

NIST 양자내성암호 공모전 Round2

코드기반암호 성능 비교 분석

한성대학교 장경배, 최승주, 권용빈, 김현지
지도교수 서화정

Contents

양자 컴퓨터와 양자 알고리즘

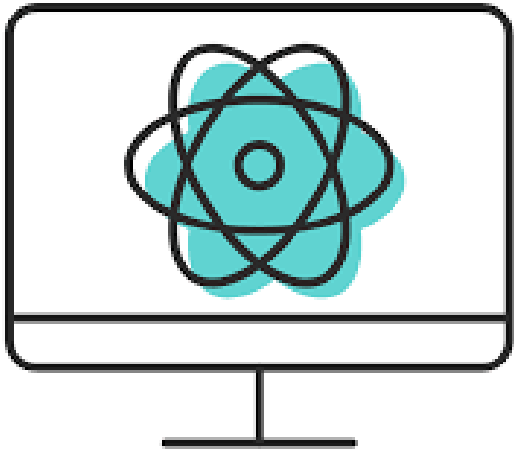
NIST 양자내성암호 표준화 공모전

코드기반 암호

성능 비교 분석

결론





양자 컴퓨터란?

기존 컴퓨터가 사용하는 일반적인 비트가 아닌
양자역학적 원리를 활용한 양자비트라는 개념을
사용하는 새로운 개념의 컴퓨터

양자 컴퓨터

양자 컴퓨터가 중요한 이유

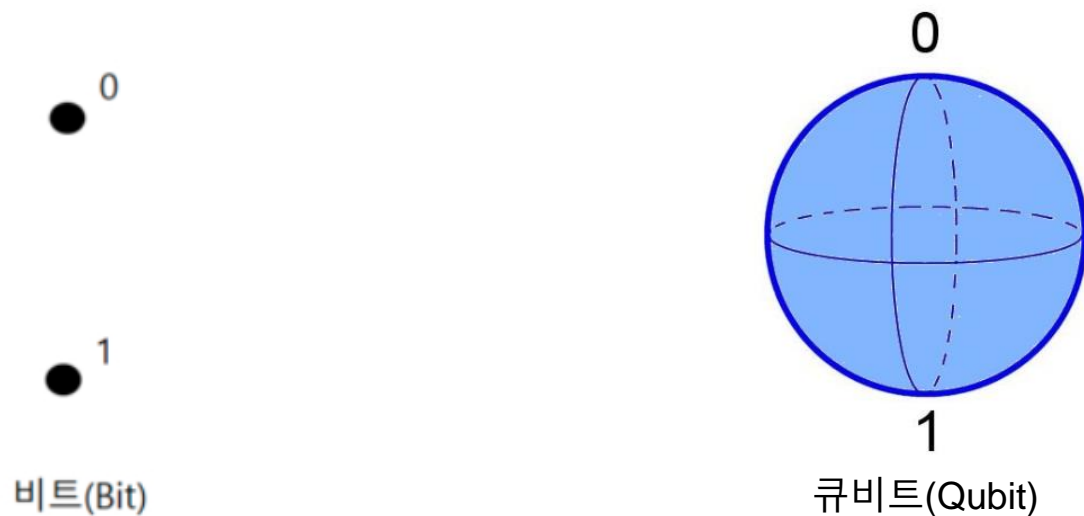
기존 컴퓨터에서 연산속도의 한계로 수행 불가능했던 문제에 대하여 뛰어난 해결능력을 가지고 있다.

양자 컴퓨터가 끼친 영향

현재 사용되는 수학적 난제에 기반한 암호시스템들은 기존 컴퓨터가 풀어내기 매우 어렵다. 하지만 양자 컴퓨터가 등장한다면 빠른 시간안에 풀어낼 수 있다.

✖ 암호체계의 붕괴 ✖

양자 컴퓨터 : 큐비트(Qubit)



0과 1 둘 중 하나를
결정하여 정보를 표현

0 이 표현될 수도 있고
1 이 표현될 수도 있다.

