Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.31, NO.3, Jun. 2021 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2021.31.3.527

NIST PQC Rainbow의 효율적 유한체 연산 구현*

김 광 식,[1†] 김 영 식[2‡]
[1,2]조선대학교 정보통신공학부 (대학원생, 교수)

Efficient Implementation of Finite Field Operations in NIST PQC Rainbow*

Gwang-Sik Kim,[1†] Young-Sik Kim[2‡]
[1,2]Chosun University (Graduate student, Professor)

요 약

본 논문에서는 미국 NIST PQC 표준화 Final List 알고리즘 중 유일한 다변수이차방정식(multivariate quadratic equation) 기반의 전자 서명인 Rainbow 알고리즘에서의 효율적인 유한체 연산 방법을 제안한다. Chou 등은 최근 Rainbow를 Cortex-M4에서 구현하기 위한 새로운 효율적 구현 방법을 제시하였다. 본 논문은 Chou 등이 제안한 방법을 개선하여 기존 대비 XOR 연산의 숫자를 13.7% 이상 감소할 수 있는 새로운 곱셈 방법을 제안한다. 또한, 테이블 룩업(Table Lookup)으로 수행되던 상에서의 역원 연산을 4x4 행렬 역원으로 치환하여 연산하는 방법을 제시한다. 또한, 새로운 구현을 RaspberryPI 3B+상에서 구현하여 성능을 측정하였다.

ABSTRACT

In this paper, we propose an efficient finite field computation method for Rainbow algorithm, which is the only multivariate quadratic-equation based digital signature among the current US NIST PQC standardization Final List algorithms. Recently, Chou et al. proposed a new efficient implementation method for Rainbow on the Cortex-M4 environment. This paper proposes a new multiplication method over the finite field that can reduce the number of XOR operations by more than 13.7% compared to the Chou et al. method. In addition, a multiplicative inversion over that can be performed by a 4x4 matrix inverse instead of the table lookup method is presented. In addition, the performance is measured by porting the software to which the new method was applied onto RaspberryPI 3B+.

Keywords: Post quantum cryptography, Rainbow, Multivariate Quadratic Equation, Finite Field Operation

I. 서 론

최근 양자 컴퓨터 기술이 급격하게 발전하면서, 양자 알고리즘을 통해 표준 공개키 암호를 해독하는 것이 현실적인 문제가 되고 있다. 국제적인 IT 기업들의 주도로 양자 컴퓨터 기술이 빠른 속도로 발전하고 있으며, 최근에는 이미 특정한 문제에 대해서는 슈퍼컴퓨터의 성능을 뛰어넘은 것으로 평가되고 있다. 따라서 현재 널리 사용되는 RSA와 타원 곡선 암호(ECC) 기반의 암호화 및 전자서명 알고리즘을 해독할 수 있는 수준의 양자 컴퓨터가 10년 이내에 등장할 것으로 예상한다[1].

이러한 문제에 대응하기 위해서 2017년부터 미국의 NIST에서는 포스트 양자 암호(Post-Quantum Cryptography) 표준화를 시작하였으며, 2020년 7월에 3라운드 진출 알고리즘을 발표하였고, 현재 각 알고리즘에 대한 평가가 진행 중이다[2].

Received(04. 27. 2021), Modified(05. 27. 2021), Accepted(05. 28. 2021)
* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.2021-0-00400, 저사양 디바이스 대상 고효율 PQC 안전성 및 성능 검증 기술 개발)
† 주저자, drains3005@naver.com
‡ 교신저자, iamyskim@chosun.ac.kr(Corresponding author)

본 논문에서는 현재 미국 NIST PQC 표준화 Final List 알고리즘 중 유일한 다변수이차방정식(multivariate quadratic equation) 기반의 전자 서명인 Rainbow에서의 효율적인 유한체 연산 방법을 제안한다[3]. Chou et al.은 최근 Rainbow를 Cortex-M4에서 구현하기 위한 새로운 효율적 구현 방법을 제시하였다[4]. 본 논문은 Chou et al.이 제안한 방법을 개선하여, Chou et al. 방식 대비 XOR 연산의 숫자를 13.7% 이상 감소시킬 수 있는 새로운 곱셈 방법을 제안한다. 또한, 테이블 룩업(Table Lookup) 방식으로 수행되던 유한체 상에서의 역원 연산을 4x4 행렬 역원으로 치환하여 수행하는 방법을 제시한다. 또한, 제안하는 방법을 Raspberry PI 3B+ 환경에서 구현하여 성능을 측정하였다.

