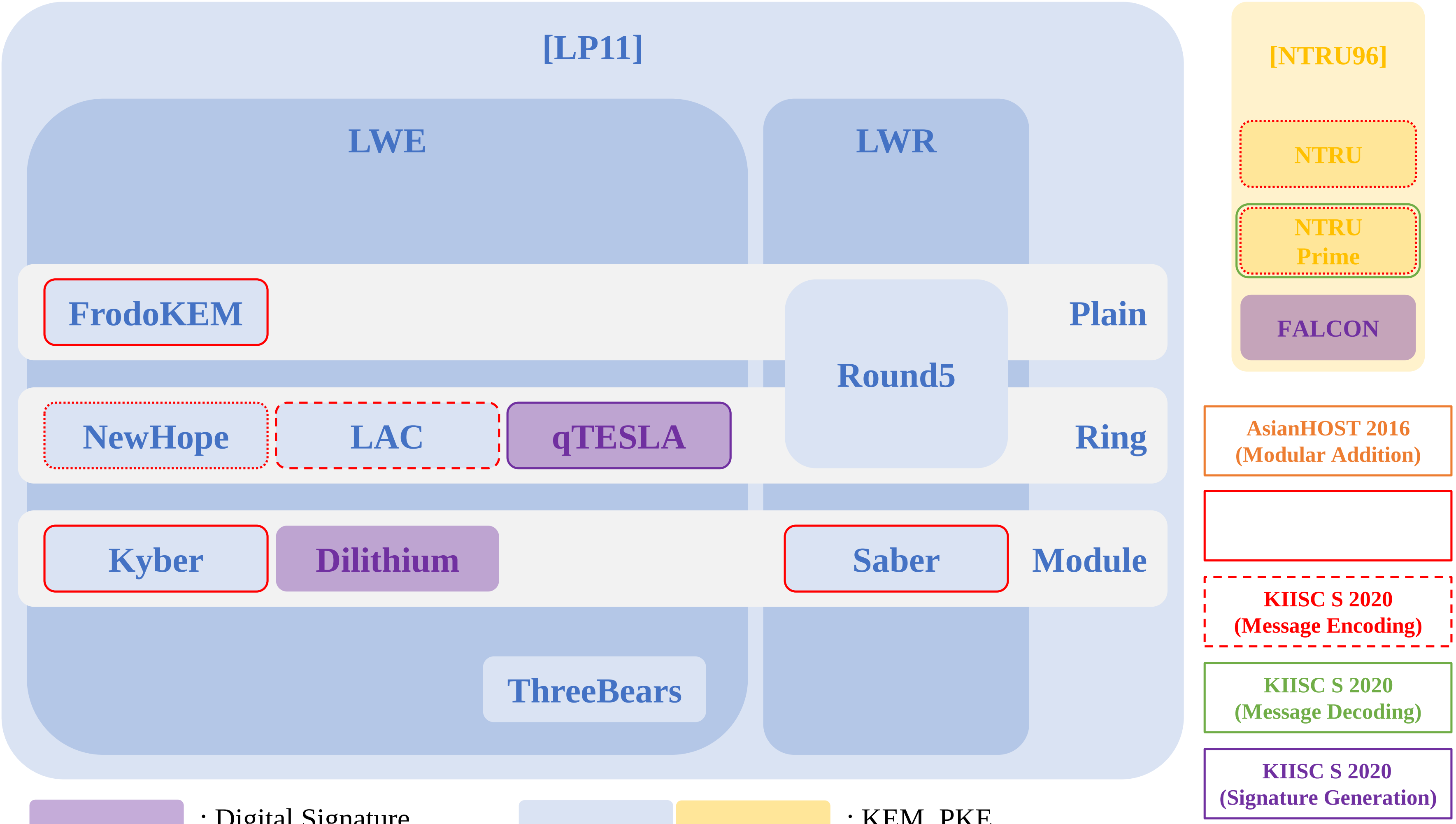
Fig. 5. Similarity between a single power trace compared to the reference traces.

SCA on Signature

	공격 대상	공격 연산	공격 유형	대응 기법
[Cryptography and Communications 2012]	NTRUSign	Sign Generation	FA	
[CHES 2016]	BLISS	Gaussian Sampling, Rejection Sampling	CA	
[FDTC 2016]	BLISS, ring-TESLA, GLP	Key Generation, Sign Generation, Verification	FA	
[SAC 2016]	BLISS, TESLA, GLP, GPV	Gaussian Sampling	CA	
[ICAR 2016]	BLISS	Polynomial multiplication, Gaussian Sampling		Blinding
[ePrint 2017]	BLISS	Gaussian Sampling	SPA, TA	
[SAC 2017]	BLISS	Rejection Sampling, Gaussian Sampling, Polynomial multiplication	SPA, SEMA, BTA	
[SAC 2017]	BLISS	Gaussian Sampling	CA	
[IACR 2018]	Dilithium, qTESLA	Sign Generation	FA	
[EUROCRYPT 2018]	GLP			Masking
[ICAR 2019]	BLISS	Rejection Sampling	TA	
[ePrint 2019]	Dilithium			Masking
[CARDIS 2019]	qTESLA			Masking

Polynomial multiplication & Sampler

NIST Round 2 Lattice



🟪 : Digital Signature 🟦 🟨 : KEM, PKE

[LP11] Richard Lindner and Chris Peikert, "**Better Key Sizes (and Attacks) for LWE-Based Encryption**", CT-RSA 2011, pp. 319-339.

[NTRU96] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman, "**NTRU: A Ring-Based Public Key Cryptosystem**", ANTS 1998, pp.267-288.

Our results

Conference	Title / Description
AsianHOST 2016	Chosen ciphertext Simple Power Analysis on software 8-bit implementation of ring-LWE encryption
KIISC S2020	NIST Round 2 후보 LAC Key Encapsulation Mechanism에 대한 신규 단일 파형 공격
	LAC / Message Encoding
	NIST Round 2 후보 격자 기반 KEM NTRU LPRime에 대한 신규 단일 파형 공격
	NTRU LPRime / Message Decoding
	NIST Round 2 후보 마스킹된 qTESLA 전자서명 알고리즘에 대한 단일 파형 공격
	qTESLA / Signature Generation

Our results 1 : AsianHOST 2016

Chosen ciphertext SPA attack on ring-LWE encryption scheme

🟥 : Private key

🟩 : Public key

Attack Scenario

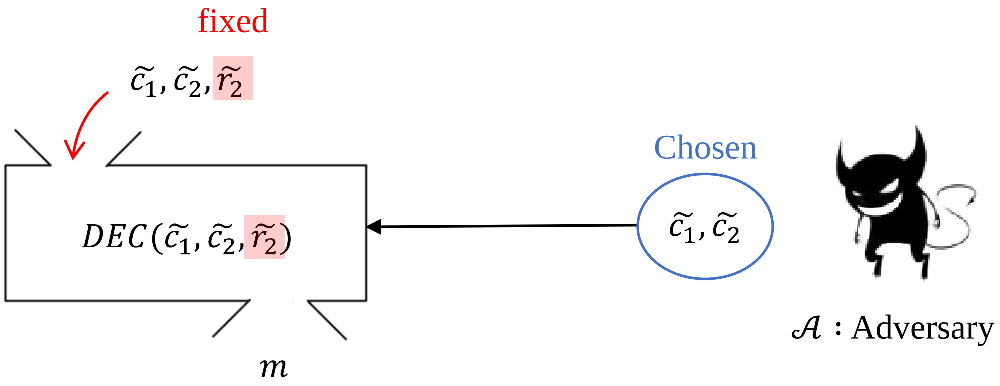
Algorithm. Pseudo-code of Decryption

Input	$(\tilde{c}_1, \tilde{c}_2)$, $(\tilde{r}_2)$
Output	Message m

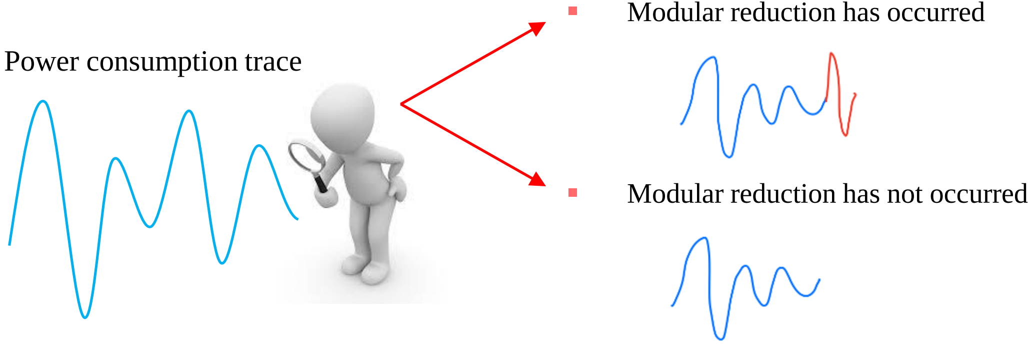
```
for $i = 0$ to $n-1$ do
    $TC_1[i] \leftarrow \tilde{c}_1[i] \times \tilde{r}_2[i]$
    if $TC_1[i] \geq q$ then
        $TC_1[i] \leftarrow TC_1[i]\ mod\ q$
    end if
    $M'[i] \leftarrow TC_1[i] + \tilde{c}_2[i]$
    if $M'[i] \geq q$ then
        $m'[i] \leftarrow M'[i]\ mod\ q$
    else
        $m'[i] \leftarrow M'[i]$
    end if
    $m' \leftarrow \mathbf{INTT}(m')$
    $m \leftarrow \mathbf{Decode}(m')$
end for
```

distinguishable

- The adversary can select different ciphertexts (= chosen ciphertexts)

