Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.29, NO.6, Dec. 2019 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2019.29.6.1285

영지식 증명을 활용한 프라이버시 보장 신용평가방법*

박 철,† 김 종 현, 이 동 훈‡
고려대학교 정보보호대학원

Privacy-Preserving Credit Scoring Using Zero-Knowledge Proofs*

Chul Park,† Jonghyun Kim, Dong Hoon Lee‡
Graduate School of Information Security, Korea University

요 약

현재의 신용평가체계에서는 신용정보회사가 개인의 신용정보를 금융기관으로부터 수집하고 이를 기반으로 신용평가를 수행한다. 하지만 이런 신용평가방법은 민감한 신용정보가 하나의 중앙기관에 집중되기에 프라이버시 침해 소지가 있으며, 외부의 공격이 성공할 경우 대규모의 개인정보가 유출될 수 있다. 본 논문에서는 이 문제를 해결하기 위해 개인이 스스로 금융기관으로부터 수집한 신용정보를 바탕으로 신용점수를 계산하고, 이 신용점수가 정상적으로 계산되었음을 영지식 증명과 블록체인으로 증명하는 프라이버시 보장 신용평가방법을 제안한다. 또한 영지식 증명에 이용된 신용정보가 금융기관에서 실제로 제공한 값인지 블록체인을 통해 확인하기 위해, 커밋된 입력에 대해 효율적으로 증명이 가능한 영지식 증명 기법을 제안한다. 이 기법은 Agrawal 등의 기법과 달리 완벽한 영지식성을 제공하며, CRS와 증명의 크기가 작고 증명과 검증 과정이 빠르다. 그리고 제안한 신용평가방법에 실제 환경과 유사한 신용점수 알고리즘을 적용하여 구현 및 실험함으로써 실제 환경에 이용 가능함을 확인하였다.

ABSTRACT

In the current credit scoring system, the credit bureau gathers credit information from financial institutions and calculates a credit score based on it. However, because all sensitive credit information is stored in one central authority, there are possibilities of privacy violations and successful external attacks can breach large amounts of personal information. To handle this problem, we propose privacy-preserving credit scoring in which a user gathers credit information from financial institutions, calculates a credit score and proves that the score is calculated correctly using a zero-knowledge proof and a blockchain. In addition, we propose a zero-knowledge proof scheme that can efficiently prove committed inputs to check whether the inputs of a zero-knowledge proof are actually provided by financial institutions with a blockchain. This scheme provides perfect zero-knowledge unlike Agrawal et al.'s scheme, short CRSs and proofs, and fast proof and verification. We confirmed that the proposed credit scoring can be used in the real world by implementing it and experimenting with a credit score algorithm which is similar to that of the real world.

Keywords: zero-knowledge proof, zk-SNARK, blockchain, privacy-preserving credit scoring

Received(10. 23. 2019), Modified(11. 18. 2019), Accepted(11. 26. 2019)

* 이 논문은 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2016-6-00599, 함수서명 설계기법 및 응용기술 연구).

* 이 논문은 금융보안원·금융정보보호협의회·금융보안포럼이 주최한 「2019 디지털 금융혁신과 금융보안 공모전」에 응모한 논문을 토대로 수정 보완한 논문임.

† 주저자, chul0101@korea.ac.kr

‡ 교신저자, donghlee@korea.ac.kr(Corresponding author)

I. 서 론

개인이 경제활동을 수행하기 위해 필요한 자금을 조달하는 방법 중 대표적인 것이 금융기관의 대출이다. 또한 금융기관의 대출은 예대차익을 통해 얻을 수 있는 금융기관의 중요한 수익 원천이기도 하다. 하지만 금융기관은 채무자가 대출을 정상적으로 상환하지 않을 위험성을 항상 고려해야 한다. 채무자가 대출을 정상적으로 상환하지 않을 때, 담보대출의 경우에는 채무자의 담보물을 매각한 후 그 가액으로부터 원금과 이자를 비교적 쉽게 회수할 수 있지만, 신용대출의 경우에는 민사소송을 통해서 채무자의 재산에 강제집행을 해야 한다. 그러나 이는 절차가 복잡하며 채무자의 재산이 부족할 경우 원금 및 이자의 회수가 어렵다는 문제가 있다.

따라서 금융기관은 신용대출을 제공할 때 개인이 성실하게 대출금을 상환할지 사전에 판단해야 하며, 이를 위한 수단이 신용정보회사가 제공하는 신용점수(credit score)이다. 신용점수는 신용정보회사가 평가한 개인의 신용도를 의미하며 이를 정확히 계산하기 위해 연체이력, 현재 대출상황, 소득 등 개인의 신용정보가 각 금융기관으로부터 신용정보회사로 수집된다[1,2].

하지만 이 방법은 개인의 신용정보가 하나의 기관에 집중되어 저장되기 때문에 내부자에 의한 프라이버시 침해 소지가 있다. 또한 외부로부터의 공격이 성공할 경우 대규모의 신용정보가 유출될 수도 있다. 실제로 2017년에는 미국의 대표적인 신용정보회사인 에퀴팩스(Equifax)에 대한 해커의 공격이 있어 1억 4천3백만 명의 개인정보가 유출되었다[3]. 이는 중앙기관에 의존하여 서비스가 제공되기 때문에 발생하는 근본적인 현상이다. 이에 따라 최근에는 영지식 증명과 블록체인을 활용하여 중앙기관 없이 신용평가를 수행함으로써 이 문제를 해결하고자 하는 아이디어가 제안되고 있다[4,5].

영지식 증명(zero-knowledge proof)은 특정한 사실이 참이라는 것을 증명하되 그 이외에는 어떤 사실도 노출하지 않는 암호학적 기법이다. 따라서 개인은 각 금융기관으로부터 스스로 신용정보를 수집하고 이를 이용하여 신용점수를 계산한 후 이에 대한 영지식 증명을 생성하면 다른 정보를 노출하지 않고 신용점수만을 대출기관에 제공할 수 있다[4,5]. 이때 신용정보를 개인에게 제공한 금융기관이 해당 신용정보에 대한 커밋먼트(commitment)를 블록체인에 공유하면, 대출기관은 영지식 증명 검증 과정에서 이를 활용하여 증명 생성 시 이용된 신용정보가 금융기관이 제공한 것과 일치하는지 확인할 수 있다[5]. 이 과정의 전체적인 흐름은 Fig. 1.과 같다.

본 논문에서는 신용평가에 영지식 증명을 활용할 수 있다는 [4,5]의 아이디어를 구체화하여 안전성 증명이 가능한 프라이버시 보장 신용평가방법을 제안한다. 또한 영지식 증명 과정에서 이용된 신용정보가 금융기관이 실제로 제공한 값인지 확인하기 위해, 커밋된 입력(committed input)에 대해 증명이 가능한 효율적인 영지식 증명 기법을 제안한다.

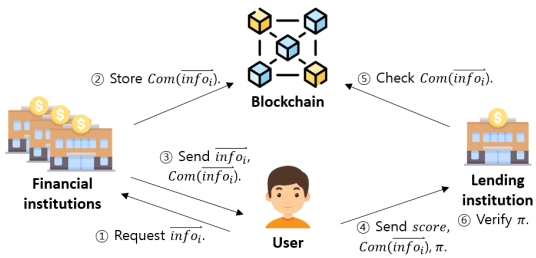

Fig. 1. Privacy-preserving credit scoring overview

1.1 기여도

본 논문에서는 영지식 증명과 블록체인을 활용한 프라이버시 보장 신용평가방법을 제안하였다. 이 신용평가방법에서는 [4,5]의 아이디어에 기반하되 한국의 금융환경을 고려하여 프로토콜을 설계하였다. 또한 금융기관이 블록체인에 저장한 신용정보의 커밋먼트를 영지식 증명의 검증뿐 아니라 감사(audit) 프로세스에서도 활용할 수 있도록 했다. 그리고 제안한 신용평가방법이 영지식 증명의 안전성(security)을 만족함을 확인하였다.

또한 Agrawal 등의 커밋된 입력에 대한 영지식 증명[6]이 완벽한 영지식성(zero-knowledge)을 만족하지 않음을 보였으며, Groth의 zk-SNARK 기법[8]과 Agrawal 등의 기법[6]을 결합 및 개선하여 완벽한 영지식성을 만족하면서 효율성이 개선된 새로운 커밋된 입력에 대한 영지식 증명을 설계하고 안전성을 증명하였다.

마지막으로 실제 환경과 유사하게 신용정보의 종류, 금융기관의 수, 신용점수 알고리즘을 설계한 후, CRS(Common Reference String) 및 증명(proof)의 크기 그리고 증명 생성(prove) 및 검증

(verify)에 소요되는 시간을 측정하여 논문에서 제안한 방법이 실제 환경에서 이용 가능하다는 것을 실험적으로 확인하였다.

1.2 관련 연구

2013년 Parno 등이 KE 가정(Knowledge of Exponent assumption)에 기반하여 효율적인 zk-SNARK 기법[7]을 Pinocchio라는 이름으로 제안하였다. 2016년에는 Groth가 Parno 등의 기법[7]보다 효율성이 개선된 기법[8]을 제안하였으며, 이는 KE 가정보다 강력한 가정인 제네릭 그룹 모델(generic group model)에서 안전성이 증명되었다. 이 모델은 공격자가 그룹(group) 연산 및 페어링(pairing) 연산을 제공하는 오라클(oracle) 질의를 제외하고는 다른 어떤 정보도 그룹 원소로부터 얻어낼 수 없다는 가정이다.

2016년 Fiore 등은 증명자가 원본 데이터에 대해 수행한 연산을 증명자가 생성한 증명과 검증자가 계산한 원본 데이터의 해쉬 값만으로 검증하는 기법[9]을 제안하였다. 하지만 이 기법은 검증 가능한 연산(verifiable computation)에 초점을 두어 설계되었기 때문에 완벽한 영지식성을 제공하지 않았다.

2018년에 Agrawal 등은 하나의 스테이트먼트(statement)를 여러 구성요소로 분할한 후 이를 커밋먼트로 결합하여 증명을 수행하는 커밋된 입출력에 대한 비대화형 영지식 증명(non-interactive zero-knowledge proof)[6]을 제안하였다. 또한 2019년에 Campanelli 등은 하나의 스테이트먼트를 구성요소로 분할한 후 커밋먼트로 결합하여 증명을 수행하는 또 다른 기법[10]을 LegoSNARK라는 이름으로 제안하였다. 이런 기법은 구성요소 별로 그 특성에 맞는 효율적인 영지식 증명 기법을 활용할 수 있다는 장점이 있다.

이 두 가지 기법은 모두 CRS의 크기가 비교적 작고 증명과 검증이 빠르다. 다만 Agrawal 등의 기법[6]은 증명의 크기가 커밋된 입력의 수 등에 비례하여 증가한다는 문제가 있으며, Campanelli 등의 기법[10]은 이를 해결하여 간결한(succinct) 증명을 이용하도록 하였다. 반면, Campanelli 등의 기법[10]은 Agrawal 등의 기법[6]에 비해 CRS의 크기가 다소 크다는 단점이 있다.

개인의 신용정보에 기반하여 신용평가를 수행하고 이를 대출에 활용하는 방법은 1950년대부터 이용되었다. 이에 따라 신용점수를 효과적으로 산정하는 방법도 기업 또는 학계에서 다양하게 연구되었으며, 통계학적 이론이 발전됨에 따라 이를 신용평가에도 적용해 왔다[1]. 영지식 증명을 통해서 신용평가를 수행하는 방법은 아직 본격적인 연구가 이루어지고 있지 않다. 최근 영지식 증명이 블록체인에서의 프라이버시를 보장하는 방법으로 관심을 받는 가운데 신용평가가 또 다른 응용 분야로 제시되고 있는 정도이다[4,5].

