PQC DSA 표준화

송민호

유튜브: https://youtu.be/LbPWpzVoK7s

Falcon Towards FN-DSA

- **FN-DSA 표준을 위해 정리한 문서**
  - 표준화를 위해 선택된 Falcon을 기반으로 함
- **Falcon의 알고리즘 및 특징에 대한 내용 포함**
  - 각 특징 별로 적합한 환경에 대한 내용 포함
  - Falcon이 가지고 있는 이점을 강조함
- **FN-DSA 표준화를 위해 필요한 내용 포함**
  - Falcon이 가지고 있는 문제점에 대한 해결책 존재
- **FN-DSA를 위한 추가 변경 사항 포함**

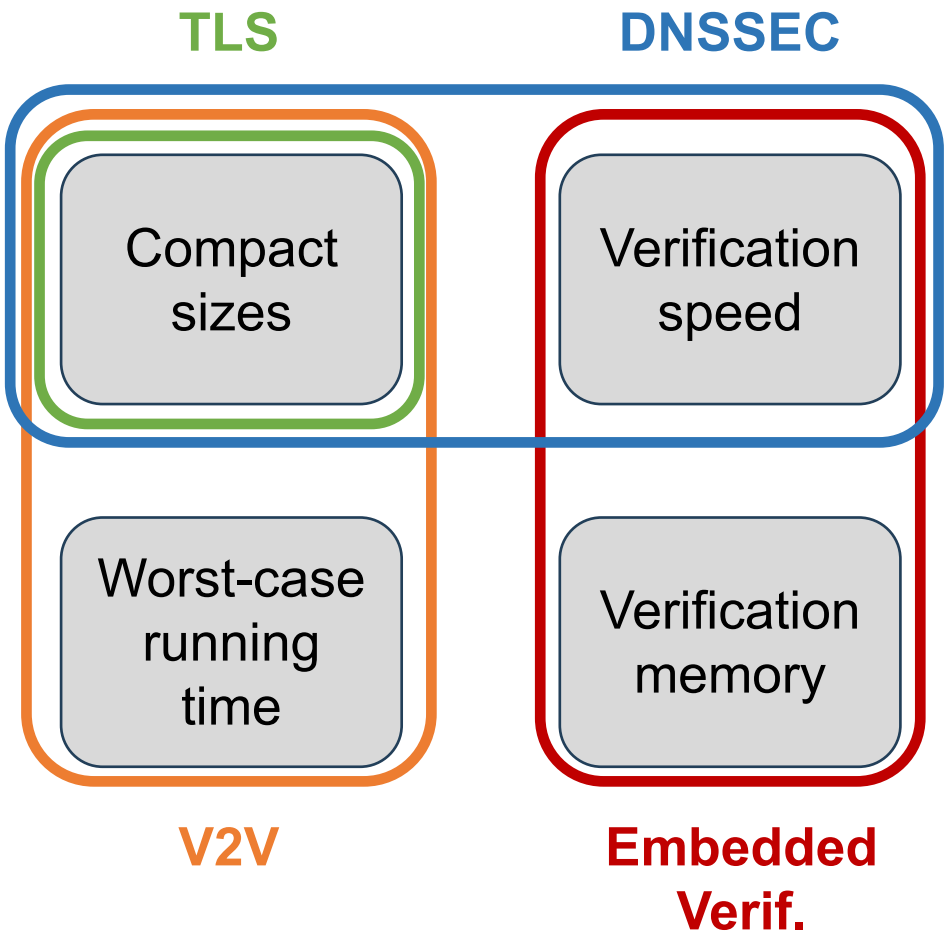
FALCON

- **격자 기반 서명 체계**

- **장점**
  - **작은 크기**
  - **매우 빠른 검증**
  - Dilithium보다는 느리지만 **빠른 서명**

- **단점**
  - 키 생성 및 서명이 **floating-point 연산**이 필요함
  - 키 생성 및 서명의 구현이 **매우 복잡함**

- **여러 특성에 따라 다양한 환경에서 적합함**



FALCON

- **키 생성**
  - 의사 난수 생성기를 사용하여 A 매트릭스 생성
  - B·A=0 조건을 만족하며 작은 계수들로 구성된 B 매트릭스 생성
  - A는 공개 키가 되고 B는 비밀 키가 됨

