논문 20-45-12-20 The Journal of Korean Institute of Communications and Information Sciences '20-12 Vol.45 No.12
https://doi.org/10.7840/kics.2020.45.12.2154

이진 Goppa 부호기반 암호시스템 구현

유 기 순•, 임 대 운°

Implementation of Binary Goppa Code-Based Cryptography

Ki-Soon Yu•, Dae-Woon Lim°

요 약

Classic McEliece은 NIST에서 진행 중인 양자 내성 공개키 암호알고리즘 3라운드 공모에서 경합 중인 암호 알고리즘 중 하나이다. 해당 알고리즘은 이진 Goppa 부호와 Niederreiter 암호시스템을 기반으로 하고 있다. 본 논문에서 우리는 Niederreiter 암호시스템과 같이 부호를 기반으로 하는 McEliece 암호시스템을 구현하였다. 구현한 암호시스템의 성능 평가를 위해 복호 시간을 측정하고 사이클 수를 분석하였다. 또한 Patterson 알고리즘과 Berlekamp-Massey 알고리즘을 구현하고, 각 알고리즘별 디코딩 사이클 수를 비교하였다.

키워드 : 이진 Goppa 부호, McEliece 암호시스템, Patterson 알고리즘, Berlekamp-Massey 알고리즘, 부호기반 암호시스템

Key Words : binary Goppa code, McEliece cryptosystem, Patterson algorithm, Berlekamp-Massey algorithm, Code-based Cryptography

ABSTRACT

Classic McEliece is among the 3rd round finalists in the Post-Quantum Cryptography(PQC) Competition held by NIST. The submitted algorithm is based on both Niederreiter cryptosystem and the binary Goppa code. In this paper, we have regenerated the McEliece cryptosystem which is a code-based cryptosystem like the Niederreiter cryptosystem. We measure the execution time and analyze the decryption cycles to evaluate the performance of our implementation. We also present a comparative assessment of Patterson and Berlekamp-Massey decoding algorithms based on the number of cycles for each algorithm.

Ⅰ. 서 론

양자컴퓨터 기술 발전과 더불어 암호알고리즘에 대한 위협이 커짐에 따라 양자컴퓨터에서 안전성을 보장함은 물론 기존 컴퓨터(classic computer)에서도 효율적 연산이 가능한 암호알고리즘에 대한 기술 확보의 일환으로 2016년 미국 표준기술원 NIST는 다양한 난제에 기반을 둔 양자 내성 공개키 암호알고리즘(Post-Quantum cryptography)을 공모하였다. 2020년 현재 3라운드를 진행 중에 있으며, 공개키 암호알고리즘 및 키 설정분야에서 Classic McEliece, CRYSTALS-KYBER, NTRU, SABER이 경합 중에 있다.

본 논문에서는 Classic McEliece에서 참고하고 있는 McEliece 암호시스템을 소프트웨어로 구현하였다. 구현한 암호시스템을 3.40GHz Intel Core(TM)

※ 본 연구는 동국대학교 교내 연구과제 지원사업(S-2020-G0041-00016)의 지원을 받아 수행되었습니다.
• First Author : Department of Information Communication Engineering, Dongguk University, ykscj39@naver.com, 학생회원
° Corresponding Author : Department of Information Communication Engineering, Dongguk University, daewoonlim@gmail.com, 종신회원
논문번호 : 202011-283-D-RU, Received November 12, 2020; Revised November 26, 2020; Accepted November 26, 2020

i7-6700 CPU에서 실행하였을 때, 1 바이트 당 암·복호화 사이클 수를 측정하고, 이진 Goppa code의 디코딩 알고리즘 별로 측정 결과를 비교한다.

