PQC SPHINCS+ 전자 서명 알고리즘의 효과적인 하드웨어 설계에 관한 연구

이용석 [1], Cansu Karakuzu[1], 백윤흥 [1]
[1]서울대학교 전기정보공학부, 서울대학교 반도체 공동연구소

yslee@sor.snu.ac.kr, krcansu@sor.snu.ac.kr, ypaek@snu.ac.kr

A Study on Efficient Hardware Design of Digital Signature Algorithm for Post-Quantum Cryptography SPHINCS+

Yongseok Lee[1], Cansu Karakuzu[1], Yunheung Paek[1]
[1]Dept. of Electrical and Computer Engineering and Inter-University Semiconductor Research Center(ISRC), Seoul National University

요 약

본 논문은 통신 시스템에 주로 사용되는 디지털 전자 서명 알고리즘 중 양자 내성 암호인 SPHINCS+ 알고리즘에 대한 효과적인 하드웨어 설계 방안에 대한 연구이다. SPHINCS+ 알고리즘은 해시 함수 기반 알고리즘으로, 많은 횟수의 해시 함수가 반복해서 사용된다. 해시 함수를 가속 연산해도, 그 횟수가 크기 때문에 SPHINCS+ 알고리즘은 다른 전자 서명 알고리즘보다 하드웨어 설계 후 큰 latency 를 가지는 특징이 있다. 이를 극복하기 위해 SPHINCS+ 알고리즘에서 사용되는 해시 함수들을 면밀하게 분석한다. 그 결과 같은 해시 함수에 대해서도 입출력 데이터 크기가 다양하게 변화하고, 서로 다른 데이터 플로우를 가지는 그 세부 차이점들을 파악하여, 이를 접목한 하드웨어 설계에 대해 논의한다.

1. 서론

양자컴퓨터의 발달로 기존에 사용되던 디지털 서명 알고리즘(Digital Signature Algorithm, DSA)을 통한 통신 시스템을 대체하는 양자 내성 암호를 적용하려는 표준화가 진행되고 있다. 미국 국립표준기술연구소(NIST)에서는 양자 내성을 가진 디지털 서명 알고리즘을 선정하고 있으며, SPHINCS+ 알고리즘은 표준화 후보 중 하나이다[1]. SPHINCS+ 알고리즘은 해시 함수 기반 서명 알고리즘이다. 해시 함수는 양자 내성을 가지는 것으로 알려져 있으면서, 기존 암호 시스템에 자주 사용되어온 함수이다. 그래서 기존 시스템을 위해 연구해왔던 결과물 혹은 개발된 해시 함수 하드웨어 가속 모듈을 사용하거나 라이브러리를 사용하는 등 활용하기 좋은 특징이 있다[2]. 또한 해시 함수 기반 서명 알고리즘은 복잡한 수학적 연산에 기반하지 않고, 해시 함수에 대한 반복적은 수행으로 보안 강도를 조정할 수 있다는 장점이 있다. 이와 더불어 SPHINCS+ 알고리즘은 다른 서명 알고리즘과 달리 32~64-Bytes 로 상대적으로 적은 양의 공개키를 가지고 있다. 이는 공개키를 상대방에 전송해서 서명을 검증하기 때문에, 통신 데이터로 추가되는 오버헤드가 작은 특징이 있다고 볼 수 있다. 이러한 장점으로 SPHINCS+ 알고리즘은 기존 통신 시스템에 추가적으로 적용하기 적합하다고 할 수 있다.

하지만 SPHINCS+ 알고리즘은 양자 내성 암호의 보안 강도를 만족하기 위해서, 한 번의 전자 서명을 생성하는 과정에서 해시 함수를 만 번 이상 반복 수행한다는 문제점이 있다. 해시 함수를 한 번 수행하는 것은 시스템에 부담되지 않는 간단한 연산이지만, 이를 반복해서 수행하는 것은 전체적인 시스템에서 많은 연산 시간을 차지하며 부하를 일으킬 수 있다. 따라서 SPHINCS+ 알고리즘에 대한 하드웨어 설계를 통해 호스트 CPU 와 별도의 하드웨어에서 가속 연산하려는 기존 논문들이 있다[3-6]. 하드웨어 설계 성능을 비교하기 위해 HLS(High Level Synthesis)방식을 사용한 논문에서는 SPHINCS+가 서명 생성 과정에서 464,961,626-Cycle 의 latency 를 가지는 것으로 나타났다. 이는 다른 양자 내성 암호 표준화 후보인 CRYSTALS-Dilithium 알고리즘이 18,525-Cycle 의

latency 를 가지는 것과 비교했을 때, 해시 함수 반복 수행에 많은 시간이 소요되는 것을 알 수 있다. 물론 하드웨어 설계 과정에서 해시 함수를 병렬적으로 확장하면 연산 시간을 줄일 수 있겠지만, 설계 자원이 증가한다는 trade-off 를 비교해야 한다. 이러한 이유로 해시 함수 기반 서명 알고리즘을 가속하는 하드웨어 설계 연구는 효율적인 하드웨어 설계 방법을 찾는 연구들이 진행되어 왔다[4-6].

본 논문에서는 이러한 기존 연구들을 바탕으로 SPHINCS+ 알고리즘에 대해 효과적으로 하드웨어 설계하는 방법을 제시하려 한다. 그 방법으로 먼저 SPHINCS+ 알고리즘에서 사용되는 해시 함수의 종류를 분석하고, 세부적인 데이터 플로우를 파악하여 최적화하는 방법에 대해 논의한다.

