래티스 기반 선형 준동형 다중서명 설계 방법[1)]

최락용* 김광조*

*카이스트 전산학부

A Design Methodology of Linearly Homomorphic Multisignature in Lattice

Rakyong Choi* Kwangjo Kim*

*School of Computing, KAIST

요 약

선형 준동형 서명이란 주어진 서명 알고리즘을 이용해 다수의 개인 서명자 S_i가 각각의 메시지 m_i에 대해서 서명 σ_i를 하고 이 정보를 서버에 저장한다고 가정할 때, 어떤 데이터 수집가가 평균 등 데이터에 대한 선형 함수 f를 요구할 경우 서버 상에서 데이터에 대한 함수 값 $f(m_1, m_2, ..., m_n)$에 대한 올바른 서명 σ_f를 계산할 수 있는 서명을 말한다. 본 논문은 기존의 선형 준동형 다중서명을 이용하여 더 효율적으로 선형 함수의 서명을 계산할 수 있는 선형 준동형 다중서명을 설계하였다.

I. 서론

점점 늘어나는 클라우드 시스템 환경으로 인해 클라우드 보안은 정보보호의 한 가지 중요한 응용분야로 대두되고 있다. 이 중 서버에서 어떤 함수를 계산할 때 실제 메시지에 대한 최소한의 정보를 제공할 수 있도록 암호화 과정이나 서명 과정에서 준동형 성질을 제공하도록 하는 연구가 2009년 Gentry의 래티스를 기반으로 하는 완전 준동형 암호에 대한 논문[1]을 시작으로 해서 활발하게 진행되고 있다.

한편, 준동형 서명에 관한 연구는 네트워크 코딩 분야에서 처음으로 선형 계산에 대해 다루어진 뒤 최초의 래티스 기반 준동형 서명으로 2011년 Boneh와 Freeman에 의해 제안된 선형 준동형 서명(이하 [BF11a])[2] 및 다항식 준동형 서명(이하 [BF11b])[3], Zhang 등이 제안한 선형 준동형 집합서명[4], 2015년 Gorbunov 등에 의해 제안된 완전 준동형 서명[5] 등이 있었으며 최근 Choi와 Kim에 의해 제안된 선형 준동형 다중서명에 대한 논문(이하 [CK16])[6]은 기존의 연구에서 개인 서명자에 의한 서명만 고려하는 단점을 보완하여 회사, 정부 등 그룹 서명자가 존재하는 실제 클라우드 시스템 보안으로 응용분야를 확장하였다.

본 논문은 기존에 알려진 준동형 서명에서 사용한 기법들을 분석하여, 효율적인 준동형 다중서명에 대해 제시한다.

1.1 논문의 구성

본 논문의 구성은 다음과 같다. 우선 II장에서는 래티스의 정의와 래티스 기반 알고리즘에는 어떤 것들이 있는지 알아보며, III장에서 기존의 래티스 기반 선형 준동형 서명과 선형 준동형 다중서명의 설계방법에 대해서 기술한다. 이후 IV장에서 새로운 선형 준동형 다중서명을 기존의 연구를 통해 설계방법을 논하고 이 때 각 참여자의 역할에 대해서 논의한다. 마지막으로 V장에서는 현재까지 진행 결과를 요약하고 추후 연구할 내용에 대해 제시한다.

1) 이 논문은 2016년도 정부(미래창조과학부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. NRF-2015R1A2A2A01006812).

II. 배경지식

래티스 기반 암호는 최근 암호학 분야에서 가장 많이 사용되는 도구로 양자 컴퓨터를 이용한 Shor의 알고리즘에 의한 공격에도 안전한 포스트 양자 암호이다. 이 장에서는 래티스의 정의와 래티스 기반 어려운 문제에 대해 설명하고 래티스 기반 알고리즘에는 어떤 것들이 있는지 기술한다.

2.1 래티스 및 래티스 기반 난제

래티스(lattice)란 덧셈 연산을 가지는 임의의 군(group) G의 이산 부분군(discrete subgroup)을 말하며, 이러한 이산 부분군을 생성해주는 생성 집합(generating set)을 기저(basis)라고 말한다. 특히, 군 G가 정수 공간 $\mathbb{Z}^m$ 상에 있을 경우 여기서 나오는 이산 부분군을 정수 래티스(integer lattice)라고 칭하며, 임의의 행렬 $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$에 대해서도 $A \cdot e = 0 \bmod q$를 만족하도록 래티스 $\Lambda_q^{\perp}(\mathbf{A})$를 생성할 수 있다.

