(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0101469
(43) 공개일자 2022년07월19일

(51) 국제특허분류(Int. Cl.)
H04L 9/32 (2006.01)
(52) CPC특허분류
H04L 9/3215 (2013.01)
H04L 9/50 (2022.05)
(21) 출원번호 10-2021-0003531
(22) 출원일자 2021년01월11일
심사청구일자 2021년01월11일

(71) 출원인
충남대학교산학협력단
대전광역시 유성구 대학로 99 (궁동, 충남대학교)
(72) 발명자
류재철
대전광역시 유성구 어은로 57 한빛아파트 132-801
김근영
대전광역시 유성구 장대로 87 퍼스트캐슬 202호
김혁진
대전광역시 서구 대덕대로168번길 82 웰리움
(74) 대리인
이은철, 김재문

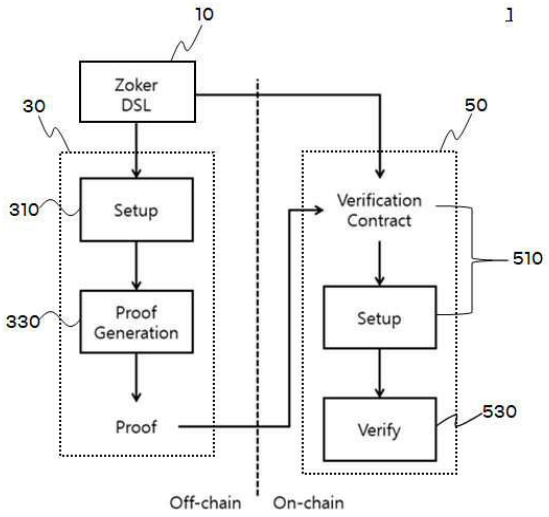
전체 청구항 수 : 총 4 항

(54) 발명의 명칭 **동적 연산을 통한 검증이 가능한 영지식 증명 시스템 및 증명 방법**

(57) 요 약

본 발명은 비밀 값에 대한 공개 없이 비밀 값에 대한 신뢰 가능 여부를 검증하는 블록체인에 적용되는 영지식 증명 시스템에 있어서, 하기 증명부 또는 하기 검증부의 수행 기준이 되는 연산을 포함한 소스 코드를 생성하며, 상기 소스 코드를 하기 증명부 또는 하기 검증부에 스마트 컨트랙트로 전달하는 소스 코드 생성부; 오프-체인(off-chain)에서 수행되며, 상기 소스 코드를 통해 상기 비밀 값에 대한 증명 값을 생성하는 증명부; 및 상기 증명부에서 생성된 상기 증명 값을 입력받고, 상기 소스 코드로부터 상기 증명 값에 따른 검증 회로를 생성하는 검증 셋업부와, 상기 검증 회로를 통해 상기 증명 값의 참/거짓 여부를 판단하는 판단부를 포함하는 검증부를 포함하며, 상기 검증부는 상기 증명 값의 검증을 온-체인(on-chain)에서 수행되어 동적 연산을 가능하게 함으로써 사용자의 편의성과 신뢰도를 높일 수 있다.

대 표 도 - 도3



(52) CPC특허분류
H04L 2209/12 (2013.01)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711102829
과제번호	2018-0-00251-003
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	정보보호핵심원천기술개발(R&D)
연구과제명	스마트 컨트랙트 프라이버시 보호 및 취약점 분석 기술개발
기 여 율	1/1
과제수행기관명	충남대학교 산학협력단
연구기간	2018.04.01 ~ 2020.12.31

명 세 서

청구범위

청구항 1

비밀 값에 대한 공개 없이 비밀 값에 대한 신뢰 가능 여부를 검증하는 블록체인에 적용되는 영지식 증명 시스템에 있어서,

하기 증명부 또는 하기 검증부의 수행 기준이 되는 연산을 포함한 소스 코드를 생성하며, 상기 소스 코드를 하기 증명부 또는 하기 검증부에 스마트 컨트랙트로 전달하는 소스 코드 생성부;

오프-체인(off-chain)에서 수행되며, 상기 소스 코드를 통해 상기 비밀 값에 대한 증명 값을 생성하는 증명부; 및

상기 증명부에서 생성된 상기 증명 값을 입력받고, 상기 소스 코드로부터 상기 증명 값에 따른 검증 회로를 생성하는 검증 셋업부와, 상기 검증 회로를 통해 상기 증명 값의 참/거짓 여부를 판단하는 판단부를 포함하는 검증부를 포함하며,

상기 검증부는 상기 증명 값의 검증을 온-체인(on-chain)에서 수행하는 것을 특징으로 하는 영지식 증명 시스템.