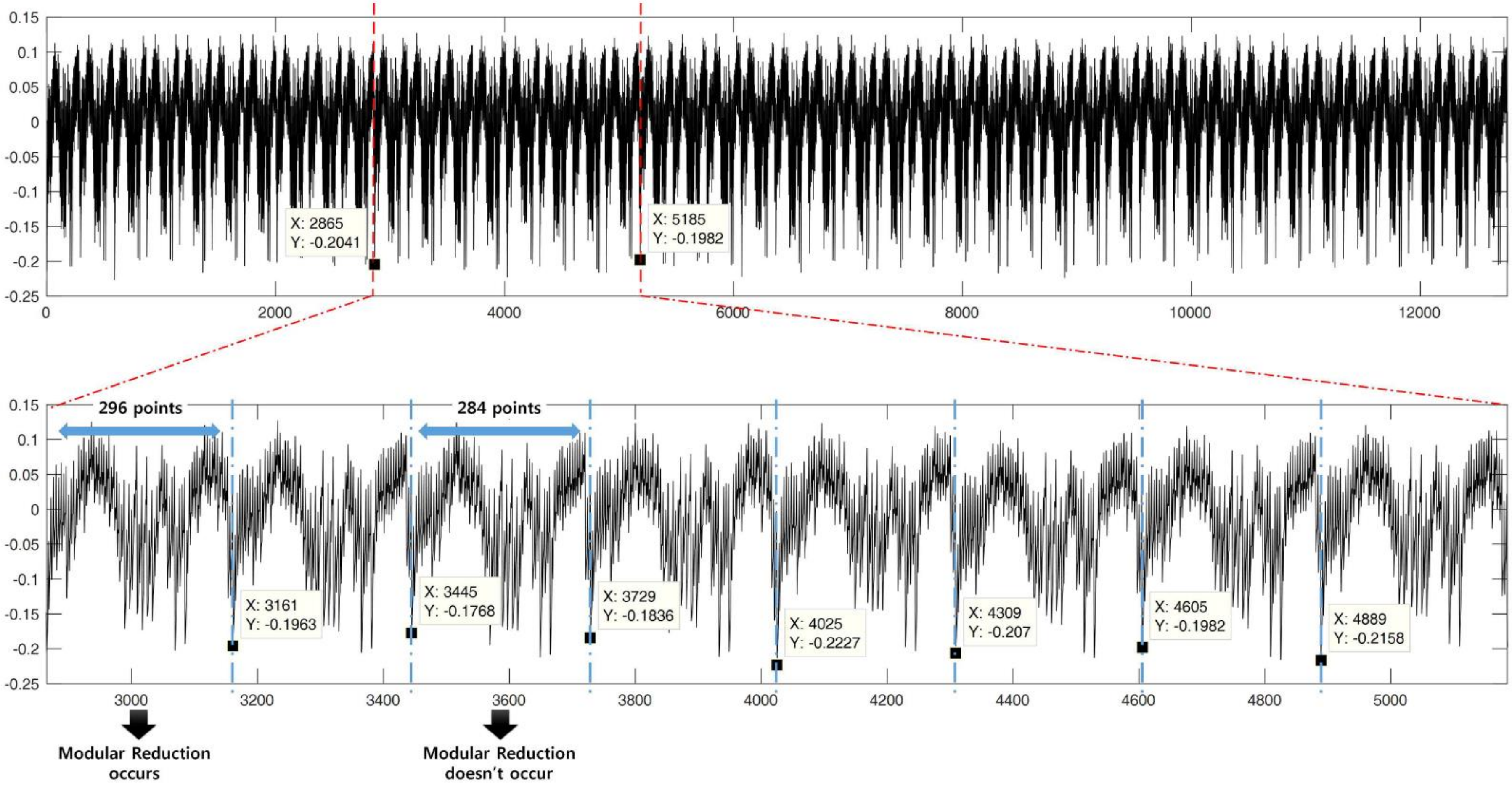


- The adversary can measure their power consumption during decryption operation on 8-bit processor



Our results 1 : AsianHOST 2016

▣ Is it possible to distinguish whether a modular operation occurs or not?

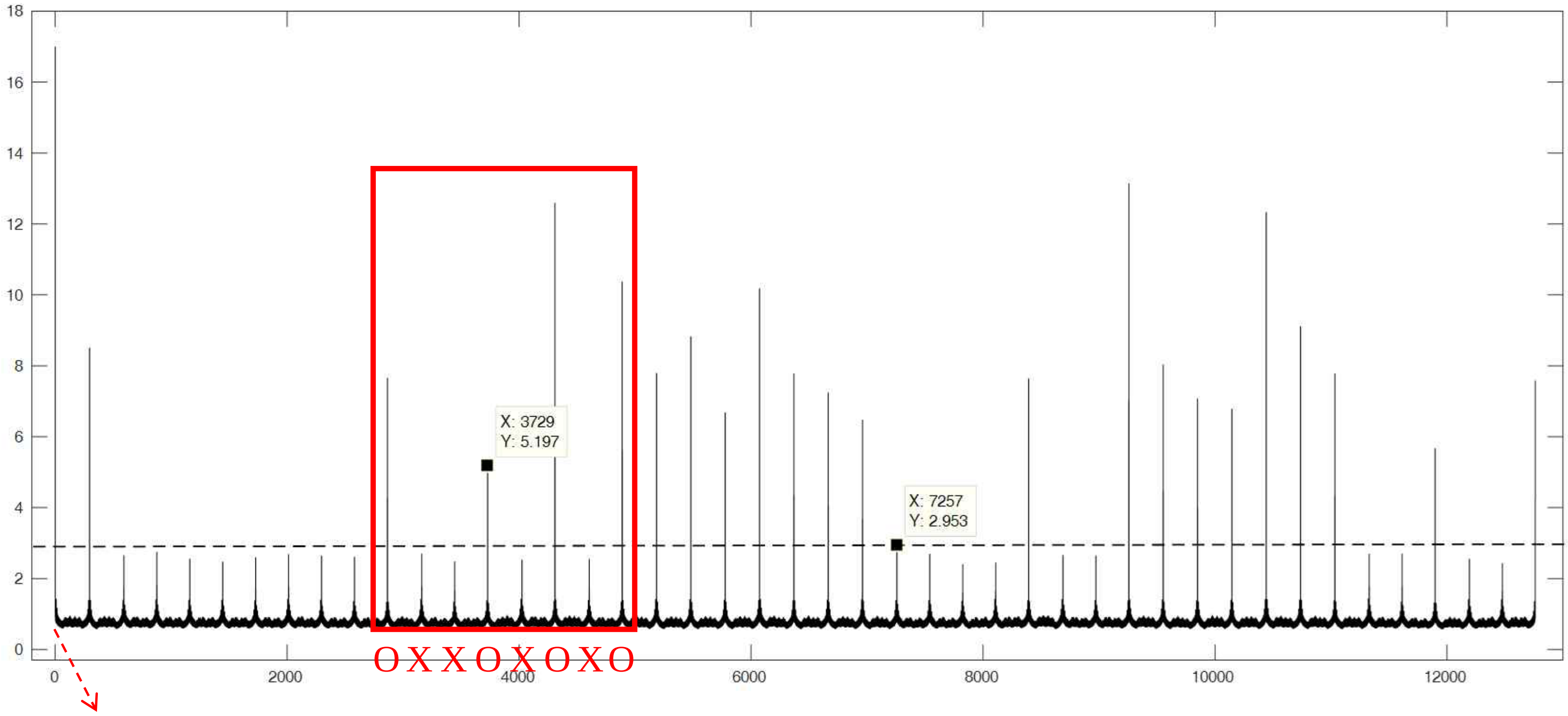


O X X O X O X O

Our results 1 : AsianHOST 2016

▣ Is it possible to distinguish whether a modular operation occurs or not?

❖ PCDC $(X, Y) = \sigma(X)/\sigma(X-Y)$

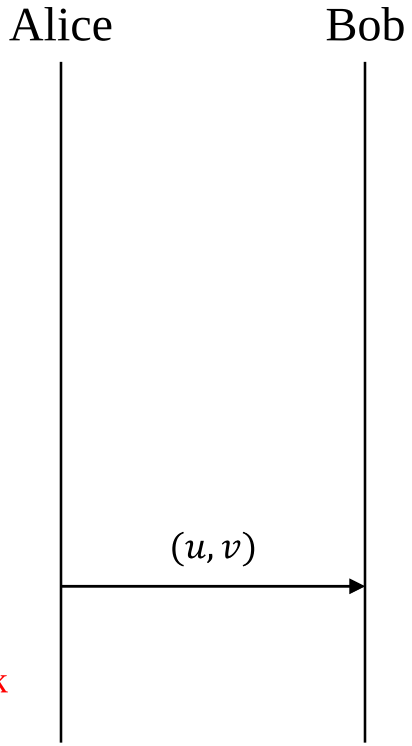


Reference Trace (O)

- 박애선, 원유승, 한동국, "8 비트 구현 Ring-LWE 암호시스템의 SPA 취약점 연구", 한국정보보호학회 논문지, 2017.
- Countermeasure : 박애선, 원유승, 한동국, "Ring-LWE 기반 공개키 암호시스템의 선택 암호문 단순전력분석 공격 대응법", 한국정보보호학회 논문지, 2017.

LAC

Attack Scenario



μ : secret message

$a \leftarrow GenA(seed)$

$\mu' \leftarrow ECC_{BCH}(\mu)$

$s', e', e'' \leftarrow sample()$

$u \leftarrow as' + e'$

$v \leftarrow bs' + e'' + Encode(\mu')$

Single-Trace Attack

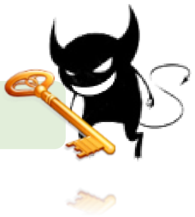
$\mu' \leftarrow Decode(v - us)$

$\mu \leftarrow ECC_{BCH}^{-1}(\mu')$

LAC : Key Generation	
Input	
Output	Public key $(b, seed)$, private key (s)

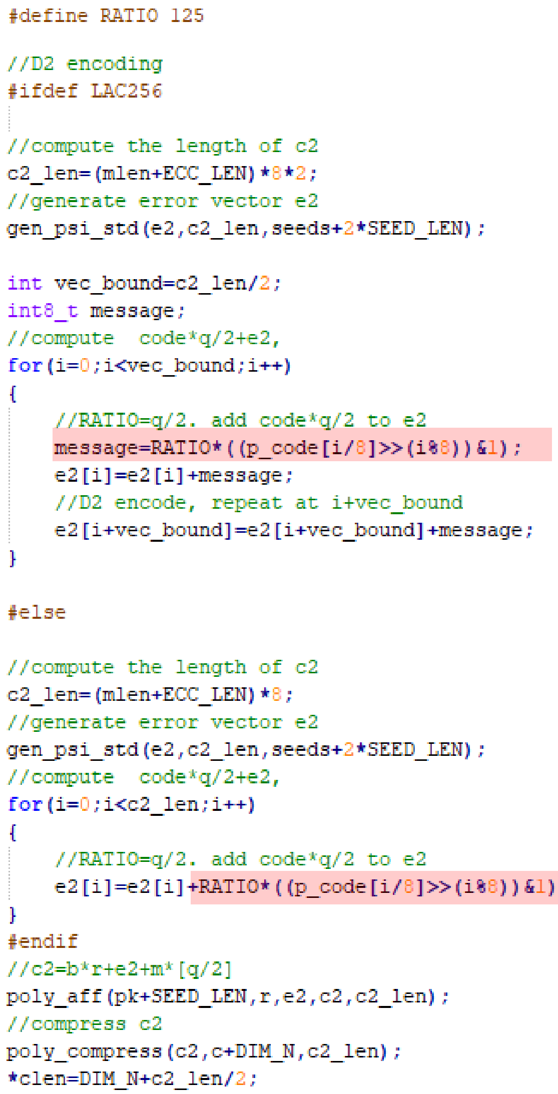
1. Generate $seed$
2. $a \leftarrow GenA\ (seed)$
3. $s, e \leftarrow sample()$
4. $b = as + e$
5. Return $(b, seed)$, (s)

$ss_{key} = HASH(\mu, u, v)$ ss_{key} : shared secret key $ss_{key} = HASH(\mu, u, v)$



