FALCON 전자서명 알고리즘의 최적화 동향

이규섭[1], 조성민[2], 서승현[3]
[1]한양대학교 전자공학과 석사과정
[2]한양대학교 전자공학과 석박통합과정
[3]한양대학교에리카 전자공학부 교수
r22jiw0n@hanyang.ac.kr, smcho3315@hanyang.ac.kr, seosh77@hanyang.ac.kr

Optimization Trends of the Falcon Digital Signature Algorithm

Gyu Sup Lee[1], Seong-Min Cho[2], Seung-Hyun Seo[3]
[1,2]Dept. of Electrical Engineering, Hanyang University
[3]School of Electrical Engineering, Hanyang University ERICA

요 약

FALCON 알고리즘은 격자기반 서명 체계로 서명 길이 및 공개키가 짧고, 서명 생성/검증 속도가 빠르다는 장점이 있다. 하지만 Fast Fourier Transform(FFT), discrete Gaussian sampling과 같은 리소스가 많이 사용되는 연산이 활용되기 때문에 최적화 연구가 필요하다. 최근에는 병렬처리 및 파이프라인 기법을 활용할 수 있는 하드웨어를 통한 최적화 및 ARM 아키텍쳐의 병렬 처리 유닛을 활용하고 메모리 접근 방식을 최소화하는 방법들이 연구되고 있다. 이에 본 논문에서는 FALCON 알고리즘 대상 최적화 연구 동향과 그 결과를 분석하고 향후 추가적으로 필요한 FALCON 최적화 구현 방안에 대해서 기술한다.

1. 서론

Shor 알고리즘의 등장과 이를 구현하기 위한 양자컴퓨터의 개발이 가속화됨에 따라 소인수 분해 및 이산로그 문제에 기반하는 기존 공개키 암호체계의 안전성이 위협받고 있다. 이로 인해 양자컴퓨팅에 내성을 갖는 암호화 알고리즘에 대한 필요성이 증가하고 있으며 2022년 미국의 NIST에서는 4개의 양자내성암호를 표준으로 선정하였다. 그 중 FALCON 알고리즘은 격자기반의 hash-and-sign 서명 체계로 서명 길이 및 공개키가 짧고, 서명 생성/검증 속도가 빠르다는 장점이 있다. 하지만 Fast Fourier Transform(FFT), discrete Gaussian sampling과 같은 리소스가 많이 사용되는 연산이 활용되기 때문에 현재까지 최적화 연구가 활발하게 진행되고 있다.

최근 병렬처리 및 파이프라인 기법을 활용할 수 있는 하드웨어를 통한 FALCON의 최적화 연구가 진행되고 있다. 또한 ARM 아키텍쳐의 병렬 처리 유닛을 활용하고 메모리 접근 방식을 최소화하는 최적화 방법들이 연구되고 있다. 이러한 연구는 FALCON 알고리즘의 실제 응용 분야에서의 활용성 및 확장성을 높이는 데 도움을 주고 있다.

이에 본 논문에서는 FALCON 알고리즘 대상 최적화 연구 동향과 그 결과를 분석하고 향후 추가적으로 필요한 Falcon 최적화 구현 방안에 대해서 기술한다.

2. 배경 지식

2.1 FALCON 알고리즘

FALCON은 Fast-Fourier Lattice-based Compact Signatures over NTRU의 줄임말로 격자기반의 hash-and-sign 서명 기법이다. FALCON의 안전성은 NTRU 격자에서 Short Integer Solution(SIS) 문제의 어려움에 기반하고 있다.

FALCON은 트리 데이터 구조, 부동소수점 연산, discrete Gaussian distribution에서의 랜덤 샘플링과 같은 복잡한 연산을 필요로 하기 때문에 구현하기 어렵고, 키 생성 과정에서 많은 연산 리소스를 사용하는 단점이 있다. 하지만 키를 생성하면 재사용이 가능하며 같은 양자내성암호 중 서명 기법인 SPHINCS+, Dilithium과 비교했을 때, <표 1>과 같이 공개키와 서명의 크기가 작다. 또한 서명 생성 및 검증 속도가 빠르며 메모리의 사용량이 적은 장점이 있다.