Ⅱ. 이진 Goppa 부호 및 부호기반 암호시스템

2.1 이진 Goppa 부호[1]

이진 Goppa 부호 $\Gamma(L, g(x))$는 확대체(extension field) F_{2^m}상에서 t차 Goppa 다항식 $g(x)$와 위치집합(location set) L에 의해 정의된다.

$$g(z) = g_0 + g_1 z + \cdots + g_t z^t = \sum_{i=0}^{t} g_i z^i \in F_{2^m}[z]$$

$$L = \{\alpha_1, \alpha_2, \cdots, \alpha_n\} \subseteq F_{2^m}$$

이 때, $g(\alpha_j) \neq 0, \forall \alpha_j \in L$이다.

이진 Goppa 부호의 부호어(codeword)는 다음을 만족하는 벡터 $\boldsymbol{c} = (c_1, c_2, \cdots, c_n)$로 구성된다.

$$R_c(z) = \sum_{i=1}^{n} \frac{c_i}{z - \alpha_i} \equiv 0 \pmod{g(z)}$$

앞서 보았듯이 $g(\alpha_i) \neq 0$이고, $z - \alpha_i \in F_{2^m}[z]/g(z)$이므로, $(z-\alpha_i)(z-\alpha_i)^{-1} \equiv 1 \pmod{g(z)}$를 만족하는 역원(inverse) $(z-\alpha_i)^{-1}$은 항상 존재한다.

이진 Goppa 부호의 파라미터 (n, k, d)는 다음을 의미한다.

- 부호어 길이 $n = |L| \leq 2^m$
- 차원 $k \geq n - mt$
- 최소거리 $d \geq t + 1$

선형부호(linear code)에서 최소해밍거리(minimum distance) d는 최소해밍무게(minimum weight)와 동일하며, d에 의해 선형부호의 오류정정능력(error-correcting capability)이 결정된다.

이진 Goppa 부호에서 d는 Goppa 다항식 $g(z)$에 따라 달라진다. $g(z)$가 중근(repeated roots)을 갖지 않는 분리 다항식(separable polynomial)이거나 기약 다항식(irreducible polynomial)이면, $d \geq 2t + 1$가 된다. 따라서 두 다항식을 사용할 경우, 정정 가능한 최대 오류 개수는 t가 된다. 단, $g(z)$가 분리 다항식이면, 디코딩 시 $g^2(z)$ 기반의 패리티 행렬이 사용된다.

표 1. 다항식 별 오류정정능력
Table 1. Error correcting capability based on polynomial type

Type of $g(z)$	Error correction capability	H for decoding
Easier	$\leq \left\lfloor \frac{t}{2} \right\rfloor$	H of $\Gamma(L, g(x))$
Separable	$\leq t$	H of $\Gamma(L, g^2(x))$
Irreducible	$\leq t$	H of $\Gamma(L, g(x))$

이진 Goppa 부호의 패리티 행렬(parity-check matrix) $H = CXY$는 다음과 같다.

$$C = \begin{pmatrix} g_t & g_{t-1} & \cdots & g_1 \\ 0 & g_t & \cdots & g_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g_t \end{pmatrix}, \quad X = \begin{pmatrix} \alpha_1^{t-1} & \alpha_2^{t-1} & \cdots & \alpha_n^{t-1} \\ \alpha_1^{t-2} & \alpha_2^{t-2} & \cdots & \alpha_n^{t-2} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix},$$

$$Y = \begin{pmatrix} g(\alpha_1)^{-1} & 0 & \cdots & 0 \\ 0 & g(\alpha_2)^{-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & g(\alpha_n)^{-1} \end{pmatrix}$$

C는 정칙행렬(non-singular matrix)이므로, 패리티 행렬로 $H = XY$만을 사용해도 된다.

생성 행렬(generator matrix) G의 행공간(row space)은 H의 영공간(null space)의 벡터로 구성된다.