이렇게 자신의 상태가
확률로서 존재하는 것이 바로 큐비트

➡ 중첩(Superposition)

양자 컴퓨터 : 큐비트 (Qubit)

관측

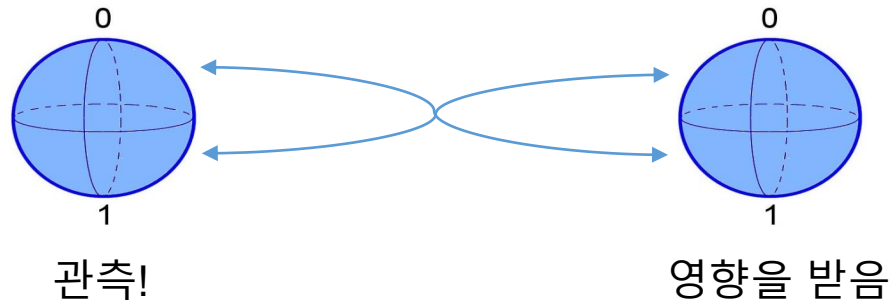
확률상태로 존재하던 큐비트의
상태를 결정하는 행위



얽힘

관측으로 인해 큐비트의 상태가 결정 되었을 때
관측된 큐비트와 얽혀 있는 다른 큐비트의 상태까지 결정되는 것

➡ 데이터가 한순간에 다른 곳으로 이동하는 것으로 보임



양자 컴퓨터 : 큐비트 (Qubit)

Bit

2bit → 00

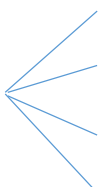
2bit → 01

2bit → 10

2bit → 11

- 한번에 한가지 정보 표현
- 직렬처리

Qubit

2 Qubit 

00
01
10
11

- 한번에 여러가지 정보 표현
- 병렬처리
- 메모리 공간 확보
- 지수승으로 증가 (예 : 3 큐비트 -> 8가지 정보 표현 가능)

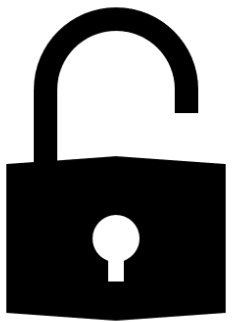
기존 컴퓨터



양자 컴퓨터



양자 알고리즘



양자 컴퓨터의 등장과 함께 나타난 새로운 양자 알고리즘

기존 컴퓨터에서 해결하기 어려운 수학적 난제들을 효율적으로 풀어냄

→ 이러한 난제에 기반한 현재 암호시스템들을 위협

양자 알고리즘 : Shor

1994년 수학자 Shor는 기존 컴퓨터에서의 난제인 **소인수분해 문제**를

효율적으로 풀어낼 수 있는 양자 알고리즘을 제안

커다란 두 소수를 곱하는 것은 쉽지만 이렇게 곱해진 매우 커다란 정수를 두 소수로 다시 분해하는 것은, 두 소수 중 하나를 모른다면 매우 어려운 일 예제) $N = pq$

$$O\left(e^{1.9(\log N)^{1/3}(\log \log N)^{2/3}}\right) \xrightarrow{\text{use Shor's Algorithm}} O\left((\log N)^2(\log \log N)(\log \log \log N)\right)$$

지수 차원의 복잡도

Before

→

다항시간내에 해결

After

결과 : 소인수 분해의 어려움에 기반한 암호시스템들을 무너뜨릴 수 있다. 예시) ~~RSA~~

양자 알고리즘 : Shor

소인수 분해를 푸는 과정에서

주기 찾기 알고리즘을 이용하여 $f(x) = a^x \bmod N$ 일 때
 $f(x + r) = f(x)$ 를 만족하는 차수 r 을 구한다.



주기 찾기 문제
(Order Finding)

$$f(x) = a^x \bmod N \quad N = 15, a = 7$$

예제

$$\begin{aligned} 7^1 &> 7 \bmod 15 = 7 \\ 7^2 &> 49 \bmod 15 = 4 \\ 7^3 &> 343 \bmod 15 = 13 \\ 7^4 &> 2401 \bmod 15 = 1 \\ 7^5 &> 16807 \bmod 15 = 7 \end{aligned}$$

답 : $r = 4$



양자 알고리즘 : Shor

Problem

$f(x + r) = f(x)$ 를 만족하는 차수 r 을 구한다.

Solution !

