효율적인 PQC TLS 통신을 위한 인증서 압축 알고리즘 분석

강정훈[1], 김제인[2], 서승현[3]
[1]한양대학교 전자공학과 석사과정
[2]한양대학교 전자공학과 박사과정
[3]한양대학교 전자공학부 교수

kjh980922@hanyang.ac.kr, rean5123@hanyang.ac.kr, seosh77@hanyang.ac.kr

Analysis of Certificate Compression Algorithms for Efficient PQC TLS Communication

Jung Hun Kang[1], Jane Kim[2], Seung-Hyun Seo[3]
[1,2]Dept. of Electrical Engineering, Hanyang University
[3]School of Electrical Engineering, Hanyang University ERICA

요 약

전 세계적으로 PQC migration 을 위한 암호 표준화, 로드맵 발표 등 많은 관심이 집중되고 있다. 그 중, 인터넷 환경에 필수적으로 사용되는 TLS 는 PQC 전환이 필수적이다. 하지만 NIST 표준으로 선정된 PQC 서명 알고리즘들은 기존 서명 알고리즘에 비해 공개키와 서명의 크기가 크다. 따라서 TLS 통신에 필요한 인증서를 PQC 서명알고리즘을 통해 생성하게 되면 통신 오버헤드가 급증하는 문제가 있다. 이에, 본 논문에서는 TLS 에 사용되는 인증서에 RFP8879 에 정의된 3 가지 압축알고리즘(Zlib, Z-standard, Brotli)과, CBOR 인코딩 기법을 사용하여 인증서 압축성능을 비교해본다.

1. 서론

최근, 양자컴퓨터로 인한 보안 위협에 대비하기 위해 RSA 암호나 타원곡선 기반 암호가 적용된 곳을 모두 찾아 PQC 알고리즘을 적용하는 PQC migration 에 관한 연구가 활발히 진행되고 있다. 특히, 전송계층보안 프로토콜인 TLS(Transport Layer Security)는 웹, IoT 등 다양한 환경에서 범용적으로 사용되고 있어 PQC 전환이 필수적이다. TLS 는 3 가지 보안요소인 암호화, 인증, 무결성을 제공하는데, 이를 위해 다양한 암호 알고리즘이 사용된다. 일반적으로 PQC 알고리즘은 기존 암호에 비해 키와 암호문이 커지기 때문에 TLS 에 사용되는 암호알고리즘을 PQC 로 전환할 경우 통신 오버헤드가 커지는 문제점이 있다.

특히, TLS Handshake 프로토콜을 위해 사용되는 인증서는 일반적으로 하나의 인증서만 사용하지 않고, 인증서에 서명한 인증기관(CA)를 인증하기 위해 certificate chain 을 형성해 사용한다, 이 경우 서명알고리즘이 한 번만 사용되는 것이 아니라, 여러 번 사용되게 된다. 따라서, 서명알고리즘으로 생성되는 서명의 크기가 커지면 certificate 의 크기는 더 커지게 된다.

[1]에 따르면, NIST 후보 서명 알고리즘들을 사용할 경우, 서명 및 키 사이즈의 증가함에 따라 인증서의 크기가 기존 알고리즘에 비해 증가한다. 예를 들어, NIST 선정 PQC 표준 서명 알고리즘인 Falcon1024 를 사용하면, 기존 ECDSA 를 사용한 인증서에 비해 크기가 6.81 배 증가하게 된다. 이러한 인증서 사이즈의 증가는 통신 오버헤드로 이어지므로 통신 오버헤드를 줄이기 위한 연구가 진행되고 있다.

그 중 TLS 통신에 큰 비중을 차지하는 인증서 크기를 줄이는 문제에 대해서 여러가지 인증서 압축 알고리즘이 제안되고 있다. 또 IETF(Internet Engineering Task Force)에서는 제안된 인증서 압축 알고리즘들을 RFC(Request For Comments)문서로 표준화하고 있다.

본 논문에서는 3 가지 표준 인증서 압축 알고리즘(Zlib, Z-standard, Brotli)으로 PQC 인증서를 압축하여 비교 및 분석한다.