청구항 2

제 1 항에 있어서,

상기 증명부는,

상기 소스 코드를 증명 회로로 변환하는 증명 셋업부; 및

상기 비밀 값을 입력으로 하여 상기 증명 회로에 대한 상기 비밀 값의 출력인 상기 증명 값을 생성하는 증명값 생성부를 포함하는 것을 특징으로 하는 영지식 증명 시스템.

청구항 3

제 2 항에 있어서,

상기 증명 셋업부 또는 상기 검증 셋업부는,

for 문의 반복 횟수, 변수의 가변 길이에 따라 상기 증명 회로 또는 상기 검증 회로를 생성하는 것을 특징으로 하는 영지식 증명 시스템.

청구항 4

비밀 값에 대한 공개 없이 비밀 값에 대한 신뢰 가능 여부를 검증하는 블록체인에 적용되는 영지식 증명 방법에 있어서,

소스 코드를 생성하는 단계;

오프-체인(off-chain)에서 상기 소스 코드를 입력받아, 상기 소스 코드를 증명 회로로 변환하는 단계;

오프-체인(off-chain)에서 상기 비밀 값을 입력으로 하여 상기 증명 회로에 대한 상기 비밀 값의 출력인 상기 증명 값을 생성하는 단계;

오프-체인(off-chain)에서 생성된 상기 증명 값과 상기 소스 코드를 전달받아, 상기 소스 코드로부터 상기 증명 값에 따른 검증 회로를 온-체인(on-chain)에서 생성하는 단계; 및

상기 검증 회로를 통해 상기 증명 값의 참/거짓 여부를 판단하는 단계를 포함하는 것을 특징으로 하는 영지식 증명 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 블록체인에 적용되는 영지식 증명에 관한 것이다.

배 경 기 술

[0003] 블록체인은 특정 내용을 블록체인에 넣고 공개함으로써 일단 공표된 내용에 대해서 수정이 불가능한 불변성을 제공하여 신용이 필요한 용도에 적용할 수 있는 기술이다. 블록체인은 합의를 통해 탈중앙화함으로써 중앙화된 시스템의 도입 없이 신뢰성을 확보할 수 있다. 다만, 이때 해당 정보를 거래를 하는 주체 간뿐만 아니라 거래원장에 속한 타인에게도 공개된다. 즉, 블록체인의 가장 큰 특징은 투명성을 전제로 한 신뢰성 확보이다.

[0004] 그러나, 블록체인에 저장되는 데이터를 공개하고 유지해야 하는 특성 때문에 프라이버시 노출의 문제가 항상 존재한다. 예를 들어, 블록체인에 저장된 트랜잭션에 의한 사용자의 잔액, 송수신 정보 및 스마트 컨트랙트의 실행 내역 등이 노출된다. 따라서 이는 블록체인이 상용화를 저해하는 저해요소로 작용한다. 이러한 문제로 인해, 증명자(prover)가 자신의 정보를 공개하지 않으면서 해당 정보가 신뢰할 수 있는 진실이라는 점만을 확인하여 익명성을 확보하면서도 신뢰성을 유지하고자 하였다.

[0005] 블록체인에서 프라이버시 문제를 해결하는 방법은 크게 블록체인 외부에서 연산을 수행하는 오프 체인(off-chain) 연산 방법과 사용자의 개인정보를 드러내지 않고 연산 수행을 암호학적으로 증명하는 방법이 있다.

[0006] 그러나, off-chain에서 연산을 수행하는 방법은, 블록체인 상에서 연산이 수행된 것이 아니므로 기존의 DApp에 비해 신뢰성이 떨어지는 문제가 있으며, 개인정보를 다루는 작업을 위해 중앙화되는 문제가 있다. 이에 신뢰성을 위해 추가적인 작업을 필요로 하는데, 추가적인 작업으로 인해 시스템이 커지게 되고, 추가적인 작업을 위해 인적 자원이 요구되어 문제된다.