벡터 $\boldsymbol{w} \in F_2^m$에 대한 신드롬(syndrome) $s(z)$는 아래와 같이 계산된다.

$$s(z) = \sum_{i=1}^{n} \frac{w_i}{z - \alpha_i} \pmod{g(z)} \tag{1}$$

$s(z) = 0$이면 $\boldsymbol{w} \in \Gamma$이다. $s(z) \neq 0$이면 $\boldsymbol{w} \notin \Gamma$이므로, 오류위치 다항식(error locator polynomial) $\sigma(z)$를 찾아 오류위치 집합 B를 구한다. $\sigma(z)$를 계산하는 대표적인 알고리즘으로 Patterson 알고리즘, Berlekamp-Massey 알고리즘 등이 있으며 상세 내용은 3장에서 소개한다.

$$B = \{i | \sigma(\alpha_i) = 0\}$$

B를 기반으로 오류벡터 $\boldsymbol{e}$를 정의한다.

$$\boldsymbol{e}=(e_1, e_2, \cdots, e_n), e_i \begin{cases} 0, i \notin B \\ 1, i \in B \end{cases}$$

끝으로 $\boldsymbol{e}$와 $\boldsymbol{w}$를 더해 오류를 정정한다.

$$\boldsymbol{c}=\boldsymbol{w}+\boldsymbol{e}$$

2.2 부호기반 암호시스템

1978년 McEliece은 오류정정부호를 활용한 공개키 암호시스템을 소개했다. McEliece 암호시스템은 정칙행렬(non-singular matrix) S와 치환행렬(permutation matrix) P를 사용해 오류정정부호의 생성 행렬 G를 숨기고, 이를 공개키로 사용한다.

$G'=SGP$로 생성한 부호어에 오류벡터를 더해 암호문을 생성하고, 오류정정부호의 디코딩 알고리즘 Dec을 사용해 암호문을 복호한다.

2.2.1 McEliece 암호시스템[2]

McEliece 암호시스템은 $k\times k$ S, $n\times n$ P, $k\times n$ G를 사용해 공개키와 개인키를 생성한다.

- 개인키: S, G, P
- 공개키: $G'=SGP, t$
- 암호화: $\boldsymbol{y}=\boldsymbol{m}G'+\boldsymbol{e}$
  * 평문 $\boldsymbol{m}\in\{0,1\}^k$
  * 오류벡터 $\boldsymbol{e}\in\{0,1\}^n, wt(\boldsymbol{e})=t$
  * 암호문 $\boldsymbol{y}\in\{0,1\}^n$
- 복호화
  1. $\boldsymbol{w}'=\boldsymbol{y}P^{-1}$
  2. $\boldsymbol{m}'=Dec(\boldsymbol{w}')$
  3. $\boldsymbol{m}=\boldsymbol{m}'S^{-1}$

2.2.2 보안수준

McEliece 암호시스템의 안전성은 선형부호에 대한 일반적인 복호화 문제(decoding problem)가 NP-complete라는 사실에 기인한다[3].

$G=[I_k|Q]$가 체계적 형태(systematic form)일 때, 길이가 n인 암호문 $\boldsymbol{y}$와 G'이 주어지면, 다음과 같은 방법으로 평문 $\boldsymbol{m}$을 얻을 수 있다.

1. $\boldsymbol{y}_k=\boldsymbol{m}G'_k+\boldsymbol{e}_k$
2. $\boldsymbol{m}=\boldsymbol{y}_kG'^{-1}_k$

주어진 암호문 $\boldsymbol{y}$에서 오류를 포함하지 않은 $\boldsymbol{y}_k$를 얻을 확률 p_k은 아래와 같다.

$$p_k=\frac{\binom{n-t}{k}}{\binom{n}{k}}$$

G'의 $k\times k$ 부분행렬 G'_k의 역행렬 G'^{-1}_k를 얻는데 필요한 계산량은 $k^a\,(2\le a\le 3)$으로, 임의의 암호문에서 개인키 없이 평문을 얻는데 필요한 work factor는 다음과 같다[4].

$$W=k^a p_k$$

암호시스템 공격에 필요한 work factor는 해당 암호시스템의 보안수준(security level)을 나타낸다.