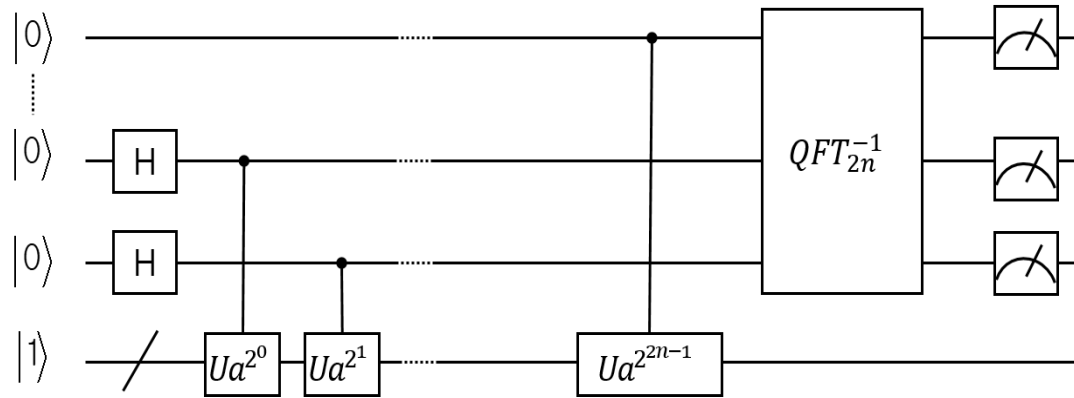
양자 컴퓨터가 여러 상태에 동시에 존재할 수 있다는 성질을 이용

함수 $f(x)$ 의 주기를 계산하기 위해서 모든 x 점에서의 함수 값을 동시에 계산한다.

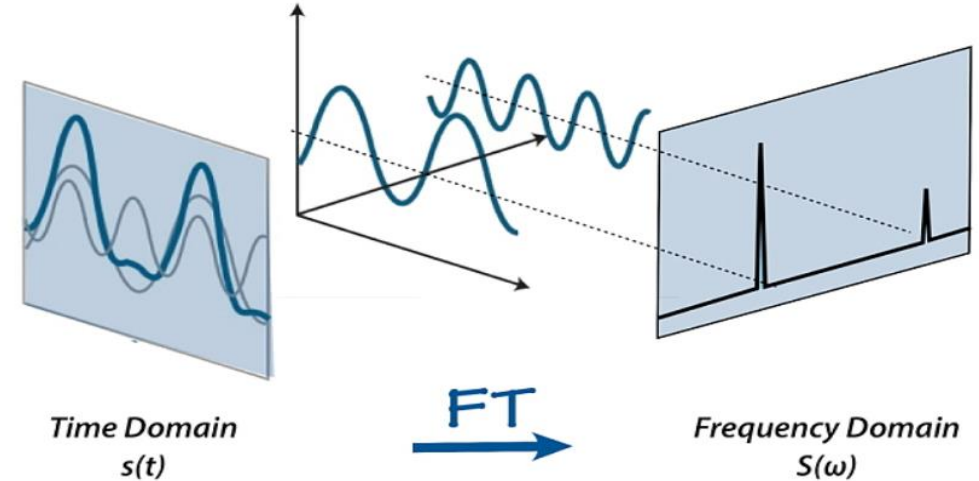
반복이 필요한 주기 찾기 작업을 한 번의 계산으로 가능하게 하여 계산 복잡도를 크게 낮춘다!

→ 쇼어 알고리즘을 이용하여 소인수분해 문제에 기반한 기존 암호시스템을 무너뜨린다.

양자 내성암호(Post Quantum Cryptography)



Shor 알고리즘 주기찾기 회로 ($f(x+r) = f(x)$)



양자 푸리에 변환

어떠한 양자 측정도 그 모든 계산 된 값을 전부 추출할 수는 없으나

함수의 출력 값의 광역적인 성질에 대해서 → **주기**

그 함수에 대한 정보를 얻어낼 수 있는 방법이 존재한다는 것

양자 알고리즘

양자컴퓨터 시대에 대한 현재 암호시스템의 상황 (NIST 발표)

유형	알고리즘	목적	영향
대칭키	AES	Encryption	키 길이 증가 필요
	SHA-2, SHA-3	Hash	출력 길이 증가 필요
공개키	RSA	Signature, Key establishment	더 이상 안전하지 않음
	ECDSA, ECDH	Signature, Key exchange	
	DSA	Signature, Key exchange	
	Diffie Hellman	Key exchange	

양자 내성암호(Post Quantum Cryptography)



Cryptography

양자역학의 발전과 빠르게 진화하는 양자컴퓨터에 대비하여
양자 컴퓨터의 계산능력에 내성을 가진 암호가 필요한 상황

양자내성암호(Post Quantum Cryptography)

NIST 양자내성암호 표준화 동향



양자내성암호(PQC)의 필요성 대두되는 상황, NIST는

PQC 표준화 공모전을 개최, 국제적으로 많은 암호알고리즘이 참가했고

제안된 알고리즘을 검증 및 평가해가며 후보 알고리즘을 점점 추려내고 있다.