[0007] 한편, 영지식 증명(zero knowledge proof) 즉, ‘지식 없이 하는 증명’ 인 암호학적으로 연산 수행을 증명하는 방법은 영지식 증명은 비밀 값은 드러내지 않고 증명하고자 하는 사실에 대한 검증만 가능한다. 영지식 증명을 사용하여 연산을 증명하며 off-chain에서 개인정보에 대한 연산과 연산을 증명하기 위한 증명 값을 생성하고 블록체인에서 증명 값을 바탕으로 개인정보를 드러내지 않고 계산 결과가 올바르게 계산되었음을 검증할 수 있다. 예를 들어 암호 화폐 Zcash는 거래 당사자의 잔액과 같은 트랜잭션 정보를 드러내지 않으면서 유효한 거래임을 증명하여 익명성을 제공한다. 영지식 증명의 검증 역할을 스마트 컨트랙트로 구현할 수 있다.

[0008] 영지식 증명은 값을 드러내지 않고 비밀 값을 가지고 있음을 증명하는 증명자(prover)와 증명자의 비밀 값없이 증명자가 값을 가지고 있음을 증명하는 검증자(verifier)로 구성된다. 증명자는 off-chain에서 개인정보에 대한 연산과 연산을 증명하기 위해 비밀 값을 바탕으로 암호화된 증명 값(proof)을 생성하고, 검증자는 블록체인에서 증명 값을 검증하여 증명자가 비밀 값을 가지고 있음을 검증한다.

[0009] 그러나, 종래의 영지식 증명은 다음과 같은 두 가지 문제를 갖는다. 첫째는 영지식 증명을 위해 Trusted-setup이 요구된다는 점이다. 증명 값을 setup 과정에서 신뢰 기관을 필수적으로 요하는데, 이로 인해, 증명자가 자신의 연산을 증명할 때 스스로 setup을 수행할 수 없고 반드시 신뢰 기관을 통해야 한다.

[0010] 둘째, 기존의 프로그래밍 언어 로직은 제한적 언어 기능만을 제공하여 영지식 증명을 이용해 스마트 컨트랙트로 직접 구현할 경우 개발 과정이 어렵다. 이를 위해 ZoKrates, zkay와 같은 도구에 대한 연구가 진행되었지만, 기반이 되는 영지식 증명의 구현이 가지는 한계와 언어적 한계를 가지고 있다. 따라서, 영지식 증명을 검증하기 위한 언어가 기존 프로그래밍 언어와 다르게 고정적인 반복 횟수의 반복문만을 지원하고 사용자가 검증 컨트랙트의 circuit이 올바르게 생성되었음을 신뢰해야 하는 문제가 있다.

[0011] 이에 본 출원인은 영지식 증명의 증명 및 검증에 있어 사용자가 증명 값을 생성할 수 있는 환경 및 검증자가 증명 값을 블록체인상에서 생성된 circuit을 통해 검증할 수 있는 영지식 증명 시스템 및 증명 방법을

개발하였다. 이를 통해, 검증자는 동적인 연산에 대해서도 신뢰성 있는 circuit으로 검증할 수 있어 사용자 편의성을 향상시키면서, 익명성과 신뢰성을 확보할 수 있음을 확인하게 되었다.

발명의 내용

해결하려는 과제

[0013] 본 발명은 setup 과정에서 별도의 신뢰 기관을 요구하지 않으며, circuit을 조건에 따라 유연하게 생성할 수 있는 개발 환경을 가진 영지식 증명 시스템 및 증명 방법을 제공하고자 한다.

[0014] 본 발명이 해결하려는 과제들은 앞에서 언급한 과제들로 제한되지 않는다. 본 발명의 다른 과제 및 장점들은 아래 설명에 의해 더욱 분명하게 이해될 것이다.

과제의 해결 수단

[0016] 상기 목적을 달성하기 위하여 본 발명은, 비밀 값에 대한 공개 없이 비밀 값에 대한 신뢰 가능 여부를 검증하는 블록체인에 적용되는 영지식 증명 시스템에 있어서, 하기 증명부 또는 하기 검증부의 수행 기준이 되는 연산을 포함한 소스 코드를 생성하며, 상기 소스 코드를 하기 증명부 또는 하기 검증부에 스마트 컨트랙트로 전달하는 소스 코드 생성부; 오프-체인(off-chain)에서 수행되며, 상기 소스 코드를 통해 상기 비밀 값에 대한 증명 값을 생성하는 증명부; 및 상기 증명부에서 생성된 상기 증명 값을 입력받고, 상기 소스 코드로부터 상기 증명 값에 따른 검증 회로를 생성하는 검증 셋업부와, 상기 검증 회로를 통해 상기 증명 값의 참/거짓 여부를 판단하는 판단부를 포함하는 검증부를 포함하며, 상기 검증부는 상기 증명 값의 검증을 온-체인(on-chain)에서 수행하는 것을 특징으로 한다.