Our results 2 : KIISC S2020

LAC : Message Encoding



```
#define RATIO 125

//D2 encoding
#ifdef LAC256

//compute the length of c2
c2_len=(mlen+ECC_LEN)*8*2;
//generate error vector e2
gen_psi_std(e2,c2_len,seeds+2*SEED_LEN);

int vec_bound=c2_len/2;
int8_t message;
//compute  code*q/2+e2,
for(i=0;i<vec_bound;i++)
{
    //RATIO=q/2. add code*q/2 to e2
    message=RATIO*((p_code[i/8]>>(i%8))&1);
    e2[i]=e2[i]+message;
    //D2 encode, repeat at i+vec_bound
    e2[i+vec_bound]=e2[i+vec_bound]+message;
}

#else

//compute the length of c2
c2_len=(mlen+ECC_LEN)*8;
//generate error vector e2
gen_psi_std(e2,c2_len,seeds+2*SEED_LEN);
//compute  code*q/2+e2,
for(i=0;i<c2_len;i++)
{
    //RATIO=q/2. add code*q/2 to e2
    e2[i]=e2[i]+RATIO*((p_code[i/8]>>(i%8))&1);
}
#endif
//c2=b*r+e2+m*[q/2]
poly_aff(pk+SEED_LEN,r,e2,c2,c2_len);
//compress c2
poly_compress(c2,c+DIM_N,c2_len);
*clen=DIM_N+c2_len/2;
```

Encapsulation

μ : secret message

$a \leftarrow GenA(seed)$

$\mu' \leftarrow ECC_{BCH}(\mu)$

$s', e', e'' \leftarrow sample()$

$u \leftarrow as' + e'$

$v \leftarrow bs' + e'' + Encode(\mu')$

Single-Trace Attack

p_code = $\mu' = (\mu \parallel ecc)$

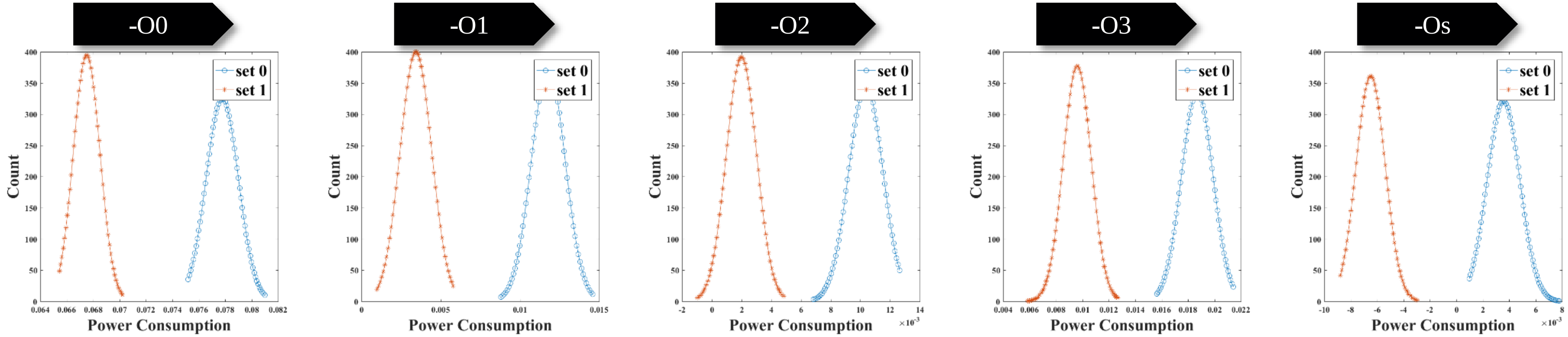
Secret Message $\mu \in \{0,1\}^I, I = 256$

Error Correcting Code $ecc \in \{0,1\}^l, l = 144$

$$message = \begin{cases} 0x00\,, if\ (p\_code[i] \gg j) \land 1 = 0 \\ 0x7d\,, if\ (p\_code[i] \gg j) \land 1 = 1 \end{cases}$$

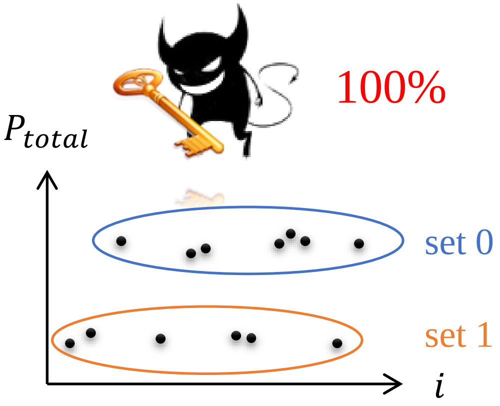
Our results 2 : KIISC S2020

LAC : Distributions of the PoIs



Optimization Level	$E(G_1)$	$E(G_2)$	$\lvert E(G_1) - E(G_2)\rvert$
-O0	7.7773e-02	6.7497e-02	1.0276e-02
-O1	1.1788e-02	3.4290e-03	8.3590e-03
-O2	1.0367e-02	1.9882e-03	8.3788e-03
-O3	1.8690e-02	9.5977e-03	9.0923e-03
-Os	3.5680e-03	-6.4983e-03	1.0066e-04

$$E(G_1) > E(G_2)$$

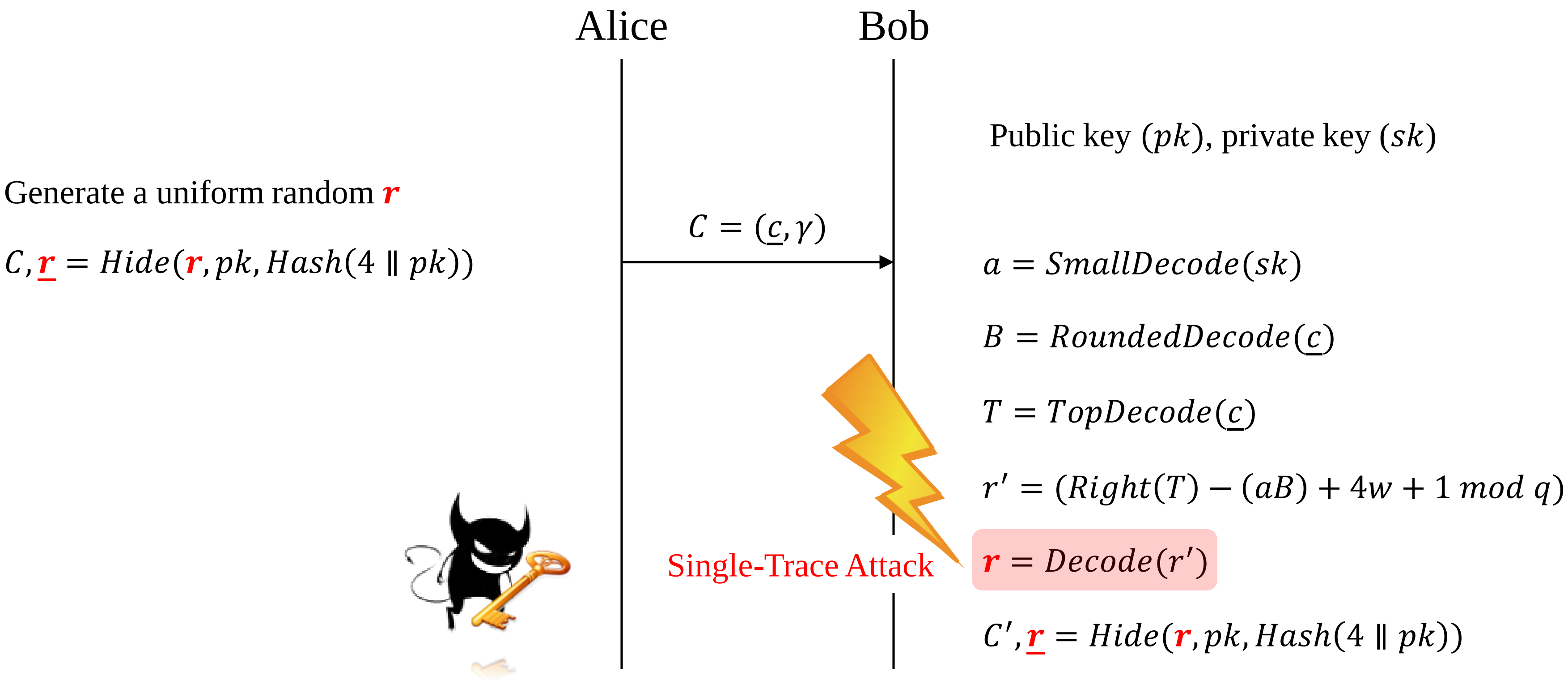


k-means clustering algorithm

Our results 3 : KIISC S2020

NTRU Prime / LPRime

Attack Scenario



Alice

Bob

Public key (pk), private key (sk)

Generate a uniform random $\mathbf{r}$

$C, \underline{\mathbf{r}} = Hide(\mathbf{r}, pk, Hash(4 \parallel pk))$

$C = (\underline{c}, \gamma)$

$a = SmallDecode(sk)$

$B = RoundedDecode(\underline{c})$

$T = TopDecode(\underline{c})$

$r' = (Right(T) - (aB) + 4w + 1 \bmod q)$

Single-Trace Attack

$\mathbf{r} = Decode(r')$

$C', \underline{\mathbf{r}} = Hide(\mathbf{r}, pk, Hash(4 \parallel pk))$

$ss_{key} = SHA512(1 \parallel \underline{r} \parallel C)$ ss_{key} : shared secret key $ss_{key} = SHA512(1 \parallel \underline{r} \parallel C)$

Our results 3 : KIISC S2020

NTRU Prime / LPRime : Message Decoding

Decapsulation

$a = SmallDecode(sk)$

$B = RoundedDecode(\underline{c})$

$T = TopDecode(\underline{c})$

$r' = (Right(T) - (aB) + 4w + 1 \bmod q)$

Single-Trace Attack

$\boldsymbol{r} = Decode(r')$

$C', \underline{\boldsymbol{r}} = Hide(\boldsymbol{r}, pk, Hash(4 \parallel pk))$

Decrypt

```
/* return -1 if x<0; otherwise return 0 */
static int int16_negative_mask(int16 x)
{
  uint16 u = x;
  u >>= 15;
  return -(int) u;
  /* alternative with gcc -fwrapv: */
  /* x>>15 compiles to CPU's arithmetic right shift */
}

/* r = Decrypt((B,T),a) */
static void Decrypt(int8 *r,const Fq *B,const int8 *T,const small *a)
{
  Fq aB[p];
  int i;

  Rq_mult_small(aB,B,a);
  for (i = 0;i < I;++i)
    r[i] = -int16_negative_mask(Fq_freeze(Right(T[i])-aB[i]+4*w+1));
}

#define XDecrypt Decrypt

/* r = ZDecrypt(C,sk) */
static void ZDecrypt(Inputs r,const unsigned char *c,const unsigned char *sk)
{
  small a[p];
  Fq B[p];
  int8 T[I];

  Small_decode(a,sk);
  Rounded_decode(B,c);
  Top_decode(T,c+Rounded_bytes);
  XDecrypt(r,B,T,a);
}
```