II. 배경지식

2.1 영지식 증명

영지식 증명은 어떤 문장이 참이라는 사실을, 그 외의 정보를 노출하지 않으면서 증명하는 기법이다. 이때 증명하는 문장을 스테이트먼트(statement), 문장이 참임을 확인할 수 있는 비밀을 위트니스(witness)라 한다. 영지식 증명의 특성은 다음과 같다.

- 완전성(completeness) : 스테이트먼트가 참일 경우, 정직한 증명자는 검증을 통과한다.
- 건전성(soundness) : 스테이트먼트가 거짓일 경우, 악의적인 증명자라도 검증을 통과할 수 없다.
- 영지식성(zero-knowledge) : 스테이트먼트가 참이라는 사실 외의 정보는 노출되지 않는다.

2.1.1 zk-SNARK

zk-SNARK(zero-knowledge Succinct Non-interactive ARgument of Knowledge)는 비대화형 영지식 증명의 하나로서 증명의 크기가 작고 검증에 소요되는 시간이 짧다는 장점이 있다. zk-SNARK의 알고리즘은 다음과 같다[11].

- KeyGen(R)→σ : KeyGen은 스테이트먼트 x와 위트니스 w 간의 관계(relation) R을 입력받아 CRS σ를 출력한다.
- Prove(R,σ,x,w)→π : Prove는 관계 R, CRS σ, 스테이트먼트 x, 위트니스 w를 입력받아 x가 참이라는 증명 π를 출력한다.
- Verify(R,σ,x,π)→0/1 : Verify는 관계 R, CRS σ, 스테이트먼트 x, 증명 π를 입력받아

검증 결과가 참일 경우 1, 그렇지 않으면 0을 출력한다.

또한 zk-SNARK가 만족해야 하는 특성은 다음과 같다[11].

- 완벽한 완전성(perfect completeness) : 관계 R에 대해, 모든 $(x,w) \in R$는 다음을 만족한다.
  $$\Pr\left[\mathrm{Verify}(R,\sigma,x,\pi)=1 : \underset{\mathrm{Prove}\ (R,\sigma,x,w)\to\pi}{(x,w)\in R}\right]=1$$
  이는 스테이트먼트가 참이면 위트니스를 이용하여 만든 증명은 반드시 검증자의 검증을 통과한다는 뜻이다.
- 계산적인 지식 건전성(computational knowledge soundness) : 모든 다항식 시간 공격자 A에 대해 다음을 만족하는 다항식 시간 익스트랙터(extractor) E가 존재한다.
  $$\Pr\left[\begin{matrix}\mathrm{Verify}(R,\sigma,x,\pi)=1 \\ (x,w)\notin R\end{matrix} : \begin{matrix}A(R,\sigma,z)\to(x,\pi) \\ E(R,\sigma,z)\to w\end{matrix}\right] \le negl(\lambda)$$
  이는 검증자의 검증을 통과할 경우 해당 스테이트먼트가 참일 뿐 아니라 증명자가 위트니스를 알고 있다는 것이며, 그렇지 않을 가능성은 다항식 시간 공격자에 대해 무시할 만하다는(negligible) 뜻이다.
- 완벽한 영지식성(perfect zero-knowledge) : 모든 다항식 시간 공격자 A에 대해서 다음을 만족한다.
  $$\Pr\left[\begin{matrix}(x,w)\in R \\ A(R,\tau,\pi)=1\end{matrix} : \begin{matrix}\mathrm{KeyGen}(R)\to\sigma \\ A(R,\sigma)\to(x,w) \\ \mathrm{Prove}(R,\sigma,x,w)\to\pi\end{matrix}\right] = \Pr\left[\begin{matrix}(x,w)\in R \\ A(R,\tau,\pi)=1\end{matrix} : \begin{matrix}\mathrm{SimKeyGen}(R)\to(\sigma,\tau) \\ A(R,\sigma)\to(x,w) \\ \mathrm{Sim\,Prove}(R,\tau,x)\to\pi\end{matrix}\right]$$
  이는 위트니스를 알고 만들어진 정상적인 증명의 분포(distribution)와 위트니스를 모르나 CRS 생성 과정에서 이용된 트랩도어(trapdoor) τ를 이용하여 시뮬레이트(simulate)된 증명의 분포가 같다는 뜻이다. 다시 말하면 이는 증명을 통해서 어떠한 정보도 노출되지 않는다고 볼 수 있다.
- 간결성(succinctness) : 증명의 크기는 $O_\lambda(1)$이며 검증에 소요되는 시간은 $O_\lambda(|x|)$이다.

본 논문에서 참조하는 기법[6-8]은 증명하고자 하는 관계를 Fig. 2.와 같이 덧셈 및 곱셈 게이트로 산술연산을 수행하는 산술회로(arithmetic circuit)로 표현한다. 이때 공개 입출력의 와이어(wire)가 스테이트먼트이고 비밀 입력과 내부의 와이어가 위트니스에 해당한다. 또한 본 논문에서 참조하는 기법[6-8]은 산술회로를 표현하는 방법 중 다음의 QAP[8]를 이용한다. 여기서 m은 와이어의 수, $t(x)$의 차수(degree)는 곱셈 게이트의 수에 대응된다.

정의 1. [Quadratic Arithmetic Program(QAP)] 유한 필드(field) $\mathbb{F}$, 자연수 m, m이하의 자연수 l, $\mathbb{F}[x]$의 원소 $u_i(x), v_i(x), w_i(x), t(x)$에 대해 QAP는 $(\mathbb{F}, l, \{u_i(x), v_i(x), w_i(x)\}_{i=0}^{m}, t(x))$으로 정의된다. 이때 $u_i(x), w_i(x), w_i(x)$의 차수는 $t(x)$보다 작아야 하며, 스테이트먼트 $(a_1,\cdots,a_l)$, 위트니스 $(a_{l+1},\cdots a_m)$에 대해 다음의 관계를 만족해야 한다.

$$\left\{\begin{matrix}(a_1,\cdots,a_l),(a_{l+1},\cdots a_m): \\ \exists h(x)\in\mathbb{F}[x]: \\ \sum_{i=0}^{m}a_iu_i(x)\cdot\sum_{i=0}^{m}a_iv_i(x)-\sum_{i=0}^{m}a_iw_i(x)=h(x)t(x)\end{matrix}\right\}$$

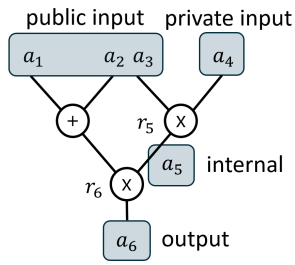

Fig. 2. Arithmetic circuit

2.1.2 시그마 프로토콜

시그마 프로토콜[12]은 증명자가 커밋 메시지를 보내면 검증자가 랜덤한 챌린지를 보내고 다시 증명자가 이에 대응되는 응답 메시지를 보냄으로써 영지식 증명을 수행한다. 이는 Fiat-Shamir 변환을 통해 비대화형으로 변환할 수 있으며[13], 변환된 기법은 랜덤 오라클 모델(random oracle model)에서 안전성이 증명된다. 또한 zk-SNARK가 산술회로 형태의 관계에 효율적인 데 반해 시그마 프로토콜은 다음의 대수적인 형태의 관계에 효율적이다[6].

$$\left\{\begin{matrix}(x_1,\cdots,x_m),(w_1,\cdots,w_n): \\ x_1=\sum_{j=1}^{n}w_jG_{1j},\ \cdots\ ,x_m=\sum_{j=1}^{n}w_jG_{mj}\end{matrix}\right\}$$

2.2 커밋먼트

커밋먼트란 증명자가 자신이 선택한 값을 커밋한 값으로서, 증명자는 커밋 이후에 선택한 값을 변경할 수 없어야 하며(binding), 증명자 이외의 사람은 커밋먼트를 보고 선택한 값을 알 수 없어야 한다(hiding). 또한 증명자는 커밋먼트 생성 시 입력된 값을 이용해서 추후 자신이 선택한 값을 필요에 따라 노출(reveal)할 수 있다.

커밋먼트 중 대표적인 기법은 페더슨 커밋먼트(Pedersen commitment)[14]이다. 증명자가 선택한 값을 x라 할 때, 난수 r을 이용하여 $\mathrm{Com}(x,r)=xG+rH$ 형태로 커밋먼트를 생성한다. 또 다른 기법은 해쉬 기반 커밋먼트이다. 이는 $\mathrm{H}(x\|r)$ 형태로 커밋먼트를 생성할 수 있다.

2.3 NILP

NILP(Non-Interactive Linear Proofs)는 증명자가 CRS의 선형결합만으로 증명을 생성하는 기법이다[8,13]. 또한 NILP에 기반하여 설계된 기법은 제네릭 그룹 모델 하에서 페어링 기반의 zk-SNARK로 변환될 수 있다[8]. 이때 비대칭 겹선형 그룹(asymmetrical bilinear group)을 이용하는 zk-SNARK로 변환되기 위해서는 CRS를 σ_1,σ_2 2개로 나눈 분할(split) NILP를 이용해야 한다. 이 σ_1,σ_2는 각각 zk-SNARK가 이용하는 $\mathbb{G}_1,\mathbb{G}_2$의 CRS로 변환된다. 분할 NILP의 알고리즘은 다음과 같다[8].

- $\mathrm{KeyGen}(R)\rightarrow\sigma$: KeyGen은 CRS $\sigma=(\sigma_1,\sigma_2)\in\mathbb{F}^{m_1}\times\mathbb{F}^{m_2}$를 출력한다.
- $\mathrm{Prove}(R,\sigma,x,w)\rightarrow\pi$: Prove는 ProofMat (R,x,w)를 통해 증명행렬 $(\Pi_1,\Pi_2)\in\mathbb{F}^{k_1\times m_1}\times\mathbb{F}^{k_2\times m_2}$를 얻은 후 증명 $\pi=(\Pi_1\sigma_1,\Pi_2\sigma_2)\in\mathbb{F}^{k_1}\times\mathbb{F}^{k_2}$를 출력한다.
- $\mathrm{Verify}(R,\sigma,x,\pi)$: Verify는 $\mathrm{Test}(R,x)$를 통해 검증행렬 $T_1,\cdots T_\eta\in\mathbb{F}^{(m_1+k_1)\times(m_2+k_2)}$를 얻은 후 $\begin{pmatrix}\sigma_1\\ \pi_1\end{pmatrix}\cdot T_i\begin{pmatrix}\sigma_2\\ \pi_2\end{pmatrix}=0\,(i\in[\eta])$ 여부를 확인하여 결과를 출력한다. 검증행렬을 이용한 위 식은 CRS 및 증명의 각 요소를 이용한 이차식에 해당되며, 따라서 η개의 T_i는 η개의 검증을 위한 이차식에 대응된다.

분할 NILP는 완벽한 완전성 및 완벽한 영지식성과 아핀 증명자 전략에 대한 통계적인 지식 건전성(statistical knowledge soundness against affine prover strategies)을 가져야 한다. 이 건전성은 모든 공격자 A에 대해 다음을 만족하는 다항식 시간 익스트랙터 E가 존재한다는 뜻이며, 이는 성공적인 증명행렬 Π으로부터 위트니스를 추출할 수 있다는 의미이다[8].

$$\Pr\left[\begin{matrix}\mathrm{Verify}(R,\sigma,x,\Pi\sigma)=1\\ (x,w)\notin R\end{matrix}\;\middle|\;\begin{matrix}A(R,z)\rightarrow(x,\Pi)\\ E(R,x,\Pi)\rightarrow w\end{matrix}\right]\le negl(\lambda)$$

분할 NILP를 그룹 원소의 스칼라(scalar) 곱셈 상에서 수행하도록 하면, 비대칭 겹선형 그룹을 이용하는 페어링 기반의 zk-SNARK로 변환할 수 있다[8]. 분할 NILP에서 증명 시 이용하는 증명행렬 연산과 검증 시 이용하는 검증행렬 연산이 zk-SNARK의 그룹 연산과 페어링 연산에 대응되고, 제네릭 그룹 모델 하에서는 공격자가 그룹 연산 및 페어링 연산 결과 외에는 다른 정보를 얻지 못하기 때문이다. 또한 분할 NILP의 공격자는 σ를 확인하지 못하고 증명행렬 Π만을 출력하는데 이는 zk-SNARK의 공격자가 σ로부터 아무런 유용한 정보를 얻지 못하는 경우와 같다[8].