진행상황

Round 1 : 자신들의 제시한 최소수용조건을 만족하는 알고리즘

Round 2 : 효율성 및 최적화구현 제시

4 NIST 양자내성암호 표준화 동향

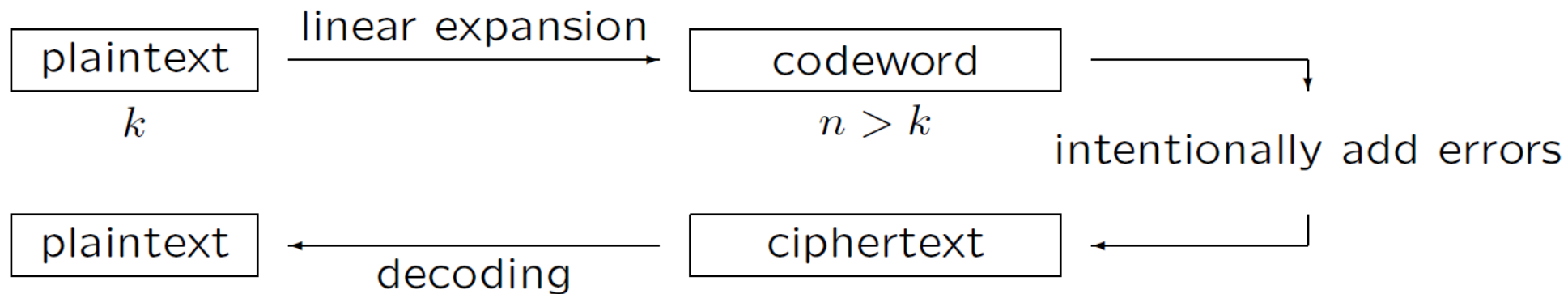
Round 1에서는 NIST가 제시한 성능 조건을 만족하는 69개의 양자내성암호 알고리즘이 선정

2019년 1월 30일, 표준화 단계가 Round 2로 진행됨에 따라 암호알고리즘의 안전성과 실용성에 대하여 평가, 69개에서 26개의 후보자로 좁혀졌다.

Round 2, 26개의 후보 알고리즘 표 →

유형	공개키암호/키관리	서명
격자	CRYSTALS-KYBER Frodo LAC Newcome NTRU NTRU Prime Round 5 SABER Three Bears	CRYSTALS-DILITHIUM FALCON qTESLA
코드	BIKE Classic McEliece HQC LEDACrypt NTS-KEM ROLLO RQC	
해쉬		SPHINCS+
다변수다항식		GeMSS LUOV MQDSS Rainbow
아이소제니	SIKE	

코드기반암호 - McEliece



- 통신채널에 적용되는 코딩이론에서 발전하여 암호시스템에도 적용됨 1978. Robert J. McEliece
- 대표적인 코드기반 암호
- 양자컴퓨터의 계산능력에 안전하다고 여겨져 다시 주목을 받음
- 40년동안 깨지지 않은 보안의 역사가 안전성을 증명
- Goppa 코드활용

NIST Round 2 - Classic McEliece

- 최초의 코드기반암호 McEliece의 업그레이드 버전
- 많은 암호팀들이 McEliece의 시스템에서 공개키의 크기를 줄이기 위해 바이너리 Goppa 코드를 다른 코드 군으로 대체 할 것을 제안했다.
 - 이러한 제안 중 많은 부분이 개인 키가 복구되는 취약점이 드러남
- 핵심강점은 보안
 - 기존의 Goppa 코드 대신 다른 코드를 사용함으로써 효율성을 늘리는 것보다 오랜기간 안전성을 지속한 Goppa 코드 사용을 고수
- 구현입장에선 키 생성, 배포비용에 투자해야 하지만 암호,복호화 속도가 빠르고 양자내성 암호 중 암호문이 매우 작은편
 - 한번 키 교환이 성립되면 효율적으로 안전한 통신이 가능