래티스를 이용한 어려운 문제로는 대표적으로 Learning With Errors(LWE) 문제와 Small Integer Solution(SIS) 문제가 있으며 각각 암호 및 서명을 만드는 데 주로 이용된다. 이 때 SIS 문제는 주어진 벡터 $a_1, a_2, \ldots, a_m \in \mathbb{Z}^m$에 대해 $\sum_{i=1}^{m} z_i a_i = 0$을 만족하는 0이 아닌 $z_1, z_2, \ldots, z_m \in \{-1, 0, 1\}$을 찾는 문제로 기저의 크기가 충분히 작은 기저를 알고 있을 경우에만 쉽게 계산할 수 있는 문제이다.

2.2 래티스 기반 알고리즘

래티스 기반 암호는 LWE 문제 및 SIS 문제를 통해 안전성을 증명하며 따라서 파라미터 설정 과정에서 크기가 작은 기저인 트랩도어(trapdoor) 행렬을 비밀키로 가져야 한다.

이러한 트랩도어를 만들어내는 트랩도어 생성 알고리즘 *TrapGen(n, l, q)*은 $\mathbb{Z}_q^{l \times n}$에서 행렬 $\mathbf{A} \in \mathbb{Z}_q^{l \times n}$와 행렬 $\mathbf{A}$의 래티스 $\Lambda_q^{\perp}(\mathbf{A})$의 작은 크기의 기저가 되는 트랩도어 행렬 $\mathbf{T}$를 추출하는 알고리즘으로 Alwen과 Peikert에[7] 의해 제안되었다.

또한 Cash 등에[8] 의해 트랩도어 생성 알고리즘에서 나온 행렬 $\mathbf{A} \in \mathbb{Z}_q^{l \times n}$와 $\mathbf{A}$의 트랩도어 행렬 $\mathbf{T}$로부터 행렬 $B = A \| A' \in \mathbb{Z}_q^{l \times (n+n')}$와 행렬 $\mathbf{A}$의 트랩도어 행렬 $\mathbf{T}$로부터 행렬 $\mathbf{B}$의 트랩도어 행렬 $\mathbf{S}$를 추출하는 기저 추출 알고리즘인 *ExtBasis(**T**, **B**)*를 제안하였다.

한편 Gentry 등은[9] 어떤 래티스 $\Lambda_q^{\perp}(\mathbf{A})$와 트랩도어 행렬 $\mathbf{T}$가 주어졌을 때, $A \cdot \sigma = u \,(\bmod q)$를 만족하도록 가우시안 분포 상에서의 벡터 σ를 생성해주는 가우시안 샘플링 알고리즘 *SamplePre(**A**, **T**, γ, **u**)*를 제안하였으며 이 알고리즘은 SIS 문제의 해를 찾는 문제로 치환할 수 있다.

III. 선형 준동형 서명

이번 장에서는 기존의 선형 준동형 다중서명과 준동형 서명에 대한 동작과정을 소개하고, 각 참여자의 역할에 대해 소개한다.

3.1 [BF11a]와 [BF11b] 방식[2-3]

임의의 서명 기법에 대해 서명자 S_i가 데이터 m_i에 대해서 서명 σ_i를 하고 서버에 저장할 때, 어떤 데이터 수집가가 평균 등 데이터 m_i에 대한 임의의 함수 f를 요구할 경우 서버 상에서 각 메시지 m_i에 대한 정보 공개 없이 메시지의 기존 서명 σ_i을 통해 함수 값 $f(m_1, m_2, \ldots, m_n)$에 대한 올바른 서명 σ_f를 계산할 수 있다면 이 서명 기법은 준동형 성질을 만족한다고 하며 만족하는 f의 차수에 따라 선형 준동형 성질, 다항식 준동형 성질, 완전 준동형 성질이라고 부른다. 또한 각 성질을 만족하는 서명 기법을 선형 준동형 서명, 다항식 준동형 서명, 또는 완전 준동형 서명이라고 부른다. 본 절에서는 이 중 [BF11b] 방식[3]은 다음과 같으며 [BF11a] 방식[2]과 달리 다항식 개수의 연산에 대해서도 선형 준동형 성질을 만족한다.

Setup(n, k): 보안 매개변수 n과 데이터 집합의 크기 k에 대해

1. $q \geq (nkp)^2$ 이 되도록 $p, q = poly(n)$을 선택하고 $l = \left\lfloor \frac{n}{6\log q} \right\rfloor$ 로 설정.

2. *TrapGen*(q, l, n) 알고리즘을 통해 나오는 행렬 $A \in F_q^{l \times n}$의 트랩도어 행렬 T_q로 잡고 $\Lambda_1 = p\mathbb{Z}^n$, $\Lambda_2 = \Lambda_q^{\perp}(A)$, $T = p \cdot T_q$로 잡으면 T는 $\Lambda_1 \cap \Lambda_2$의 기저가 됨.