<표 1> FALCON, Dilithium, SPHINCS+의 공개키, 서명 크기 비교

parameter set	NIST level	|public key|	|signature|
FALCON-512	I	897	652
SPHINCS+ 128s		32	7,856
SPHINCS+ 128f		32	17,088
DILITHIUM-2	II	1,312	2,420
DILITHIUM-3	III	1,952	3,293
SPHINCS+ 192s		48	16,224
SPHINCS+ 192f		48	35,664
FALCON-1024	V	1,793	1,261
DILITHIUM-5		2,592	4,595
SPHINCS+ 256s		64	29,792
SPHINCS+ 256f		64	49,856

2.2 ARM 아키텍쳐

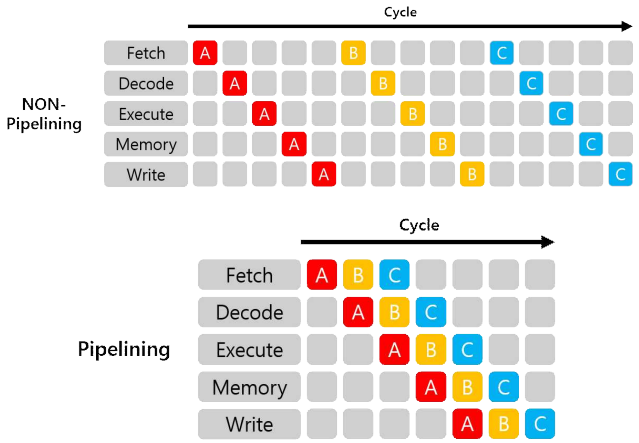
ARM 아키텍쳐는 임베디드 기기에서 많이 사용되는 RISC 프로세서이다. 최근에는 모바일, 테블릿 등 주요 MCU(Micro Controller Unit)로 사용되고 있으며 양자내성암호 임베디드 환경에서의 구현 및 가속화에 사용되고 있다. 특히 ARM 시리즈 중 ARMv8-A 버전을 활용한 최적화 연구가 진행되고 있는데, 이는 병렬 처리 모듈인 NEON 엔진과 Floating-Point(FP) 연산을 효율적으로 처리할 수 있는 FP 명령어 집합을 활용하여 최적화하고 있다. NEON은 Advanced Single Instruction Multiple Data (ASIMD) 확장의 대체 이름으로, 병렬 산술 연산이 가능한 명령어를 지원하며 Single Instruction Single Data (SISD) 명령어에 비해 64비트 피연산자의 경우 2배, 8비트 피연산자의 경우 16배까지 가속이 가능하다. 또한 ARM cortex-M4 플랫폼에서 메모리 사용량을 최소화하여 제한된 리소스 내에서 FALCON을 최적화하는 연구가 진행되고 있다.

2.3 FPGA 및 GPU

FPGA(Field Programmable Gate Array)는 프로그래밍이 가능한 집적 회로로 사용자가 원하는대로 설계가 가능하며 특정 알고리즘에 맞는 하드웨어 구현이 가능하다.

FPGA를 활용한 최적화의 가장 큰 특징은 병렬처리와 (그림 1)과 같은 파이프라이닝의 활용이다. 이를 활용해 고성능 연산처리를 가능하게 하며 효율적으로 복잡한 알고리즘을 최적화할 수 있다. 이외에 높은 유연성을 바탕으로 영상처리, 신호처리 등 다양한 분야에 사용되고 있다.

GPU(Graphics Processing Unit)는 빠른 영상 처리를 위해 개발된 프로세서로, 최근에는 다수의 코어를 활용한 병렬처리, 부동 소수점 연산 등 특정 단순 계산의 가속화가 가능해져 최적화 연구에 활용되고 있다.



(그림 1) 파이프라이닝 예시 그림

3. 최적화 기법

본 장에서는 FALCON의 최적화 연구들을 <표 2>와 같이 플랫폼에 따라 분류하여 기술한다.

<표 2> FALCON의 최적화 연구 분류 표

최적화 플랫폼	관련 연구
ARM 아키텍쳐	[1] [2] [3]
FPGA	[4]
GPU	[5]

3.1 ARM 아키텍쳐를 활용한 최적화 기법

ARM 아키텍쳐를 활용하여 FALCON을 최적화한 연구들은 다음과 같다.

2019년, Tobias Oder 등은 메모리가 제한적인 플랫폼인 ARM coretex-M4F에 FALCON을 메모리 최적화하여 구현함으로써 reference 구현에 비해 동적메모리를 43% 감소시켰다 [1]. 가장 많은 메모리를 차지하는 ffsampling(fast Fourier Sampling) 구현을 최적화하여 임시 메모리를 8KB로 줄였으며 플랫폼에 맞게 메모리를 최소화했다.

2022년, YOUNGBEOM KIM 등은 처음으로 ARMv8 환경에서 효율적인 FALCON 소프트웨어 구현을 제시했다 [2]. FALCON의 성능을 최대화하기 위해 <표 3>과 같이 다항식 곱셈에 사용되는 주요 연산인 FFT, NTT(Number Theoretic Transform) 연산을 병렬 처리 유닛인 NEON 엔진과 벡터 명령어를 활용하여 최적화했다. 또한

NEON 엔진에 사용 가능한 레지스터를 최대한 활용하여 불필요한 메모리 접근 횟수를 줄임으로써 ARMv8 MCU에서의 메모리 최적화 기법을 제시했다. 결과적으로 키 생성 과정에서 15.1%, 서명 과정에서 16.5%, 검증 과정에서 65.4%의 성능향상을 보였다.

