Grover on CHAM : 병렬 CHAM 최적화 구현

양유진*, 장경배*, 김현지*, 임세진*, 서화정**

*한성대학교 (대학원생)

**한성대학교 (교수)

Grover on CHAM : Optimization Implementation of CHAM in Parallel

Yu-Jin Yang*, Kyung-bae Jang*, Hyun-Ji Kim*,

Se-Jin Lim*, Hwa-Jeong Seo**

*Hansung University(Graduate student)

**Hansung University(Professor)

요 약

국제 대기업의 투자에 따른 양자 컴퓨터의 성능 향상과 전수조사 복잡도를 $\sqrt{}$만큼 낮춰주는 Grover 알고리즘의 등장은 대칭키 암호의 위협을 야기할 수 있다. 양자 후 암호화 시스템의 보안 강도를 추정할 때 사용되는 Grover 알고리즘의 키 검색 비용은 양자 회로의 최적화 여부와 관련이 있다. 대칭키를 기반으로 두고 있는 경량 블록 암호를 최적화하기 위한 행렬 분해 연구들이 꾸준히 제시되고 있는데 이 방법들을 경량 블록 암호양자 회로 구현에 적용한다면 양자 회로 최적화에 큰 도움이 될 수 있다. 본 논문에서는 경량 블록 암호 CHAM의 키 스케줄에 PLU 분해 방법을 적용하여 양자 회로를 최적화한 후, 각 방법마다 추정한 Grover 알고리즘 공격 비용을 비교 및 평가한다.

I. 서론

국제 대기업들의 적극적인 투자에 따른 양자 컴퓨터의 성능 향상은 현재의 암호화 시스템의 위협을 야기할 가능성을 높여준다. 특히, 전수조사 복잡도를 $\sqrt{}$만큼 감소시켜주는 검색 알고리즘인 Grover 알고리즘을 대칭키 암호의 비밀키 전수조사에 적용하면 대칭키 암호의 안전성이 크게 떨어질 수 있다. 이에 NIST는 양자 후 암호화 시스템의 보안 강도를 추정할 수 있게 AES의 비밀키 공격에 사용되는 Grover 알고리즘 비용을 기준으로 제시하였다[1]. 보안 강도를 측정하기 위해 추정하는 대칭키 암호에 대한 Grover 공격의 최종 비용은 암호의 효율적인 양자 회로 구현과 관련이 있기에 양자 회로를 최적화 구현하는 것이 중요하다.

대부분이 대칭키 암호인 경량 블록 암호는 비슷한 성능에 비용을 줄이는 것이 필수적이기에 경량 환경에서 더 효율적으로 사용하기 위하여 대칭키를 선형계층에서 최적화 구현하는 방법들이 꾸준히 논의되고 있다. PLU 분해, 가우스-조던 소거법 등 행렬 분해를 활용한 방법들을 대칭키에 적용하여 최적화하는 연구들[2]이 이에 해당한다. 따라서 다양한 행렬 분해 방법들을 경량 블록 암호 양자 회로 구현에도 적용한다면 회로 최적화와 Grover 공격의 최종 비용 절약에 큰 도움이 될 수 있을 것이다.

본 논문에서는 국산 경량 블록 암호 중 하나인 CHAM의 키 스케줄에 PLU 분해 방법을 적용하여 양자 회로를 최적화하고, 추정한 Grover 알고리즘 공격 비용을 비교 및 평가한다.

II. 관련연구

2.1 CHAM

ICISC'17에서 발표된 CHAM은 국산 초경량 블록 암호로, 저사양 사물인터넷 플랫폼을 대상으로 하며 Addition, Rotation, XOR 연산을 사용한다[3]. [4]는 CHAM의 양자 회로를 최적화

한 논문으로 depth를 줄이기 위하여 추가큐비트를 할당하고 병렬 연산을 진행하였다.

2.2 양자 게이트

기본적인 양자 게이트는 4가지가 있다. SWAP 게이트는 입력으로 들어온 두 큐비트의 값을 교환하는 역할을 수행한다. 큐비트를 반전시키는 X 게이트는 논리 게이트의 NOT 게이트와 유사하다. CNOT 게이트는 제어 큐비트가 1일 경우 대상 큐비트를 반전시켜주는 연산을 수행한다. Toffoli 게이트는 제어 큐비트 2개가 모두 1일 경우 대상 큐비트를 반전시킨다.

2.3 PLU 분해(PLU Decomposition)

PLU 분해는 순열 P, 하삼각행렬 L, 상삼각행렬 U로 구성되어 있다. 행렬 L은 하삼각행렬을 제외한 부분의 값이 0이고, U는 상삼각행렬을 제외한 부분이 모두 0이다. 순열 P 전에 수행하는 LU 분해는 임의의 행렬 A를 행렬 L, U의 곱으로 표현하는 것을 의미한다. 이는 연립선형방정식의 해나 역행렬을 구하는 데 사용될 수 있다.