2.4 신용평가방법

신용정보란 고객이 대출을 정상적으로 상환할지 판단하기 위해 이용되는 정보이며 대표적인 예로 연체이력, 현재 대출상황, 소득 등이 있다. 신용정보회사는 기존에 대출을 정상적으로 상환한 고객과 그렇지 않은 고객에 대한 신용정보를 수집한 후, 새로운 고객이 대출을 정상적으로 상환할지를 그 고객의 신용정보로 판단한다. 이런 과정이 신용평가이며, 신용평가 결과로 고객이 대출을 정상적으로 상환할 가능성을 점수로 나타낸 것이 신용점수이다[1,2].

신용평가는 기계학습(machine learning)의 분류(classification)로 볼 수 있다. 실제로 신용점수 계산방법으로 선형 회귀(linear regression), 로지스틱 회귀(logistic regression), 결정 트리(decision tree), 신경망(neural nets) 등 다양한 기계학습 방법 등이 제시되고 있다[1].

이 중 가장 널리 쓰이는 방법은 로지스틱 회귀이다[15]. 로지스틱 회귀는 분류 결과가 0과 1의 이진 형태로 나타나는 문제에 쓰이는 방법으로서 성공 확률 p에 대해서 오즈(odds) $\ln(\frac{p}{1-p})$가 다음과 같이 독립변수 x_i에 대해 선형 상관관계를 나타낸다는 가정을 하고 있다.

$$\ln(\frac{p}{1-p}) = a + b_1x_1 + b_2x_2 \cdots + b_nx_n$$

실제 신용정보 원본값이 로지스틱 회귀에 그대로 이용되기는 어렵다. 신용정보 중 누락된 값 또는 이상치(outlier)가 결과를 왜곡할 가능성이 있기 때문이다. 따라서 각 신용정보를 일정한 계급으로 나눈 후 그 계급의 대푯값들을 독립변수 x_i로 이용하는 방법이 주로 쓰인다. 대푯값으로 주로 이용되는 것은 WOE(Weight Of Evidence)이다. WOE는 성공한 사례 중 해당 계급이 차지하는 비율과 실패한 사례 중 해당 계급이 차지하는 비율을 확인하여 다음과 같이 계산한다[16].

$$WOE = \ln(\frac{distribution\ of\ good}{distribution\ of\ bad})$$

최종적으로 신용점수는 이용하기 쉽게 정수로 표현되며 다음과 같이 오즈를 스케일해서 계산한다[15].

$$score = \ln(\frac{p}{1-p}) \times factor + offset$$

III. 커밋된 입력에 대한 비대화형 영지식 증명

커밋된 입력에 대한 비대화형 영지식 증명이란 일반적인 비대화형 영지식 증명의 기능에 더해, 커밋먼트 생성에 이용된 값이 영지식 증명 과정에 실제로 이용되었는지 함께 증명하는 기법이다. 이 기법은 일반적인 영지식 증명에 이용되는 공개 입력 x, 공개 출력 y, 위트니스 w, 함수 f뿐 아니라 커밋된 입력 a_i, 페더슨 커밋먼트의 난수 r_i, 페더슨 커밋먼트 C_i를 함께 이용하여 다음의 관계를 증명한다.

$$\left\{ \begin{pmatrix} \left(\{C_i\}_{i\in[n]}, x, y\right), \\ \left(\{a_i, r_i\}_{i\in[n]}, w\right): \\ C_i = a_iG + r_iH (i\in[n]), \\ f(x, w, a_1, \cdots, a_n) = y \end{pmatrix} \right\}$$

또한 이 기법은 4.1의 프라이버시 보장 신용평가방법에서 영지식 증명 과정에 이용된 신용정보가 금융기관이 실제로 제공한 값인지 블록체인의 커밋먼트를 통해 확인할 때 활용할 수 있다.

3.1 Agrawal 등의 기법 및 분석

3.1.1 기 법

Agrawal 등의 기법[6]에서는 Parno 등의 기법[7]을 수정하여 커밋된 입출력의 영지식 증명을 함께 수행한다. Agrawal 등의 기법[6]에서는 위트니스만으로 이루어진 Parno 등의 기법[7]의 증명에 커밋된 입력과 커밋된 출력에 대한 증명 요소를 각각 추가한 후, 해당 부분이 외부의 페더슨 커밋먼트와 같은 값을 이용한다는 사실을 비대화형으로 변환된 시그마 프로토콜로 증명한다. 이는 증명에 이용된 커밋된 입출력과 페더슨 커밋먼트에 이용된 값이 모두 그룹 원소에 곱해진 스칼라 값 형태이기 때문에 가능하다. Agrawal 등[6]은 커밋된 입출력에 대한 증명을 모두 수행하는 기법과 커밋된 입력에 대한 증명만 수행하는 기법을 제안하였다. 이 중 본 논문에서는 커밋된 입력에 대한 증명만을 활용한다.

이 기법은 비대칭 겹선형 그룹 $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, \widetilde{G})$을 이용한다. 소수 p를 위수(order)로 하는 순환 그룹(cyclic group) $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$에 대해, $G, \widetilde{G}$는 $\mathbb{G}_1, \mathbb{G}_2$의 생성자(generator)이고 $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$는 페어링 연산이다. 또한 공개 입출력, 위트니스, 커밋된 입력의 와이어 번호의 집합을 각각 I_{pub}, I_{mid}, I_{com}로 표기하며, $I_{all} = \{0\} \cup I_{pub} \cup I_{mid} \cup I_{com}$이고, QAP를 위해 정의 1.의 표기법을 이용한다고 하자.

KeyGen 알고리즘은 관계를 나타내는 산술회로인 $u_i(x), v_i(x), w_i(x), h(x), t(x) (i \in I_{all})$를 입력받고 난수 $r_u, r_v, \alpha_u, \alpha_v, \alpha_w, \beta, \gamma, s \in \mathbb{F}^*$ 및 $r_w = r_u r_v$를 생성하여 σ를 계산 후 출력한다. Prove 알고리즘은 σ와 $\{a_i\}_{i\in I_{all}}$를 입력받고 π_{snark}의 영지식성을 위한 난수 $\delta_u, \delta_v, \delta_w \in \mathbb{F}$를 생성한 후 증명을 계산하여 출력하는데, 이는 zk-SNARK의 증명 π_{snark}와 시그마 프로토콜의 증명 π_{in}로 구성된다. Verify 알고리즘은 $\sigma, \{a_i\}_{i\in I_{pub}}, \pi_{snark}, \pi_{in}$를 입력받고 각각 페어링 연산과 시그마 프로토콜을 이용하여 π_{snark}, π_{in}을 검증한 후 그 결과를 출력한다.

π_{snark}는 다음과 같으며 $\{a_i\}_{i\in I_{com}}, \{a_i\}_{i\in I_{mid}}, h(x)$의 x^i항 계수인 h_i, 난수 $\delta_u, \delta_v, \delta_w$ 그리고 σ를 이용하여 계산된다.

$$\pi_{snark} = (U_{com}, U_{mid}, \widetilde{V_{com}}, \widetilde{V_{mid}}, W_{com}, W_{mid}, \widetilde{H}, \widetilde{U_{com}}', \widetilde{U_{mid}}', V_{com}', V_{mid}', \widetilde{W_{com}}', \widetilde{W_{mid}}', K_{com}, K_{mid})$$

π_{snark}의 요소 중 커밋된 입력에 대한 증명 요소와 $\widetilde{H}$ 는 다음을 의미하며, 나머지 위트니스에 대한 증명 요소도 $i \in I_{mid}$를 이용하여 동일하게 생성한다.

$$U_{com} = \sum_{i \in I_{com}} a_i u_i(s) r_u G + \delta_u t(s) r_u G$$
$$\widetilde{V_{com}} = \sum_{i \in I_{com}} a_i v_i(s) r_v \widetilde{G} + \delta_v t(s) r_v \widetilde{G}$$
$$W_{com} = \sum_{i \in I_{com}} a_i w_i(s) r_w G + \delta_w t(s) r_w G$$
$$\widetilde{U_{com}}' = \sum_{i \in I_{com}} a_i \alpha_u u_i(s) r_u \widetilde{G} + \delta_u t(s) r_u \widetilde{G}$$
$$V_{com}' = \sum_{i \in I_{com}} a_i \alpha_v v_i(s) r_v G + \delta_v t(s) r_v G$$
$$\widetilde{W_{com}}' = \sum_{i \in I_{com}} a_i \alpha_w w_i(s) r_w \widetilde{G} + \delta_w t(s) r_w \widetilde{G}$$
$$K_{com} = \sum_{i \in I_{com}} a_i (\beta u_i(s) r_u + \beta v_i(s) r_v + \beta w_i(s) r_w) G + \delta_u \beta t(s) r_u G + \delta_v \beta t(s) r_v G + \delta_w \beta t(s) r_w G$$
$$\widetilde{H} = \sum_{i=0}^{n-1} h_i s^i \widetilde{G}$$

π_{in}은 다음 관계에 대한 시그마 프로토콜 증명이다. 여기서 $\{C_i\}_{i \in I_{com}}$는 $\{a_i\}_{i \in I_{com}}$의 페더슨 커밋먼트이며, $\{s_i\}_{i \in I_{com}}$는 $\{C_i\}_{i \in I_{com}}$를 위한 난수이다. 또한 $G_i = u_i(s) r_u G (i \in I_{com})$, $G_0 = t(s) r_u G$는 σ의 일부이며, H_1, H_2는 $\mathbb{G}_1$의 생성자이다.

$$\left\{ \begin{pmatrix} U_{com}, \{C_i\}_{i \in I_{com}} \end{pmatrix}, \begin{pmatrix} \{a_i\}_{i \in I_{com}}, \delta_u, \{s_i\}_{i \in I_{com}} \end{pmatrix} : \begin{matrix} U_{com} = \sum_{i \in I_{com}} a_i G_i + \delta_u G_0, \\ C_i = a_i H_1 + s_i H_2 (i \in I_{com}) \end{matrix} \right\}$$

π_{in}을 생성하고 검증하는 구체적인 방법은 [6]의 Fig. 5.에 설명된 comEq을 참조한다.

3.1.2 분 석

완벽한 영지식성을 가지기 위해서는 2.1.1에서 언급한 바와 같이 Prove가 스테이트먼트와 위트니스를 모두 안 상태에서 만든 증명과 SimProve가 스테이트먼트만 안 상태에서 만든 증명의 분포가 같아야 한다[11].