II. 배경 지식

Rainbow는 Ding과 Schmidt에 의해서 2004년에 처음 제안된 알고리즘으로, 다중 세그먼트 Unbalanced Oil and Vinegar (UOV) 구조를 사용하는 키 교환 알고리즘이다. 현재 다변수이차방정식 기반의 전자서명 시스템으로, 유일하게 NIST PQC 3라운드 최종 후보에 이름을 올리고 있다. NIST PQC 3라운드에서 다변수이차방정식에 기반을 둔 또 다른 알고리즘으로는 Alternates 알고리즘으로 남아 있는 GeMSS가 있다.

Rainbow 파라미터는 공개키 크기가 보안 수준에 따라 258kB에서 1,885kB의 크기를 가진다. 이때 세부적으로는 F_{16}에서의 연산이 주를 이룬다.

최근 Riera는 NIST PQC 표준화 2라운드 Rainbow에 대한 Level 1 파라미터 집합을 최적화한 결과를 제시하였고[5], 테이블 룩업을 통해 F_{16} 상에서의 곱의 역원을 구하는 방법을 제시하였다. 구현 과정에서 LUT를 사용하면 상수 시간 연산이 보장되지 않고 연산 속도가 상대적으로 느린 문제가 있다. 효율적인 구현을 위해서는 서명 과정에서의 역행렬 계산의 효율적 구현이 매우 중요하다.

MQ 공개키 암호는 $K=F_q$에서 주요 연산이 이루어지고 이것을 기저체(base field)라 한다. Rainbow의 Level I에서 사용하는 기저체는 F_{16}이다. $n>m$일 때 공개키는 다음과 같이 $K^n \rightarrow K^m$으로의 사상으로 정의된다.

$$P = T \circ Q \circ S$$

여기서 T와 S는 Rainbow에서는 선형 사상으로 사용하고, 다음과 같이 나타낼 수 있다.

$$S: w \rightarrow x = M_S w$$
$$T: y \rightarrow z = M_T y$$

그리고 $Q: x \rightarrow y$는 중심사상(central map)이라 하며, 이차 다항식으로 구성되어 있고 역행렬 계산이 가능하도록 설계된다. 대부분의 MQ 기반의 암호는 Q를 어떻게 만드는지에 따라 특성이 구분될 수 있다. 보안을 위해서는 공개키 P로부터 개인키에 해당하는 각각의 요소 행렬을 구분해내는 것이 어려워야 한다.

Rainbow에서는 네 개의 정수 q, v_1, o_1, o_2가 중요한 파라미터로 사용된다. 중심사상 Q에서는 "oil"에 해당하는 변수와 "vinegar"에 해당하는 변수로 나누어진다. 첫 번째 세그먼트에서 vinegar 변수들은 $i \in V_1 = \{1,...,v_1\}$에 대응하는 x_i이고, oil 변수들은 $i \in O_1 = \{v_1+1,...,v_2 := v_1+o_1\}$에 대응하는 x_i이다.

