오픈소스 기반 격자 방식 PQC 알고리즘 분석

김민하, 문학준, 우사이먼성일

성균관대학교 소프트웨어학과

Open-Source Code Analysis on Lattice-Based Post Quantum Cryptography

Minha Kim, Hakjun Moon, Simon S. Woo

Computer Science and Engineering, Sungkyunkwan University

요 약

양자 컴퓨터가 개발됨에 따라, 현재 사용되는 RSA 등의 비대칭 키 암호 방식은 양자 컴퓨터의 빠른 연산 속도와 양자 알고리즘에 의해 모두 다항 시간 안에 해독 가능하게 된다. 오래 지나지 않아 상용화될 것으로 예상되는 양자 컴퓨터에 대응할 필요성이 제시되었고, 이에 따라 새로운 암호 체계인 양자 내성 암호가 제안되었다. 본 논문에서는 양자 내성 암호 중 몇 가지 격자 방식 알고리즘의 구현 기법과 알고리즘별 특징을 분석하고 그 실용성의 정도를 알아보기 위해 대표적인 보안 프로토콜을 사용한 테스트를 진행 후 결과를 분석하였다. 더불어 테스트한 알고리즘 중 보안 위기에 가장 효과적으로 대응할 수 있는 PQC 알고리즘을 살펴본다.

I. 서론

HTTPS를 비롯해 현재 사용되고 있는 인터넷 서비스는 사용자에 대한 암호화 및 보안이 필수적이다. 현재 HTTPS에서 사용되는 소인수분해를 이용한 RSA와 블록체인에서 사용되는 타원곡선 암호를 비롯한 비대칭 키 암호의 안전성은 수학 문제를 시간 내에 풀 수 없음에 기반하고 있다. 대표적인 양자 알고리즘인 Shor의 알고리즘이나 Grover의 알고리즘에 의하여 양자 컴퓨터를 이용하면 이러한 난제들을 다항 시간 내에 풀 수 있고, 결과적으로 인증서 등의 암호를 해독하는 것이 가능해진다. 양자 컴퓨터는 빠르면 5년에서 15년 안에 상용화될 것으로 예측되고 있다. 따라서 양자 컴퓨터의 상용화 이전에 통신의 안전성을 보장하기 위하여 양자 컴퓨터가 암호를 풀기에도 충분히 오래 걸리는 암호 알고리즘인 양자 내성 암호(PQC, Post Quantum Cryptography)를 개발해야 한다.

미국 표준기술연구소인 NIST(National Institute of Standards and Technology)에서는 양자 내성 암호의 표준화를 위하여 2016년부터 여러 후보 알고리즘을 대상으로 공모하고 있다. NIST의 주 요구사항은 양자 컴퓨팅 환경에서 AES/SHA 알고리즘 이상의 안전성 보장, 부채널 분석 공격에 대한 저항성과 소형 기기에서의 구현 가능성 등이다. 이와 관련하여 본 논문은 현재 NIST Round 3의 후보인 Dilithium, Falcon과 Round 2에서 탈락한 후보인 qTESLA 알고리즘의 특징을 키 크기, 계산 시간, 통신 시간, 보안성과 범용성을 기준으로 하여 비교 분석하였다.

Open Quantum Safe(OQS)에서는 양자 내성 암호의 개발과 시제품 제작 지원을 목적으로 하여 여러 가지 오픈소스 샘플을 배포하고 있다. 본 논문에서는 OQS에서 제공한 세 가지 오픈소스 openssl, openssh, openvpn를 이용하여 세 가지 알고리즘(qTESLA, Dilithium, Falcon)에 대하여 테스트를 수행하였다.