- **서명**
  - 해시 값 H(msg)를 사용하여 c·A=H(msg)를 만족하는 챌린지 c 계산
  - 비밀 키 B를 사용하여 c와 가까운 벡터 v를 격자 $\mathcal{L}$(B)에서 찾음
  - 서명 값 s는 c와 v의 차이로 정의됨

- **검증**
  - 서명 값 s가 충분히 작은지 확인. 작지 않으면 다시 시작
  - s·A와 H(msg)가 일치하는지 확인

Keygen(1^λ)

1. Gen. matrices **A**, **B** s.t.:
   - **A** is pseudorandom
   - $\mathbf{B} \cdot \mathbf{A} = 0$
   - **B** has small coefficients
2. pk := **A**, sk := **B**

Sign(msg, sk B)

1. Compute **c** such that $\mathbf{c} \cdot \mathbf{A} = H(\text{msg})$
2. **v** ← vector in $\mathcal{L}$(**B**), close to **c**
3. sig := $\mathbf{s} = (\mathbf{c} - \mathbf{v})$

Verify(msg, pk A, sig s)

Check (**s** short) & $(\mathbf{s} \cdot \mathbf{A} = H(\text{msg}))$

적합한 환경 – V2V, TLS

- **Bindel, N et al., NDSS(2022):** Drive(Quantum) Safe! – Towards Post-Quantum Security for V2V communications
  - **V2V 통신에 사용할 수 있는 하이브리드 인증 프로토콜 제안**
    - 기존 암호와 PQC의 맞춤형 융합
  - **하이브리드 디자인 사용 조건**
    - 5개 이하의 조각으로 인증서를 보낼 수 있어야함
    - **PQC 중 Falcon이 유일하게 가능**
  - **기존 통신보다 최대 0.39ms 지연**

- **Sikeridis et al., NDSS(2020):** Post-Quantum Authentication in TLS 1.3: A Performance Study
  - **실제 네트워크 환경에서 TLS1.3 연결을 위한 대기시간 및 세션 처리량 조사**
  - 실시간 네트워크 환경에서 Dilithium 및 Falcon이 제일 성능 높음
    - 서버에서 **Floating point** 하드웨어를 사용할 수 있는 경우 웹에는 **Falcon이 더 적합**

Bindel, Nina, et al. "Drive (Quantum) Safe! â€" Towards Post-Quantum Security for V2V Communications." *IACR Cryptol. ePrint Arch.* 2022 (2022): 483.
Sikeridis, Dimitrios, Panos Kampanakis, and Michael Devetsikiotis. "Post-quantum authentication in TLS 1.3: a performance study." *Cryptology ePrint Archive* (2020).

적합한 환경 – Embedded verif.

- **Beckwith et al., NIST Conference(2022):** FPGA Energy Consumption of Post-Quantum Cryptography
  - **FPGA 환경에서의 PQC DSA 성능 및 리소스 활용도 평가**
    - Falcon은 Dilithium 및 SPHINCS+와 비교하여 에너지 소비 측면에서 이점을 갖음
    - **가장 낮은 에너지 소비, 가장 높은 처리량, 가장 낮은 전송크기 제공**

- **Gonzalez et al., PQCRYPTO(2021):** Verifying Post-Quantum Signatures in 8 kB of RAM
  - **8kB 메모리를 가졌다고 가정한 ARM Cortex-M3 환경에서의 PQC DSA 평가**
    - Falcon의 전체 메모리 공간은 6.5kB로 적합

Beckwith, Luke, J. Kaps, and Kris Gaj. "Fpqa energy consumption of post-quantum cryptography." *Fourth PQC Standardization Conference*. Vol. 23. 2022
Gonzalez, Ruben, et al. "Verifying post-quantum signatures in 8 kB of RAM." *Post-Quantum Cryptography: 12th International Workshop, PQCrypto 2021, Daejeon, South Korea, July 20–22, 202 1, Proceedings 12*. Springer International Publishing, 2021..