두 번째 세그먼트에서 vinegar 변수들은 인덱스 집합 $V_2 = \{1,...,v_2 = v_1+o_1\}$이고 oil 변수들은 다음과 같은 인덱스 집합으로 정의된다.

$$O_2 = \{v_2+1,...,n = v_3 = v_2+o_2 = v_1+o_1+o_2\}$$

이때 중심사상 Q는 $m = o_1+o_2$ 개의 이차 방정식 $y = (y_{v_1+1},...,y_n) = (q_{v_1+1}(x),...,q_n(x))$를 갖고 있다. 여기서 각 이차방정식은 다음과 같이 정의된다.

$$y_k = q_k(x) = \sum_{i=1}^{v_1}\sum_{j=1}^{v_2} \alpha_{i,j}^{(k)} x_i x_j, \text{ 여기서 } k = O_1$$

$$y_k = q_k(x) = \sum_{i=1}^{v_2}\sum_{j=1}^{n} \alpha_{i,j}^{(k)} x_i x_j, \text{ 여기서 } k = O_2$$

k가 O_1에 있을 때 모든 이차 방정식 q_k에는 i와 j가 모두 O_1에 있는 교차항 $x_i x_j$가 없다. 그래서 $v_1 < i \le v_2$인 첫 번째 단계에서 모든 y_i에 대해 그리고 모든 vinegar 변수 $j \le v_1$인 x_j에 대해, 선형 방정식을 풀어서 대응되는 oil 변수들을 쉽게 계산할 수 있다. 마찬가지로 모든 k가 O_2에 있는 모든 이차

방정식 q_k에 대해 i와 j가 모두 O_2에 있는 교차항 $x_i x_j$가 없다. 그래서 $v_2 < i \leq n$인 두 번째 단계에서 모든 y_i에서 $j \leq v_2$인 모든 vinegar 변수 x_j가 주어져 있을 때 선형방정식을 풀어 $x_{v_{2+1}},...,x_{v_n}$을 쉽게 계산할 수 있다.

Q에 대한 x의 역상 y는 다음과 같이 구할 수 있다.

1) 랜덤 추측을 통해 초기 vinegar 변수 $\tilde{x}=(x_1,..,x_{v_1})$를 만든 후 이 값과 $(y_{v_1+1},...,y_{v_2})$로부터 Gauss 소거법을 통해 $(x_{v_{1+1}},...,x_{v_2})$를 구한다. 만일 해가 없다면 처음부터 다시 한다.
2) $\tilde{x}=(x_1,...,x_{v_2})$와 $(y_{v_2+1},...,y_n)$을 이용해서 다시 한번 Gauss 소거법을 통해서 $(x_{v_2+1},...,x_n)$을 구한다. 만일 해가 없다면 처음부터 다시 한다.

2.1 Rainbow에서의 서명 과정

Rainbow 알고리즘은 키 생성, 서명, 검증 세 단계로 구성되고, 이 중에서 키 생성은 서명과 검증과 비교하면 자주 일어나지는 않는다.

1) 키 생성

사용자는 가역 행렬 S, T, 그리고 Q로 구성된 비밀키를 랜덤하게 선택한다. 그런 후 $P=T \circ Q \circ S$를 공개키로 계산한다. S^{-1}, T^{-1} 그리고 Q의 각 요소 값들은 비밀키로 한다.

2) 서명

서명자는 먼저 메시지의 해시값 $z \in K^m$를 계산한다. 비밀키를 사용해 서명자는 $y=T^{-1}(z)$, $x=Q^{-1}(y)$ 그리고 $w=S^{-1}(x)$를 차례로 계산한다. 이때 값 w가 메시지의 서명이다.

3) 검증

메시지의 서명 w를 검증하기 위해 사용자는 $P(w)=z$를 계산한 후에 메시지의 해시값이 z와 같은지를 검증한다.

III. 제안하는 방법

3.1 Rainbow의 효율적 F_{16}상의 곱셈 연산

Rainbow에서는 기저체로 F_{16}을 사용한다. 본 장에서는 Rainbow의 F_{16}상에서의 곱셈 연산을 직접 구현함으로써 테이블 룩업 방식보다 성능을 높이는 방법을 제시한다. 특히 시간차에 기반한 부채널 공격에 내재적 저항성을 갖도록 값에 상관없이 항상 일정한 시간을 갖는 조건 분기가 없는 연산 방법이 필요하다. F_{16}연산을 가속함으로써 전체 서명(signing) 시간을 단축시킬 수 있다.