Stern 알고리즘은 가장 널리 알려진 McEliece 암호시스템 공격 기법으로, 다음 행렬에서 최소해밍무게를 가지는 벡터, 즉 오류벡터를 찾아 암호문을 해독한다.

$$\begin{pmatrix} G' \\ \boldsymbol{m}G'+\boldsymbol{e} \end{pmatrix}$$

Stern 알고리즘의 work factor는 아래와 같다. 이때 정수 p, l은 $0<p\le t$, $0\le l\le n-k$이다[6,7].

$$W=\frac{\left(\frac{1}{2}(n-k)^2(n+k)+2\binom{k/2}{p}pl+\binom{k/2}{p}^2 p(n-k)/2^l\right)}{\left(\binom{k/2}{p}^2\binom{n-k-l}{t-2p}/\binom{n}{t}\right)}$$

Bernstein-Lange-Peters는 [5]에서 Stern 알고리즘의 연산을 보다 효율적으로 할 수 있는 기법을 소개했다. 해당 기법은 $n=1024, k=524, t=50$에 대한 기존 Stern 알고리즘의 work factor $2^{66.2}$를 $2^{60.55}$로 줄였다. 해당 기법을 한 번 수행할 때 요구되는 연산량은 아래와 같다[6].

$$\begin{aligned} &(n-1)\left((k-1)\left(1-\frac{1}{q}\right)+(q-r)\right)\frac{c}{r} \\ &+\left(\left(\frac{k}{2}-p+1\right)+2\binom{k/2}{p}(q-1)^p\right)l \\ &+\frac{q}{q-1}(t-2p+1)2p\left(1+\frac{q-2}{q-1}\right)\frac{\binom{k/2}{p}^2(q-1)^{2p}}{q^l} \end{aligned} \tag{2}$$

여기서 정수 $r\le c$, 정수 c는 알고리즘에서 G'에

대한 Gaussian reduction 수행 시 교환(swap) 가능한 열의 개수를 의미한다. 적절한 c와 관련한 상세내용은 [5]에서 확인할 수 있다. Bernstein-Lange-Peters의 work factor는 알고리즘을 한번 수행했을 때 요구되는 연산량과 암호문 해독에 필요한 알고리즘의 평균수행 횟수로 구해진다. Bernstein-Lange-Peters 알고리즘은 이전 작업에 의존하여 반복적으로 수행되기 때문에, 오류를 찾는 과정을 $t+2$단계 정의하고,

- 0: 선택 해제 된 k열에 오류가 없음
- 1: 선택 해제 된 k열에 1개의 오류가 있음
- …
- t: 선택 해제 된 k열에 t개의 오류가 있음
- Done: 공격 성공

알고리즘을 수행을 때 다른 단계로 이동할 확률을 계산하여 평균수행 횟수를 구한다. 단계 u에서 $u+d$ 단계까지 알고리즘을 반복할 확률은 아래 같다.

$$\sum_i \frac{\binom{t-u}{i}\binom{n-k-t+u}{c-i}\binom{u}{d+i}\binom{k-u}{c-d-i}}{\binom{n-k}{c}\binom{k}{c}} \quad (3)$$

단, $2p$단계는 아래 확률로 정의하고 Done 단계로 이동한다[6].

$$\beta = \frac{\binom{k/2}{p}^2}{\binom{k}{2p}} \frac{\binom{n-k-t+2p}{l}}{\binom{n-k}{l}}$$

Bernstein-Lange-Peters의 work factor는 아래와 같다. 본 논문의 보안수준은 Bernstein-Lange-Peters의 work factor를 기준으로 한다.

$$W = (2) \times (3)$$

Ⅲ. 디코딩 알고리즘 및 구현

본 논문은 이진 Goppa 부호를 사용해 McEliece 암호시스템을 구현하고 그 성능을 측정한다.

3.1 이진 Goppa 부호의 디코딩 알고리즘

앞서 언급하였듯이 수식 (1)의 결과가 $s(z) \neq 0$일 경우, 이진 Goppa 부호의 다양한 디코딩 알고리즘을 사용해 부호어에 포함된 오류를 정정할 수 있다.