NIST Round 2 - Classic McEliece

Classic McEliece 성능 비교

단위 : ms

	Intel NUC			Raspberry Pii		
	KeyGen	Encaps	Decaps	KeyGen	Encaps	Decaps
mceliece348864	164.82	0.0696	23.88	1780.94	0.84	247.94
mceliece460896	339.86	0.15	56.6	3864.43	2.52	630.32
mceliece6688128	939.42	0.20	108.20	8987.07	2.74	9893.79
mceliece6960119	638.54	0.47	103.28	7003.93	6.64	1138.87
mceliece8192128	534.11	0.23	140.84	5738.65	3.05	1709.16

NIST Round 2 - BIKE

- BIKE의 모든 버전에서는 Bit-Flipping 디코더를 통해 복호를 효율적으로 수행하는 QC-MDPC 코드를 사용
 - Bit-Flipping 디코더는 가장 틀린 위치가 무엇인지 추정하고, 뒤집어서 그 결과가 이전보다 좋았는지 작은 syndrome weight 관찰
- Round 1에서는 비교적 높은 복호 실패율을 보여줬음
- Round 2에서는 향상된 Backflip 디코더라는 향상된 디코딩 기술을 적용함으로써 무시가능한 수준의 복호실패율을 달성함
- BIKE는 CPA - 1, 2, 3, CCA - 1, 2, 3 버전을 제공
 - 버전 1 : 빠른 임시 키를 사용
 - 버전 2 : 키 생성에 약간 더 높은 비용을 요구하지만 더 안전한 정적 키 사용

BIKE-2의 정적키 선택을 더 추천한다고 함.
그 이유는 사실 Classic McEliece가 주장하는 이유와 같
음

NIST Round 2 - BIKE

BIKE-2의 정적키 선택을 더 추천한다고 함.
그 이유는 Classic McEliece가 주장하는 이유와 같음

		Intel NUC			Raspberry Pi		
		KeyGen	Encaps	Decaps	KeyGen	Encaps	Decaps
CPA	BIKE-1 CPA(LEVEL1)	0.26	0.22	1.49	2.68	2.69	11.8
	BIKE-1 CPA(LEVEL3)	0.66	0.57	3.99	7.10	6.99	2.94
	BIKE-1 CPA(LEVEL5)	1.00	0.89	8.97	10.98	10.89	73.8
	BIKE-2 CPA(LEVEL1)	1.74	0.129	1.41	30.79	1.62	10.83
	BIKE-2 CPA(LEVEL3)	6.69	0.31	3.75	80.86	3.72	23.93
	BIKE-2 CPA(LEVEL5)	11.41	0.48	8.56	131.81	6.26	66.51
	BIKE-3 CPA(LEVEL1)	0.23	0.27	1.83	1.51	2.95	12.24
	BIKE-3 CPA(LEVEL3)	0.41	0.64	4.17	4.07	8.04	31.37
	BIKE-3 CPA(LEVEL5)	0.86	1.30	99.03	8.51	16.74	79.88
CCA	BIKE-1 CCA(LEVEL1)	0.37	0.28	1.85	3.40	3.46	19.14
	BIKE-1 CCA(LEVEL3)	0.89	0.73	4.40	9.08	9.70	49.17
	BIKE-1 CCA(LEVEL5)	1.74	1.60	9.45	20.53	20.97	113.41
	BIKE-2 CCA(LEVEL1)	2.28	0.16	1.61	37.86	1.86	16.11

BIKE-2 CCA(LEVEL3)	8.96	0.38	3.72	105.13	4.79	38.95
BIKE-2 CCA(LEVEL5)	18.62	0.83	8.02	231.88	10.79	93.15
BIKE-3 CCA(LEVEL1)	0.24	0.29	1.95	1.88	3.81	18.26
BIKE-3 CCA(LEVEL3)	0.62	8.09	4.51	5.21	10.66	50.94
BIKE-3 CCA(LEVEL5)	1.12	1.87	10.21	12.53	24.98	122.57

NIST Round 2 - NTS-KEM

NTS-KEM 성능 비교

	Intel NUC			Raspberry Pi		
	KeyGen	Encap	Decap	KeyGen	Encap	Decap
nts_kem_12_64	55.96	0.12	1.71	291.66	1.28	9.8
nts_kem_13_80	159.37	0.28	3.39	958.46	2.99	17.80
nts_kem_13_136	277.88	0.37	6.80	2513.81	3.892	35.57