- Prove 알고리즘 : Prove가 출력한 증명 $\pi_{snark} = (U_{com}, U_{mid}, \cdots)$에서 처음 U_{com}, U_{mid}는 같은 난수 δ_u를 이용하여 $\delta_u t(s) G$가 더해져 있는 형태이다. 이는 스테이트먼트, 위트니스, 커밋된 입력이 고정되어 있을 때, 서로 다른 2개의 상수에 같은 난수가 더해진 것으로 볼 수 있다. 이 경우에 U_{com}가 결정되면 순차적으로 δ_u와 U_{mid}가 자동 결정되므로, U_{com}와 U_{mid}의 분포는 독립(independent)이지 않으며 또한 균일(uniform)하지 않다.
- SimProve 알고리즘 : 먼저 다음의 표기를 정의하자.
  $u_{pub}(x) = \sum_{i \in I_{pub}} a_i u_i(x)$
  $u_{com}(x) = \sum_{i \in I_{com}} a_i u_i(x) + \delta_u t(x)$
  $u_{mid}(x) = \sum_{i \in I_{mid}} a_i u_i(x) + \delta_u t(x)$
  $u(x) = u_0(x) + u_{pub}(x) + u_{com}(x) + u_{mid}(x)$
  SimProve는 검증식을 통과시키기 위해 $u(x)v(x)$가 $t(x)$로 나누어떨어지도록 $u(x)$를 랜덤하게 선택한다. 또한 $u_{com}(x)$를 랜덤하게 선택한 후, 공개된 $u_0(x), u_{pub}(x)$를 이용하여 $u_{mid}(x) = u(x) - u_0(x) - u_{pub}(x) - u_{com}(x)$를 계산한다. 마지막으로 SimProve는 자신이 알고 있는 트랩도어인 s를 $u_{com}(x)$와 $u_{mid}(x)$에 대입한 후, 그 결과로 $r_u G$에 스칼라 곱셈을 하여 U_{com}와 U_{mid}를 위트니스 없이 계산한다.
  $u_{com}(x)$는 랜덤하게 선택되었으므로, 여기에 s가 대입된 후 $r_u G$가 곱해진 결과인 U_{com}은 역시 랜덤하다. 또한 $u_{com}(x)$와 독립적으로 랜덤하게 선택된 $u(x)$에서 $u_0(x) + u_{pub}(x) + u_{com}(x)$을 뺀 $u_{mid}(x)$는 $u_{com}(x)$와 독립적으로 랜덤하며, 여기에 s가 대입된 후 $r_u G$가 곱해진 결과인 U_{mid} 역시 U_{com}과 독립적으로 랜덤하다. 결국 U_{com}와 U_{mid}의 분포는 서로 다른 랜덤에 기반하므로 독립이고 균일하다.

위와 같이 Agrawal 등의 기법[6]은 Prove와 SimProve의 출력의 분포가 다르므로 완벽한 영지식성을 만족하지 않는다. 이는 증명 중 커밋된 입력과 위트니스에 대한 요소에 같은 난수가 이용되기 때문이다. 따라서 완벽한 영지식성을 가지기 위해서는 커밋된 입력에 대한 요소인 $U_{com}, \widetilde{V_{com}}, W_{com}, \widetilde{U_{com}}', V_{com}', \widetilde{W_{com}}', K_{com}$에는 난수 $\delta_{u,com}, \delta_{v,com}, \delta_{w,com}$을 이용하고 위트니스를 위한 나머지 요소에는 다른 난수 $\delta_{u,mid}, \delta_{v,mid}, \delta_{w,mid}$를 이용하여야 한다.

3.2 제안하는 기법

본 절에서는 Groth의 zk-SNARK 기법[8]과 비대화형으로 변환된 시그마 프로토콜을 결합하여, 커밋된 입력에 대한 비대화형 영지식 증명 기법을 제안한다. 이 기법은 Agrawal 등의 기법[6]과 달리 완벽한 영지식성을 제공하며, CRS와 증명의 크기가 작고 증명과 검증 과정이 빠르다. 기법의 설계 및 안전성 증명은 Groth의 기법[8]과 동일하게 제네릭 그룹 모델에서 수행하며, Groth의 기법[8]과 동일한 안전성 증명 흐름을 따른다. 이를 위해 3.2.1에서는 커밋된 입력에 대한 분할 NILP 기법을 제안하고 안전성을 증명한다. 3.2.2에서는 zk-SNARK로 변환된 3.2.1의 분할 NILP 기법과 시그마 프로토콜을 결합하여, 커밋된 입력에 대한 비대화형 영지식 증명 기법을 제안하고 안전성을 증명한다.

3.2.1 분할 NILP 기법

먼저 커밋된 입력에 대한 분할 NILP 기법을 정의하고자 한다. 이를 위해 A,B,C 3개의 요소로 구성된 Groth의 분할 NILP 기법[8]의 증명에 커밋된 입력에 대한 요소 D를 추가하고, 완전성, 건전성, 영지식성이 보장되도록 CRS 및 증명 요소를 수정하였다. 이는 공개 입출력 $\{a_i\}_{i\in I_{pub}}$, 위트니스 $\{a_i\}_{i\in I_{mid}}$, 커밋된 입력 $\{a_i\}_{i\in I_{com}}$, 함수 f에 대해 다음의 관계를 증명한다.

$$\left\{\left(\{a_i\}_{i\in I_{pub}}\right),\left(\{a_i\}_{i\in I_{mid}},\{a_i\}_{i\in I_{com}}\right): f(\{a_i\}_{i\in I_{pub}},\{a_i\}_{i\in I_{mid}},\{a_i\}_{i\in I_{com}})=\text{true}\right\}$$

제안하는 분할 NILP 기법의 각 알고리즘은 다음과 같다. 여기서 $\beta u_i(s)+\alpha v_i(s)+w_i(s)$는 K_i로 축약하였으며 m은 와이어의 수, n은 곱셈 게이트의 수이다.

- KeyGen$(R)\rightarrow\sigma$: 난수 $\alpha,\beta,\gamma,\delta,\epsilon,s\in\mathbb{F}^*$를 생성한 후 이를 트랩도어 $\tau=(\alpha,\beta,\gamma,\delta,\epsilon,s)$로 둔다. 그리고 다음과 같이 $\sigma=(\sigma_1,\sigma_2)$를 출력한다.

$$\sigma_1=\left(\alpha,\beta,\delta,\epsilon,\{s^i\}_{i=0}^{n-1},\left\{\frac{K_i}{\gamma}\right\}_{i\in\{0\}\cup I_{pub}},\left\{\frac{K_i}{\delta}\right\}_{i\in I_{mid}},\left\{\frac{K_i}{\epsilon}\right\}_{i\in I_{com}},\left\{\frac{s^it(s)}{\delta}\right\}_{i=0}^{n-2}\right)$$

$$\sigma_2=\left(\beta,\gamma,\delta,\epsilon,\{s^i\}_{i=0}^{n-1}\right)$$

- Prove$(R,\sigma,\{a_i\}_{i\in I_{all}})\rightarrow\pi$: 난수 $r_A,r_B,r_D\in\mathbb{F}^*$를 생성한 후 $\Pi_1\sigma_1=(A,C,D),\Pi_2\sigma_2=(B)$를 만족하는 $\Pi_1\in\mathbb{F}^{3\times(m+2n+4)}$, $\Pi_2\in\mathbb{F}^{1\times(n+4)}$를 계산하고 $\pi=(\Pi_1\sigma_1,\Pi_2\sigma_2)$를 출력한다.

$$A=\alpha+\sum_{i\in I_{all}}a_iu_i(s)+r_A\delta$$
$$B=\beta+\sum_{i\in I_{all}}a_iv_i(s)+r_B\delta$$
$$C=\frac{\sum_{i\in I_{mid}}a_iK_i+h(s)t(s)}{\delta}+Ar_B+Br_A-r_Ar_B\delta-r_D\epsilon$$
$$D=\frac{\sum_{i\in I_{com}}a_iK_i}{\epsilon}+r_D\delta$$

- Verify$(R,\sigma,\{a_i\}_{i\in I_{pub}},\pi)\rightarrow 0/1$: $\begin{pmatrix}\sigma_1\\ \pi_1\end{pmatrix}\cdot T\begin{pmatrix}\sigma_2\\ \pi_2\end{pmatrix}=0$가 다음의 검증식에 해당되도록 검증행렬 T를 생성하고, 검증이 통과되는지 확인한다.

$$AB=\alpha\beta+\frac{\sum_{i\in\{0\}\cup I_{pub}}a_iK_i}{\gamma}\gamma+C\delta+D\epsilon$$

- SimProve$(R,\tau,\{a_i\}_{i\in I_{pub}})\rightarrow\pi$: 난수 $A,B,D\in\mathbb{F}$를 생성한 후 다음의 C를 계산 후 $\pi=((A,C,D),(B))$를 출력한다.

$$C=\frac{AB-\alpha\beta-\sum_{i\in\{0\}\cup I_{pub}}a_iK_i-D\epsilon}{\delta}$$

정리 1. 위 기법은 완벽한 완전성, 완벽한 영지식성, 아핀 증명자 전략에 대한 통계적인 지식 건전성을 가진다.

증명. Prove가 출력한 증명을 Verify에 입력하면 항상 성공하므로 완벽한 완전성을 가진다. Prove가 출력한 실제 증명과 SimProve가 출력한 시뮬레이트된 증명 모두에 대해 A,B,D는 독립적이고 균일하게 랜덤하며 C는 검증을 통과하도록 생성된 값이므로 두 증명은 같은 분포를 보인다. 따라서 완벽한 영지식성을 가진다.

아핀 증명자 전략에 대해 통계적인 지식 건전성을 가지는지 살펴보자. 아핀 증명자 전략을 이용할 때, π_1의 첫 번째 요소인 A는 σ_1를 선형결합하여 다음과 같이 나타낼 수 있다.

$$A_\alpha\alpha+A_\beta\beta+A_\delta\delta+A_\epsilon\epsilon+A(s)+\sum_{i\in\{0\}\cup I_{pub}}A_i\frac{K_i}{\gamma}+\sum_{i\in I_{mid}}A_i\frac{K_i}{\delta}+\sum_{i\in I_{com}}A_i\frac{K_i}{\epsilon}+A_h(s)\frac{t(s)}{\delta}$$

여기서 $A_\alpha, A_\beta, A_\delta, A_\epsilon, A_i$와 $A(s), A_h(s)$의 각 계수는 Π_1의 첫 번째 행에 해당한다. π_1의 두 번째, 세 번째 요소인 C, D도 유사하다. π_2에 해당하는 B는 σ_2를 선형결합하여 $B_\beta\beta + B_\gamma\gamma + B_\delta\delta + B_\epsilon\epsilon + B(s)$와 같이 나타낼 수 있다.

Schwartz-Zippel 보조정리(lemma)[11]에 따르면 Verify의 검증식이 랜덤한 값인 $\alpha, \beta, \gamma, \delta, \epsilon, s$을 변수로 한 항등식이 되지 않으면 증명자가 무시할 만한 성공 확률을 갖는다. 항등식이 되기 위한 조건을 찾도록 하자.