디코딩 알고리즘으로 얻은 오류위치 다항식 $\sigma(z)$는

$$\sigma(z) = \prod_{i \in B}(z - \alpha_i)$$

오류 비트의 위치에 해당하는 위치집합 L의 원소의 곱으로 만들어 지므로, 오류 개수에 따라 차수 $\deg \sigma(z) \leq t$가 정해진다.

3.1.1 Patterson 알고리즘

Patterson 알고리즘은 가장 많이 알려진 이진 Goppa 부호의 디코딩 알고리즘으로, 최대 t개의 오류를 정정할 수 있다.

Algorithm 1 Patterson algorithm[1]

Inputs: $\boldsymbol{w}, H$
Outputs: $\sigma(z)$
Step 1. $s(x) \leftarrow \boldsymbol{w}H^T$
Step 2. $s(z)h(z) \equiv 1 (\mathrm{mod}\, g(z))$
Step 3. $d(z) \leftarrow \sqrt{h(z)+z} (\mathrm{mod}\, g(z))$
Step 4. $a(z), b(z) \leftarrow a(z) \equiv b(z)d(z) (\mathrm{mod}\, g(z))$
Step 5. $\sigma(z) \leftarrow a^2(z) + b^2(z)z$

Algorithm 1의 Step 3에서 $a(z)$와 $b(z)$는 $\sigma(z)$의 짝수 차수 다항식(even polynomial)과 홀수 차수 다항식(odd polynomial)에 해당하며, $\deg a(z) \leq t/2$, $\deg b(z) \leq (t-1)/2$이다.

3.1.2 Berlekamp-Massey 알고리즘

Berlekamp-Massey 알고리즘은 $g^2(z)$를 이용해 최대 t개의 오류를 정정할 수 있다. 디코딩 시 Berlekamp-Massey 알고리즘의 패리티 행렬은 다음과 같다[8].

$$H = XY = \begin{bmatrix} \frac{\alpha_1^{2t-1}}{g(\alpha_1)} & \frac{\alpha_2^{2t-1}}{g(\alpha_2)} & \cdots & \frac{\alpha_n^{2t-1}}{g(\alpha_n)} \\ \frac{\alpha_1^{2t-2}}{g(\alpha_1)} & \frac{\alpha_2^{2t-2}}{g(\alpha_2)} & \cdots & \frac{\alpha_n^{2t-2}}{g(\alpha_n)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{g(\alpha_1)} & \frac{1}{g(\alpha_2)} & \cdots & \frac{1}{g(\alpha_n)} \end{bmatrix}$$

Algorithm 2에서 $\xi=1$이면 오류벡터의 첫 번째 비트 $e_1=1$, $\xi\neq 1$면 $e_1=0$으로 설정한다. $\sigma(z)$는 $\epsilon(z)$의 상반방정식(reciprocal polynomial)이다.

3.2 구현방법

F_{2^m}의 원소는 차수가 $m-1$보다 작은 이진 다항식으로 표현되므로, 우리는 원소를 m-비트의 이진수로 나타낼 수 있다. 2의 나머지(modulo) 연산에서 $1\equiv -1 \bmod 2$이므로 뺄셈은 덧셈과 같고, 덧셈연산의 결과는 XOR연산의 결과와 동일하다.

Algorithm 2 Berlekamp-Massey algorithm[10]

Inputs: $s(z)=\boldsymbol{w}H^T$
Outputs: $\sigma(z)$, ξ
Initialize:
$\epsilon(z)=1, \beta(z)=x$
$\delta=1, L=R=\xi=0$
for $i=0$ to $2t-1$
$d=\sum_{j=0}^{\min(t,i)} \epsilon_j s_{i-j}$
$\phi(z)=\delta\epsilon(z)+d\beta(z)$
if $d=0$ or $i<2L$ then
$R=R+1$
$\beta(z)=z\beta(z)$
else
$R=0$
$\beta(z)=z\epsilon(z)$
$L=i-L+1$
$\delta=d$
end if
$\epsilon(z)=\phi(z)$
end for
if $\deg(\sigma(z))<t-\frac{R}{2}$ then
$\xi=1$
end if
$\sigma(z)=z^{t-\xi}\epsilon(z^{-1})$

따라서 이진 다항식의 덧셈과 뺄셈은 XOR연산으로 얻을 수 있다. 이진 다항식의 곱셈과 나머지 연산은 Bit-Shift와 XOR연산으로 계산 가능하다. F_{2^m}에서 주어진 이진 다항식의 차수가 m보다 큰 경우, 원시 다항식(primitive polynomial)의 나머지 연산을 통해 m보다 작은 차수의 다항식으로 만든다.