III. 제안기법

본 논문에서는 [4]의 키 스케줄을 개선하였다. 파라미터간 과정이 유사하기 때문에 CHAM 64-128을 예시로 들어 설명한다.

3.1 CHAM의 Key Schedule

CHAM의 round key RK 행렬은 CHAM의 키 스케줄 Equation 1에 의해 생성되었다. i의 범위는 $(0 \le i \le 8)$이고, $RK_0 \sim RK_{15}$까지 존재한다[3].

$$RK[i] = K[i] \oplus (K[i] \lll 1) \oplus (K[i] \lll 8),$$
$$RK[(i+k/w) \oplus 1] = K[i] \oplus (K[i] \lll 1) \oplus (K[i] \lll 11) \cdots (1)$$

RK는 인덱스에 따라 2가지 방식으로 생성되므로 총 2개의 행렬이 나온다. 예를 들어, $RK_{0\sim7}$의 첫 번째 행은 Equation 1의 첫 번째 식에 따라 [1000000010000001]로 구성된다. $i=1$이고 $K[i]$가 1열에 있을 때, $\lll 1$한 것이 16열의 1이고, $\lll 8$한 값이 8열의 1이다. CHAM-64/128의 경우 워드 크기 n이 16이므로 위 과정을 거쳐 나온 행렬은 16x16의 크기를 가진다. 나머지 파라미터에 따른 RK 행렬들도 sage math를 활용하여 만들었다.

3.2 PLU 분해를 적용한 CHAM 키스케줄

PLU 분해를 활용하여 RK를 생성하기 위해선 3.1에서 언급한 두 행렬에 따른 P, L, U 행렬을 각각 얻어야 한다. 이 또한 sage math의 LU() 함수를 활용하면 쉽게 획득할 수 있다.

Algorithm 1: Quantum circuit implementation of Apply_PLU
Input: key $K_{0\sim7}$, word size n, upper triangular matrix $U_{1\sim2}$, lower triangular matrix $L_{1\sim2}$
Output: Round key $RK_{0\sim7}$

```
1: Transform K0~7 :
   // Apply_U
2:   for i = 0 to n - 2
3:     for j = 0 to n - i - 2
4:       if U1~2[(i * n) + 1 + i + j] == 1
5:         CNOT (K[i + j + 1], K[i])
   // Apply_L
6:   for i = 0 to n - 2
7:     for j = n - 1 to i + 1 step -1
8:       if L1~2[n * (n - 1 - i) + n - 1 - j] == 1
9:         CNOT (K[n - 1 - j], K[n - 1 - i])
   // Apply_P
10:  if RK0~7 == True
11:    SWAP (K[12], K[11]), SWAP (K[11], K[13])
12:    SWAP (K[11], K[10]), SWAP (K[10], K[14])
13:    SWAP (K[10], K[9]), SWAP (K[9], K[15])
14:  return RK0~7
15:  else
16:    SWAP (K[11], K[13]), SWAP (K[10], K[9])
17:    SWAP (K[7], K[8]), SWAP (K[5], K[6])
18:  return RK8~15
19: Reverse(transform K0~7) // Reverse operation
```

(알고리즘 1) Apply_PLU 양자 회로 구현

알고리즘 1은 PLU 분해를 활용한 RK 생성을 양자 회로로 구현한 것이다. 각 행렬에서 n열은 K[n-1]을 나타낸다. U와 L의 경우 입력 행렬만 다를 뿐 적용하는 알고리즘은 동일하기에 같은 연산이 쓰는 반면 P는 RK 인덱스에 따라 각각 다른 SWAP 게이트가 적용된다. $RK_{0\sim7}$에 U_1, L_1이 입력으로 사용되고 $RK_{8\sim15}$은 U_2, L_2가 입력으로 사용된다.

U 과정은 행렬의 1행에서 16행 방향으로 진행되며 행에서는 행렬 U에서 주대각선 기준으로 대각항의 위쪽 항들만 의미가 있다. 따라서 오른쪽을 탐색하며 1인 값이 나오면 해당하는 값 K[1+i+j]를 제어큐비트로 주대각선 K[i]를 대상큐비트로 두어 CNOT을 적용한다.

L은 반대로 행렬의 16행에서 1행 방향으로 진행되고, U와 마찬가지로 주대각선의 열에 다른 열

의 값들이 XOR 된다. line 6~9가 이를 수행하는 과정으로 U와 마찬가지로 주대각선 기준 왼쪽을 탐색하며 1인 값이 나오면 해당하는 값과 주대각선 값을 CNOT 한다.

line 10~14와 line 15~18은 P를 수행하는 부분으로 각각 $RK_{0\sim7}$, $RK_{8\sim15}$를 반환하기 위해 SWAP 게이트를 활용하여 자리를 교환한다. line 19는 키 $K_{0\sim7}$를 재활용하기 위하여 Transform 범위에 포함된 양자 게이트들에 대해 reverse 연산을 수행하는 부분이다.