II. 알고리즘

2.1 격자 방식 암호 알고리즘

격자 방식 양자 내성 암호(Lattice-based Post Quantum Cryptography)는 NIST Round 2의 26개 후보 중 9개로 가장 높은 비율을 차지하였다. 격자 방식 양자 내성 암호(이하 격자 암호)는 암호화하기 쉽지만 brute force 방식으로 해독하기 어려운 비대칭 키 암호의 원리를 사용한다. 격자 암호 안전성의 기반은 Shortest Vector Problem(SVP), Closest Vector Problem(CVP) 등의 NP-hard 문제이다. SVP는 격자 위의 가장 짧은 벡터를 찾는 문제, CVP는 주어진 벡터와 가장 가까운 격자 위의 벡터를 찾는 문제이다. n차원 격자에서 연립방정식을 푸는 문제로, 모든 연산이 행렬의 곱셈과 덧셈이기 때문에 계산의 효율성이 높다.

2.2 암호 알고리즘 구현 기법

1) LWE and R-LWE problem

Learning With Errors problem(LWE)[3]은 격자 암호의 트랩도어로 사용되는, 격자 암호의 양자 안전성을 보장하는 알고리즘이다. LWE 문제는 n차원 격자 위에서 여러 개의 벡터가 주어졌을 때 특정 조건을 만족하는 해를 구하는 문제이다. Search version과

Decision version으로 구분되지만, 결과적으로 동치이다.

Algorithm 1 : Learning With Errors

● Search LWE : Find $s \in \mathbb{Z}_q^n$ given noisy random inner products

$$\overrightarrow{a_1} \leftarrow \mathbb{Z}_q^n,\ b_1 = \langle s, \overrightarrow{a_1} \rangle + e_1$$
$$\overrightarrow{a_2} \leftarrow \mathbb{Z}_q^n,\ b_2 = \langle s, \overrightarrow{a_2} \rangle + e_2$$
$$\cdots$$

where $e_i \leftarrow \mathcal{X}, \mathcal{X}$ Gaussian over $\mathbb{Z}$ with width αq. $(\alpha q > \sqrt{n})$

$$\mathbb{Z}_q^n \times \mathbb{Z}^m \xrightarrow{\frac{s \in \mathbb{Z}_q^{n \times m}}{(s,e) \to s^t A + e}} \mathbb{Z}_q^n$$

● Decision LWE : Distinguish $(A, b^t = s^t A + e^t)$ from uniform (A, b^t), where $A = (\overrightarrow{a_1}, \ldots, \overrightarrow{a_m})$.

● Search LWE ⟺ Decision LWE.

다음 Algorithm 2는 LWE를 활용한 공개 키/비밀 키 생성 알고리즘이다. 비밀 키는 격자 위의 특정 점 s, 공개 키는 s를 특정 수식에 넣어 계산한 벡터이다.

Algorithm 2 : Public-key Cryptosystem using LWE

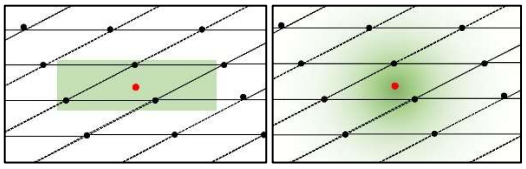
● $A \leftarrow \mathbb{Z}_q^n\ (random\ n \times m\ matrix\ over\ \mathbb{Z}_q)$ is open public.
● $Secret\ key\ of\ Alice: s \leftarrow \mathbb{Z}_q^n$
● $Public\ key\ of\ Alice: b^t = s^t A + e^t.$
● Bob's plaintext : $x \leftarrow \{0,1\}^m$

Bob sends to Alice
$$u = Ax, \qquad u' = b^t x + bit \cdot q/2.$$
Alice decodes $u' - s^t u \approx bit \cdot q/2.$

Ring-learning with errors problem(R-LWE)은 정수 이산 격자 $\mathbb{Z}_q$ 원소 기반의 LWE 문제를 특정 Ring 위에서의 격자인 $\mathbb{R}_q$ 원소 기반으로 확장한 문제이다.

2) Sampling

Gaussian sampling은 정규분포 상에서 pseudo-random한 값을 추출하는 방법이다. 격자 상에서는 격자 위 무작위 한 점을 고르는 방법으로, qTESLA와 Falcon 알고리즘에서는 Gaussian sampling을 이용하여 키 s를 생성한다.