예를 들어, F_{2^3}에서 $(x^2+x+1)(x^2+1)$이 주어졌을 때 결과는 $11100\oplus 111=11011$, 즉 x^4+x^3+x+1이다. 얻어진 이진 다항식의 차수가 $m=3$보다 크므로 $x^4+x^3+x+1 \bmod x^3+x+1$를 수행하여, 해당 이진 다항식을 x^2+x로 계산한다. 이는 원시 다항식 x^3+x+1을 Bit-Shift하고, 주어진 이진 다항식과 XOR연산하는 과정을 반복 수행하여 $11011\oplus 10110=1101, 1101\oplus 1011=110$을 얻을 수 있다.

본 논문에서는 F_{2^m}에서 보다 빠른 곱셈 및 나머지 연산을 위해 이진 다항식을 거듭제곱 형태로 표현할 때의 지수 값을 배열 형태로 저장하고, 이진 다항식의 곱셈을 해당 지수간의 덧셈 $\alpha^i \cdot \alpha^j=\alpha^{i+j}$으로 계산한다. 앞선 예에서 $\alpha^5=x^2+x+1, \alpha^6=x^2+1$이므로, $\alpha^5\cdot\alpha^6=\alpha^{11}$로 계산된다. 나머지 연산은 $\alpha^{i \bmod (2^m-1)}$로 계산되므로, $\alpha^{11 \bmod 7}=\alpha^4$와 같다. 이때, $(i \bmod (2^m-1))=((i/2^m)+(i \bmod 2^m))$과 같다.

구현한 암호시스템은 이진 Goppa 부호의 오류정정 능력을 높이기 위해 기약다항식을 Goppa 다항식으로 사용하였다. 이때 임의로 생성한 다항식이 기약다항식임을 판별하기 위해 Ben-Or 알고리즘으로 인수분해 가능성을 확인하였다. 부호어에서 메시지를 수월하게 추출하기 위해 메시지와 패리티가 명확히 구분되는 체계적 형태의 생성 행렬을 사용하였다.

Patterson 알고리즘의 Step 1에서 다항식의 곱셈에 대한 역원 $h(z)$과 Step 4에서 $a(z), b(z)$은 확장 유클리드 알고리즘(Extended Euclidean Algorithm)으로 계산했다. Step 3에서 제곱근(square root)은

Algorithm 3 Alternate square root algorithm [8]

Inputs: $f(z)=f_0+f_1z+\cdots+f_{t-1}z^{t-1}$
Outputs: $\sqrt{f(z)}$
$\sqrt{f(z)}\equiv \sum_{i=0}^{\lfloor \frac{t-1}{2} \rfloor} f_{2i}^{2^{m-1}} z^i + \sum_{i=0}^{\lfloor \frac{t}{2}-1 \rfloor} f_{2i+1}^{2^{m-1}} z^i \sqrt{z} \pmod{g(z)}$
where $\sqrt{z}\equiv (z)^{2^{mt-1}} \pmod{g(z)}$

$p(z)^{2^{mt-1}} \equiv \sqrt{p(z)} \bmod g(z)$로 계산되며, 이때 Step 3의 시간 복잡도(complexity)는 $O(mt^2)$이다. 복호화 속도 향상을 위해 Algorithm 3을 참고하여 $z^i\sqrt{z}$를 미리 계산한 후 저장하고, 복호 시 이를 참고하는 방식으로 Step 3의 복잡도를 $O(t)$로 줄였다.