Secret Message $r \in \{0,1\}^I, I = 256$

$$r = (r_0, \cdots, r_{I-1})_2, r_i = r[i]$$

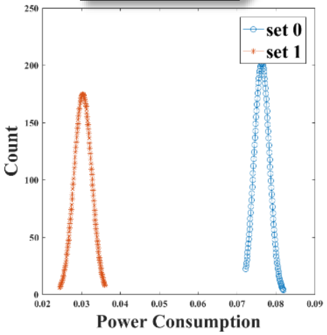
$$int16\_negative\_mask() = \begin{cases} 0x0000, if\ Fqfreeze() \geq 0 \\ 0xffff, if\ Fqfreeze() < 0 \end{cases}$$

$$r[i] = \begin{cases} 0, if\ \ int16\_negative\_mask() = 0x0000 \\ 1, if\ \ int16\_negative\_mask() = 0xffff \end{cases}$$

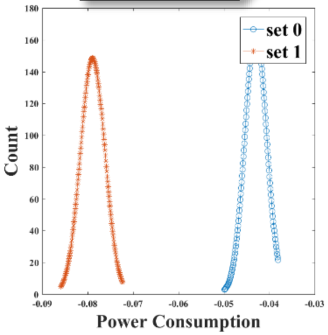
Our results 3 : KIISC S2020

NTRU Prime / LPRime : Distributions of the PoIs

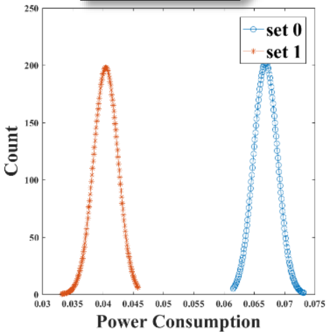
-O0



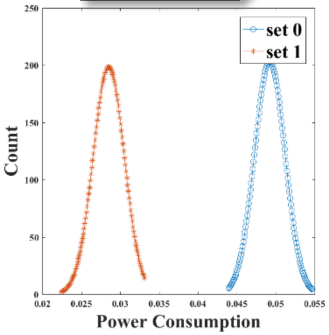
-O1



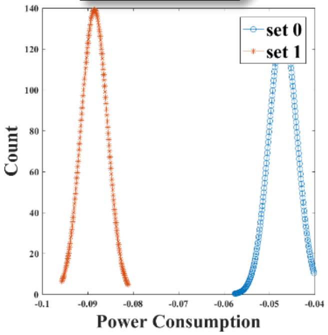
-O2



-O3

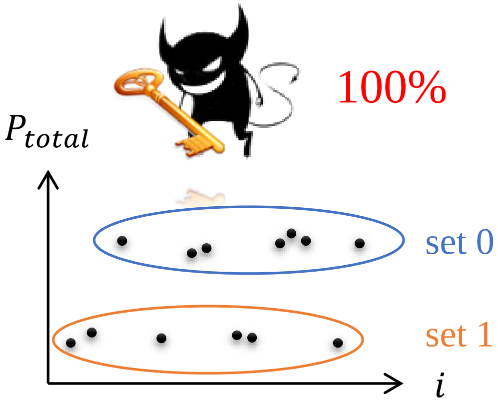


-Os



Optimization Level	$E(G_1)$	$E(G_2)$	$\lvert E(G_1) - E(G_2) \rvert$
-O0	7.6411e-02	3.0403e-02	4.6008e-02
-O1	-4.2961e-02	-7.8910e-02	3.5949e-02
-O2	-6.6894e-02	4.0483e-02	2.6411e-02
-O3	4.9231e-02	2.8530e-02	2.0701e-02
-Os	-4.6974e-02	-8.8565e-02	4.1591e-02

$E(G_1) > E(G_2)$



k-means clustering algorithm

Our results 4 : KIISC S2020

qTESLA

Attack Scenario

$N+1$ Share. $x = x_0 \oplus \cdots \oplus x_N, (x_i)_{0 \le i \le N}$

Public key (pk), Secret key (sk)

Message m, Signature ($Sign$)

$(y_i)_{0\le i\le N} \leftarrow Sample()$

$(v_i)_{0\le i\le N} \leftarrow a \cdot (y_i)_{0\le i\le N}$

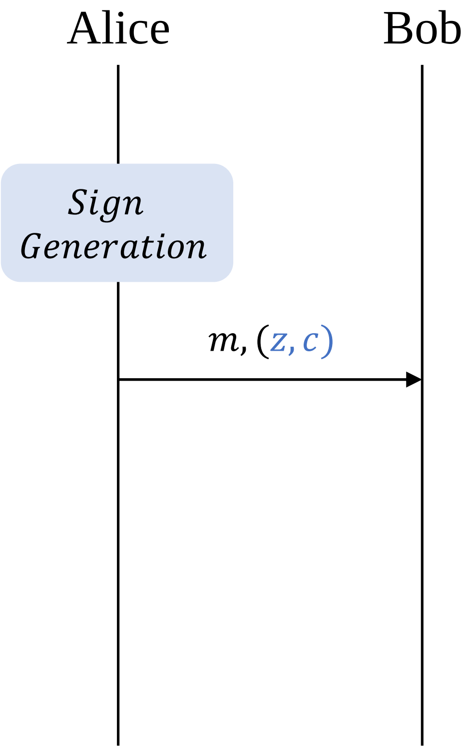
$c \leftarrow Enc(H([(v_i)_{0\le i\le N}]_M, m))$

$(z_i)_{0\le i\le N} \leftarrow (y_i)_{0\le i\le N} + (s_i)_{0\le i\le N} \cdot c$

$(w_i)_{0\le i\le N} \leftarrow (v_i)_{0\le i\le N} - (e_i)_{0\le i\le N} \cdot c$

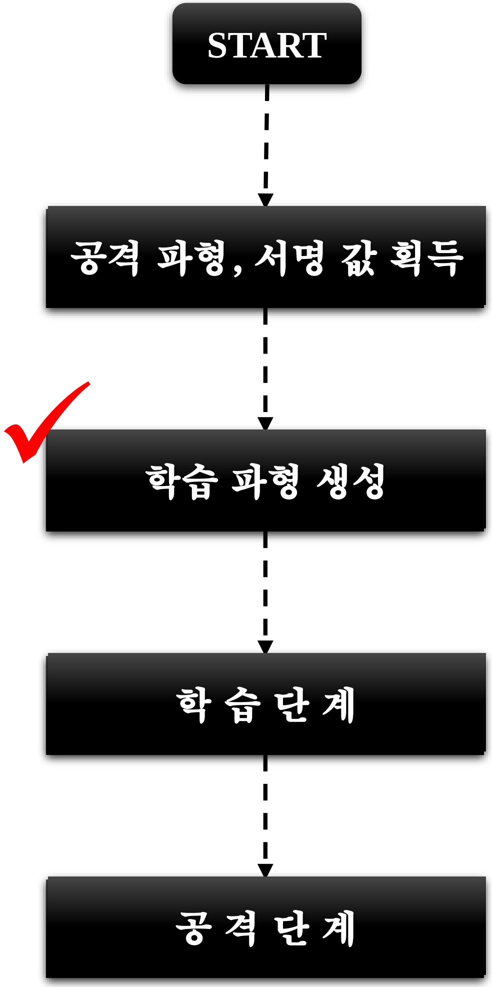
$z \leftarrow (z_i)_{0\le i\le N}$

Single-Trace Attack



Our results 4 : KIISC S2020

qTESLA : ML-based TA (Machine Learning based Template Attack)



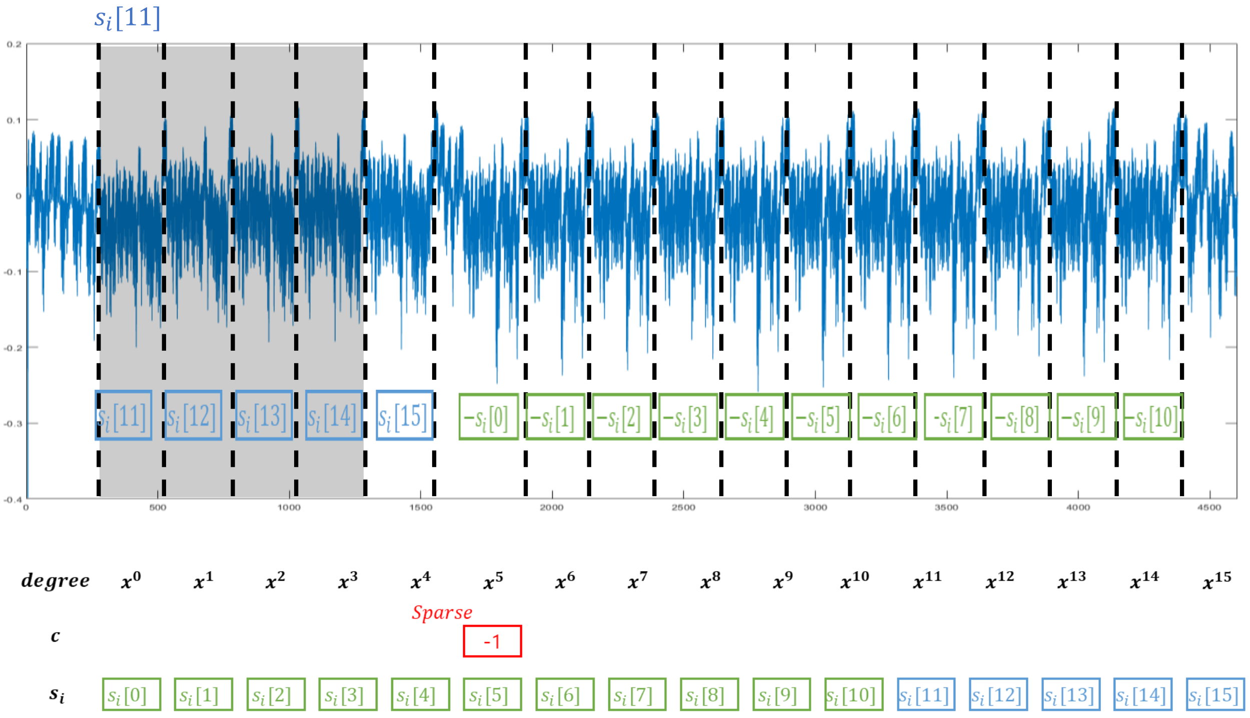
START

공격 파형, 서명 값 획득

✓ 학습 파형 생성

학습 단계

공격 단계



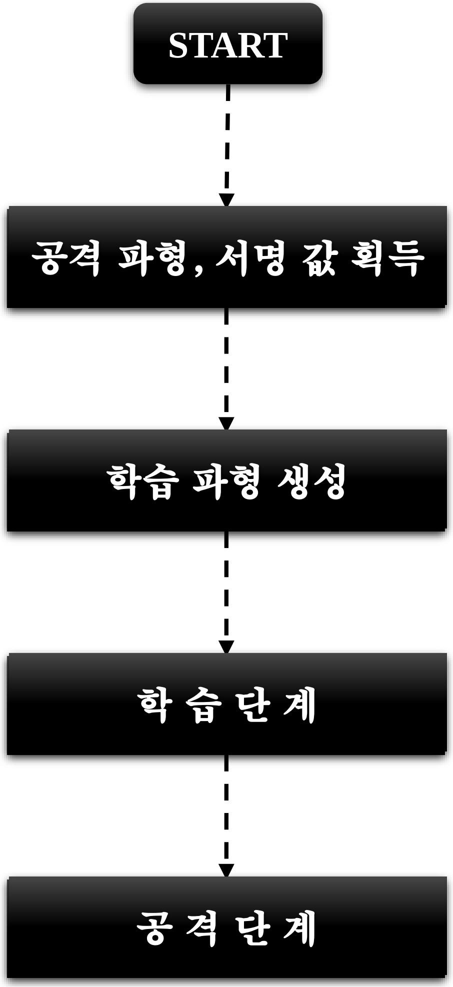
- **학습 파형 (곱셈 연산 4개)**
- **Label : $s_i[j]$ value**
- **Model : MLP**