[그림 1] 격자 위에서의 Gaussian sampling

R-LWE의 hardness를 기반으로 하는 Gaussian sampling을 사용한 격자 암호의 기본 알고리즘은 다음과 같다.

Algorithm 3 : Hash signature using Gaussian sampling

● KeyGen(1^n): let (A,B) be TrapGen(1^n), where $A \in \mathbb{Z}_q^{n \times m}$ and $B \in \mathbb{Z}_q^{n \times m}$ is a short basis of $\Lambda_q^{\perp}(A)$. The verification key is A and the signing key is B.

● Sign(B, m) : if (m, Sign(m)) is in local storage, output Sign(m). Otherwise:
1. find $c \in \mathbb{Z}_q^m$ such that $A \times c^t = H(m)$
2. Sample $\mathbf{v} \leftarrow$Sample($\mathbf{B}, \sigma$,c).
3. $s(m) \leftarrow c - v$. Store (m,s(m)) and output s(m).

● Verify(A, m, s) : if $A \times s^t = H(m)$ and $\| s \| \leq \sigma\sqrt{2\pi m}$, accept. Else, reject.

III. PQC 암호 알고리즘 비교

3.1 qTESLA

R-LWE에 의해 알고리즘의 안전성이 보장된다. 가장 복잡한 부분인 Gaussian sampling이 키 생성 부분에서만 사용되어 300여 줄의 C언어 코드로도 매우 간단하게 구현할 수 있다. 상수 시간으로 구현이 가능하며, secret memory와 branch에 접근하는 것을 피함으로써, 암호 연산 과정에서 장치가 소비하는 전력이나 전자파 등의 부가적인 요소를 사용하여 비밀키를 도출해내는 기법인 부채널 분석 공격(side-channel analysis attacks)과 key substitution 공격에 안전하다. 단점은 키 크기가 지나치게 큰 것과, 서명과 검증 과정이 오래 걸리기 때문에 IoT 기기나 임베디드 시스템 등 제한적인 환경에는 부적합하다는 것이다.

3.2 Falcon

Falcon 알고리즘은 키 선택 알고리즘으로 Gaussian samling을, 서명 생성 알고리즘으로 고속 푸리에 샘플링을 사용하였다. Ring-LWE에 의해 안전성이 보장된다. 공개 키와 서명의 크기가 모두 작기 때문에, 빠른 키 생성과 검증이 가능하다. 모듈화된 디자인으로 보안성을 조절할 수 있으며, 필요에 따라 격자도 NTRU가 아닌 다른 격자를 선택할 수 있다. 단점으로는 키 생성 과정과 푸리에 샘플링 방법이 모두 복잡하여 상수 시간으로 구현하지 못하며, 53비트의 부동소수점 연산을 사용하기 때문에 소형 기기에는 부적합할 수 있다. 또한 부동소수점 연산으로 인하여 부채널 분석 공격에 취약하다는 문제도 있다.

3.3 Dilithium

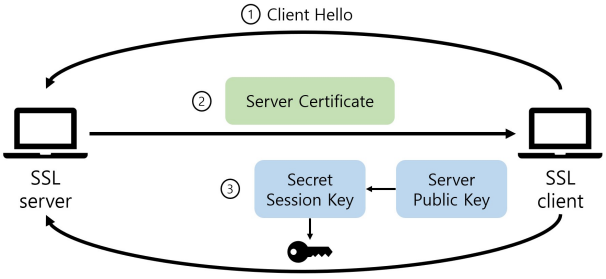
Dilithium 알고리즘 역시 다른 알고리즘과 마찬가지로 Ring-LWE에 의해 안전성을 보장한다. Gaussian sampling을 사용하는 것이 부채널 분석 공격에 상대적으로 안전한 방법이지만, Dilithium의 개발자들은 부채널 분석 공격을 막기 위해 정교한 방법을 사용해도 모든 경우에 대하여 대응하지 못할 것이라 생각하여 uniform sampling을 대신 사용하였다. Gaussian

sampling의 복잡한 연산을 사용하지 않기 때문에 상수 시간 안에 구현할 수 있다는 것이 장점이다. 또한 키와 서명의 크기가 작으며, 모듈화된 구현으로 필요에 따라 보안성을 변경할 수 있다.