2. SPHINCS+ 알고리즘의 세부 연산 분석 결과

연산 측정 환경으로 NIST 에서 PQC 알고리즘 성능 비교를 위해 추천하고 있는 Cortex M4 보드 환경에서 168MHz 동작 속도로 사용하였다. 공개된 레퍼런스 코드를 SPHINCS+-SHAKE-128s-simple 파라미터로 설정하고, 타겟 보드에서 수행하며 clock cycle 을 측정하였다. SPHINCS+ 알고리즘에서 사용되는 해시 함수 종류는 SHAKE, SHA2, Haraka 가 있으며, 본 논문에서는 SHAKE 해시 함수에 대해 분석하였다. SHAKE 해시 함수는 SHA3 해시 함수와 비슷하지만, 출력 데이터 크기를 조정할 수 있다는 차이점이 있다.

SPHINCS+ functions	Clock Cycles
Keypair Generation	2,353,825,792
Signature Generation	15,373,480,135
Signature Verification	182,687,874

<표 1> Cortex M4 성능 측정 결과

표 1 에서 볼 수 있듯이, 서명 생성 과정이 가장 많은 시간을 차지하고 있다. 그 중 Keypair 생성 과정 또한 많은 연산 시간을 차지하고 있는데 아래 표 2 는 Keypair 생성 과정에서 99% 연산 시간을 차지하는 WOTS+ 공개키 생성 과정을 분석한 결과이다.

WOTS+ process	Percentage
WOTS+ sk generation	6.13%
WOTS+ chain generation	93.12%
Compression	0.74%

<표 2> WOTS+ public key generation 과정에서 함수별 연산 시간 비율

WOTS+ 공개키 생성 과정은 세 가지 연산으로 이루어져 있는데, 그 중 WOTS+ chain 생성 과정은 특히 반복적으로 해시 함수에 대한 연산이 수행되는 과정이다. 하지면 이 함수의 데이터 플로우를 분석해보면 다른 해시 함수들과 다른 특징이 있는 것을 알 수 있다. 바로 w 파라미터에 대해 반복 수행하면서, 피드백 방식으로 연산이 수행된다는 점이다. 즉, 해시 함수의 결과 데이터가 다시 해시 함수의 입력 데이터로 사용된다. 다른 점은 concatenation 을 통해 주소 값을 새로 추가하는 점이 달라진다. 하지만 대부분의 데이터는 그대로 재사용 가능하며, 주소 값의 일부만 변경된다. 이러한 특징은 하드웨어 설계 과정에서 데이터 이동을 최소화할 수 있는 특징이다.

Hash functions
H_msg(R, PK.seed, PK.root, M)
PRF(SEED, ADRS)
PRF_msg(SK.prf, OptRand, M)
F(PK.seed, ADRS, M1)
H(PK.seed, ADRS, M1‖M2)
Th_l(PK.seed, ADRS, M)

<표 3> SPHINCS+ 알고리즘에서 사용되는 해시 함수 리스트

이러한 특징은 모든 연산들이 해시 함수를 사용하고 있지만, 표 3 과 같이 해시 함수는 모두 같은 입출력 데이터를 가지고 있지 않다. 따라서 입출력 데이터의 길이에 따라서 해시 함수 수행 시간의 차이가 발생한다. 또한 해시 함수를 사용하는 개별 함수에 따라서 입출력 데이터 플로우가 변화한다. 따라서 같은 해시 함수라 해도 다른 방식으로 연산들이 수행되기 때문에, 효과적인 하드웨어 설계를 위해서는 이에 대한 고려가 필요하다.

3. 결론

본 논문에서는 SPHINCS+ 알고리즘을 Cortex M4 임베디드 보드에서 성능 분석한 결과를 바탕으로, 해시 함수를 효과적으로 하드웨어 설계하는 방법에 대해 논의하였다. 사용되는 해시 함수의 종류 6 가지를 분석하고, 각자 세부적인 데이터 플로우가 다르다는 점을 착안하여 서로 다른 방법으로 최적화하는 방법을 고려해야 할 것이다. 특히 WOTS+ 공개키 생성에 많은 연산 시간이 소요되었는데, 그 과정에서 발생하는 피드백 데이터를 활용하는 방식이 크게 효과를 발휘할 것으로 예상된다.

ACKNOWLEDGEMENT

이 논문은 2023 년도 BK21 FOUR 정보기술 미래인재 교육연구단에 의하여 지원되었음. 본 연구는 IDEC 에서 EDA Tool 을 지원받아 수행하였습니다. 이 논문은

2023 년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2020-0-01840, 스마트폰의 내부데이터 접근 및 보호 기술 분석)

참고문헌

[1] https://csrc.nist.gov/projects/post-quantum-cryptography

[2] Hülsing, Andreas, et al. "Hash-based signatures: An outline for a new standard." NIST Workshop on Cybersecurity in a Post-Quantum World. 2015.

[3] Basu, Kanad, et al. "Nist post-quantum cryptography-a hardware evaluation study." Cryptology ePrint Archive (2019).

[4] Berthet, Quentin, et al. "An area-efficient SPHINCS+ post-quantum signature coprocessor." 2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). IEEE, 2021.

[5] Amiet, Dorian, et al. "Fpga-based sphincs+ implementations: Mind the glitch." 2020 23rd Euromicro Conference on Digital System Design (DSD). IEEE, 2020.

[6] Amiet, Dorian, Andreas Curiger, and Paul Zbinden. "FPGA-based accelerator for post-quantum signature scheme SPHINCS-256." IACR Transactions on Cryptographic Hardware and Embedded Systems (2018): 18-39.