$s_i[j]$: 16bit

상위 8bit	하위 8bit

Our results 4 : KIISC S2020

qTESLA : ML-based TA (Machine Learning based Template Attack)


START

공격 파형, 서명 값 획득

학습 파형 생성

학습 단계

공격 단계

Layer	Node (in, out)	Kernel initializer
Input Layer	(x, x)	-
Batch Normalization	(x, x)	-
Dense	$(x, 64)$	he_uniform
Relu	$(64,64)$	-
Batch Normalization	$(64, 64)$	-
Dense	$(64,64)$	he_uniform
Relu	$(64,64)$	-
Batch Normalization	$(64, 64)$	-
Dense	$(64, 256)$	he_uniform
Softmax	$(256, 256)$	-

- Input Normalization: all values are within the range of -1 and 1
- Loss function: categorical_crossentropy
- Optimizer: Nadam (lr=0.002, epsilon=1e-08)
- Label encoding: one-hot encoding
- Batch size and epochs: 32 and maximum 100, respectively
- #Trace: (training T, validation V)

100%

Masked qTESLA
Secret (s, e)

Single-Trace Attack

$$s = s_0 \oplus s_1 \oplus s_2 \oplus \cdots \oplus s_N$$

$$t = a \cdot s + e \xrightarrow{\qquad} e = t - a \cdot s$$

NIST Round 2 Code

QC code based cryptosystem

Algorithm		Purpose	Code
BIKE		KEM	QC-MDPC
HQC		KEM-DEM	QC & BCH
RQC		KEM-DEM	QC & Gabidulin
LEDAcrypt	LEDAkem	KEM	QC-LDPC
	LEDApkc	KEM	QC-LDPC

other code based cryptosystem

Algorithm	Purpose	Code
Classic McEliece	KEM	Goppa
NTS-KEM	KEM	Goppa
ROLLO(LAKE, LOCKER, Ouroboros-R)	KEM-DEM	LRPC

Code

QC code

Goppa code

QC-LDPC code

Reed-Solomon codes

QC-MDPC code

MDPC codes

LDPC codes

- **Quasi-Cyclic** code
  for saving memory (small key sizes)

SCA on KEM/PKE

	공격 대상	공격 연산	공격 유형	대응 기법
[PQCrypto 2008]	McEliece	Decryption (Patterson algorithm / Error locator polynomial)	TA	Raise degree
		Key Generation (Generation of parity-check matrix)	PA	Masking
		Decryption (Permutation)	MA	Constant-time operation, only public input dependent access
[ICISC 2009]	McEliece	Decryption (Patterson algorithm / Error locator polynomial)	TA	Regular operation
[PQCrypto 2010]	McEliece	Decryption (Patterson algorithm / Secret permutation)	TA	Regular operation
[PQCrypto 2010]	McEliece	Decryption (Permutation, syndrome computation, syndrome inversion)	SPA	Shuffling, insert dummy, masking
FA [FutureTech 2010]				
[J. Cryptographic Engineering 2011]	McEliece	Decryption (Patterson algorithm / Error locator polynomial)	SPA	Regular operation
[J. Cryptographic Engineering 2011]	McEliece	Decryption (Root finding)	TA / FA	Regular operation, blinding, masking
[J. Cryptographic Engineering 2011]	McEliece / Niederreiter	Decryption (Patterson algorithm / Error locator polynomial)	TA	
[PQCrypto 2013]	McEliece	Decryption (Syndrome inversion)	TA^C	
[RadioElektronika 2015]	McEliece	Decryption (Syndrome computation)	SPA	Regular operation

Patterson algorithm

SCA on KEM/PKE

	공격 대상	공격 연산	공격 유형	대응 기법
[RadioElektronika 2016]	McEliece	Decryption (Permutation)	DPA	Masking
[J. Computers, Communications & Control 2017]	McEliece	Decryption (Patterson algorithm / Error locator polynomial)	TA	Regular operation
survey [ICCCC 2018]				
[PQCrypto 2014]	QC-MDPC McEliece	Encryption (Matrix multiplication) / Decryption (Key rotation)	TA, SPA	Regular operation
[ACNS 2015]	QC-MDPC McEliece	Decryption (Syndrome computation / key rotation)	DPA + Algebraic	Masking
[IEEE Transactions 2016]	QC-MDPC McEliece	Decryption (Syndrome computation / key rotation)	DPA + Algebraic	Masking
[SAC 2016]	QC-MDPC McEliece	Decryption (Syndrome computation, decoder)		Masking
[CHES 2016]	QC-MDPC	Decryption (Syndrome computation)		Regular operation
[CHES 2017]	QC-MDPC	Decryption (Syndrome computation)	DPA + Algebraic	Masking
[TIIS 2019]	QC-MDPC	Decryption (Syndrome computation)		Masking
[TCHES 2019]	QC	Decryption (Syndrome computation)	DPA, SPA	Masking, hiding(mentioned)

▣ [CHES 2017] A side-channel assisted cryptanalytic attack against QcBits

Syndrome computation $\boldsymbol{H} \cdot \boldsymbol{c}^{\top}$

2014 Timing Attack

(Simple Power Analysis)

TA Countermeasure

2016 Constant-Time Implementation

2017 Differential Power Analysis on Constant-Time Implementation

Limitation: It could not completely recover accurate secret indices, requiring further solving linear equations to obtain entire secret information

	8-bit	16-bit	32-bit	64-bit
80-bit security	0.4 seconds	15 seconds	16 hours	**≈ 530 years**
128-bit security	2 seconds	4 minutes	≈ 7 days	**≈ 790,000 years**

It is not feasible on 64-bit processor

Our results 5 : TCHES 2019

▣ Multiple-Trace Attack on Constant-Time Multiplication

8-bit word

$$d = (\underbrace{d_7d_6d_5d_4d_3}_{\text{Correlation Occurring Position}}\underbrace{d_2d_1d_0}_{\text{Correlation Power Analysis}})_2$$

Word unit rotation

$$result = \begin{cases} unrotated & ,if\ d_i = 0 \\ rotated & ,if\ d_i = 1 \end{cases}$$

$$result = \begin{cases} (rotated \,\&\, 0x00) \oplus (unrotated \,\&\, 0xff) = unrotated & ,if\ d_i = 0 \\ (rotated \,\&\, 0xff) \oplus (unrotated \,\&\, 0x00) = rotated & ,if\ d_i = 1 \end{cases}$$

Bit rotation

$$result = (\ll_{8-L})|(\gg_L)$$

$$0 \le L = (d_2d_1d_0)_2 < 8$$

Our results 5 : TCHES 2019

▣ Multiple-Trace Attack on the Word Unit Rotation

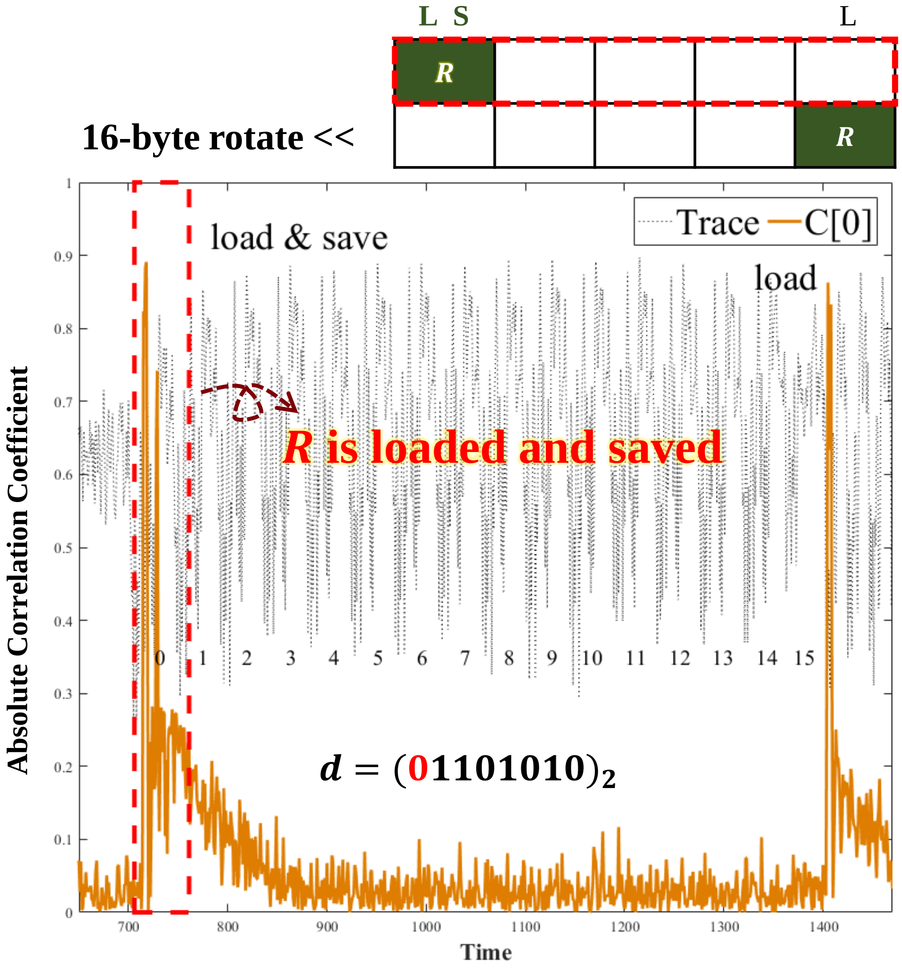
Property 1.

$$result = \begin{cases} unrotated & , if\ d_i = 0 \\ rotated & , if\ d_i = 1 \end{cases}$$

$$d = (\overbrace{d_7 d_6 d_5 d_4}^{target} d_3 d_2 d_1 d_0)_2, d_i \in \{0,1\}$$