$\alpha\beta$의 계수가 1이므로 $A_\alpha B_\beta = 1$이다. 이때 일반성을 잃지 않고 $A_\alpha = B_\beta = 1$로 간주할 수 있다. β^2의 계수가 0이므로 $A_\beta = 0$이다. 이제 A와 B를 다음과 같이 단순화할 수 있다.

$$A = \alpha + A_\delta\delta + A_\epsilon\epsilon + A(s) + \cdots$$
$$B = \beta + B_\gamma\gamma + B_\delta\delta + B_\epsilon\epsilon + B(s)$$

여기서 $\frac{\beta}{\gamma}, \frac{\beta}{\delta}, \frac{\beta}{\epsilon}$ 항을 이용하면 각각 $\sum_{i\in\{0\}\cup I_{pub}} A_i K_i = 0$, $\sum_{i\in I_{mid}} A_i K_i + A_h(s)t(s) = 0$, $\sum_{i\in I_{com}} A_i K_i = 0$을 얻을 수 있다. 따라서

$$A = \alpha + A_\delta\delta + A_\epsilon\epsilon + A(s)$$
$$B = \beta + B_\gamma\gamma + B_\delta\delta + B_\epsilon\epsilon + B(s)$$

이다. Verify의 검증식에서 α와 $\{s^i\}_{i=0}^{n-1}$과 관련된 항만을 추려낸 식과 β와 $\{s^i\}_{i=0}^{n-1}$과 관련된 항만을 추려낸 식은 각각 다음과 같다.

$$\alpha B(s) = \sum_{i\in\{0\}\cup I_{pub}} a_i\alpha v_i(s) + \sum_{i\in I_{mid}} C_i\alpha v_i(s) + \sum_{i\in I_{com}} D_i\alpha v_i(s)$$
$$\beta A(s) = \sum_{i\in\{0\}\cup I_{pub}} a_i\beta u_i(s) + \sum_{i\in I_{mid}} C_i\beta u_i(s) + \sum_{i\in I_{com}} D_i\beta u_i(s)$$

$i\in I_{mid}$에 대해 $C_i = a_i$, $i\in I_{com}$에 대해 $D_i = a_i$로 정의하면 다음의 식을 얻는다.

$$A(s) = \sum_{i\in I_{all}} a_i u_i(s),\ B(s) = \sum_{i\in I_{all}} a_i v_i(s)$$

최종적으로 $\{s^i\}_{i=0}^{n-1}$와 관련된 항만을 추려내면 다음과 같다.

$$\sum_{i\in I_{all}} a_i u_i(s) \cdot \sum_{i\in I_{all}} a_i v_i(s) = \sum_{i\in I_{all}} a_i w_i(s) + C_h(s)t(s)$$

이를 통해 $C_i = a_i (i\in I_{mid})$는 위트니스이며 $D_i = a_i (i\in I_{com})$는 커밋된 입력임을 알 수 있다. Π_1, Π_2에서 위트니스와 커밋된 입력을 추출할 수 있으므로 아핀 증명자 전략에 대해 통계적인 지식 건전성을 가진다.□

3.2.2 커밋된 입력에 대한 비대화형 영지식 증명

이제 커밋된 입력에 대한 비대화형 영지식 증명을 정의하고자 한다. 이를 위해 3.2.1의 분할 NILP 기법을 zk-SNARK로 변환하고 커밋된 입력과 외부 커밋먼트에 이용된 값이 같다는 것을 증명하기 위해 비대화형으로 변환된 시그마 프로토콜을 이용하였다. 이는 커밋먼트 $\{C_i\}_{i\in I_{com}}$, 공개 입출력 $\{a_i\}_{i\in I_{pub}}$, 위트니스 $\{a_i\}_{i\in I_{mid}}$, 커밋된 입력 $\{a_i\}_{i\in I_{com}}$, 커밋먼트의 난수 $\{s_i\}_{i\in I_{com}}$, 함수 f에 대해 다음의 관계를 증명한다.

$$\left\{\begin{pmatrix}\{C_i\}_{i\in I_{com}}, \{a_i\}_{i\in I_{pub}}\end{pmatrix}, \begin{pmatrix}\{a_i\}_{i\in I_{mid}}, \{a_i, s_i\}_{i\in I_{com}}\end{pmatrix} : \begin{array}{l} C_i = a_i G + s_i H (i\in I_{com}), \\ f(\{a_i\}_{i\in I_{pub}}, \{a_i\}_{i\in I_{mid}}, \{a_i\}_{i\in I_{com}}) = \text{true}\end{array}\right\}$$

각 알고리즘은 다음과 같다. 비대칭 겹선형 그룹$(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, G, \widetilde{G})$을 이용하였으며 소수 p를 위수로 하는 순환 그룹 $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$에 대해, $G, \widetilde{G}$는 $\mathbb{G}_1, \mathbb{G}_2$의 생성자이고 $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$는 페어링 연산이다.

- KeyGen$(R) \to \sigma$: 난수 $\alpha, \beta, \gamma, \delta, \epsilon, s \in \mathbb{F}^*$를 생성한 후 이를 트랩도어 $\tau = (\alpha, \beta, \gamma, \delta, \epsilon, s)$로 둔다. 그리고 다음과 같이 $\sigma = (\sigma_1, \sigma_2)$를 출력한다. σ_1는 $\mathbb{G}_1$, σ_2는 $\mathbb{G}_2$의 원소이다.

$$\sigma_1 = \begin{pmatrix} \alpha G, \beta G, \delta G, \epsilon G, \{s^i G\}_{i=0}^{n-1}, \left\{\frac{K_i}{\gamma} G\right\}_{i\in\{0\}\cup I_{pub}}, \\ \left\{\frac{K_i}{\delta} G\right\}_{i\in I_{mid}}, \left\{\frac{K_i}{\epsilon} G\right\}_{i\in I_{com}}, \left\{\frac{s^i t(s)}{\delta} G\right\}_{i=0}^{n-2} \end{pmatrix}$$
$$\sigma_2 = \left(\beta\widetilde{G}, \gamma\widetilde{G}, \delta\widetilde{G}, \epsilon\widetilde{G}, \{s^i\widetilde{G}\}_{i=0}^{n-1}\right)$$

- Prove$(R, \sigma, \{a_i\}_{i\in I_{all}}) \to \pi$: 난수 $r_A, r_B, r_D \in \mathbb{F}$를 생성한 후 $\pi_{snark} = (A, B, C, D)$를 계산한다.

$$A = (\alpha + \sum_{i\in I_{all}} a_i u_i(s) + r_A\delta) G$$
$$B = (\beta + \sum_{i\in I_{all}} a_i v_i(s) + r_B\delta) \widetilde{G}$$
$$C = \begin{pmatrix} \frac{\sum_{i\in I_{mid}} a_i K_i + h(s)t(s)}{\delta} \\ + r_B(\alpha + \sum_{i\in I_{all}} a_i u_i(s)) \\ + r_A(\beta + \sum_{i\in I_{all}} a_i v_i(s)) \\ + r_A r_B\delta - r_D\epsilon \end{pmatrix} G$$
$$D = \left(\frac{\sum_{i\in I_{com}} a_i K_i}{\epsilon} + r_D\delta\right) G$$

그리고 다음 관계에 대한 시그마 프로토콜 증명 π_{in}을 계산한다. 여기서 $\{C_i\}_{i\in I_{com}}$는 $\{a_i\}_{i\in I_{com}}$의 페더슨 커밋먼트이고, $\{s_i\}_{i\in I_{com}}$는 $\{C_i\}_{i\in I_{com}}$를 위한 난수이다. 또한 $G_i=\left\{\frac{K_i}{\epsilon}G\right\}_{i\in I_{com}}$ 및 $G_0=\delta G$는 σ의 일부이며, H_1, H_2는 $\mathbb{G}_1$의 생성자이다.

$$\left\{\begin{array}{l}\left(D,\{C_i\}_{i\in I_{com}}\right),\\ \left(\{a_i\}_{i\in I_{com}}, r_D, \{s_i\}_{i\in I_{com}}\right):\\ D=\sum_{i\in I_{com}} a_i G_i + r_D G_0,\\ C_i = a_i H_1 + s_i H_2 (i\in I_{com})\end{array}\right\}$$

π_{in}을 계산하는 구체적인 방법은 [6]의 Fig. 5.에 설명된 comEq를 참조한다. Prove는 최종적으로 $\pi=(\pi_{snark},\pi_{in})$를 출력한다.

- $\mathrm{Verify}(R,\sigma,\{a_i\}_{i\in I_{pub}},\pi)\rightarrow 0/1$: $\pi_{snark}=(A,B,C,D)$에 대해 다음의 검증식이 통과되는지 확인한다.

$$e(A,B)=e(\alpha G,\beta\widetilde{G})\cdot e\left(\frac{\sum_{i\in\{0\}\cup I_{pub}} a_i K_i}{\gamma}G,\gamma\widetilde{G}\right)\cdot e(C,\delta\widetilde{G})\cdot e(D,\epsilon\widetilde{G})$$

그리고 시그마 프로토콜의 검증 과정을 통해 π_{in}이 올바른지 확인한다. π_{in}를 검증하는 구체적인 방법은 [6]의 Fig. 5.에 설명된 comEq를 참조한다.

- $\mathrm{SimProve}(R,\tau,\{a_i\}_{i\in I_{pub}})\rightarrow\pi$: 난수 $r_A, r_B, r_D\in\mathbb{F}$를 생성한 후 $A=r_A G, B=r_B\widetilde{G}, D=r_D G$를 계산한다. 그리고 트랩도어 τ를 이용하여 다음과 같이 C를 계산하고 $\pi_{snark}=(A,B,C,D)$로 둔다.

$$C=\frac{r_A r_B-\alpha\beta-\sum_{i\in\{0\}\cup I_{pub}} a_i K_i - r_D\epsilon}{\delta}G$$

또한 시그마 프로토콜의 영지식성을 이용하여 시뮬레이트된 π_{in}을 계산한 후 최종적으로 $\pi=(\pi_{snark},\pi_{in})$를 출력한다. π_{in}을 시뮬레이트하는 구체적인 방법은 [6]의 C.1에 설명된 comEq의 안전성 증명을 참조한다.

정리 2. 위 기법은 제네릭 그룹 모델과 랜덤 오라클 모델을 모두 가정할 때, 완벽한 완전성, 완벽한 영지식성, 계산적인 지식 건전성을 갖는다.

증명. 제네릭 그룹 모델의 공격자는 가정에 따라 CRS로부터 어떠한 유용한 정보도 얻을 수 없다. 따라서 분할 NILP 기법의 공격자 역시 CRS로부터 어떠한 유용한 정보도 얻을 수 없어야, 분할 NILP 기법이 zk-SNARK로 변환될 수 있다. 이는 분할 NILP 기법의 공격자 입장에서 CRS의 값과 관계없이 검증 결과가 같다는 의미이며, 다시 말하면 동일한 관계 R에 대해 랜덤하게 생성된 CRS인 $(\sigma_1,\sigma_2),(\sigma_1',\sigma_2')$와 모든 공격자가 출력한 행렬 T에 대해, $\Pr[\sigma_1\cdot T\sigma_2=0\Leftrightarrow\sigma_1'\cdot T\sigma_2'=0]\approx 1$이 성립한다는 뜻이다[8]. Schwartz-Zippel 보조정리[11]에 따라, 3.2.1의 $\sigma_1\cdot T\sigma_2=0$이 랜덤한 값인 $\alpha,\beta,\gamma,\delta,\epsilon,s$를 변수로 한 항등식이 되지 않으면 $\sigma_1\cdot T\sigma_2=0$인 경우는 무시할 만하다. 따라서 $\sigma_1\cdot T\sigma_2=0$인 경우, 이는 항등식이 되어야 하며, 이때 $\sigma_1'\cdot T\sigma_2'=0$ 역시 성립한다. 따라서 3.2.1의 (σ_1,σ_2)는 어떠한 유용한 정보도 노출하지 않는다. 3.2.1의 기법은 완벽한 완전성, 완벽한 영지식성, 아핀 증명자 전략에 대한 통계적인 지식 건전성을 가지고 (σ_1,σ_2)에서 어떠한 유용한 정보도 노출하지 않으므로, 3.2.1의 기법에 기반하여 설계된 π_{snark}는 [8]의 보조정리 1.에 따라 제네릭 그룹 모델에서 완벽한 완전성 및 완벽한 영지식성과 다항식 횟수의 그룹 연산을 수행하는 공격자에 대해 통계적인 지식 건전성을 갖는다. π_{in}은 랜덤 오라클 모델에서 [6]의 C.1에 따라 완벽한 완전성 및 완벽한 영지식성과 계산적인 지식 건전성을 갖는다. 따라서 위 기법은 랜덤 오라클 모델과 제네릭 그룹 모델을 모두 가정할 때, 완벽한 완전성, 완벽한 영지식성, 계산적인 지식 건전성을 갖는다. □

3.3 성능 분석

본 절에서는 실험을 통해 3.2.2에서 제안한 기법의 성능을 다른 기법과 비교하고자 한다. 이를 위해 5.1의 구현을 활용하였으며 5.2에서 실험을 위해 가정한 신용정보의 종류와 실험 환경을 이용하였다.