3. $\nu = p \cdot \sqrt{n \log q} \cdot \log n$, $H: \{0,1\}^* \rightarrow F_q^l$으로 잡고 공개키 $pk = (\Lambda_1, \Lambda_2, \nu, k, H)$와 비밀키 $sk = T$를 출력.

Sign(sk, τ, m_i, i): 태그 τ를 가진 데이터 집합 내의 각 데이터 m_i에 대한 해쉬 값 $H(\tau \| i) = \alpha_i$를 구하고 이를 통해 $t \bmod p = m$과 $A \cdot t \bmod q = \alpha_i$를 만족하는 $t \in \mathbb{Z}^n$를 찾고 *SamplePre*$(\Lambda_1 \cap \Lambda_2, T, \nu, t)$를 이용하여 서명 σ를 출력.

Verify(pk, τ, m, σ, f): 함수 f를 통해 만들어진 데이터 m과 서명 σ에 대해 $\sigma \bmod p = m$, $A \cdot \sigma \bmod q = \alpha_i$, $\| \sigma \| \leq k \cdot \frac{p}{2} \cdot \nu\sqrt{n}$을 모두 만족하면 1 (수용), 어느 하나라도 만족하지 않으면 0 (거절)을 출력.

Eval$(pk, \tau, f, \{\sigma_i\})$: 같은 태그 τ를 가지는 서명 σ_i와 $f(m_i) = \sum_{i=1}^{k} c_i m_i$에 대해 $\sigma = \sum_{i=1}^{k} c_i \sigma_i$를 출력.

3.2 [CK16] 방식[6]

Choi와 Kim의 논문은 그룹 서명자의 서명을 하기 위해 초기에 데이터 m_i을 분배했으며 *RandBasis* 알고리즘을 통해 한 여러 서명자가 있어 각 서명자마다 서로 다른 비밀키를 공유할 수 있는 특징을 가지며 다음과 같이 동작한다.

Setup$(n, g,$ **params**$)$: g명의 그룹 서명자가 있을 때, *TrapGen(n, l, 2q)* 알고리즘을 g번 실행하여 행렬 A_i와 트랩도어 행렬 T_i를 잡고, $A = A_1 \| A_2 \| \cdots \| A_g$, $H: \{0,1\}^* \rightarrow \mathbb{Z}_q^{l \times n}$로 잡고 각 그룹 내의 서명자의 공개키 $pk = (A, H)$와 비밀키 $sk = T_i$를 출력.

PreShare(g, m): *SamplePre*$(A, T, \gamma, 0)$ 알고리즘을 이용해 메시지 m에 각각의 멤버에 노이즈 $e_1, e_2, \cdots, e_g$를 더하여 $m_1, m_2, \cdots, m_g$로 그룹 내의 각 서명자에게 분배하는 알고리즘. 이 때, $m = \sum_{i=1}^{g} m_i$이 됨.

Sign(sk_i, id, m_i): 그룹 내 각 서명자의 서명 알고리즘으로 $B = A \| H(id)$로 잡고 B의 트랩도어 S_i를 *ExtBasis*(T_i, B) 알고리즘을 통해 찾고 *SamplePre*$(B, S_i, \gamma, q \cdot m_i)$ 알고리즘을 이용해 서명 σ_i를 출력.

Combine$(pk, id, g, \{\sigma_i\}_{i=1}^{g})$: 메시지 m의 서명을 계산하는 알고리즘으로 에러가 더해진 각각의 메시지 m_i에 대한 서명을 σ_i라고 할 때, m의 서명 $\sigma = \sum_{i=1}^{q} \sigma_i$를 계산하는 알고리즘이다.

LinCom$(pk, id, \{g_j, \sigma_j\}_{j=1}^{L_g})$: 같은 태그 id를 가지는 서명 σ_j에 대해 α_j가 σ_j에 대한 가중치라고 할 때, $\sigma_{lin} = \sum_{j=1}^{L_g} \alpha_j \sigma_j$를 계산하는 알고리즘이다. 이 때, L_g는 $\sum_{j=1}^{L_g} \alpha_j g_j \leq L$을 만족하도록 해야 한다.

Verify$(pk, id,$ **y**$, \sigma)$: 데이터 **y**와 서명 σ $\| \sigma \| \leq L \cdot \gamma\sqrt{2m}$, $B \cdot \sigma = q \cdot y$를 만족하는 올바른 서명인지 검증하는 알고리즘.

IV. 새로운 선형 준동형 다중서명

[BF11b] 방식[3]에서 제안된 선형 준동형 서명 논문을 기존의 선형 준동형 다중 서명에 적용하면 더 많은 선형 함수를 만족할 수 있으며 선형 함수값을 계산한 이후의 서명의 크기 또한 전체 데이터 집합의 로그에 비례하므로 상대적으로 작은 값을 가지게 된다.