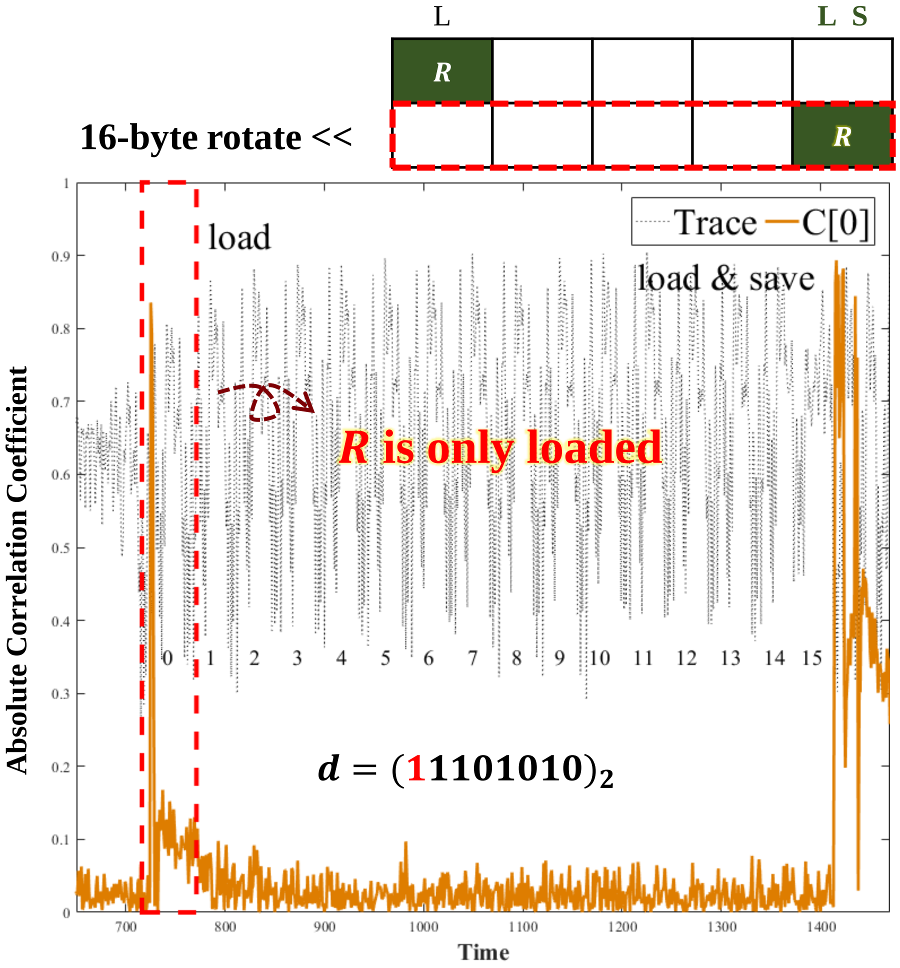
$$R \in_{Random} \{0,1\}^8$$

8-bit word



R is loaded and saved

$d = (01101010)_2$



R is only loaded

$d = (11101010)_2$

Our results 5 : TCHES 2019

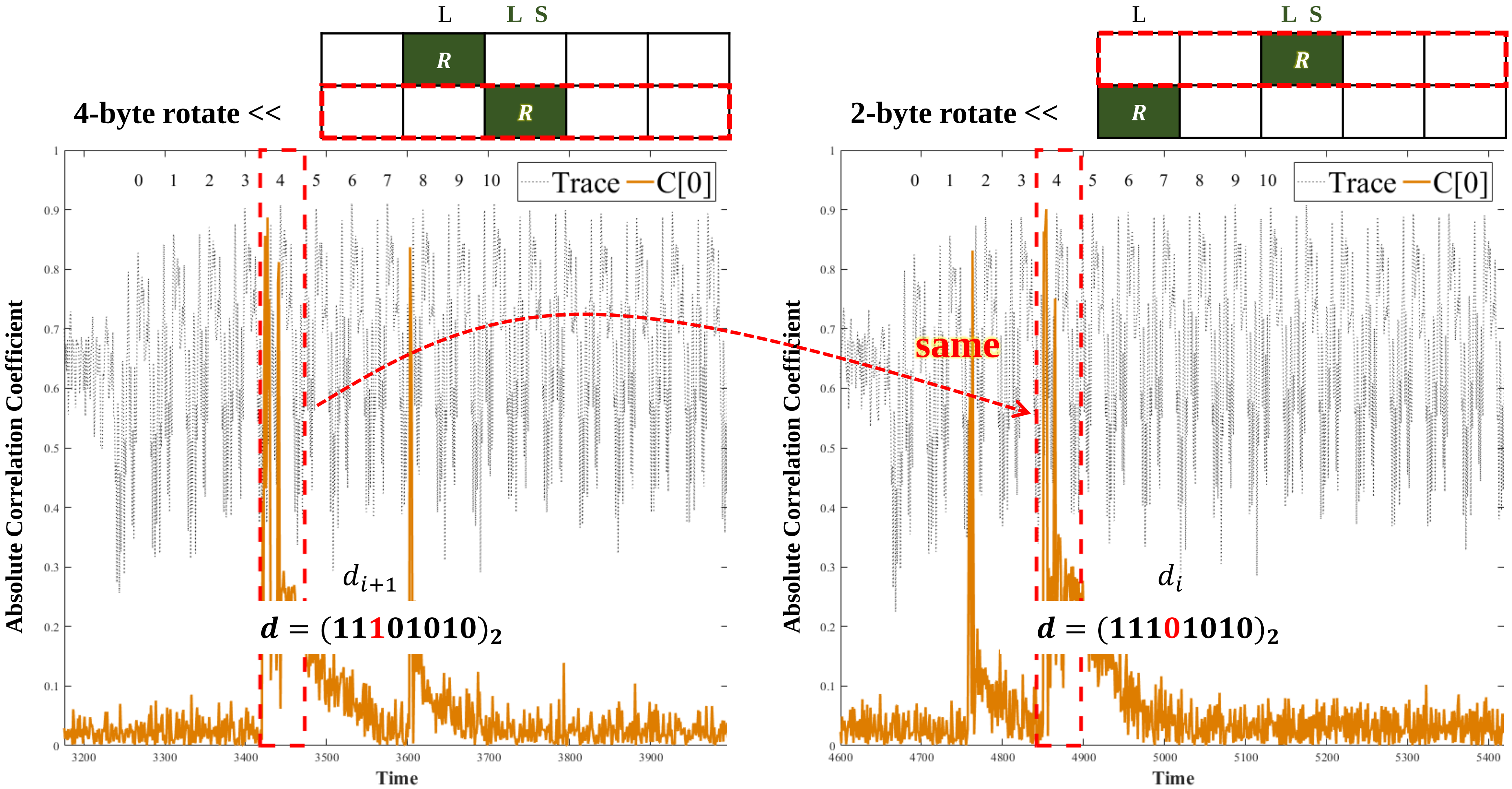
▣ Multiple-Trace Attack on the Word Unit Rotation

Property 2. $result = \begin{cases} unrotated & , if\ d_i = 0 \\ rotated & , if\ d_i = 1 \end{cases}$

$d = (\overbrace{d_7d_6d_5d_4}^{\text{target}}d_3d_2d_1d_0)_2, d_i \in \{0,1\}$

$R \in_{Random} \{0,1\}^8$

8-bit word



Our results 5 : TCHES 2019

Multiple-Trace Attack on the Bit Rotation

$$\boldsymbol{d} = (\boldsymbol{d_7 d_6 d_5 d_4 d_3} \overbrace{\boldsymbol{d_2 d_1 d_0}}^{\text{target}})_2, \boldsymbol{d_i} \in \{\boldsymbol{0},\boldsymbol{1}\}$$

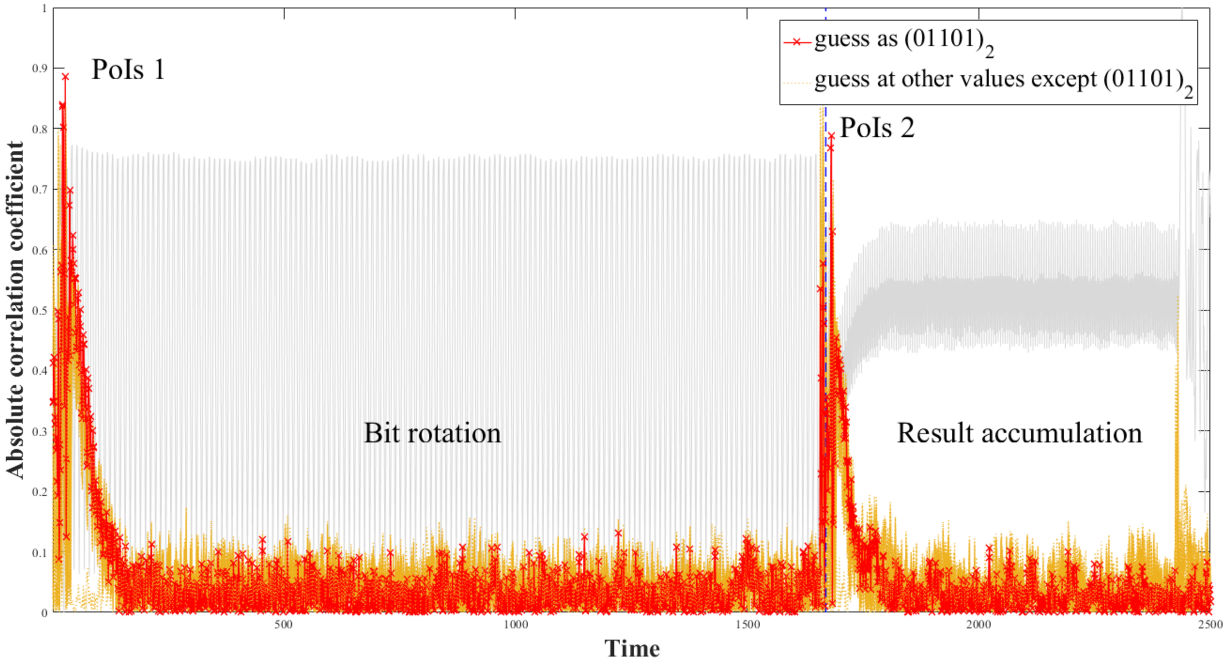
8-bit word

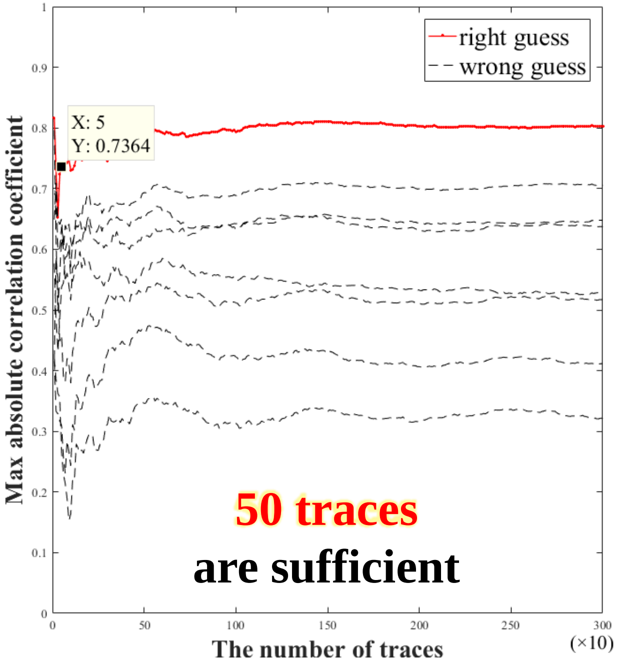
$$\boldsymbol{result} = (\ll_{(8-L)})|(\gg_{L})$$

$$0 \le \boldsymbol{L} = (\boldsymbol{d_2 d_1 d_0})_2 < 8$$

- **Guess the *L* value from 0 to 7**

  and calculate Pearson's correlation coefficient between traces and $\boldsymbol{result}$ values