적합한 환경 – DNSSEC

- **Muller et al., ACM(2020):** Retrofitting Post-Quantum Cryptography in Internet Protocols: A Case Study of DNSSEC
  - **DNSSEC에서 적합한 PQC 분석**
    - **Falcon-512**의 성능이 현재 사용하는 알고리즘에 제일 가깝고 **DNSSEC의 요구 사항 충족**

- **Goertzen et al., (2023):** Post-Quantum Signatures in DNSSEC via Request-Based Fragmentation
  - **128비트 보안 수준에서 표준화를 위해 NIST 선정 DSA 평가**
  - **시뮬레이션된 네트워크에서 접근 방식 구현 및 성능 평가**
    - **Falcon-512**를 현재 DNSSEC용으로 표준화할 수 있는 **가장 적합한 PQC**로 보고 있음

Müller, Moritz, et al. "Retrofitting post-quantum cryptography in internet protocols: a case study of DNSSEC." *ACM SIGCOMM Computer Communication Review* 50.4 (2020): 49-57.
Goertzen, Jason, and Douglas Stebila. "Post-quantum signatures in DNSSEC via request-based fragmentation." *International Conference on Post-Quantum Cryptography*. Cham: Springer Nature Switzerland, 2023.

FN-DSA 표준화를 위해

- **키 생성 및 서명 과정에 floating-point arithmetic(FPA) 필요**
- **FPA에 대한 완화 필요**
  - 키 생성: Hawk처럼 **fixed-point arithmetic** 사용
  - 서명: **Antrag** 사용

- **Antrag**
  - **Falcon의 단점을 완화시킬 수 있는 새로운 키 생성 알고리즘**
  - **효율적인 trapdoor 제공**
    - Fast Fourier sampler -> Hybrid sampler
    - **서명 과정을 단순하게 할 수 있음**
  - **FPA의 분석이 쉬워지고 제거 가능**
  - **문제점**
    - **표준화하기엔 너무 최근에 나옴**
    - **Antrag 사용시 보안성이 떨어지는지 확인이 안됨**

Espitau, Thomas, et al. "Antrag: Annular NTRU Trapdoor Generation: Making Mitaka as Secure as Falcon." *International Conference on the Theory and Application of Cryptology and Informatio n Security*. Singapore: Springer Nature Singapore, 2023.

FN-DSA에 대한 추가 변경사항 제안

- **더 나은 보안성을 제공하기 위함**
  - BUFF 변환
    - $h = H(salt \parallel msg)$ 대신에 $h = H(H(pk) \parallel salt \parallel msg)$ 계산

- **위조 어려움**
  - $B_\infty \approx 840$을 사용하여 조건 $\| s \|_\infty \leq B_\infty$ 추가

- **Worst-case 실행 시간이 중요한 applications에 적합**
  - 서명 재시작 비율을 매우 작게 설정

전체 성능에 미치는 영향은 매우 적음

FIPS 204 STATUS UPDATE

- **FIPS 204는 ML-DSA 표준을 위해 작성된 문서**
  - 표준화를 위해 선택된 CRYSTALS-Dilithium을 기반으로 함
- **FIPS 204 초안과 달라진 내용 포함**
- **FIPS204에 대해 NIST의 변경 예정 내용 포함**
  - 표준의 실질적인 사용과 적응을 용이하기 위해 변경 예정
- **FIPS204에 대해 변경하지 않을 내용 포함**
  - 변경 후 실용성 및 보안적 이점이 크게 달라지지 않을 것으로 판단하여 NIST는 해당 내용들에 대한 변경을 거절

FIPS 204 – ML-DSA

- **CRYSTALS-Dilithium을 기반으로 하는 ML-DSA 표준 지정**
  - 격자 기반 DSA
  - Fiat-Shamir wit aborts paradigm
  - NTT 적용 가능한 환 사용 → $F_q[x]/\langle x^{256}+1\rangle$

- **ML-DSA는 NIST 기본 승인 서명 방식이 될 것으로 예상**
  - 작은 서명 및 키 크기
  - 빠른 키 생성, 서명, 검증
  - 부동 소수점 연산이 필요하지 않음

ML-DSA 디자인

- **skDecode를 사용하여 비밀 키 $(S_1, S_2, \ldots)$ 확장**
- **ExpandA를 사용하여 matrix A 확장**
- **메시지 생성: $\mu \leftarrow H(tr|M)$**
- **유효한 서명 $(\tilde{c}, z, h)$이 생성될 때까지 Rejection Sampling 루프 수행**
  - $\mathbf{y}$ ←ExpandMask(“Per-sig-random”, “Counter”)
  - $\tilde{c}$ ←H(HighBits(Ay), μ)
  - c ←SampleInBall($\tilde{c}$)
  - $z \leftarrow y + c\, S_1$
  - h ←MakeHint(…)
- **SigEncode를 사용하여 서명 패킹**