[0017] 바람직하게, 상기 증명부는, 상기 소스 코드를 증명 회로로 변환하는 증명 셋업부; 및 상기 비밀 값을 입력으로 하여 상기 증명 회로에 대한 상기 비밀 값의 출력인 상기 증명 값을 생성하는 증명값 생성부를 포함할 수 있다.

[0018] 바람직하게, 상기 증명 셋업부 또는 상기 검증 셋업부는, for 문의 반복 횟수, 변수의 가변 길이에 따라 상기 증명 회로 또는 상기 검증 회로를 생성할 수 있다.

[0019] 또한, 상기 목적을 달성하기 위하여 본 발명은, 비밀 값에 대한 공개 없이 비밀 값에 대한 신뢰 가능 여부를 검증하는 블록체인에 적용되는 영지식 증명 방법에 있어서, 소스 코드를 생성하는 단계; 오프-체인(off-chain)에서 상기 소스 코드를 입력받아, 상기 소스 코드를 증명 회로로 변환하는 단계; 오프-체인(off-chain)에서 상기 비밀 값을 입력으로 하여 상기 증명 회로에 대한 상기 비밀 값의 출력인 상기 증명 값을 생성하는 단계; 오프-체인(off-chain)에서 생성된 상기 증명 값과 상기 소스 코드를 전달받아, 상기 소스 코드로부터 상기 증명 값에 따른 검증 회로를 온-체인(on-chain)에서 생성하는 단계; 및 상기 검증 회로를 통해 상기 증명 값의 참/거짓 여부를 판단하는 단계를 포함하는 것을 다른 특징으로 한다.

발명의 효과

[0021] 본 발명에 따르면 블록체인상(on-chain)에서 검증 컨트랙트 함수의 연산 정보를 이용해 circuit을 생성하여 setup을 수행할 수 있다. 이때, 본 발명은 setup 과정에서 신뢰 기관을 요구하지 않으며 검증 시간이 짧고, 언어 환경을 개선하여 개발자에게 일반 프로그래밍 언어와 비슷한 수준의 개발 환경을 제공할 수 있다.

[0022] 이를 통해, circuit이 올바르게 계산되었다고 무조건 신뢰하지 않고 입력값에 적절하게 circuit을 생성하여 검증할 수 있으므로 사용자의 신뢰도를 높일 수 있는 장점이 있다. 또한, loop의 횟수가 늘어나거나 변수의 길이가 늘어나는 등 증명하고자 하는 statement가 변경되는 경우에도 검증 컨트랙트를 새롭게 생성하지 배포하지 않고 기존 검증 컨트랙트를 그대로 이용할 수 있어 사용자 편의성을 높이는 효과가 있다.

도면의 간단한 설명

[0024] 도 1은 종래의 영지식 증명 중 하나인 ZoKrates을 나타낸다.

도 2는 ZKBoo 알고리즘의 구조를 나타낸다.

도 3은 본 발명의 실시예로서 영지식 증명 시스템을 나타낸다.

도 4는 Verifiable Computation 개념은 나타낸 것이다.

도 5는 증명부의 증명 셋업부와 증명 값 생성부를 통해 증명 값을 생성하는 과정을 나타낸다.

도 6은 본 발명에 따른 다른 실시예로서, 비밀 값에 대한 공개 없이 비밀 값에 대한 신뢰 가능 여부를 검증하는 블록체인에 적용되는 영지식 증명 방법을 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0025] 이하, 첨부된 도면들에 기재된 내용들을 참조하여 본 발명을 상세히 설명한다. 다만, 본 발명이 예시적 실시 예들에 의해 제한되거나 한정되는 것은 아니다. 각 도면에 제시된 동일 참조부호는 실질적으로 동일한 기능을 수행하는 부재를 나타낸다.