<표 1 >압축 알고리즘별 PQC 인증서 크기 비교

알고리즘	Cert Size (Bytes)	Zlib		Z-standard		Brotli	
		Size	압축률	Size	압축률	Size	압축률
FALCON512	2,602	1,981	(23.86%)	1,993	(23.40%)	1,962	(24.60%)
FALCON1024	4,641	3,532	(23.89%)	3,531	(23.91%)	3,501	(24.56%)
DILITHIUM2	5,571	4,214	(24.35%)	4,238	(23.93%)	4,226	(24.14%)
DILITHIUM3	7,623	5,790	(24.05%)	5,803	(23.88)	5,753	(24.53%)
DILITHIUM5	10,251	7,759	(24.31%)	7,775	(24.15)	7,707	(24.82%)
Sphincs-sha2-128fast	23,678	18,006	(23.95%)	17,918	(24.33)	17,874	(24.51%)
Sphincs-sha2-128small	11,174	8,490	(24.02%)	8,471	(24.19)	8,428	(24.57%)
Sphincs-shake-128fast	23,678	18,010	(23.94%)	17,921	(24.31)	17,878	(24.50%)
Sphincs-sha2-192fast	48,853	37,152	(23.95%)	36,931	(24.40)	36,892	(24.48%)

2. 관련연구

2.1 압축 알고리즘

TLS 통신의 효율성과 지연시간을 줄이기 위해서는 TLS handshake 과정에서 교환하는 데이터의 양을 줄이는 것이 효과적이다. 특히 TLS handshake 과정에서는 하나의 인증서만 전송하는 것이 아니라, 인증서 체인(Certificate Chain)을 전송하는 경우가 대다수이다. 따라서, 인증서 체인의 크기를 줄이기 위해 인증서를 압축하여 전송하는 연구가 진행되었고, 다양한 압축 알고리즘들이 제안되었다. IETF 에서 세가지 알고리즘(Zlib, Brotli, Zstandard)을 선정하여 표준으로 제정하여 RFC8879[2]에서 소개하고 있다.

인증서 압축을 위한 압축알고리즘은 일부 손실이 허용되는 이미지 압축 알고리즘과는 다르게 손실이 없는 무손실 압축알고리즘을 사용해야 한다. 일정한 window size 로 데이터를 나누고, 그 안에 공통된 표현이나, 데이터를 메타데이터로 저장하고 압축해제를 한 후에 압축된 데이터의 체크 섬을 확인하는 형태로 동작한다.

2.2 Zlib

Zlib[3]은, 압축알고리즘 중 가장 널리 사용되는 LZ77 알고리즘을 통해 압축하고, 중복되는 내용에 대한 포인터를 허프만 코딩(Huffman coding)를 통해 한 번 더 압축한다. 또한, 압축 정도에 따라 compress level 을 0 부터 3 까지 조절할 수 있다.

2.3 Z-standard

Z-standard[4]는 메타(Meta)에서 제안한 범용 압축 알고리즘으로, Zlib 과 같은 LZ77 압축 알고리즘을 기반으로 동작한다. 특히, 압축률에 따라 압축 속도를 조절할 수 있는 기능을 바탕으로, 작은 데이터를 빠르고 효율적으로 압축하는 dictionary compression 기능을 제공한다. API 를 다양한 언어를 지원하도록 설계하였다.

2.3 Brotli

Brotli[5] 압축 알고리즘은 구글에서 제안한 범용 무손실 압축 알고리즘으로, 허프만 코딩과 2 차 컨텍스트 모델링의 조합을 사용해 현재 사용가능한 범용 압축 알고리즘(Zlib, Zstandard)와 비슷한 압축속도를 보이지만, 더 높은 압축 효율을 보인다.

3. 인증서 압축 알고리즘 별 성능 측정

이 장에서는 PQC 인증서 생성을 위해 사용한 LibOQS 라이브러리에 대하여 설명하고, 이후 실험 방법과 측정 결과에 대해 서술한다.

3.1. LibOQS

LibOQS[6]는 PQClean 을 기반으로 TLS 에 적용할 수 있는 PQC 를 지원하는 Open Quantum Safe 의 라이브러리이다. NIST 에서 선정한 PQC 4 종(Kyber, Dilitium, Falcon, Sphincs+)와 그 외 PQC candidates 를 OpenSSL 과 연동할 수 있도록 라이브러리와 테스트 코드를 제공한다. 본 논문에서는 LibOQS 에서 지원하는 TLS 용 PQC 인증서 9 종(Falcon 512, Falcon1024, Dilitium2, Dilitium3, Dilitium5, Sphincs-128fast, Sphincs-128small, Sphincs-192fast, Sphincs-shake-128fast)에 대하여 실험을 진행하였다.

3.2 실험 방법

앞서 다룬 세가지 압축 알고리즘의 성능을 실험을 통해 측정, 비교하였다. 먼저, LibOQS 라이브러리에서 제공하는 TLS 용 PQC 인증서를 X509 타입으로 생성한다. 그 다음 인증서 크기를 측정하여 압축률을 아래와 같이 계산하였다.