NIST Round 2 - HQC

HQC 성능 비교

- 2018 년, HQC는 보안이 깨진 이력이 있음
 - Breaking the Hardness Assumption and IND-CPA Security of HQC Submitted to NIST PQC Project, Zhen Liu, Yanbin Pan, and Tianyuan Xie, CANS, LNCS 11124, pp. 344–356, 2018
- QC 코드는 암호체계에서 매우 유용하며 상당히 큰 키의 사이즈를 줄인다. 하지만 앞의 공격사례의 경우가 우려되는 것이 사실

	Intel NUC			Raspberry Pi		
	KeyGen	Encap	Decap	KeyGen	Encap	Decap
hqc-128-1	3.96	0.76	1.24	50.65	9.02	15.30
hqc-192-1	0.93	1.83	2.82	11.06	24.12	37.09
hqc-192-2	1.01	1.94	2.93	12.68	25.94	39.97
hqc-256-1	1.30	2.56	3.92	16.64	35.57	51.61
hqc-256-2	1.59	3.11	4.73	17.85	34.61	52.12
hqc-256-3	1.80	3.60	5.46	19.13	40.	60.05

NIST Round 2 - RQC

RQC 성능 비교

	Intel NUC			Raspberry Pi		
	KeyGen	Encaps	Decaps	KeyGen	Encaps	Decaps
rqc128	0.31	0.55	2.61	2.64	4.65	27.03
rqc192	0.46	0.91	5.75	4.19	8.50	64.91
rqc256	0.86	1.75	11.99	7.67	16.98	126.34

NIST Round 2 - ROLLO

ROLLO 성능 비교

	Intel NUC			Raspberry Pi		
	KeyGen	Encaps	Decaps	KeyGen	Encaps	Decaps
Rollo-I-128	0.84	0.17	0.53	8.19	1.21	4.27
Rollo-I-192	1.62	0.21	0.99	10.18	1.48	6.96
Rollo-I-256	1.70	0.31	1.55	18.94	2.48	12.46
Rollo-II-128	7.16	0.82	2.23	73.94	6.89	19.84
Rollo-II-192	7.70	0.92	2.62	89.42	7.97	24.07
Rollo-II-256	7.77	1.03	2.91	93.77	8.69	27.02
Rollo-III-128	0.16	0.30	0.49	1.67	2.62	3.98
Rollo-III-192	0.20	0.41	0.96	2.19	3.56	7.53
Rollo-III-256	0.37	0.73	1.65	3.46	5.94	12.66

NIST Round 2 - LEDAcrypt

LEDAcrypt 성능 비교

	Intel NUC			Raspberry Pi		
	KeyGen	Encap	Decap	KeyGen	Encap	Decap
LEDAcrypt ephemeral keys 1-2	8.66	0.737	3.041	79.62	7.89	29.91
LEDAcrypt ephemeral keys 1-3	2.641	0.548	3.301	28.16	5.71	34.09
LEDAcrypt ephemeral keys 1-4	2.599	0.706	5.250	27.95	7.63	53.12
LEDAcrypt ephemeral keys 3-2	21.237	1.344	7.945	208.15	12.94	80.64
LEDAcrypt ephemeral keys 3-3	8.801	1.429	8.078.	86.61	15.07	81.22
LEDAcrypt ephemeral keys 3-4	7.907	1.863	12.023	81.02	17.13	122.37
LEDAcrypt ephemeral keys 5-2	43.430	2.675	13.400	417.12	29.05	134.20
LEDAcrypt ephemeral keys 5-3	25.234	2.778	17.188	256.30	29.97	173.21
LEDAcrypt ephemeral keys 5-4	18.651	3.350	17.561	180.91	36.33	179.01
LEDAcrypt Longterm keys	390.139	2.811	3.466	3911.10	30.22	36.07