[0026] 본 발명의 목적 및 효과는 하기의 설명에 의해서 자연스럽게 이해되거나 보다 분명해 질 수 있으며, 하기의 기재만으로 본 발명의 목적 및 효과가 제한되는 것은 아니다. 또한, 본 발명을 설명함에 있어서 본 발명과 관련된 공지 기술에 대한 구체적인 설명이, 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명을 생략하기로 한다.

[0027] 영지식 증명은 어떠한 사실을 증명하고 검증할 것인지, 숨기고자 하는 비밀 값이 무엇인지를 정의하는 암호학적 프로토콜이다. 예를 들어, 인증 시스템에서 영지식 증명을 적용할 경우 인증에서 요구되는 패스워드나 개인정보를 드러내지 않고, 값을 알고 있음을 증명하는 방식으로 구성할 수 있다. 영지식 증명은 증명자와 검증자 간의 프로토콜에서 증명자가 비밀 값을 드러내지 않고 검증자가 증명하고자 하는 사실에 대한 문장(statement)의 참/거짓 여부만을 확인시킨다. 증명자와 검증자에게 증명하고자 하는 함수가 공유되었을 때, 증명자가 함수의 입력으로 문장과 비밀 값(witness)을 입력으로 하여 증명 값을 생성하면, 검증자가 검증과정에서 비밀 값과 관련된 어떠한 값도 알 수 없어야 한다.

[0028] 도 1은 종래의 영지식 증명 중 하나인 ZoKrates을 나타낸다. ZoKrates는 circuit에 대한 setup 과정을 off-chain에서 수행하고 검증 컨트랙트(verification contract)에서 setup의 결과를 공유한다. 사용자가 자신의 비밀 값을 이용해 statement를 증명할 때 ZoKrates 언어와 setup에서 만들어낸 Proving key와 함께 비밀 값을 통해 proof를 생성한다. 값의 검증은 Setup에서 생성된 Verification Key를 통해 검증 컨트랙트를 생성해 블록체인에 배포한 후 블록체인상에서 proof의 참/거짓 여부를 검증할 수 있다.

[0029] 그러나 ZoKrates 프로토콜을 이용할 경우 검증 컨트랙트는 proof의 검증만을 수행하기 때문에 블록체인 상에서의 비용을 줄일 수 있지만, setup 과정에서 지정한 circuit에 대해서만은 검증을 수행할 수 없어 동적인 연산의 검증을 할 수 없다는 문제가 있다.

표 1

[0030]

scheme	Verification time	proof size	Trusted setup
SNARK	l	1	O
Bulletproofs [15]	$n\log(n)$	$\log(n)$	X
ZKBoo [16]	$c\log(c)+h\log(h)$	$\sqrt{n}$	X

[0031] 표 1은 영지식 증명 implementation 성능을 비교한 것이다. n은 연산을 표현한 gate의 수이며, c는 sub-circuit의 개수이다. h는 sub-circuit의 폭을 표현하는 값이고 l은 instance size를 의미한다. SNARK은 검증 시간과 proof 크기가 circuit의 크기에 비례하지 않아 복잡한 연산의 검증을 수행하더라도 항상 작은 검증 시간과 proof을 가지는 장점이 있지만 Trusted setup을 요구한다. 그에 반해 Bulletproofs는 Trusted setup을 요구하지 않고 proof size는 비교적 작지만 검증 시간이 오래 걸리는 문제가 있다. ZKBoo는 Trusted setup을 요구하지 않는 알고리즘 중 검증 시간이 gate에 비례하지 않아 비교적 빠른 장점이 있지만 proof의 크기가 매우 큰 문

제가 있다.

[0032] 표 1를 참고하면 Trusted setup을 사용하는 SNARK에 비해 Trusted setup을 사용하지 않는 알고리즘들은 검증시간 또는 proof 크기 면에서 성능이 떨어지는 것을 확인할 수 있다. 그러나 Trusted setup을 사용하지 않는 알고리즘은 setup 과정에서 신뢰 기관을 필요로 하지 않기 때문에 증명자가 setup을 수행하더라도 fake proof 문제가 발생하지 않는다는 장점을 가진다.

[0033] 한편, 블록체인은 P2P 네트워크로 구성된 탈중앙화 방식을 지향하기 때문에, 프라이버시 보호를 위해 영지식 증명을 도입하는 경우 비대화형 방식이면서 신뢰 기관을 요구하지 않는 영지식 증명이 적합하다. 또한 검증 비용이 적으면서, 증거 크기가 작은 방향의 알고리즘을 선정할 필요가 있다.