FIPS 204 변경 사항

- **Buff 보안 강도를 높이기 위해 tr 및 $\tilde{c}$의 길이를 늘림**
  - tr: 메시지 생성 단계에서 해시 함수의 입력으로 사용되는 값
  - $\tilde{c}$: 서명 과정의 Rejection sampling 루프에서 생성되는 해시 함수의 출력 값

- **”Per-sig-random”을 생성하는 기본 방법이 “결정론적”에서 “Hedged”로 변경**
  - Hedged: 난수를 생성할 때 **비결정론적 요소**(예:무작위 값)를 추가

- **SampleInBall에서 처음 256비트만 사용하지 않고 $\tilde{c}$ 전체 사용**
  - **보안상의 차이는 없음**

- **ExpandMask에서 offset이 아닌 처음부터 SHAKE 결과 사용**
  - **SHAKE 출력 비트 재사용 방지**

- **HintBitUnpack에서 누락된 검사 수정**
  - Strong Unforgeability(SUF-CMA)에 대한 확인 필요

FIPS 204 변경 예정 사항

- **표준의 실질적인 사용과 적응을 용이하기 위해 변경**
  - **불확정 출력 길이를 가지는 SHAKE 사용**
    - 다양한 출력 길이를 필요로 하는 어플리케이션에서 유연성을 제공
  - **해시 함수의 바이트 문자열 처리**
    - **해시 함수의 입출력 처리가 명확해지며 구현의 일관성을 보장**
  - **검증 과정에서 hint의 가중치에 대한 불필요한 검사 제거**
    - Hint Unpacking에서 이미 hint의 가중치가 충분히 작음을 보장함
    - **코드 간소화 및 성능 최적화 가능**
  - **더 낮은 수준의 비난수화된 API 제공**
    - 테스트를 위해 랜덤 값을 내부 키 생성 및 서명 함수의 입력으로 처리할 수 있도록 허용
    - 테스트 과정에서의 **유연성 제공하여 다양한 시나리오를 검증할 수 있음**

FIPS 204 변경하지 않을 사항

- **다양한 의견이 있었으나 NIST가 검토 후 변경하지 않기로 결정**
  - **샘플링 과정에서 XOF를 DRBG로 교체**
    - XOF는 이미 널리 사용되고 있고 DRBG로의 교체는 **큰 이점이 없음**
  - **샘플링 과정에서 XOF를 RNG로 교체**
    - RNG는 예측 가능성이 높아 XOF의 **보안성과 일관성을 대체할 수 없음**
  - **SHAKE를 사용한 모든 해싱을 SHA2로 교체**
    - SHAKE는 다양한 출력 길이를 필요로 하는 과정에서 **더 유연하게 사용 가능**
  - **키 생성 시 개인 랜덤 시드 크기를 32바이트 이상으로 증가**
    - 현재의 32바이트의 크기가 충분하다고 판단
    - 단순 크기를 증가시키는 것이 **큰 보안적 이점을 제공하지 않을 수 있음**

FIPS 205 STATUS UPDATE

- **FIPS 205는 SLH-DSA 표준을 위해 작성된 문서**
  - 표준화를 위해 선택된 SPHINCS+를 기반으로 함
- **여러 comment에 대한 내용 포함**
  - 일부 comment에 대해서는 NIST가 합의 후 변경할 예정
  - 대부분의 comment에 대해서는 변경 예정 없음
- **FIPS 205에 대해 변경하지 않을 사항 포함**
- **변경 사항 중 pre-hash에 대한 내용 포함**
  - NIST는 모든 서명에 pre-hash를 지정할 예정

FIPS 205 – SLH-DSA

- **SLH-DSA(Stateless Hash-Based Signature)**
  - SPHINCS+를 기반으로 하는 SLH-DSA 표준 지정

- **최대 2^{64}개의 서명을 생성하고 검증할 수 있는 보안 제공**

- **매우 큰 서명 크기**
  - 8 KiB – 50 KiB

- **느린 서명 속도, 빠른 검증 속도**

FIPS 205 변경하지 않을 사항

- **SLH-DSA는 12개의 파라미터를 가지고 있음**
  - 다양한 보안성 카테고리
  - SHA256 vs SHAKE
  - 느리고 작은 서명 vs 크고 빠른 서명