커밋된 입력에 대한 증명을 하는 방법으로는 3.2.2의 기법, Agrawal 등의 기법[6], Campanelli 등의 기법[10]과 같이 영지식 기법 내에서 커밋된 입력에 대한 증명을 함께 수행하는 방법이 있다. 이와 같은 방법의 성능 측정을 위해 커밋먼트를 다음과 같이 생성하였으며, 5.1의 구현대로 BN(Barreto-Naehrig) 타원곡선[17]을 이용하면 커밋먼트의 크기는 32바이트이다.

Table 1. Performance of commitment link scheme

	CRS size	Proof size	Prover comp.	Verifier comp.
[12]	-	$m+1\mathbb{G}_1$, $m+l+1\mathbb{F}$	$m+2l$ $+1E$	$2m+2l$ $+2E$
[18]	$m+l+1\mathbb{G}_1$, $m+2\mathbb{G}_2$	$1\mathbb{G}_1$	$m+l$ $+1E$	mE, $m+2P$

m : commitment, l : input, $\mathbb{G}_{1,2}$: group element, $\mathbb{F}$: field element, E : exponentiation, P : pairing

$$\mathrm{Com}(x_1,\cdots,x_n,r) = rG_0 + \sum_{i=1}^{n} x_i G_i$$

3.2.2의 기법과 Agrawal 등의 기법[6]은 시그마 프로토콜[12]을, Campanelli 등의 기법[10]은 Kiltz 등의 기법[18]을 이용하여 커밋먼트 생성 시 입력된 값이 같다는 증명을 수행한다. 이 증명을 위해 추가적으로 소요되는 CRS 및 증명의 크기와 증명 및 검증시간은 Table 1.과 같다. 여기서 m과 l은 4.1의 프라이버시 보장 신용평가방법에서 각각 금융기관과 신용정보의 수에 대응된다.

커밋된 입력에 대한 증명을 하는 다른 방법으로는 산술회로 내에 커밋먼트 기법을 직접 구현하는 방법이 있다. 이와 같은 방법의 성능 측정을 위해 Parno 등의 기법[7]과 Groth의 기법[8]에서 이용하는 산술회로 내에 Ajtai 해쉬 알고리즘[19]으로 커밋먼트 기법을 구현하였다. 이 해쉬 알고리즘은 입력 $x \in \{0,1\}^m$, 랜덤행렬 $A \in \mathbb{Z}_q^{n\times m}$에 대해 $\mathrm{H}(x) = Ax \pmod q$와 같이 정의되는데, Kosba 등[20]은 $q \approx 2^{254}$에 대해 $n=1,2,3,4$ 별로 최적의 m을 찾고 그에 대한 충돌 저항성(collision resistance) 보안수준(security level)을 분석하였다. 최근의 가이드라인[21]에서 요구하는 최소 보안수준은 112비트이므로, 본 실험에서는 위 분석 결과 중에서 이를 충족하는 162비트 보안수준 파라미터 $n=4, m=2032$를 이용하였다. 또한 커밋먼트는 다음과 같이 생성하였으며 커밋먼트의 크기는 32바이트$\times n$=128바이트이다.

$$\mathrm{Com}(x_1,\cdots,x_n,r) = \mathrm{H}(x_1\|x_2\|\cdots\|x_n\|r)$$

일반적으로 널리 쓰이는 SHA-256의 경우, 많은 수의 비트 연산을 수행하므로 산술회로로 표현하면 비효율적이다. 반면 Ajtai 해쉬 알고리즘[19]에서는 A의 각 행과 x의 곱을 산술회로의 곱셈 게이트 하나로 표현할 수 있으므로 zk-SNARK를 이용할 때 효율적이다[20]. 따라서 본 실험에서는 Ajtai 해쉬 알고리즘[19]을 이용하여 산술회로 내에 구현된 커밋먼트 기법을 활용하였다.

Table 2. Performance of NIZK on committed input

	CRS size	Proof size	Com. size	Proving time	Verifying time
3.2.2	170kiB	56.6kiB	3.13kiB	0.34sec	0.30sec
[6]	234kiB	57.0kiB	3.13kiB	0.35sec	0.30sec
[10,8]	232kiB	197B	3.13kiB	0.06sec	0.04sec
[7,19]	21.0MiB	296B	12.5kiB	0.22sec	0.003sec
[8,19]	13.0MiB	131B	12.5kiB	0.19sec	<0.001sec

Number of commitments : 100

	CRS size	Proof size	Com. size	Proving time	Verifying time
3.2.2	339kiB	113kiB	6.25kiB	0.66sec	0.59sec
[6]	468kiB	113kiB	6.25kiB	0.69sec	0.60sec
[10,8]	462kiB	197B	6.25kiB	0.10sec	0.09sec
[7,19]	42.0MiB	296B	25.0kiB	0.44sec	0.003sec
[8,19]	26.0MiB	131B	25.0kiB	0.37sec	<0.001sec

Number of commitments : 200

측정된 성능은 Table 2.와 같다. 처음 2개의 행은 3.2.2의 기법, Agrawal 등의 기법[6]의 결과이며, 그다음 행은 Campanelli 등의 기법[10]을 이용하되 내부 구성요소로 Groth의 기법[8]을 이용한 결과이다. 마지막 2개의 행은 Parno 등의 기법[7]과 Groth의 기법[8]을 이용하되, Ajtai 해쉬 알고리즘[19]으로 커밋먼트를 구현한 결과이다.

성능 확인 결과 증명에 소요되는 시간은 Campanelli 등의 기법[10]이 다소 빠르며 나머지 기법은 비슷하다. 다만 증명 소요시간이 1초 미만이기 때문에, 그 차이가 실제 서비스에 유의미한 영향을 미치는 수준은 아니다.

CRS의 크기는 3.2.2의 기법, Agrawal 등의 기법[6], Campanelli 등의 기법[10]이 다른 기법에 비해 작고, 증명의 크기는 간결한 증명을 사용하지 않는 3.2.2의 기법 및 Agrawal 등의 기법[6]이 다른 기법보다 큰 것을 확인할 수 있다. 개인의 단말기로 CRS가 전송되어야 하는 4.1의 신용평가방법을 고려할 때, CRS와 증명의 크기를 합친 전체 통신량은 3.2.2의 기법, Campanelli 등의 기법[10], Agrawal 등의 기법[6] 등의 순으로 유리하다. 따라서 3.2.2의 기법이 서비스에 적합하나, 증명 소요시간이나 증명의 크기를 중요하게 고려해야 할 경우는 Campanelli 등의 기법[10]도 적용 가능하다.

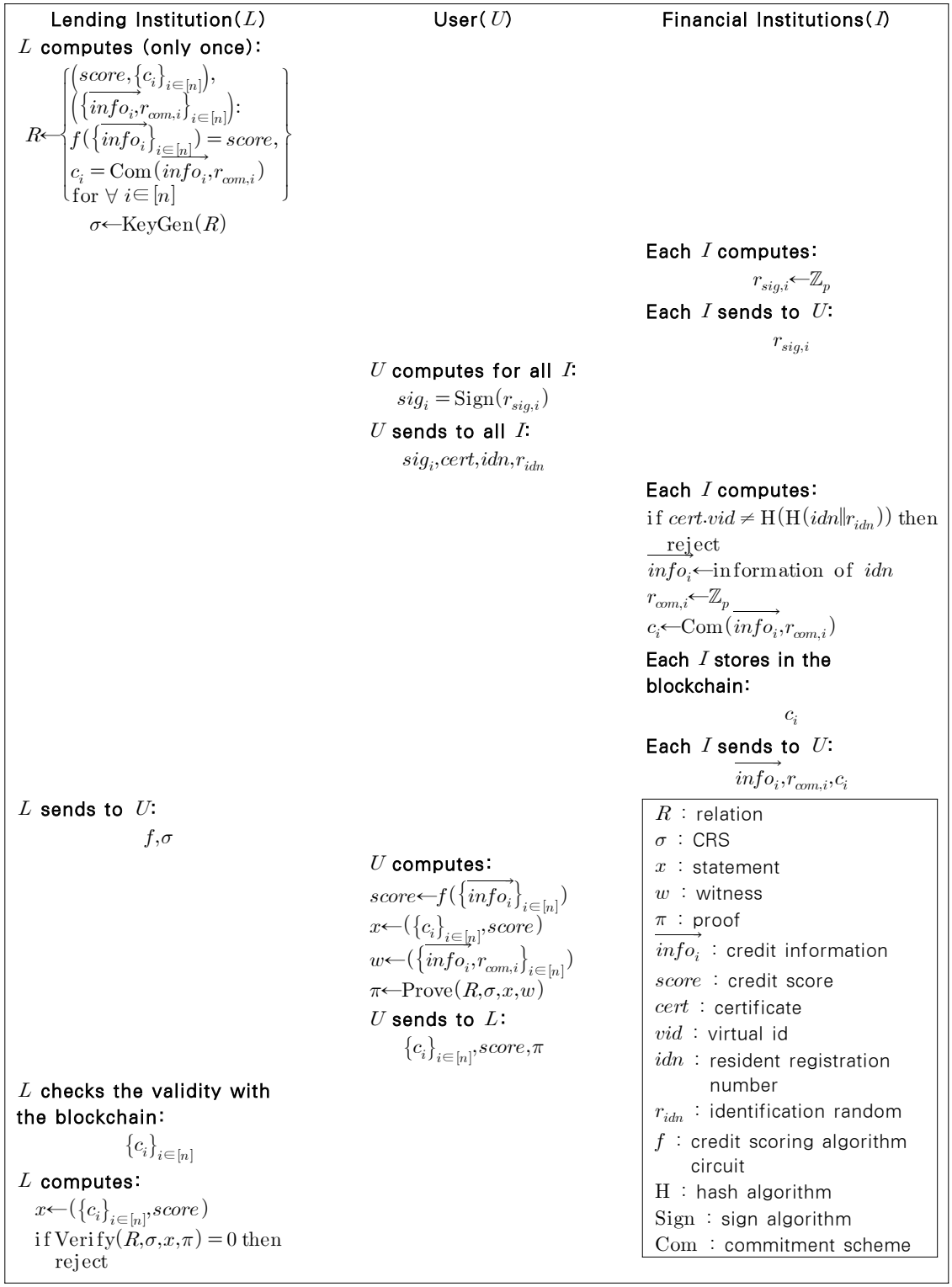

Fig. 3. Privacy-preserving credit scoring protocol

IV. 프라이버시 보장 신용평가방법

4.1 제안하는 방법

영지식 증명을 이용하여 프라이버시를 보장하도록 본 논문에서 제안하는 신용평가방법은 개인이 각 금융기관으로부터 신용정보를 수집하는 신용정보 수집 단계, 개인이 신용점수를 계산하고 증명을 생성하는 신용점수 계산단계, 대출기관이 신용점수에 대한 증명을 검증하는 신용점수 검증단계로 구성된다. 전체적인 흐름은 Fig. 3.과 같다.

한국의 금융환경에서 가장 널리 쓰이는 인증수단은 공인인증서이며, 이는 공인인증기관이 전자서명법에 따라 개인의 신원을 확인하고 발급한 공개키 인증서이다. 공인인증기관은 주민등록번호 idn를 이용하여 계산된 vid를 공인인증서에 포함시켜, 공인인증서의 소유자가 특정 주민등록번호를 이용하는지 확인할 수 있도록 한다.

본 논문의 신용평가방법에서는 개인의 신원확인을 위해 공인인증서를 이용하였다. 공인인증서는 공인인증기관이라는 중앙기관에 의존하여 발급되지만, 일단 발급된 공인인증서는 중앙기관의 개입 없이 전자서명에 이용할 수 있다. 본 논문의 신용평가방법에서는 이미 금융거래를 위해 발급된 공인인증서를 활용만 하므로 추가적인 중앙기관을 필요로 하지 않는다.