50 traces are sufficient

Our results 5 : TCHES 2019

▣ Single-Trace Attack on Constant-Time Multiplication

$$\boldsymbol{d} = (\underbrace{d_7d_6d_5d_4d_3}_{\text{Key Bit-dependent Attack}}\underbrace{d_2d_1d_0}_{\text{Simple Power Analysis}})_2$$

8-bit word

Word unit rotation

$$result = \begin{cases} unrotated & , if\ d_i = 0 \\ rotated & , if\ d_i = 1 \end{cases}$$

$$result = \begin{cases} (rotated \,\&\, 0x00) \oplus (unrotated \,\&\, 0xff) = unrotated & , if\ d_i = 0 \\ (rotated \,\&\, 0xff) \oplus (unrotated \,\&\, 0x00) = rotated & , if\ d_i = 1 \end{cases}$$

Bit rotation

$$result = (\ll_{8-L})|(\gg_L)$$

$$0 \leq L = (d_2d_1d_0)_2 < 8$$

Our results 5 : TCHES 2019

▣ Single-Trace Attack on the Word Unit Rotation

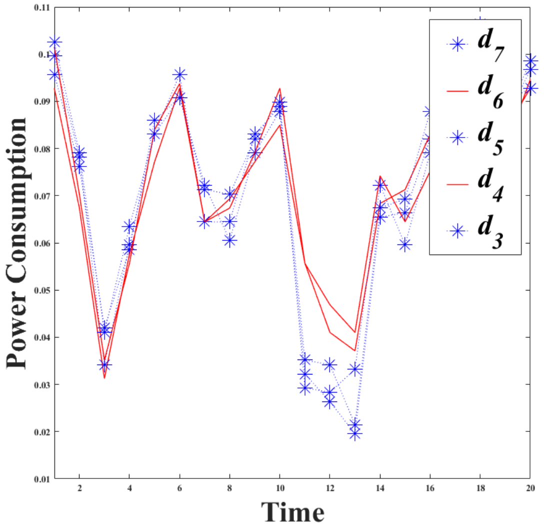
target

$d = (\overbrace{d_7d_6d_5d_4d_3}^{}d_2d_1d_0)_2, d_i \in \{0,1\}$

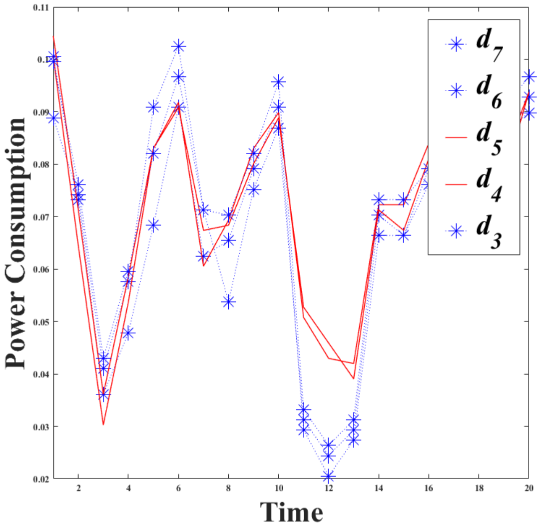
8-bit word

- $d = (d_7d_6d_5d_4d_3d_2d_1d_0)_2, d_i \in \{0,1\}$

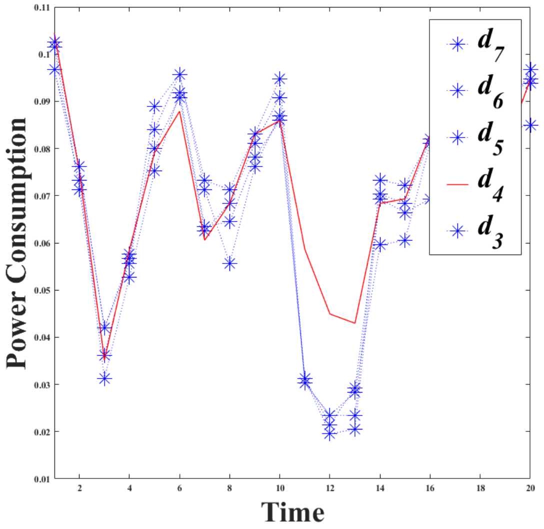
$$result = \begin{cases} (rotated \,\&\, 0x00) \oplus (unrotated \,\&\, 0xff) = unrotated & , if\ d_i = 0 \\ (rotated \,\&\, 0xff) \oplus (unrotated \,\&\, 0x00) = rotated & , if\ d_i = 1 \end{cases}$$



$169 = (10101001)_2$



$201 = (11001001)_2$



$233 = (11101001)_2$

Key Bit-dependent Property

✓ $W = 8$

$$mask = \begin{cases} 0x00 & , if\ d_i = 0 \\ 0xff & , if\ d_i = 1 \end{cases}$$

- K-means clustering
- Fuzzy k-means clustering
- EM (Expectation-maximization)

- 심보연, 권지훈, 최규영, 조지훈, 한동국, "마스킹된 상수 시간 곱셈을 사용하는 QC 코드 기반 암호 알고리즘에 대한 단일 파형 분석", 한국정보보호학회 하계학술대회, 2019.

Our results 5 : TCHES 2019

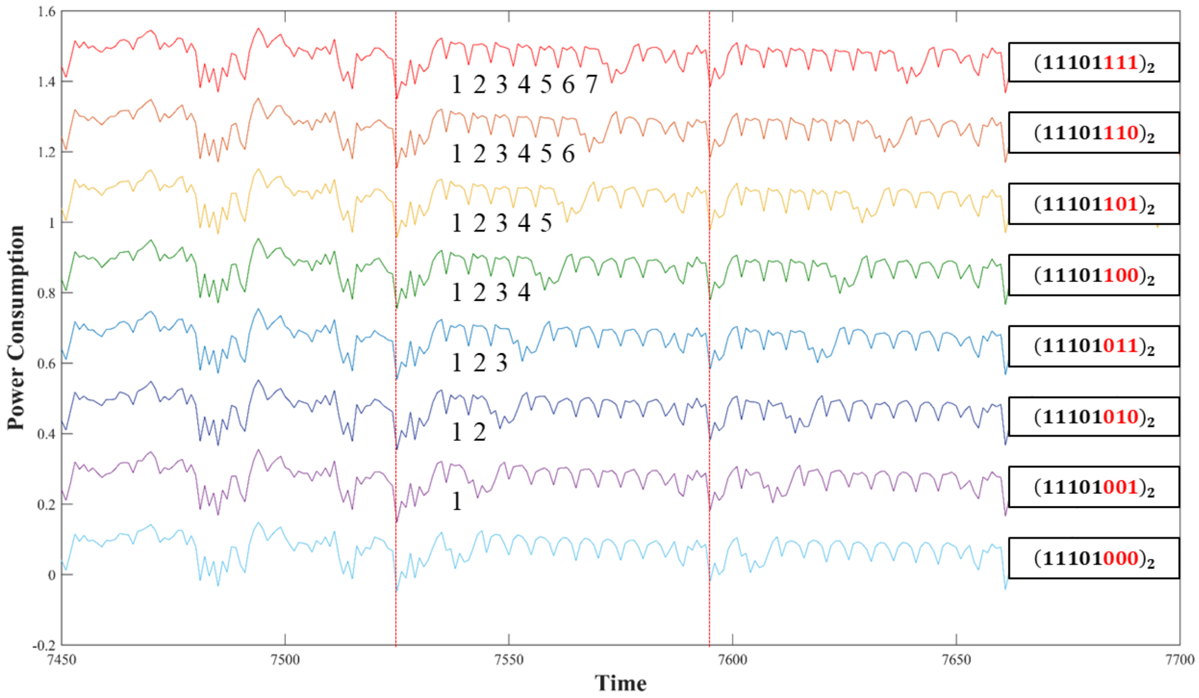
▣ Single-Trace Attack on the Bit Rotation

$$d = \underbrace{(d_7 d_6 d_5 d_4 d_3}\overbrace{d_2 d_1 d_0}^{\text{target}})_2, d_i \in \{0,1\}$$

8-bit word

$$result = (\lll_{8-L}) | (\ggg_{L})$$

$$0 \leq L = (d_2 d_1 d_0)_2 < 8$$



Our results 5 : TCHES 2019

▣ Single-Trace Attack on the Bit Rotation

$$\boldsymbol{d} = (d_7d_6d_5d_4d_3\overbrace{d_2d_1d_0}^{\text{target}})_2, d_i \in \{0,1\}$$

8-bit word

$$\boldsymbol{result} = (\ll_{8-L})|(\gg_{L})$$

$$0 \le L = (d_2d_1d_0)_2 < 8$$

	Bit rotate	Left shift	Right shift	SPA
AVR 8-bit word	Single bit shift instructions	$(8-L)$ times ($(8-L)$ clock cycles)	L times (L clock cycles)	O
MSP 16-bit word	Single bit shift instructions	$(8-L)$ times ($(8-L)$ clock cycles)	L times (L clock cycles)	O
ARM 32-bit word	Multiple bit shift instructions (ex. barrel shifter)	One clock	One clock	X
64-bit word	Multiple bit shift instructions (ex. barrel shifter)	One clock	One clock	X

✓ In the cases of 32-bit and 64-bit, we need to solve linear equations to find accurate indices

Vulnerable operations

Loops whose bound is input-dependent

```
int j = 4;
for(i = 0; i < j; i++)
  ...

for(i = 0; i < length; i++)
  ... break;

for(i = 0; i < length; i++)
  ... exit;
```

```
int max_length = 6;
int j = 4;
for(i = 0; i < max_length; i++)
  determiner = ((i - j) & mask) >> 31;
```

Branches whose condition is input-dependent

```
if(i == 1)
  ...a=b;


if( j != 0)
  ...v=x;
else
  ...v=v;
```

```
determiner = !(((i-1) & mask) >> 31);
a = b * determiner;
determiner = -(((- j) & mask) >> 31);
v = x * determiner + v * !determiner;
```

input-dependent memory access

```
for(i = 0; i < length; i++)
  if(i == index)
    ...
    a[i]=b[2i+1];
```

```
for(i = 0; i < length; i++)
  xorVal = i ^ index;
  determiner = (xorVal & 1) − 1;
  out = b[i] & determiner;
```

- ✓ Constant-time BCH Error-Correcting Code, ePrint 2019-155
- ✓ Timing Attack on HQC and Countermeasure, ePrint 2019-909

```
mask = 0xff; // 8-bit processor
```

NIST Round 2 Multivariate

Algorithm	Purpose	
GeMMS	Signature	HFE
LUOV	Signature	UOV
Rainbow	Signature	UOV
MQDSS	Signature	Fiat-Shamir