- **파라미터 중 일부를 제거하자는 의견이 많음**
  - 명확한 합의가 없어서 NIST는 **12개 파라미터를 유지하기로 결정**

- **더 작은 파라미터를 추가하자는 의견이 있음**
  - 더 작고 빠른 서명을 위해 2^{64} 대신 2^{20} 이나 2^{30} 추가

- **다양한 의견이 제기되었으나 변경 계획이 없다고 답변**
  - SHA-256, SHA-512 조합이 아닌 SHA-512만 사용
  - 성능 개선을 위한 SHAKE256 대신 TurboSHAKE256 사용

PRE-HASH 서명 지정

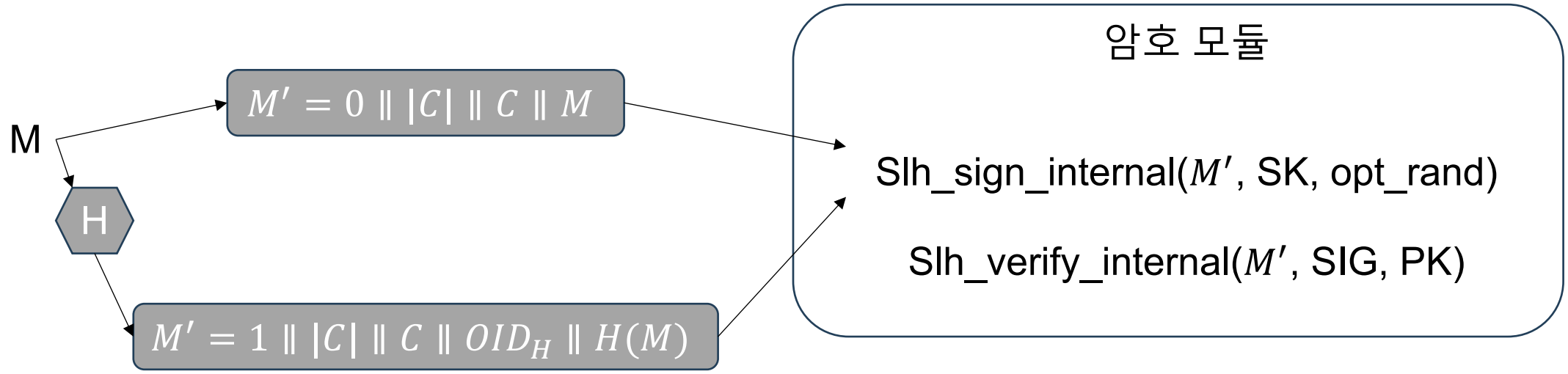
- **NIST는 모든 서명에 대해 Pre-hash를 지정할 계획**
  - Pre-hash: 서명하기 전에 메시지를 해시함수로 변환하는 과정
  - Pre-hash 사용시 해시 테이블에서 특정 값을 검색할 때 해시를 여러 번 계산하는 것을 피할 수 있음
    - Pre-hash 버전과 pure-hash 버전 간의 도메인 분리
    - 외부 해시의 OID 통합
    - 외부 해시 출력은 보안 강도의 2배 이상

- **OID 정의 시 NIST는 pre-hash에 대한 옵션 수 제한할 계획**
  - SLH-DSA-SHA2-{128,192,256}{s,f}-with-SHA512-prehash
  - SLH-DSA-SHAKE-{128}{s,f}-with-SHAKE128-prehash
  - SLH-DSA-SHAKE-{192,256}{s,f}-with-SHAKE256-prehash

  - **SHA512가 SHA256보다 다양한 플랫폼에서 더 빠르기 때문에 SLH-DSA-SHA2-128{s,f}..에서 SHA512 선호**

PRE-HASH 과정

- **메시지 해싱**
  - 서명하려는 메시지를 해시 함수를 사용하여 해싱
- **해시 값 서명**
  - 생성된 해시 값을 DSA로 서명
- **서명 검증**
  - 검증할 때도 동일한 해시 함수를 사용하여 메시지를 해싱한 후, 해당 해시 값과 서명 검증



Q & A