NIST Round 2 - LEDAcrypt

LEDAcrypt 성능 비교

LEDAcrypt Longterm keys 1-2(2^{128})	581.784	3.905	3.746	5722.01	41.01	38.91
LEDAcrypt Longterm keys 3-2(2^{64})	926.896	4.602	8.505	9273.56	45.12	85.15
LEDAcrypt Longterm keys 3-2(2^{192})	2135.948	9.984	9.274	20059.0	101.16	95.65
LEDAcrypt Longterm keys 5-2(2^{64})	3665.445	7.818	14.634	38056.9	80.12	149.94
LEDAcrypt Longterm keys 5-2(2^{256})	4467.717	18.410	17.787	42680.2	186.55	178.07
LEDAcrypt PKC 1-2(2^{64})	302.381	2.885	3.283	3126.65	29.01	33.73
LEDAcrypt PKC 1-2(2^{128})	432.130	4.125	4.303	4123.01	43.21	44.81
LEDAcrypt PKC 3-2(2^{64})	926.896	4.602	8.505	8971.91	48.72	87.15
LEDAcrypt PKC 3-2(2^{192})	2135.948	9.984	9.274	2317.43	102.31	93.01
LEDAcrypt PKC 5-2(2^{64})	4415.511	7.954	16.393	42458.7	81.22	162.53
LEDAcrypt PKC 5-2(2^{256})	4555.365	18.819	16.856	45355.2	189.27	170.51

결론

- 어떤 코드를 사용하는가?

- Goppa 코드 : 역사를 내세운 뛰어난 보안성이 강점, 하지만 효율성이 떨어짐
- QC 시리즈, MDPC, LDPC 등의 새로운 코드 : 성능이 Goppa 코드에 비해 비교적 높음, 하지만 검증된 기간이 길지 않기 때문에 지속적인 안전성 검증이 필요할 것

- 구현 환경에 따른 암호 적합성 평가

- 각 코드기반암호의 성능평가를 고성능 Intel 프로세서와 저전력 모바일 ARM 프로세서에서 수행
 - Intel NUC 와 같은 고사양의 환경에서는 디바이스의 메모리 공간이 여유롭기 때문에 코드기반암호의 단점인 대규모의 큰 키를 수용할 수 있음

- 구현 시나리오 : 고성능 Intel 프로세서

- 우선 키 생성, 배포비용에 투자를 수행하고 나면 개인키, 공개키 쌍은 오랜 기간 동안 배치될 수 있음. 따라서 코드기반암호 중 보수적인 Goppa 코드를 사용하여 키 생성 비용에 많은 투자가 필요하지만 상대적으로 준수한 암호, 복호화 성능을 가지고 있는 Classic McEliece와 NTS-KEM이 적합해 보임

결론

• 저전력 ARM 프로세서

- 성능평가 결과, ARM 프로세서 상에서 키 생성, 암호화/복호화 속도가 Intel 프로세서보다 약 10배 느림
 - 암호의 효율성이 떨어질수록 이 10 배의 수치는 더 큼, 때문에 저전력 환경 디바이스에 느린 속도의 암호를 탑재하기엔 무리가 있음
- 이렇게 제한된 상황에 어떤 암호가 적합한지는 통신과정에서 암호문을 얼마나 자주 교환 하느냐가 매우 중요
 - 주기적으로 잦은 통신이 필요하다면 우선 느린 암호, 복호화 속도를 가진 후보는 고려해 볼 수도 없음
 - 저장 공간이 허용하는 선에서 안전하고 장기간 사용 가능한 키를 사용한 빠른 통신이 이상적
하지만 여건이 충족되지 않는다면 ROLLO와 RQC와 같이 효율성을 중시한 코드기반암호가 최선일 것
- 코드기반암호를 저전력 환경에서 사용하기 위해선 QC시리즈 코드가 새롭게 등장하였듯 더 많은 연구가 필요
- 혹은 코드기반암호가 아닌 보다 더 효율적인 다른 기법의 양자내성암호 선택도 고려해야 함

결론

- 각 암호의 특성과 성능의 관계에 대하여 분석해 본 결과,
이 두 가지 환경에서조차 사용되기 적합한 암호들이 구분되었음
- 하지만 실제로 더욱 다양한 환경의 디바이스가 존재하기 때문에 각 분야에 알맞은
양자내성암호가 필요할 것 따라서 이번 NIST의 표준화 작업에선 하나가 아닌 여러가지의
양자내성암호가 다양한 분야에서 각자의 특성에 맞게 표준화되어 활용될 것
- 우리는 다가오는 이 양자 후 시대를 위한 NIST의 표준화 작업에 관심을 갖고 지켜봐야 하며
양자내성암호에 대한 지속적인 연구 또한 필요

Q & A

감사합니다