4.1.1 신용정보 수집단계

개인은 전자서명의 재전송(replay)을 막기 위해 각 금융기관으로부터 난수 r_{sig}를 수신받고 이 값을 공인인증서 개인키로 서명하여 금융기관으로 전송한다. 이때 개인키에 함께 저장된 본인확인용 난수 r_{idn}와 주민등록번호 idn을 함께 전송하면 수신한 금융기관은 $vid = \mathrm{H}(\mathrm{H}(idn \| r_{idn}))$를 계산한 후 이 값이 공인인증서에 포함된 vid와 같은지 확인하여 주민등록번호의 소유자 여부를 확인할 수 있다[22]. 그다음 금융기관은 해당 주민등록번호에 대응되는 신용정보로 커밋먼트를 생성한 후 블록체인에 저장하고 신용정보와 커밋먼트를 개인에게 전송한다.

이때 개인은 자신이 거래하지 않는 금융기관과도 동일하게 위 과정을 수행하여야 하며, 이 경우에 해당 금융기관은 전달할 정보가 없으므로 신용정보 내에 그 사실을 별도의 구분 값으로 명시한다. 이는 개인이 자신에게 불리한 정보를 누락시키는 것을 막기 위해, 모든 금융기관의 신용정보 및 커밋먼트가 증명 생성에 이용되었는지 대출기관이 검증하기 때문이다.

4.1.2 신용점수 계산단계

개인은 대출을 받고자 하는 기관으로부터 신용점수 생성을 위한 산술회로와 CRS를 수신받는다. 개인은 산술회로를 이용하여 신용점수를 계산한 후, 커밋먼트와 신용점수를 스테이트먼트, 금융기관으로부터 받은 신용정보를 위트니스로 하여 수신받은 CRS로 증명을 생성한다. 최종적으로 신용정보를 제외하고 신용점수, 커밋먼트, 증명만을 대출기관에게 전달한다.

4.1.3 신용점수 검증단계

대출기관은 모든 금융기관의 커밋먼트가 전달되었는지와 해당 금융기관이 그 커밋먼트를 최신 블록에 게시하였는지 점검한다. 그리고 CRS, 커밋먼트, 신용점수를 이용하여 수신받은 증명이 올바른지 검증한다. 검증에 성공할 경우 개인이 보낸 신용점수가 올바른 것이므로 이를 대출심사에 활용할 수 있다.

4.2 신용평가 파라미터 결정

2.4의 신용평가방법을 이용할 경우, 로지스틱 회귀 계수 및 계급별 WOE와 같은 파라미터(parameter)는 여러 개인의 신용정보 및 대출 정상 상환 여부를 수집하여 분석 후 결정된다. 기존에는 신용정보회사가 이런 정보를 직접 수집한 후 파라미터를 결정하였다. 하지만 본 논문의 방법에서는 프라이버시를 보장하기 위해 각 금융기관으로부터 별도로 익명화(anonymization)된 신용정보와 대출 정상 상환 여부를 수집하여 파라미터를 결정하여야 한다. 이 분석작업은 대출기관 스스로 수행할 수도 있고 별도의 기관이 수행한 후 각 대출기관에 제공할 수도 있다.

4.3 블록체인을 이용한 감사 프로세스

4.1의 방법은 각 금융기관이 개인에게 보내는 신용정보가 올바른 경우에만 정상적으로 동작한다. 하지만 이것을 강제하기는 쉽지 않다. 이를 검증할 수

있는 별도의 중앙기관이 존재하지 않으며, 특정 개인이 해당 금융기관의 고객일 경우 그 개인에게 유리하게 거짓 정보를 제공할 유인이 존재하기 때문이다. 이 경우, 블록체인에 저장된 커밋먼트를 감사기록으로 활용하면 이 현상을 최소화할 수 있으며 그 프로세스는 다음과 같다.

먼저 신뢰할 수 있는 제3자로 감사인을 지정하고 감사인은 주기적으로 블록체인에 저장된 커밋먼트의 무작위 표본을 뽑는다. 그리고 표본으로 뽑힌 커밋먼트를 저장한 금융기관에게 커밋먼트 생성 시 입력된 신용정보 및 난수와 신용정보에 대한 증빙자료를 제시하도록 요구한다. 금융기관이 요청받은 값을 감사인에게 전달하면 감사인은 신용정보 및 난수가 해당 커밋먼트와 일치하는지와 증빙자료가 올바른지를 확인하고 만약 올바르지 않으면 해당 금융기관에게 벌칙을 부과한다. 그리고 이 모든 작업이 끝나면 감사인은 전달받은 값을 안전하게 폐기한다. 비록 표본 비율이 전체 데이터에 비해 극히 일부라 하더라도 이런 프로세스를 통해서 금융기관이 올바른 정보를 개인에게 제공하도록 강제성을 부여할 수 있다.

4.4 커밋된 입력 검증의 효율화

영지식 증명과 블록체인을 이용하여 프라이버시를 보장하는 서비스를 제공할 때는 영지식 증명을 통해 증명하는 사실이 블록체인에 저장된 기존 사실들과 모순되지 않음을 함께 보여야 한다. 이를 위해 각 사실에 대한 커밋먼트를 블록체인에 저장하고 새로운 영지식 증명 과정에서 기존 커밋먼트와 모순되지 않는 위트니스를 이용한다는 것을 함께 증명한다.

Zerocash[23]는 비트코인에 기반을 둔 익명 암호화폐이다. Zerocash에서는 자신이 정당하게 소유한 코인만을 이용하게 하기 위해, 코인 소비를 위한 영지식 증명 시 블록체인에 저장된 자신의 코인에 대한 커밋먼트와 모순이 되지 않음을 함께 증명하게 한다. 다만 해당 영지식 증명이 어떤 커밋먼트를 이용하는지를 검증자가 직접 알게 되면 거래 그래프(transaction graph)를 알 수 있으므로 거래와 관련된 정보가 간접적으로 유출될 수 있다. 따라서 Zerocash에서는 영지식 증명 시에 특정 커밋먼트를 직접 스테이트먼트로 활용하는 것이 아니라 머클트리(merkle tree)의 루트만을 스테이트먼트로 활용하고 관련된 커밋먼트의 머클경로(merkle path)를 위트니스로 활용한다. 이 방법을 통해서 구체적으로 어떤 커밋먼트를 이용하는지는 드러나지 않지만 실제 블록체인 상에 존재하는 커밋먼트를 이용한다는 사실 자체는 증명할 수 있다. 다만 위 방법은 산술회로 내에 머클 트리에 대한 검증 로직이 포함되어야 하므로 CRS의 크기가 커지고 증명에 소요되는 시간이 늘어난다는 단점이 있다.

4.1의 방법에서도 신용점수에 대한 영지식 증명을 생성할 때, 블록체인에 저장된 금융기관의 커밋먼트를 이용한다는 것을 함께 증명하여야 한다. 다만 이 경우는 어떤 커밋먼트가 이용되는지 직접 노출되어도 추가로 유출되는 정보가 없다. 커밋먼트는 일회성으로 이용되기 때문에 그래프가 생기지 않으며, 영지식 증명 생성 시 고객 여부에 상관없이 모든 금융기관의 커밋먼트가 이용되기 때문이다. 따라서 4.1의 방법에서는 추가적인 정보의 유출 없이 직접 커밋먼트를 스테이트먼트로 이용함으로써 영지식 증명을 효율적으로 수행할 수 있다.

4.5 안전성 증명

4.1에서 대출기관에 제공하는 증명의 안전성은 일반적인 영지식 증명의 안전성과 같이 완전성, 건전성, 영지식성을 기준으로 판단할 수 있다.

4.5.1 완전성(completeness)

4.1에서 쓰인 영지식 증명, 커밋먼트 기법은 완벽한 완전성을 가지므로 이를 결합하여 대출기관에 제공한 증명도 역시 완전성을 가진다.

4.5.2 건전성(soundness)

4.1에서 쓰인 영지식 증명과 커밋먼트는 각각 계산적인 건전성과 계산적인 바인딩 속성을 가진다. 또한 블록체인에 커밋먼트를 저장하기 위한 금융기관 서명은 계산적으로 위조불가능하며(unforgeable), 전체 노드 51% 이상의 연산 능력을 점유하는 등 극단적인 조건이 만족되지 않으면 블록체인의 임의적인 변경이 불가능하다. 그리고 모든 금융기관에서 제공한 신용정보가 영지식 증명 생성에 이용되도록 하였으므로 유리한 신용점수를 위해 특정 신용정보를 누락시키는 것이 불가능하다. 따라서 대출기관에 제공한 증명은 건전성을 가진다.

Table 3. Credit information used for credit scoring

Name	Info Source	Info Type	Bit Width	Aggregation Method	Note
Sector	F	-	4		0 : Bank, 1 : Non-bank, 2 : Saving bank, 3 : Not a customer
Loan history					
- Number of overdue loans	F	C	6	Summation	(1)
- Total days delinquent	F	D	10	Summation	(2)
- Total days after payments of all overdue loans	F	D	10	Minimum	(3)
- Total amount of overdue loans	F	A	20	Summation	(4)
Current Loan					
- Number of loans	F	C	6	Summation	(5)
- Initial amount of credit loans	F	A	20	Summation	(6)
- Initial amount of secured loans	F	A	20	Summation	(7)
- Remaining balance of credit loans	F	A	20	Summation	(8)
- Remaining balance of secured loans	F	A	20	Summation	(9)
- Total amount due in credit card	F	A	20	Summation	(10)
- Remaining balance of credit card loans	F	A	20	Summation	(11)
Credit account age	F	D	12	Maximum	(12)
Percentage of recent loans	F	P	4	Average	(13)
Usage pattern of credit card	F	P	4	Average	(14)
Usage pattern of check card	F	P	4	Average	(15)
Income	P	A	20	-	(16)
User diligence	P	P	4	-	(17)
User tendency toward credit transaction	P	P	4	-	(18)
Credit transaction education	P	P	4	-	(19)
Long-term overdue experience	P	-	4	-	0 : No 1 : Yes

Info Source : F - Financial institutions, P - Public agencies
Info Type : C - Count, D - Number of days, A - Amount of money, P - Point(0-9)
Note : (1) to (5), (10) to (19), (8) divided by (6), (9) divided by (7) is used for logistic regression.

4.5.3 영지식성(zero-knowledge)

4.1에서 쓰인 영지식 증명은 완벽한 영지식성을 가지며 커밋먼트 기법도 완벽한 하이딩 속성을 가진다. 또한 개인이 대출기관에 제공한 증명과 각 금융기관에서 생성한 커밋먼트 간의 연결(link)을 통해서도 아무런 정보가 노출되지 않는다. 따라서 대출기관에 제공한 증명은 영지식성을 가진다.

V. 구현 및 실험

5.1 구 현

본 논문에서 제안한 신용평가방법을 구현하기 위해 공개 소프트웨어인 libsnark[24]와 xJsnark[25]를 이용하였다. libsnark는 zk-SNARK 라이브러리 중 가장 널리 쓰이며 Parno 등의 기법[7]과 Groth의 기법[8]을 포함한 다양한 zk-SNARK 기법을 지원한다. 성능 측정을 위해 libsnark를 이용하여 3.2.2의 기법, Agrawal 등의 기법[6], Campanelli 등의 기법[10]을 구현하였다. 이때 이용한 보안수준은 128비트이며 이에 맞추어 BN 타원곡선을 선택하였다.

xJsnark는 Java와 유사한 고수준(high-level) 자체 언어를 제공하는데 이 언어를 이용하여 코드를 작성하면 산술회로를 자동으로 생성해 준다. 성능 측정을 위해 xJsnark를 이용하여 신용평가 알고리즘을 구현하였으며 특히 xJsnark에서 제공하는 기능 중 산술연산(+,−,×,÷)과 비교연산을 이용하였다. 다만 부동소수점 연산은 제공되지 않으므로 내부적으

Table 4. Credit scoring performance

Condition	CRS size	Commitment size	Circuit size (compressed)	Proof size	Proving time	Verifying time
$n=100, m=5$	5.24MiB	3.16kiB	406kiB	56.7kiB	0.79sec	0.29sec
$n=100, m=10$	6.02MiB	3.16kiB	433kiB	56.7kiB	1.00sec	0.29sec
$n=200, m=5$	9.35MiB	6.28kiB	776kiB	113kiB	1.85sec	0.57sec
$n=200, m=10$	10.1MiB	6.28kiB	804kiB	113kiB	1.90sec	0.59sec

n : Number of financial institutions, m : Number of classes for each information

로 소수점 계산이 필요한 경우는 필요한 정확도에 따라 10^n을 곱한 상태에서 계산하고 최종적으로 다시 10^n을 나누도록 했다. 또한 자동 생성된 산술회로에서 일부 불필요한 부분을 수동으로 삭제하여 효율성을 높였다.