Berlekamp-Massey 알고리즘의 경우 오류벡터의 첫 번째 비트가 오류인 경우 해당 오류를 찾지 못하는 문제점이 있다. Algorithm 2는 이러한 문제점을 개선한 것으로 이를 참조하여 구현했다.

Patterson 알고리즘과 Berlekamp-Massey 알고리즘으로 계산한 $\sigma(z)$의 근을 찾는 대표적인 방법으로 L의 원소를 모두 대입하는 방법, Horner's 기법, Additive FFT 알고리즘이 있다. 각 방법의 복잡도는 $O(nt^2)$, $O(nt)$, $O(n\log n)$으로 본 논문에서는 Additive FFT 알고리즘을 사용하였다[9]. 해당 알고리즘은 유한체상에서 $z^n - z$의 근을 주어진 다항식에 대입했을 때의 결과를 벡터 형식으로 반환한다. Additive FFT 알고리즘으로 얻은 벡터의 원소 중 0에 해당하는 근의 위치정보를 기반으로 오류벡터를 생성한다.

3.3 성능 측정

암호시스템에서 평문 1 바이트를 암·복호화 하는데 필요한 사이클 수는 다음과 같이 계산된다.

cycles per byte = (1 블록 암·복호 시 사이클 수)/
(블록크기 / 8)

- Processer: Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz
- RAM: 4.0GB
- OS: Ubuntu 18.04
- Compile: gcc 7.5.0

위와 같은 환경에서 구현한 암호시스템을 실행했을 때 사이클 수를 측정하였다.

<그림 1>은 구현한 Patterson 알고리즘의 단계별 사이클 수 측정 결과를 보여준다. <그림 2>는 구현한 암호시스템의 암호문을 평문으로 복호 시 사이클 수를 측정한 결과이다. 이는 암호문 $\boldsymbol{y}$에서 평문 $\boldsymbol{m}$을 얻을 때까지 사이클 수를 측정한 것으로, Patterson 알고리즘과 Berlekamp-Massey 알고리즘별로 측정하였다.

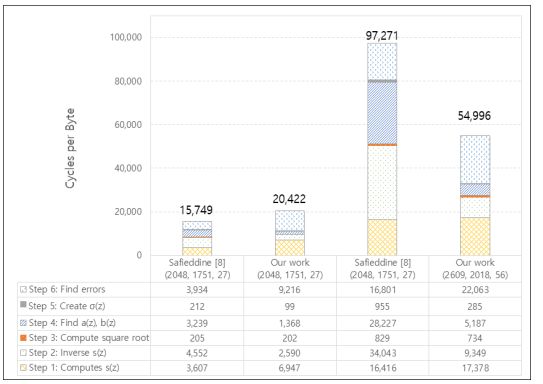

그림 1. Patterson 알고리즘 단계별 사이클 수
Fig. 1. Cycles for each step of Patterson

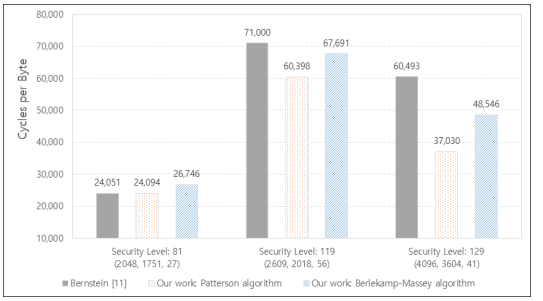

그림 2. McEliece 암호시스템의 복호화 사이클 수
Fig. 2. Decryption cycles for McEliece cryptosystem

Ⅳ. 결 론

McEliece 암호시스템은 평문의 암·복호화를 위해 오류정정부호를 사용하는 부호기반 암호시스템으로, 양자 컴퓨터를 이용한 공격에도 내성을 갖는다.