2023년, Duc Tri Nguyen and Kris Gaj는 NEON 명령어 체계를 지원하는 ARMv8 프로세서들에서 FALCON의 서명 생성 및 검증을 최적화하였다 [3]. 해당 연구에서는 NEON 명령어를 사용하여 FFT 구현의 twiddle factor 테이블 크기를 기존 16KB에서 4KB로 줄였다. 또한 twiddle factor의 모든 경우에 대한 완전 탐색 경계 분석을 제시함으로써 NTT 연산에서의 Barrett reduction 수를 최소화했다.

<표 3> Jetson Xavier(ARMv8.2)에서의 FALCON 알고리즘 별 FFT, NTT 연산 사용량

ALG	ALL	FFT	NTT	Others
KeyGen	13,056,817 (100%)	3,394,772 **(26%)**	2,350,227 (18%)	7,311,817 (56%)
Sign	580,694 (100%)	278,733 **(48%)**	- (0%)	301,960 (52%)
Verify	48,023 (100%)	- (0%)	31,214 **(65%)**	16,808 (35%)

3.2 FPGA를 활용한 최적화 기법

2023년, Emre Karabulut과 Aydin Aysu는 Xilinx SoC FPGA를 활용하여 FALCON의 서명 생성 과정의 72%를 차지하는 discrete Gaussian sampling을 최적화하는 기법을 제시했다 [4]. 해당 연구에서 제안한 최적화 구조는 sampling 과정에서 필요한 부동소수점 연산을 독립적인 2개의 파트로 나누고 각각 하드웨어와 소프트웨어가 연산을 진행한다. 이를 통해 병렬로 처리될 수 있게 유연한 환경을 설계했으며 sampling을 9.83배 가속화하고 전체 서명 체계를 2.7배 가속화할 수 있음을 보였다.

3.3 GPU를 활용한 최적화 기법

2023년, Wai-Kong Lee 등은 FALCON과 FALCON의 연산을 병렬로 처리할 수 있게 만든 MITAKA에 대해서 GPU로 최적화하는 연구를 진행했다 [5]. 기존 연구와 유사하게 FALCON에서 가장 많은 시간을 소비하는 sampling 과정을 반복적으로 병렬처리할 수 있는 구조를 제안했으며, GPU에서 비용이 많이 드는 재귀함수 호출을 사용하지 않고 서명 생성을 구현했다. 이를 통해 CPU에서 최적화된 AVX2 구현보다 20배 이상 빠르게 구현하는 기법을 제시했다.

4. 결론 및 향후 연구 방향

본 논문에서는 FALCON 알고리즘의 ARM 아키텍쳐, FPGA, 그리고 GPU 플랫폼에 대한 최적화 동향을 분석하였다. 이러한 분석을 통해 각 플랫폼이 공통적으로 병렬 처리 기법이 활용되고 있다는 것을 확인하였다.

FALCON 알고리즘에는 다양한 최적화 기법들이 적용될 수 있다. 특히, 키 생성 과정에서 가장 많은 시간 차지하는 FFT 연산은 최적화가 필요하다고 생각한다. 본 연구에서 제시된 [3]의 내용에 따르면, FFT 연산은 전체 키 생성 과정 중 가장 많은 시간과 리소스를 차지한다. 따라서 이 부분의 최적화는 FALCON 알고리즘의 성능 향상에 크게 기여할 것으로 보인다.

본 논문은 FALCON 알고리즘 최적화에 대한 현행 동향을 중심으로 하였으며, 향후 FFT 연산을 포함한 다른 연산들에 대한 최적화 연구가 계속 진행될 것으로 예상된다.

Acknowledgement

본 연구는 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2021R1A2C1095591)

참고문헌

[1] Oder, Tobias, et al. "Towards practical microcontroller implementation of the signature scheme Falcon." Post-Quantum Cryptography: 10th International Conference, PQCrypto 2019, Chongqing, China, May 8–10, 2019 Revised Selected Papers 10. Springer International Publishing, 2019.

[2] Kim, Youngbeom, Jingyo Song, and Seog Chung Seo. "Accelerating falcon on armv8." IEEE Access 10 (2022): 44446-44460.

[3] Nguyen, Duc Tri, and Kris Gaj. "Fast falcon signature generation and verification using armv8 neon instructions." International Conference on Cryptology in Africa. Cham: Springer Nature

Switzerland, 2023.
[4] Karabulut, Emre, and Aydin Aysu. "A Hardware-Software Co-Design for the Discrete Gaussian Sampling of FALCON Digital Signature." Cryptology ePrint Archive (2023).
[5] Lee, Wai-Kong, et al. "High Throughput Lattice-based Signatures on GPUs: Comparing Falcon and Mitaka." Cryptology ePrint Archive (2023).