5.2 실 험

본 논문에서 제안한 신용평가방법이 실제 이용 가능한 성능을 가지는지 검증하기 위해 실험을 수행하였다. 실험을 위해 이용되는 신용정보는 KCB의 개인신용평가체계 공시[2]를 참고하되, 환경에 맞게 일부를 수정하여 Table 3.과 같이 정의하였다. 상단의 정보는 총 n개의 금융기관에서 각각 제공하고 하단의 정보는 한 개의 공공기관에서 제공한다고 가정하였다. 또한 신용점수 계산방식은 2.4의 로지스틱 회귀를 이용하였다.

본 실험에서는 금융기관이 제공하는 각 신용정보를 3개의 업권으로 나누어서 집계하였다. 이는 제1금융권, 제2금융권, 저축은행권 여부에 따라 대출 및 연체가 신용점수에 미치는 영향이 다르기 때문이다. 그리고 집계된 각 신용정보가 m개의 계급 중 어디에 속하는지 확인하여 해당하는 WOE를 이용하였으며, 각 WOE에 로지스틱 회귀 계수를 곱하여 합산되었다. 이때 고객이 일반 고객인 경우와 장기연체고객인 경우를 분리하여 별도의 로지스틱 회귀 계수 및 계급별 WOE를 이용하도록 하였으며, 이 값들은 4.2의 과정을 거쳐 이미 결정되었다고 가정하였다. 최종적으로 오즈 $\ln(\frac{p}{1-p})$와 신용점수를 계산하기 위한 식은 다음과 같다[16].

- 일반 고객인 경우:

$$\ln(\frac{p}{1-p}) = a+(b_{\text{업권}A,1}WOE_{\text{업권}A,1}+\cdots+b_{\text{업권}A,l}WOE_{\text{업권}A,l})+(b_{\text{업권}B,1}WOE_{\text{업권}B,1}+\cdots+b_{\text{업권}B,l}WOE_{\text{업권}B,l})+\cdots+(b_{\text{기타},1}WOE_{\text{기타},1}+\cdots+b_{\text{기타},k}WOE_{\text{기타},k})$$

- 장기연체경험 고객인 경우 :

$$\ln(\frac{p}{1-p}) = c+(d_{\text{업권}A,1}WOE_{\text{업권}A,1}+\cdots+d_{\text{업권}A,l}WOE_{\text{업권}A,l})+(d_{\text{업권}B,1}WOE_{\text{업권}B,1}+\cdots+d_{\text{업권}B,l}WOE_{\text{업권}B,l})+\cdots+(d_{\text{기타},1}WOE_{\text{기타},1}+\cdots+d_{\text{기타},k}WOE_{\text{기타},k})$$

- 신용점수 :

$$score = \ln(\frac{p}{1-p})\times factor + offset$$

위 과정에서 신용정보를 집계하거나 오즈 및 신용점수를 계산할 때는 xJsnark의 산술연산을, 집계된 신용정보로 계급별 WOE를 계산할 때는 xJsnark의 비교연산을 주로 이용하였다. 또한 신용점수 계산 시 이용된 신용정보가 커밋먼트와 일치하는지 검증하기 위해서 효율적인 3.2.2의 기법을 이용하였다.

금융기관의 수 n은 [2]에 공시된 이용기관 수 183을 감안하여 200으로 하였으며 계급의 수 m은 임의로 10으로 정했다. Intel Core i7-8700K 3.70GHz RAM 16GB 환경에서 실험을 수행한 결과는 Table 4.와 같다. 본 논문의 방법에서는 개인의 단말에서 증명을 생성하기 때문에, CRS 및 산술회로가 개인의 단말로 다운로드되고 생성된 증명이 다시 개인의 단말로부터 업로드되어야 한다. $n=200, m=10$일 때, 대출기관과의 총 통신량은

11MiB 정도이며 증명 생성 시간은 2초 미만이므로 충분히 실 환경에서 이용 가능하다는 것을 확인할 수 있다.

VI. 결 론

본 논문에서는 영지식 증명과 블록체인을 이용함으로써 프라이버시를 보장하며 중앙기관에 의존하지 않는 신용평가방법을 제안하였다. 또한 영지식 증명 생성 시 입력된 값이 블록체인의 커밋먼트와 일치하는지 검증하는 과정을 효율화하기 위해 커밋된 입력에 대해 효율적으로 증명이 가능한 영지식 증명 기법을 제안하였다. 마지막으로 실제 환경과 유사하게 신용정보 및 신용점수 알고리즘을 설계한 후 실험을 수행하여 제안한 방법이 실제 환경에서 이용 가능함을 확인하였다.

중앙기관에 의존하지 않는 신용평가방법은 프라이버시 보장 외에도 개인정보보호법 및 신용정보법의 의무사항 준수에 필요한 비용을 최소화하고 대출기관별로 자신만의 신용평가 알고리즘을 이용할 수 있다는 장점이 있다. 또한 영지식 증명과 블록체인을 결합할 경우, 중앙기관에 의존하는 다른 서비스들도 중앙기관 없이 제공되도록 변경할 수 있다. 다만 현재의 영지식 증명은 대량의 데이터와 많은 연산량이 필요할 경우 그 효율이 낮을 수 있으므로 서비스의 특성을 분석하여 채택 여부를 결정해야 한다.

References

[1] H.A. Abdou and J. Pointon, "Credit scoring, statistical techniques and evaluation criteria: a review of the literature," Intelligent Systems in Accounting, Finance and Management, vol. 18, no. 2-3, pp. 59-88, Apr. 2011.

[2] Korea Credit Bureau, "Public disclosure of personal credit scoring model," http://www.koreacb.com/kr/etc/policy_scoring, Oct. 2019.

[3] Interwork Technologies, "The Equifax cyber attack - how it happened and how to protect yourself," https://interwork.com/equifax-cyber-attack-happened-protect, Oct. 2019.

[4] Keep Network, "How to kill Equifax," https://blog.keep.network/how-to-kill-equifax-9b0222af5f88, Oct. 2019.

[5] QEDIT-Medium, "Trustless computing on private data," https://medium.com/qed-it/trustless-computing-on-private-data-6dc2deac306b, Oct. 2019.

[6] S. Agrawal, C. Ganesh, and P. Mohassel, "Non-interactive zero-knowledge proofs for composite statements," CRYPTO 2018, pp. 643-673, Aug. 2018.

[7] B. Parno, J. Howell, C. Gentry, and M. Raykova, "Pinocchio: nearly practical verifiable computation," IEEE Symposium on Security and Privacy 2013, pp. 238-252, May 2013.

[8] J. Groth, "On the size of pairing-based non-interactive arguments," EUROCRYPT 2016, pp. 305-326, May 2016.

[9] D. Fiore, C. Fournet, E. Ghosh, M. Kohlweiss, O. Ohrimenko, and B. Parno, "Hash first, argue later: adaptive verifiable computations on outsourced data," ACM Conference on Computer and Communications Security 2016, pp. 1304-1316, Oct. 2016.

[10] M. Campanelli, D. Fiore, and A. Querol, "LegoSNARK: modular design and composition of succinct zero-knowledge proofs," ACM Conference on Computer and Communications Security 2019, pp. 2075-2092, Nov. 2019.

[11] N. Bitansky, A. Chiesa, Y. Ishai, O. Paneth, and R. Ostrovsky, "Succinct non-interactive arguments via linear interactive proofs," Theory of Cryptography Conference 2013, pp. 315-333, Mar. 2013.

[12] Home page of Ivan Damgård, "On Σ-protocols," http://www.cs.au.dk/~ivan/Sigma.pdf, Oct. 2019.

[13] A. Fiat and A. Shamir, "How to prove

yourself: practical solutions to identification and signature problems," CRYPTO '86, pp. 186-194, Aug. 1986.
[14] T.P. Pedersen, "Non-interactive and information-theoretic secure verifiable secret sharing," CRYPTO '91, pp. 129-140, Aug. 1991.
[15] E. Mays, Handbook of credit scoring, 1st Ed., Glenlake Publishing Company, pp. 91-92, Mar. 2001.
[16] B.W. Yap, S.H. Ong, and N.H.M. Husain, "Using data mining to improve assessment of credit worthiness via credit scoring models," Expert Systems with Applications, vol. 38, no. 10, pp. 13274-13283, Sep. 2011.
[17] P.S. Barreto and M. Naehrig, "Pairing-friendly elliptic curves of prime order," Selected Areas in Cryptography 2005, pp. 319-331, Mar. 2005.
[18] E. Kiltz and H. Wee, "Quasi-adaptive NIZK for linear subspaces revisited," EUROCRYPT 2015, pp. 101-128, Apr. 2015.
[19] M. Ajtai, "Generating hard instances of lattice problems (extended abstract)," ACM Symposium on Theory of Computing '96, pp. 99-108, Jul. 1996.
[20] A. Kosba, Z. Zhao, A. Miller, Y. Qian, H. Chan, C. Papamanthou, R. Pass, A. Shelat, and E. Shi, "C∅C∅: a framework for building composable zero-knowledge proofs," IACR ePrint 2015-1093, Nov. 2015.
[21] Korea Internet & Security Agency, "Guideline for using cryptographic algorithms and key lengths," KISA-GD-2018-0034, Dec. 2018.
[22] Korea Internet & Security Agency, "Subscriber identification based on virtual ID," KCAC.TS.SIVID, Sep. 2009.
[23] E.B. Sasson, A. Chiesa, C. Garman, M. Green, I. Miers, E. Tromer, and M. Virza, "Zerocash: decentralized anonymous payments from Bitcoin," IEEE Symposium on Security and Privacy 2014, pp. 459-474, May 2014.
[24] GitHub, "libsnark: a C++ library for zkSNARK proofs," https://github.com/scipr-lab/libsnark, Oct. 2019.
[25] GitHub, "xJsnark," https://github.com/akosba/xjsnark, Oct. 2019.

〈저자소개〉

박 철 (Chul Park) 학생회원
2007년 2월: 서울대학교 컴퓨터공학부 졸업
2018년 3월~현재: 고려대학교 정보보호대학원 석사과정
〈관심분야〉 암호 프로토콜, 암호이론, 영지식 증명, 금융보안

김 종 현 (Jonghyun Kim) 학생회원
2014년 2월: 성균관대학교 수학과 졸업
2014년 3월~현재: 고려대학교 정보보호대학원 석박사 통합과정
〈관심분야〉 암호 프로토콜, 암호이론, 함수 암호

이 동 훈 (Dong Hoon Lee) 종신회원
1983년 8월: 고려대학교 경제학과 졸업
1987년 12월: Oklahoma University 전산학과 석사 졸업
1992년 5월: Oklahoma University 전산학과 박사 졸업
1993년 3월~1997년 2월: 고려대학교 전산학과 조교수
1997년 3월~2001년 2월: 고려대학교 전산학과 부교수
2001년 3월~현재: 고려대학교 정보보호대학원 교수
〈관심분야〉 암호 프로토콜, 암호이론, USN이론, 키 교환, 익명성 연구, PET 기술