IV. 오픈소스 기반 PQC 알고리즘 분석

4.1 Openssl

Secure Socket Layer(SSL)은 웹 서버간 안전하게 데이터 전송을 하기 위해 고안된 프로토콜로 현재 버전의 명칭은 Transport Layer Security(TLS)로 바뀌었다. SSL(TLS) 환경에서 서버와 클라이언트는 서로의 신원을 확인하기 위해 핸드셰이크 (handshake) 과정을 거친다. 서버는 클라이언트에 인증서를 보내고, 클라이언트는 이를 검증한다. 비대칭 키 암호를 이용하여 서버-클라이언트 간에 서로 같은 세션 키를 생성하여 보안 통신을 구성하는 것이 개략적인 과정이다.



[그림 2] SSL authentication process

Openssl에서 알고리즘별 공개키/개인키, X.509 인증서 생성, 메시지 암/복호화 및 서버-클라이언트간 접속을 확인하였으며, openssl의 라이브러리 함수 중 하나인 speed를 이용하여 RSA512와 양자 암호의 알고리즘별 key size, signature 생성 시간, verification 시간을 확인하였다.

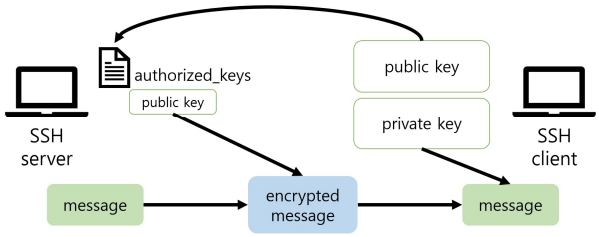
	Key size	Sign(s)	Verify(s)
RSA512	1704	0.000040s	0.000004s
dilithium2	5490	0.0001s	0.00001s
falcon512	3036	0.0049s	0.00002s
qteslapi	27322	0.0038s	0.0008s

[표 1] Comparison of RSA, qTESLA, Dilithium, and Falcon in Openssl key/signature generation

RSA와 비교했을 때 모든 양자 암호의 키 크기, 그리고 sign과 verify에 걸리는 시간이 더 큼을 확인할 수 있다. Dilithium은 키 크기에 비해 연산 속도가 빠르다. qTESLA의 키 크기가 가장 크고, sign 과정에서는 Falcon이 다른 알고리즘보다 오래 걸린다는 것을 확인할 수 있다.

4.2 Openssh

Secure Shell(SSH)은 안전한 원격 접속을 위한 프로토콜이다. SSH 연결은 크게 두 단계로 구성되어 있는데, 첫 번째 단계에서 클라이언트와 서버는 Diffie-Hellman과 같은 키 교환 알고리즘을 사용하여 연결 전체를 암호화하는 대칭키를 만들고 나눠 가진다. 그 후, [그림 3]과 같이 RSA와 같은 인증 알고리즘으로 만들어진 비대칭 키 쌍을 이용하여 클라이언트를 인증하고 서버에 접근하는 것을 승인한다. 본 연구에서는 인증 알고리즘을 양자 내성 암호 알고리즘으로 바꾸어 원격 서버에 접속하고 알고리즘간 효율성을 비교해보았다.



[그림 3] SSH authentication process

[표 2]는 RSA, qTESLA, Dilithium, 그리고 Falcon 알고리즘을 이용한 SSH 원격 연결 속도와 key size를 비교한 것이다. 키 크기는 qTESLA가 압도적으로 크며, 같은 보안 등급의 알고리즘끼리 비교할 때 Dilithium보다 Falcon이 근소한 차이로 더 작다. 연결 속도의 경우 qTESLA가 가장 느리며 Dilithium과 Falcon은 비슷하다. 또한 RSA512와 비교했을 때 키 크기의 차이는 크나 연결 속도가 비슷하다는 점에서 인증 알고리즘을 양자 내성 암호 알고리즘으로 바꾸는 것이 현실성 없지는 않다고 판단한다. 하지만 추후 키 크기를 획기적으로 줄일 수 있는 방법에 대한 연구가 필요할 것이다.