SCA on Signature

- [ITTC 2004] On the importance of protecting delta; in SFLASH against side channel attacks
- FA [The Computer Journal 2017] On the importance of checking multivariate public key cryptography for side-channel attacks: The case of enTTS scheme
- FA [Future Generation Computer System 2018] Side-channel security analysis of UOV signature for cloud-based Internet of Things
- [TCHES 2018] Side-Channel Attacks on Post-Quantum Signature Schemes based on Multivariate Quadratic Equations

Our results 7 : TCHES 2018

CPA against Rainbow Signature

- **Non-invasive attack on MQ-based signature schemes**

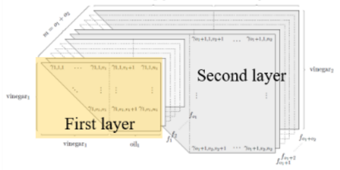
분석 대상

Rainbow Signature generation

Invert S

$$\begin{pmatrix} s'_{11} & s'_{12} & \cdots & s'_{1m} \\ s'_{21} & \ddots & \ddots & s'_{2m} \\ \vdots & \ddots & \ddots & \vdots \\ s'_{m1} & s'_{m2} & \cdots & s'_{mm} \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$$

Invert $\mathcal{F}$



Invert T

$$\begin{pmatrix} t'_{11} & t'_{12} & \cdots & t'_{1n} \\ t'_{21} & \ddots & \ddots & t'_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ t'_{n1} & t'_{n2} & \cdots & t'_{nn} \end{pmatrix} \begin{pmatrix} y'_1 \\ y'_2 \\ \vdots \\ y'_n \end{pmatrix}$$

기존 연구

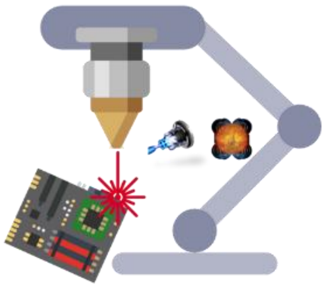
[The Computer Journal 2017]

CPA + Fault injection

✓ **(Goal) Recovery the secret maps** $S, \mathcal{F}, T$

enTTS **implementation**

with random linear maps



Fault Injection

공격 가정 완화

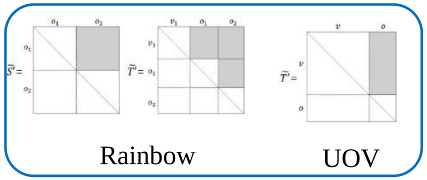
제안 방법

CPA

✓ **(Goal) Recovery the secret maps** $S, \mathcal{F}, T$

Rainbow and UOV **implementation**

with equivalent keys



CPA + Algebraic

✓ **(Goal) forged signature**

Rainbow **implementation**

with random linear maps

Conclusion

KEM/Encryption

(9) Lattice-based	
FrodoKEM	LWE
LAC	RLWE
NewHope	RLWE
Round5	LWR/RLWR
Crystals-Kyber	MLWE
Saber	MLWR
Three Bears	IMLWE
NTRU	NTRU
NTRU Prime	NTRU

(7) Code-based	
Classic McEliece	Goppa
NTS-KEM	Goppa
BIKE	Short Hamming
HQC	Short Hamming
LEDAcrypt	Short Hamming
RQC	Low rank
ROLLO	Low rank

(1) Isogeny	
SIKE	Isogeny

Signature

(3) Lattice-based	
qTESLA	Fiat-Shamir
Crystals-Dilithium	Fiat-Shamir
FALCON	Hash then sign

(4) Multivariate	
GeMMS	HFE
LUOV	UOV
Rainbow	UOV
MQDSS	Fiat-Shamir

(2) Symmetric-based	
SPHINCS+	Hash
Picnic	ZKP

부채널 분석에 대한 안전성 검증 필요

Q & A

PA	Power Analysis
EMA	Electromagnetic Analysis
MA	Microarchitectural Attack
SPA	Simple Power Analysis
SPA^P	Simple Power Analysis with Profiling
SEMA	Simple Electromagnetic Analysis
TA	Timing Attack
TA^C	Cache Timing Attack ($\in$ MA)
TA^P	Template Attack ($\in SPA^P$)
CA	Collision Attack
DPA	Differential Power Analysis
CPA	Correlation Power Analysis
CEMA	Correlation Electromagnetic Analysis
FA	Fault Attack
DFA	Differential Fault Attack
CBA	Cold Boot Attack

SCA on KEM/PKE

- [CT-RSA 2007] Timing attacks on NTRUEncrypt via variation in the number of hash calls
- [RFID Security 2008] Power analysis on NTRU implementations for RFIDs: First results
- [IEICE 2010] Countermeasures against Power Analysis Attacks for NTRU Public Key Cryptosystem
- [IEICE 2011] Fault analysis of the NTRUEncrypt cryptosystem
- [Cryptography and Communications 2012] Fault analysis of the NTRUSign digital signature scheme
- [Microprocessors and Microsystems 2013] First-order collision attack on protected NTRU cryptosystem
- [TIIS 2013] Power analysis attacks and countermeasures on NTRU-based wireless body area networks
- [ePrint 2014] Compact and Side Channel Secure Discrete Gaussian Sampling
- [CHES 2015] A masked Ring-LWE implementation
- [AsianHOST 2016] Chosen ciphertext Simple Power Analysis on software 8-bit implementation of ring-LWE encryption
- [J. Cryptographic Engineering 2016] Masking ring-LWE
- [PQCrypto 2016] Additively Homomorphic Ring-LWE Masking
- [CHES 2017] Single-Trace Side-Channel Attacks on Masked Lattice-Based Encryption
- [Computers and Electrical Engineering 2017] Random key rotation: Side-channel countermeasure of NTRU cryptosystem for resource-limited devices
- [Applied Science 2018] Single Trace Analysis on Constant Time CDT Sampler and Its Countermeasure
- [Applied Science 2018] Single Trace Side Channel Analysis on NTRU Implementation

SCA on KEM/PKE

- [DATE 2018] Binary Ring-LWE hardware with power side-channel countermeasures
- [GLSVLS 2018] Physical Protection of Lattice-Based Cryptography: Challenges and Solutions
- [IEEE Trans. on Computer 2018] Loop-Abort Faults on Lattice-Based Signature Schemes and Key Exchange Protocols
- [HOST 2018] Horizontal Side-Channel Vulnerabilities of Post-Quantum Key Exchange Protocols
- [SAC 2018] Assessing the feasibility of single trace power analysis of Frodo
- [TCHES 2018] Cold Boot Attacks on Ring & Module-LWE Under the NTT
- [TCHES 2018] Differential fault attacks on deterministic lattice signatures
- [CT-RSA 2019] Assessment of the Key-Reuse Resilience of NewHope
- [ePrint 2019] Correlation Power Analysis on NTRU Prime and Related Countermeasures = [TCHES 2020] Power Analysis of NTRU Prime
- [Latincrypt 2019] More Practical Single-Trace Attacks on the Number Theoretic Transform
- [TIS 2019] Timing Attacks on Error Correcting Codes in Post-Quantum Schemes
- [ePrint 2020] LWE with Side Information; Attacks and Concrete Security Estimation
- [PQCrypto 2020] Defeating NewHope with a Single Trace
- [TCHES 2020] Power Analysis of NTRU Prime
- [TCHES 2020] Generic Side-channel attacks on CCA-secure lattice-based PKE and KEM schemes

SCA on Signature

countermeasure / qTESLA, Dilithium 공격

- [Cryptography and Communications 2012] Fault analysis of the NTRUSign digital signature scheme
- [CHES 2016] Flush, Gauss, and Reload - A Cache Attack on the BLISS Lattice-Based Signature Scheme
- [[FDTC 2016] Lattice-Based Signature Schemes and Their Sensitivity to Fault Attacks
- [SAC 2016] Loop-Abort Faults on Lattice-Based Fiat-Shamir and Hash-and-Sign Signatures
- [ICAR 2016] Arithmetic Coding and Blinding Countermeasures for Lattice Signatures
- [ePrint 2017] Analyzing the Shuffling Side-Channel Countermeasure for Lattice-Based Signatures
- [SAC 2017] Side-Channel Attacks on BLISS Lattice-Based Signatures
- [SAC 2017] To BLISS-B or not to be - Attacking strong Swan's Implementation of Post-Quantum Signatures
- [IACR 2018] Differential Fault Attacks on Deterministic Lattice Signatures
- [EUROCRYPT 2018] Masking the GLP Lattice-Based Signature Scheme at Any Order
- [ICAR 2019] One Bit It Takes A Devastating Timing Attack on BLISS's Non-Constant Time Sign Flips
- [ePrint 2019] Masking Dilithium : Efficient Implementation and Side-Channel Evaluation
- [CARDIS 2019] An Efficient and Provable Masked Implementation of qTESLA

SCA on KEM/PKE

- [PQCrypto 2008] Side channels in the McEliece PKC
- [ICISC 2009] A timing attack against patterson algorithm in the McEliece PKC
- [PQCrypto 2010] A timing attack against the Secret Permutation in the McEliece PKC
- [PQCrypto 2010] Practical power analysis attacks on software implementations of McEliece
- [FutureTech 2010] MecEliece/Niederreiter PKC: sensitivity to fault injection
- [J. Cryptographic Engineering 2011] A simple power analysis attack on a McEliece cryptoprocessor
- [J. Cryptographic Engineering 2011] Message-aimed side channel and fault attacks against a public-key cryptosystems with homomorphic properties
- [J. Cryptographic Engineering 2011] Side-channel attacks on the McEliece and Niedereiter public-key cryptosystems
- [PQCrypto 2013] Timing attacks against the syndrome inversion in code-based cryptosystems
- [RadioElektronika 2015] Countermeasure against the SPA Attack on an Embedded McEliece Cryptosystem

SCA on KEM/PKE

- [RadioElektronika 2016] Differential power analysis attacks on the secure bit permutation in the McEliece cryptosystem
- [J. Computers, Communications & Control 2017] Improved Timing Attacks against the Secret Permutation in the McEliece PKC
- survey [ICCCC 2018] Survey on cryptanalysis of code-based cryptography: From theoretical to physical attacks
- [PQCrypto 2014] Towards side-channel resistant implementations of QC-MDPC McEliece encryption on constrained devices
- [ACNS 2015] Differential Power Analysis of a McEliece Cryptosystem
- [IEEE Transactions 2016] Horizontal and Vertical Side Channel Analysis of a McEliece Cryptosystem
- [SAC 2016] Masking Large Keys in Hardware: A Masked Implementation of McEliece
- [CHES 2016] QcBits: Constant-Time Small-Key Code-Based Cryptography
- [CHES 2017] A side-channel assisted cryptanalytic attack against QcBits
- [TIIS 2019] Higher-Order Masking Scheme against DPA Attack in Practice: McEliece Cryptosystem Based on QD-MDPC Code
- [TCHES 2019] Novel Side-Channel Attacks on Quasi-Cyclic Code-Based Cryptography