Setup$(n,$ **params**$)$: 서버 상에서 서명을 위해 필요한 공개키와 서명키를 설정하는 알고리

증으로 [BF11b]의 방식과 동일함.

PreShare(g, m_i): 신뢰할 수 있는 딜러가 데이터 m을 그룹 내 서명자들에게 분배하는 알고리즘으로 [CK16]의 방식과 동일함.

Sign$(sk, \tau, m_{(i,j)}, i)$: 각 서명자가 태그 τ를 가진 데이터 집합 내의 각 데이터 m_i에 대한 해쉬 값 $H(\tau \| i) = \alpha_i$를 구한 뒤 대해 $t_{(i,j)} \bmod p = m_{(i,j)}$과 $A \cdot t_{(i,j)} \bmod q = \alpha_i$를 만족하는 $t_{(i,j)} \in \mathbb{Z}^n$를 찾고 $SamplePre(\Lambda_1 \cap \Lambda_2, T, \nu, t_{(i,j)})$를 이용하여 서명 $\sigma_{(i,j)}$를 출력하는 알고리즘.

Combine$(pk, \tau, g, \{\sigma_{(i,j)}\}_{j=1}^{g})$: 데이터 m_i에 대한 각 서명자의 서명 $\sigma_{(i,j)}$로부터 서명 $\sigma_i = \sum_{j=1}^{g} \sigma_{(i,j)}$를 찾아내는 알고리즘.

Eval$(pk, \tau, f, \{\sigma_i\})$: 같은 태그 τ를 가지는 서명 σ_i와 $f(m_i) = \sum_{i=1}^{k} c_i m_i$에 대해 $\sigma = \sum_{i=1}^{k} c_i \sigma_i$를 출력하는 알고리즘.

Verify(pk, τ, m, σ, f): 주어진 서명이 올바른 서명인지 검증하는 알고리즘으로 [BF11b]의 방식과 동일함.

위에 제안된 서명의 정확성 및 위조불가능성은 [BF11b]의 방식[3]에서 나온 서명의 정확성과 위조불가능성에 의해 자명하며 딜러의 계산량을 제외하면 그룹 서명자를 가지면서도 각 사용자의 서명과정에서의 계산량이 [BF11b]의 방식과 동일하다.

V. 결론

본 논문에서 우리는 기존에 소개된 래티스 기반 선형 준동형 서명 및 선형 준동형 다중서명에 대해 소개한 뒤, 기존보다 효율적인 선형 준동형 다중서명을 설계하였다.

추후 진행할 연구로는 제안한 서명 기법의 안전성 증명을 위해 위조불가능성의 엄밀한 증명을 진행하고자 하며, 또한 [CK16]의 방식과 같이 각 서명자가 서로 다른 비밀키를 가지는 서명 기법에 대한 연구도 진행하고자 한다.

[참고문헌]

[1] C. Gentry, "A Fully Homomorphic Encryption Scheme," Doctoral dissertation, Stanford University, 2009.

[2] D. Boneh and D. M. Freeman, "Linearly Homomorphic Signatures over Binary Fields and New Tools for Lattice-based Signatures," Public Key Cryptography - PKC 2011, Springer Berlin Heidelberg, 2011, pp. 1-16.

[3] D. Boneh and D. M. Freeman. "Homomorphic signatures for polynomial functions," Advances in Cryptology - EUROCRYPT 2011. Springer Berlin Heidelberg, 2011. pp. 149-168.

[4] P. Zhang, J. Yu, and T. Wang, "A Homomorphic Aggregate Signature Scheme Based on Lattice," Chinese Journal of Electronics, 21(4), 2012, pp. 701-704.

[5] S. Gorbunov, V. Vaikuntanathan, and D. Wichs, "Leveled Fully Homomorphic Signatures from Standard Lattices," Proceedings of the 47th Annual ACM Symposium on Theory of Computing (STOC 2015), ACM, 2015, pp. 469-477

[6] R. Choi and K. Kim, "Lattice-based Multi-signature with Linear Homomorphism," 2016 Symposium on Cryptography and Information Security (SCIS 2016), 1D1-3, 2016.

[7] J. Alwen and C. Peikert, "Generating Shorter Bases for Hard Random Lattices," Theory of Computing Systems, 48(3), Springer Berlin Heidelberg, 2011, pp. 535-553.

[8] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. "Bonsai Trees, or How to Delegate a Lattice Basis," Journal of Cryptology, 25(4), 2012, pp. 601-639.

[9] C. Gentry, C. Peikert, and V. Vaikuntanathan, "Trapdoors for hard lattices and new cryptographic constructions," Proceedings of the 40th annual ACM Symposium on Theory of Computing, ACM, 2008, pp. 197-206.