[0034] 따라서, 본 발명의 실시예에서는 Trusted setup 가정을 사용하지 않는 알고리즘 중 Bulletproofs보다 비교적 검증 시간이 짧은 ZKBoo 알고리즘을 이용해 증명자가 자신의 증명하려는 statement에 대해서 스마트 컨트랙트에서 setup을 수행하여 동적인 연산 검증을 지원하는 방법을 제안한다

[0035] 도 2는 ZKBoo 알고리즘의 구조를 나타낸다.

[0036] 먼저, ZKBoo 알고리즘은 영지식 증명 implementation 중 하나로 Multi-Party Computation과 SHA256 해시 알고리즘을 기반으로 구현될 수 있다. 도 2를 참고하면 입력 값을 3-party가 계산할 수 있도록 share로 나누고 연산을 수행하여 각 party의 한 gate의 실행 결과를 자신과 다른 party에게 나누어 주는 (2, 3)-decomposition을 수행할 수 있다.

[0037] 연산 도중 계산된 witness 값들 중 한 party의 값만을 공개하고 나머지 party의 값은 해시 값인 SHA256 결과만을 공개한다. 이것으로 인해 계산의 결과는 확인할 수 있지만 전체 party의 값이 공개되지 않았기 때문에 입력 값에 대해서는 검증자가 알 수 없는 영지식 증명이 가능하다. 또한 setup 과정에서 사용되는 랜덤 값이 proof 검증에 영향을 주지 않기 때문에 Trusted setup이 필요하지 않다.

[0038] 하지만 ZKBoo를 이용할 때 2-party의 witness 값에 대한 해시 값을 proof으로 모두 사용할 경우 사용자가 입력해야 하는 proof의 크기가 매우 커질 수 있다. 때문에 전체 해시 값을 입력하는 것이 아닌 초기 2-party의 해시 값과 witness 값만을 commitment로 공개하고 이후의 값은 검증 시에 계산하도록 구현되어야 한다.

[0039] 도 3은 본 발명의 실시예로서 영지식 증명 시스템(1)을 나타낸다. 본 발명의 영지식 증명 시스템(1)은 비밀 값에 대한 공개 없이 비밀 값에 대한 신뢰 가능 여부를 검증하는 블록체인에 적용될 수 있다. 비밀 값은 증명자가 공개하지 않고자 하는 정보이며, 검증자는 본 발명의 영지식 증명 시스템(1)의 검증을 통해 증명자가 제공하는 증명 값을 검증함으로써 증명자의 비밀 값의 신뢰 여부를 판단할 수 있다. 영지식 증명 시스템(1)은 소스 코드 생성부(10), 증명부(30) 및 검증부(50)를 포함할 수 있다. 본 발명의 실시예로서 영지식 증명 시스템(1)을 ‘Zoker’라 명명하며, 이하 Zoker 프로토콜을 도 3을 통해 설명한다. 본 발명은 Zoker DSL(Domain Specific Language)를 이용해 개발자가 프로그램을 작성할 수 있도록 개발 환경을 제공하므로 Zoker DSL을 이용해 영지식 증명을 생성할 수 있다.

[0040] 소스 코드 생성부(10)는 증명부(30) 또는 검증부(50)의 수행 기준이 되는 연산을 포함한 소스 코드를 생성할 수 있으며, 소스 코드를 증명부(30) 또는 검증부(50)에 영지식 증명을 검증하기 위한 스마트 컨트랙트로 전달할 수 있다. 전달한 스마트 컨트랙트(소스 코드)는 증명자의 증명 값을 검증하기 위한 검증 컨트랙트(Verification Contract)로서 검증자에게 공개될 수 있다. 도 4를 통해 소스 코드 생성부(10)를 구체적적으로 설명한다.