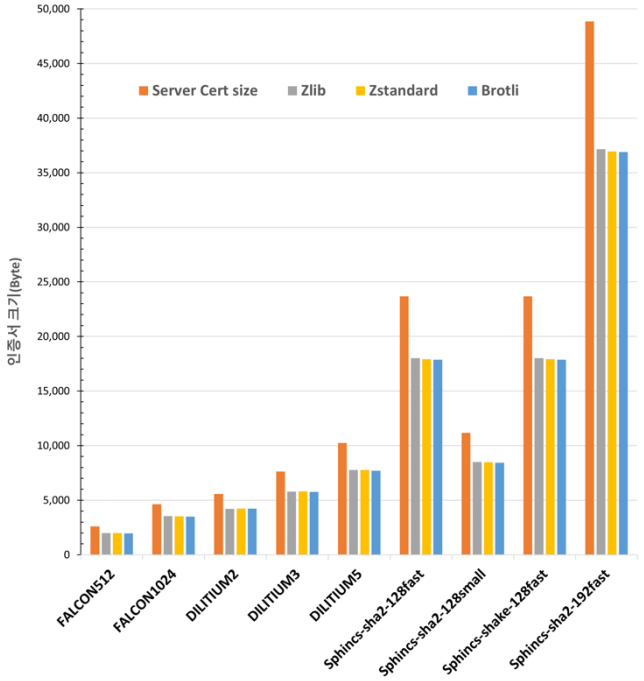
$$\frac{(\textit{기존 인증서 크기} - \textit{압축 후 인증서 크기})}{\textit{기존 인증서 크기}} \times \mathbf{100}(\%)$$

먼저, OpenSSL 에서 제공하는 세가지의 표준 압축알고

리즘(Zlib, Z-standard, Brotli)을 파이썬 기반의 코드로 작성하여 압축하고, 인증서 사이즈를 압축 전과 후로 나누어 측정하였다.

3.3 측정 결과

<표 1>은 LibOQS 에서 제공하는 NIST 표준 PQC 서명 알고리즘으로 생성한 1-chain 인증서를 압축 알고리즘별로 압축한 결과 인증서 크기를 나타낸 표이다. 괄호안의 숫자는 최적화 알고리즘 적용 전과 후의 압축률을 나타낸다 압축 알고리즘에 따른 압축률의 차이는 크지 않았지만, 전반적으로 24%전후의 압축률을 보였다.



<그림 1> 압축 알고리즘별 PQC 인증서 크기

<그림 1>은 PQC 서명 알고리즘을 통해 생성한 X509 타입의 인증서를 각각의 최적화 알고리즘을 적용한 후 차트로 나타내었다. 압축 대상 인증서의 크기가 크면 클수록 인증서 크기가 줄어드는 폭이 크지만, 압축률 측면에서는 아주 근소한 차이만 보였다.

4. 결론

PQC 마이그레이션 관점에서 볼 때, 현재 사용하는 프로토콜을 그대로 IoT 환경등에 적용하는 데에는 증가한 서명 및 공개키 크기로 인해 제약이 따를 것이다. 본 논문을 통해 TLS 를 위한 PQC 인증서에 압축 알고리즘을 적용하면 평균적으로 24%가량의 인증서 크기를 줄일 수 있음을 확인하였다.

ACKNOWEDGEMENT

이 논문은 2023 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No.RS-2023-00225201, 국방 무인이동체 역이용 방지 제어권 보호기술 개발)

참고문헌

[1] Raavi, Manohar, et al. "Performance characterization of post-quantum digital certificates." 2021 International Conference on Computer Communications and Networks (ICCCN). IEEE, 2021.

[2] Ghedini, A. and Vasiliev, V., 2020. RFC 8879 TLS Certificate Compression.

[3] Deutsch, P. and J-L. Gailly "ZLIB Compressed Data Format Specification version 3.3" RFC 1950 DOI 10.17487/RFC1950 <https://www.rfc-editor.org/info/rfc1950>

[4] Z-standard, Meta, 2023 년 9 월 27 일 접속, http://facebook.github.io/zstd/

[5] Alakuijala, Jyrki, et al. "Brotli: A general-purpose data compressor." *ACM Transactions on Information Systems (TOIS)* 37.1 (2018): 1-30.

[6] Michael Baentsch, Christian Paquin, Richard Levitte, Basil Hess, Julian Segeth, Alex Zaslavsky, Will Childs-Klein, Thomas Bailleux, “liboqs oqsprovider”, Github, https://github.com/open-quantum-safe/oqs-provider.git