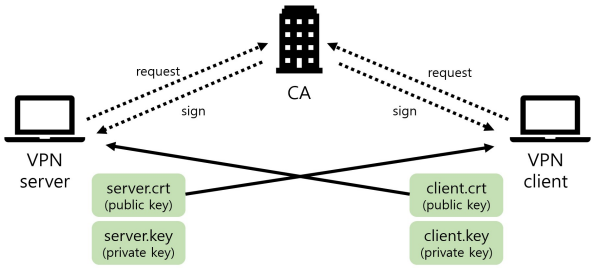
	Public key size	Private key size	Connection time(ms)
RSA512	397	1823	103
qTesla-p-Ⅰ	19898	47551	106
qTesla-p-Ⅲ	51304	120949	112
Dilithium-2	1640	7231	105
Dilithium-4	2408	10217	104
Falcon-512	1255	4395	102
Falcon-1024	2452	8205	104

[표 2] Comparison of RSA, qTESLA, Dilithium, and Falcon in Openssh authentication

4.3 Openvpn

Virtual Private Network(VPN)은 인터넷과 같은 공중망을 이용하여 둘 이상의 네트워크를 잇는 가상 터널을 만든 후 암호화된 데이터를 전송할 수 있도록 구성된 네트워크이다. VPN 연결은 크게 두 단계로 이루어진다. 첫번째 단계에서는 서버와 클라이언트 사이의 인증이 이루어지는데, 이에 앞서 Openssl 키 생성 알고리즘으로 Certificate Authority(CA)의 인증서와 키를 만들어야 한다. [그림 4]는 VPN 인증의 전 과정을 도식화한 것이다. VPN 통신은 대칭키로 암호화되기 때문에 상호간의 인증이 끝난 서버와 클라이언트는 차례로 사용할 대칭키 암호화 알고리즘과 키

교환 알고리즘을 합의한다. 본 연구에서는 인증 과정의 키 생성 알고리즘을 양자 내성 암호 알고리즘으로 바꾸어 원격 서버에 접속하고 알고리즘 간 효율성을 비교하였다.



[그림 4] VPN authentication process

[표 3]은 RSA, qTESLA, Dilithium, 그리고 Falcon 알고리즘을 이용하여 VPN 연결을 했을 때 생성된 서버의 인증서(server.crt)와 키(server.key) 크기를 비교한 것이다. 예상된 결과와 같이 qTESLA 알고리즘으로 생성된 키 크기가 다른 알고리즘에 비해 5배 이상 크다.

	Certificate size	Key size
RSA512	1216	1704
qTesla-p-Ⅰ	24186	27322
qTesla-p-Ⅲ	60253	68922
Dilithium-2	4913	5490
Dilithium-4	7481	7700
Falcon-512	2614	3036
Falcon-1024	4669	5636

[표 3] Comparison of RSA, qTESLA, Dilithium, and Falcon in Openvpn authentication

4.3 NIST Round 3 결과와 알고리즘 상호 비교

키 크기	qTESLA >> Dilithium ≈ Falcon
통신 시간	qTESLA > Dilithium ≈ Falcon
계산 시간	Falcon> qTESLA > Dilithium
보안성	qTESLA > Dilithium > Falcon
범용성	Dilithium > Falcon > qTESLA

[표 4] Overall comparison of qTESLA, Dilithium, and Falcon

[표 4]는 qTESLA, Dilithium, Falcon의 세 가지 알고리즘을 여러 기준으로 비교하여 정리한 것이다. qTESLA 알고리즘으로 생성된 키 크기가 가장 크며, 푸리에 샘플링을 사용하는 Falcon의 계산 시간이 가장 길다. 또한 부채널 분석 공격에 저항성이 있고, 키의 크기가 가장 큰 qTESLA의 보안성이 가장 뛰어나다.