IV. 평가

<표 1> CHAM Quantum Resources Comparison by Methods Applied to Key-schedule

method	CHAM	qubits	CNOT	Depth
prev[4]	64/128	204	27,120	17,035
	128/128	292	58,040	37,766
	128/256	420	70,080	45,252
PLU	64/128	195	35,280	17,092
	128/128	259	81,280	37,878
	128/256	387	95,536	45,014

<표 2> Comparison of Cost of Grover key search by Methods Applied to Key-schedule for CHAM

method	CHAM	qubits	Total gates	Total depth	Cost
prev[4]	64/128	409	1.206×2^{80}	1.633×2^{78}	1.970×2^{158}
	128/128	293	1.280×2^{81}	1.810×2^{79}	1.159×2^{161}
	128/256	841	1.542×2^{145}	1.085×2^{144}	1.672×2^{289}
PLU	64/128	391	1.402×2^{80}	1.639×2^{78}	1.148×2^{159}
	128/128	260	1.559×2^{81}	1.816×2^{79}	1.415×2^{161}
	128/256	775	1.871×2^{145}	1.079×2^{144}	1.009×2^{290}

<표 1>은 CHAM의 모든 파라미터에 대한 양자 회로 구현 비용을 나타내며 위에서부터 기존의 방법, PLU 분해가 사용된 것이다. 다른 게이트 비용들은 동일하기에 비교를 위해 값이 다른 항목들만 가져왔다. 기존의 CHAM 구현과 비교해보면 PLU는 in-place로 회로가 구성되었기 때문에 보조 큐비트를 추가로 사용하지 않아 큐비트 수를 줄일 수 있었다.

Oracle은 암호화 연산과 reverse 연산을 차례로 동작하는데 여기에 각각 CHAM 양자 회로가 적용되기에 Oracle을 한 번 수행하는 데 발생되는 비용은 큐비트를 제외하고 <표 1> × 2이다. CHAM에 Grover 공격을 시도하면 Oracle이 $\sqrt{2^{128}}$번 반복되기에 최종 비용은 <표 1> × 2 × 2^{64}로 추정하며 그 결과가 <표 2>이다.

NIST에서 제공하는 양자 후 보안 강도 기준[1]에 따르면 128/256은 Level 3에 해당하는 2^{333}을 만족하였으나 64/128과 128/128은 Level 1에 해당하는 2^{170} 을 넘지 못했다. 그러나, [1]에서 추후 공격 비용이 크게 감소된 양자 공격이 등장할 경우 추정한 비용들이 보수적으로 간주되어야 한다고 언급되어있기에 충족하지 못한 값들에 대해서 보수적으로 평가하여야 한다. [1]의 기준이 되었던 AES에 대한 최신 Grover 공격 비용 추정 연구[5]에 따르면 Level1이 2^{157}이기에 2가지 방법 모두 보안 강도를 만족한다고 볼 수 있다.

V. 결론

본 논문에서는 병렬로 구현되었던 기존의 CHAM 양자 회로를 최적화하기 위하여 키 스케줄에 PLU 분해를 적용하여 얻은 양자 자원을 [4]와 비교하였다. 기존의 방법보다는 새로운 방법들이 큐비트 관점에서 효율적이라고 볼 수 있다. 추후에는 PLU 분해 외에도 추가적으로 회로를 최적화할 수 있는 방안을 찾아 경량 블록 암호에 적용하고 비교하여 최적의 방법을 찾을 예정이다.

VI. Acknowledgment

This work was supported by Institute for Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government(MSIT) (<Q|Crypton>, No.2019-0-00033, Study on Quantum Security Evaluation of Cryptography based on Computational Quantum Complexity, 100%).

[참고문헌]

[1] NIST, Submission requirements and evaluation criteria for the post-quantum cryptography standardization process

,https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf.

[2] Van Hoof, I. Space-efficient quantum multiplication of polynomials for binary finite fields with sub-quadratic Toffoli gate count. arXiv preprint arXiv:1910.02849, 2019.

[3] B.W.Koo, D.Y.Roh, H.J.Kim, Y.H.Jung, D.G.Lee and D.S.Kwon, CHAM: A family of lightweight block ciphers for resource-constrained devices, International Conference on Information Security and Cryptology, pp. 3-25, 2017.

[4] K.B.Jang, G.J.Song, H.J.Kim, H.D. Kwon, H.J.Kim, & H.J.Seo, Parallel quantum addition for Korean block cipher. Cryptology ePrint Archive, 2021.

[5] K.B. Jang, A. Baksi, G.J. Song, H.J. Kim, H.J. Seo, A. Chattopadhyay, Quantum Analysis of AES, Cryptology ePrint Archive, 2022.