[5]에서는 가속을 위해 bitslicing을 통해서 연산 성능을 높이는 방법을 제안하였다. 이 방법은 원래 McEliece 암호에서 성능 향상을 위해 적용된 방법을 Rainbow에 접목한 것이다[6].

F_{16}연산은 4비트 크기의 원소를 다루는 연산으로 32비트 MCU환경에서는 하나의 변수에 4비트 원소 8개를 배열하는 것이 가능하다. 실제 곱셈 연산에서 각 비트마다 비트 단위의 연산이 가능하므로 32개의 원소를 모아서 한 변수에 대입하고 각 비트에 대해서 MCU가 제공하는 bit단위의 AND/XOR을 적용하면 32개의 연산을 동시에 병렬로 처리할 수가 있다. 본 논문에서 제안하는 방법도 같은 방법에 의해 성능을 높이는 것이 가능하며, 제안하는 방법에 사용되는 AND/XOR 연산을 bitsplicing으로 구현하면 병렬화를 통한 가속 효과를 마찬가지로 얻을 수 있다.

Rainbow에서는 AES의 S-box에서 널리 사용되던 방식인 tower field 형태로 유한체 연산이 수행된다. 이때 F_{16}은 부분 체 F_4를 사용해서 다음과 같이 나타낼 수 있다.

$$F_{16} := F_4[y]/(y^2+y+\beta)$$

여기서 다시 F_4는 하위체 F_2를 사용해서 다음과 같이 나타낼 수 있다.

$$F_4 := F_2[x]/(x^2+x+1)$$

F_4상의 두 원소를 $a=a_1x+a_0$와 $b=b_1x+b_0$라 하자. 그러면 두 원소의 곱은 다음과 같이 나타낼 수 있다.

$$c = c_1x + c_0 = ab = (a_1x + a_0)(b_1x + b_0)$$
$$ab = (a_1b_0 + a_0b_1 + a_1b_1)x + (a_0b_0 + a_1b_1)$$

F_{16}상의 두 원소의 곱 ef는 두 개의 F_4상의 원소의 곱셈을 이용해서 구할 수 있다. 여기서 $e = e_1y + e_0$, $f = f_1y + f_0$, $e_i, f_i \in F_4$이다. 그러면 $g_i \in F_4$에 대해 $ef = g_1y + g_0$로 나타낼 수 있고 $c_i \in F_2$에 대해

$$g_0 = c_1x + c_0,\ g_1 = c_3x + c_2$$

로 나타낼 수 있다. 이때 $a_i, b_i \in F_2$에 대해

$$e_1 = a_3x + a_2,\ e_0 = a_1x + a_0$$
$$f_1 = b_3x + b_2,\ f_0 = b_1x + b_0$$

라 하면, F_{16}상에서의 두 원소 e와 f의 곱은 다음과 같은 관계가 성립한다.

$$\begin{aligned}
c_0 &= a_0b_0 + a_1b_1 + a_3b_2 + a_2b_3 + a_3b_3\\
c_1 &= a_1b_0 + a_0b_1 + a_1b_1 + a_2b_2 + a_3b_2 + a_2b_3\\
c_2 &= a_2b_0 + a_3b_1 + a_0b_2 + a_2b_2 + a_1b_2 + a_3b_3\\
c_3 &= a_3b_0 + a_2b_1 + a_3b_1 + a_1b_2 + a_3b_2 + a_0b_3 + a_1b_3\\
&\quad + a_2b_3 + a_3b_3
\end{aligned} \tag{1}$$

이것을 다시 반복되는 연산에 따라 분류하면 다음과 같이 변경할 수 있다.

$$\begin{aligned}
t_0 &= a_3(b_2 + b_2)\\
t_1 &= a_1b_1 + a_2b_3\\
t_2 &= a_3b_1 + a_1b_2\\
c_0 &= a_0b_0 + t_2 + t_0\\
c_1 &= a_1b_0 + a_0b_1 + t_1 + (a_2 + a_3)b_2\\
c_2 &= a_2b_0 + a_0b_2 + t_2 + a_2b_2 + a_3b_3\\
c_3 &= a_3b_0 + a_2b_1 + (a_0 + a_1 + a_3)b_3 + t_0
\end{aligned} \tag{2}$$