NIST Round 3에서 Dilithium과 Falcon은 후보로 선정된 반면 qTESLA는 탈락되었다. qTESLA가 탈락한 이유가 상세하게 설명되어 있지는 않지만, 그 이유를 추론해 보자면 보안성 이외의 지표에서 모두 저조한 결과를 보이기 때문이라고 생각한다. 다른 양자 암호 알고리즘의 10배 이상, RSA와 비교하면 최대 100배 크기에 달하는 키를 생성하기 위하여 Gaussian sampling 등 비용이 높은 연산에 의존하며 많은 자원을 소비하는 것이 주 원인일 것이다.

Dilithium 알고리즘은 비록 부채널 분석 공격에 대한 일부 취약점이 존재하며 Falcon과 비교하면 개인키의 크기가 근소하게 크지만, 키와 서명 생성, 검증의 속도가 모두 빠르고 부동 소수점 연산을 사용하지 않아 연산량이 적어 소형 기기 등 제한적인 환경에도 적용하기 쉽다. 이러한 특성으로 인하여 Dilithium이 NIST Round 3에서도 좋은 결과를 얻을 것이라 예상한다.

Ⅴ. 결론

본 논문에서는 격자 방식 양자 내성 암호를 구현하는 이론적 배경인 LWE problem과 Gaussian Sampling에 대하여 살펴보았다. 또한 오픈소스 보안 프로토콜에 기반하여 qTESLA, Dilithium, Falcon 알고리즘을 테스트하였으며, 분석 결과를 바탕으로 알고리즘의 장단점을 평가하였다. 세 가지 알고리즘을 키 크기, 통신 시간, 계산 시간, 보안성, 그리고 범용성의 기준에서 평가해 보았을 때 Dilithium이 가장 효율적인 알고리즘이라고 판단한다. Dilithium은 키 크기가 작고 통신 및 계산 시간이 작다는 장점이 있다. 하지만 Gaussian sampling을 사용하지 않았기 때문에 부채널 분석 공격에 대한 취약점이 존재할 가능성이 높다. 부채널 분석 공격과 같은 보안 취약점을 극복하기 위한 지속적인 연구가 진행된다면, 기존 RSA와 같은 알고리즘을 양자 컴퓨터로 인한 보안 위협에서 벗어난 실용적인 알고리즘으로 대체할 수 있을 것이다.

[참고 문헌]

[1] Thomas Prest. Gaussian sampling in lattice-based cryptography. Cryptography and Security [cs.CR]. Ecole normale supérieure - ENS PARIS, 2015. English.

[2] Hall, Michael, and Raj Jain. "Performance analysis of openvpn on a consumer grade router." *cse. wustl. edu* (2008).

[3] DP Chi, JW Choi, J San Kim, T Kim - IACR Cryptol. “Lattice based cryptography for beginners.” ePrint Arch., 2015.

[4] Ellingwood, Justin. "Understanding the SSH Encryption and Connection Process." *URL: https://www. digitalocean. com/community/tutorials/understanding-the-ssh-encryption-and-connection-process (visitedon03/11/2017)* (2014).

[5] Feilner, Markus. *OpenVPN: Building and integrating virtual private networks*. Packt Publishing Ltd, 2006.

[6] Fouque, P.A. Falcon : Fast-Fourier Lattice-based Compact Signatures over NTRU specificatins v1.0. di.ens.fr, 2018.

[7] Alkim, Erdem. The Lattice-Based Digital Signature Scheme qTESLA. qtesla.org, 2020.

[8] Ducas, Léo. CRYSTALS-Dilithium Algorithm Specificati--on and Supporting Documantation. pq-crystals.org, 2017.

[9] Moody, D. “Round 2 of the NIST PQC Competition-What was NIST Thinking?,” PQCrypto 2019, 2019.