본 논문에서 우리는 이진 Goppa 부호를 사용하는 McEliece 암호시스템을 구현하였다. 이때 이진 Goppa 부호의 디코딩 알고리즘에 따른 암호시스템의 복호화 성능을 비교하기 위해 Patterson 알고리즘과 Berlekamp-Massey 알고리즘을 각각 구현하였다.

Patterson 알고리즘 구현 시 전처리를 통해 제곱근 계산의 복잡도 $O(mt^2)$를 $O(t)$로 줄였으며, Additive FFT 알고리즘을 사용하여 오류를 찾는 복잡도 $O(nt^2)$를 $O(n\log n)$로 줄였다.

측정결과 Patterson 알고리즘을 이용한 암호시스템의 복호화 속도가 Berlekamp-Massey 알고리즘 기반 암호시스템의 복호화 속도보다 빨랐다. 또한, 이진 Goppa 부호의 파라미터 값이 증가할수록 복호화 사이클 수 역시 증가하였으며, 특히 Goppa 다항식의 차수 t, 즉 암호문 생성 시 추가되는 오류 개수에 따른

복호화 사이클 수의 변화가 가장 컸다.

References

[1] E. Jochemsz, "*Goppa Codes & the McEliece Cryptosystem*," Amsterdam: Vrije Universiteit Amsterdam, 2002.

[2] R. J. McEliece, "*A public-key cryptosystem based on algebraic coding theory*," JPL DSN Progress Report, pp. 114-116, 1978.

[3] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg, "On the intractability of certain coding problems," *IEEE Trans. Inf. Theory*, vol. 24, no. 3, pp. 384-386, 1978.

[4] C. Adams and H. Meijer, "Security related comments regarding McEliece's public-key cryptosystem," *IEEE Trans. Inf.*, vol. 35, no. 2, pp. 454-455, Mar. 1989.

[5] D. J. Berstein, T. Lange, and C. Peters, "Attacking and defending the McEliece cryptosystem," *PQCrypto*, pp. 31-46, 2008.

[6] C. Peters, "Information-set decoding for linear codes over Fq," in *PQCrypto 2010 LNCS*, vol. 6061, pp. 81-94. Springer, Heidelberg, 2010.

[7] D. J. Bernstein, T. Lange, C. Peters, and H. van Tilborg, "Explicit bounds for generic decoding algorithms for code-based cryptography," in *Pre-proc. WCC 2009*, pp. 168-180, 2009.

[8] R. Safieddine and A. Desmarais, "Comparison of different decoding algorithms for binary goppa codes," Univ. of Lyon, 2014. Downloaded from http://www.cayrel.net/?Implementation-of-Goppa-codes.

[9] S. Gao and T. Mateer, "Additive fast fourier transforms over finite fields," *IEEE Trans. Inf. Theory*, vol. 56, no. 12, pp. 6265-6272, 2010.

[10] M. Albrecht, C. Cid, K. G. Paterson, C. J. Tjhai, and M. Tomlinson, "NTS-KEM," *NIST Submission*, 2017, [online] Available: https://nts-kem.io/.

[11] D. J. Bernstein, T. Chou, and P. Schwabe, "McBits: Fast constant-time code-based cryptography," in *CHES 2013*, vol. 8086 of LNCS, pp. 250-272, 2013.

유 기 순 (Ki-Soon Yu)

2007년 2월 : 안동대학교 컴퓨터공학과 학사

2015년 2월 : 동국대학교 정보보호학과 석사

2015년 3월~현재 : 동국대학교 정보통신공학과 박사과정

<관심분야> 포스트양자암호, 제어시스템 보안

임 대 운 (Dea-Woon Lim)

1994년 8월 : KAIST 전기및전자공학사 학사

1997년 2월 : KAIST 전기및전자공학사 석사

2002년 8월 : 서울대학교 전기컴퓨터공학부 박사

1995년 9월~2002년 8월 : LS산전 중앙연구소 선임연구원

2006년 9월~현재 : 동국대학교 정보통신공학과 교수

<관심분야> 무선통신, 부호이론, 신호설계, 암호 및 보안

[OCID:0000-0002-4661-7044]