[5]에서 Chou et al.이 제안한 방법에서 필요한 연산의 개수는 16개의 AND 연산과 22개의 XOR 연산이었으나, 본 논문에서는 식 (2)와 같이 연산 순서를 재조정함으로써 16개의 AND 연산을 유지하면서 XOR 연산을 19개로 감소시키는 것이 가능하였다. 이를 통해 F_{16}상에서의 유한체 연산을 위해 기존 대비 cycle 수 92.1%로 감소시키는 효과를 얻을 수 있다. 이 결과를 Table 1에 비교하였다.

Table 1. Comparison of Required Computation

Algorithm	AND	XOR	Ratio
Chou et al.	16	16	100%
I Compressed	22	19	86.3%

또한, 곱셈은 논리곱 AND에 대응되고 +는 논리합 XOR에 대응되고, 두 개의 연산은 모두 carry가 없이 처리할 수 있다. 따라서 변수 32비트 변수 R_3, R_2, R_1, R_0에 32개의 4비트 변수를 대입하여 bitsplicing을 통한 병렬 연산이 가능하다.

예를 들어 $R_i = (r_{i,31}, r_{i,30}, r_{i,29}, \ldots, r_{i,1}, r_{i,0})$라 가정하면 F_4상의 원소 $e_j = (e_{3,j}, e_{2,j}, e_{1,j}, e_{0,j})$는 $e_{i,j} = r_{i,j}$에 대응되도록 하면, $e_i = (r_{3,i}, r_{2,i}, r_{1,i}, r_{0,i})$이다. Chou et al.은 McEliece 암호를 위해 제안된 방법을 응용하여 Interleaving 방식으로 네 개의 변수를 사용해서 비트 단위 데이터 저장 방법을 제안하였고[6], 이는 32개의 기본 연산이 아닌 28개의 연산으로 구현할 수 있음을 보였다. 본 논문에서 제안하는 방법도 같은 최적화가 그대로 적용될 수 있다.

3.2 Rainbow의 효율적 F_{16}상의 곱의 역원 연산

곱셈의 역원은 $ab = c$에서 주어진 a에 대해 $c = (0,0,0,1)$이 되는 b를 찾는 문제로 정의할 수 있다. 곱셈의 관계에서 식 (1)은 다시 아래와 같은 행렬로 표현할 수 있다.

$$\begin{bmatrix}
a_0 & a_1 & a_3 & a_2 + a_3\\
a_1 & a_0 + a_1 & a_2 + a_3 & a_2\\
a_2 & a_3 & a_0 + a_2 & a_1 + a_3\\
a_3 & a_2 + a_3 & a_1 + a_3 & a_0 + a_1 + a_2 + a_3
\end{bmatrix}
\begin{bmatrix} b_0\\ b_1\\ b_2\\ b_3 \end{bmatrix} =
\begin{bmatrix} 1\\ 0\\ 0\\ 0 \end{bmatrix}$$

즉, 좌측 행렬을 A라 하면 $Ab = c$에서 $b = A^{-1}c$를 연산하는 문제로 바꿀 수 있다. 이때 4x4 행렬 $A = [a_{i,j}]$(여기서 $0 \le i, j \le 3$)는 F_2상에서의 Gauss 소거법을 통해 구현할 수 있으며, F_2 상의 연산을 통해 구현하는 것이 가능하다.

Table 2. Performance of Rainbow on RaspberryPi 3B+

Algorithm		Key Gen	Sign	Verify
I Classic	Previous	87.3ms	88.9ms	90.0ms
	Proposed	80.8ms	81.8ms	82.8ms
I Compressed	Previous	95.4ms	51.0ms	16.4ms
	Proposed	87.8ms	46.9ms	15.1ms

IV. 구현 및 성능평가

본 논문에서 제안한 방법을 Raspberry PI 3B+에서 실제로 구현하여 성능을 평가하였다. 이때 구현에 사용된 알고리즘은 Rainbow 알고리즘 Level I Classic 버전과 Level I Compressed Cyclic 버전이다. 두 가지 방식 모두 본 논문에서 제안한 F_{16}상에서의 연산을 핵심 연산으로 사용한다. 구현을 통해 연산 시간을 측정한 결과는 Table 2에 나타내었다.

Table 2에서 나타난 것처럼 Rainbow 알고리즘은 Level I에 해당하는 알고리즘의 경우 키생성, 서명, 및 검증을 Classic의 경우 80ms대에서 수행할 수 있었고 Compressed를 적용한 경우 verification 속도는 15ms로 비교적 경량 환경에서 매우 효율적 연산이 가능함을 확인할 수 있다.

V. 결 론

본 논문에서는 현재 미국 NIST PQC 표준화 Final List 전자 서명인 Rainbow 알고리즘에서의 효율적인 유한체 연산 방법을 제안한다[3]. 본 논문에서는 Chou et al.이 최근에 제안한 Rainbow의 효율적 구현 방법을 더 개선하여 더 효율적인 곱셈 연산이 가능하도록 만들 수 있었다. 또한, 유한체 상에서의 역원 연산도 테이블 룩업이 아닌 4x4 행렬 역원을 통해 계산할 수 있도록 하였다. F_2상에서 연산이 이루어지기 때문에 Gauss 소거법을 통해 고속 구현이 가능한 방법이다.

실제 성능평가는 RaspberryPI 3B+ 환경에서 구현하여 연산 시간을 측정하는 방법으로 이루어졌다. 향후 최근 IoT 연산을 위한 경량 환경으로 활용되는 Cortex M4 상에서 실제 구현을 통해 Chou et al. 이 제안한 방법과 같은 환경에서 직접 비교 연구를 수행할 것이다.

References

[1] W. Beullens, J.-P. D'Anvers, A. Hulsing, T. Lange, L. PAnny, C. de Saint Guilhem, N. P. Smart, "Post-Quantum Cryptography - Current state and quantum mitigation," ENISA Report, vol. 2, pp. 3-29, May 2021,

[2] NIST, the US National Institute of Standards and Technology. Post-quantum cryptography standardization project. https://csrc.nist.gov/Projects/post-quantum-cryptography, accessed at June 4, 2021.

[3] J. Ding, M.-S. Chen, M. Kannwischer, J. Patarin, A. Petzoldt, D. Schmidt, and B.-Y. Yang. "Rainbow," submission to the NIST post-quantum cryptography project, 2020.

[4] J. M. Riera, "Performance Analysis of Rainbow on ARM Cortex-M4," Bachelor's Thesis, Technische Universität München, 2019.

[5] T. Chou, M.J. Kannwischer, and B.-Y. Yang, "Rainbow on Cortex-M4," IACR eprint, 2021/532.

[6] T. Chou. "Mcbits revisited," In Proc CHES 2017, pp. 213 - 231, 2017.

〈저 자 소 개〉

김 광 식 (Kwang-Sik Kim) 정회원
2015년 2월: 조선대학교 정보통신공학과 졸업
2017년 2월: 조선대학교 정보통신공학과 석사
2017년 3월~현재: 조선대학교 정보통신공학부 박사과정
〈관심분야〉 양자내성암호, 부채널 분석

김 영 식 (Young-Sik Kim) 종신회원
2001년 2월: 서울대학교 전기공학부 공학사
2003년 2월: 서울대학교 전기컴퓨터공학부 공학석사
2007년 2월: 서울대학교 전기컴퓨터공학부 공학박사
2007년 3월~2010년8월: 삼성전자 시스템 LSI 사업부 책임연구원
2010년 9월~현재: 조선대학교 정보통신공학부 교수
〈관심분야〉 양자내성암호, 부채널 분석, 완전동형암호, 스마트카 보안, 스마트공장 보안