[0041] 도 4는 Verifiable Computation 개념은 나타낸 것이다. 연산 검증기법은 도 4와 같이 증명자와 검증자가 공통적으로 수행할 연산 f를 공유한다. 연산 f에 x를 입력으로 한 결과 y를 직접 실행하지 않고 올바른 계산임을 증명 및 검증할 수 있도록 구성하기 위해서는 산술 회로 (Arithmetic Circuit) 또는 부울 회로 (Boolean Circuit)로 표현되어야 하는데, 이렇게 표현된 산술/부울 회로를 대화형 혹은 비대화형 프로토콜을 이용해 검증자는 증명자의 계산량보다 적은 계산량으로 Circuit의 수행 결과를 검증할 수 있어야 한다. 비밀 값을 입력으로 하여 Circuit의 실행 도중 계산된 값인 Witness를 통해 생성한 증거 Proof 또는 arguments를 전송하여 검증자가 효율적으로 연산이 올바르게 수행되었음을 납득할 수 있도록 구성하는 것이 중요하다. 이 때 검증자가 연산의 특정 입력에 대하여 비대화형 Proof of Knowledge이고 동시에 증거의 크기가 문장 또는 회로의 크기에 비해 매우 작은 (예: $\log(|x|)$) 경우 SNARK(Succinct Non-interactive ARgument of Knowledge)라고 한다.

[0042] 이때 종래의 영지식 증명에 사용되는 알고리즘은 증명하는 statement의 setup 과정에서 랜덤 값으로 인해 신뢰 기관을 요구하는 Trusted setup을 가정한다. 전술한 바와 같이 Trusted setup 가정은 성능은 개선하지만 setup

단계에서 신뢰 기관을 요구하고, 이때 setup을 수행하는 사람이 악의적인 의도를 가진다면 statement를 만족시키는 비밀 값을 알지 못하더라도 검증자의 검증 결과를 참으로 만들 수 있는 fake proof 문제가 발생할 수 있다.

[0043] 반면, 본 발명은 이하 후술하는 바와 같이, 증명 회로 또는 검증 회로를 변경할 수 있어 fake proof와 같은 문제 생성을 방지할 수 있다. 보다 상세하게, 연산은 Zoker DSL의 함수에 정해져 있지만 for 문의 반복 횟수, 변수의 가변 길이에 의해 수학적인 증명 회로 또는 검증 회로는 변경될 수 있다. 따라서, 증명자 또는 검증자의 입력값에 맞는 회로 생성을 위해 setup을 수행할 수 있다.

[0044] 증명부(30)는 오프-체인(off-chain)에서 수행되며, 소스 코드를 통해 비밀 값에 대한 증명 값을 생성할 수 있다. 증명부(30)는 소스 코드를 증명 회로로 변환하는 증명 셋업부(310)와 비밀 값을 입력으로 하여 증명 회로에 대한 비밀 값의 출력인 증명 값을 생성하는 증명 값 생성부(330)를 포함할 수 있다. 증명 셋업부(310)는 for 문의 반복 횟수, 변수의 가변 길이에 따라 증명 회로를 생성할 수 있다.

[0045] 도 5는 증명부(30)의 증명 셋업부(310)와 증명 값 생성부(330)를 통해 증명 값을 생성하는 과정을 나타낸다. 본 발명의 증명부(30)는 off-chain이 아닌 on-chain에서 수행될 수 있다. 증명 셋업부(310)는 도 5의 Frontend와 대응하는 구성이며, 증명 값 생성부(330)는 Backend에 대응될 수 있다. 증명 셋업부(310)는 소스 코드 생성부(10)에서 생성된 소스 코드를 입력받아 circuit 또는 gate로 변경할 수 있으며, 이때 증명 셋업부(310)는 편리한 구현을 위하여 소스 코드를 IR (Intermediate Representation) 언어로 변경한 후 circuit으로 변환할 수 있다. 여기에서 생성되는 circuit은 증명 회로일 수 있다. 증명 셋업부(310)에서 생성된 증명 회로는 증명 값 생성부(330)에서 영지식 증명 암호 알고리즘의 implementation에 따라 수학적 알고리즘을 이용하여 판단부(530)에서 판단하는 도중에 비밀 값을 공개하지 않고도 검증을 할 수 있도록 변경될 수 있다.

[0046] 증명 값 생성부(330)는 증명 셋업부(310)에서 생성된 증명 회로를 이용해 증명 값(proof)을 만들 수 있다. 증명 값 생성부(330)는 변경한 증명 회로에 증명자의 비밀 값을 입력으로 하여 증명 값을 생성할 수 있고, 증명 값을 검증함으로써 비밀 값의 공개 없이 계산 결과가 올바른 연산 수행을 통해 산출되었음을 확인할 수 있다. 이 때 값을 드러내지 않고 연산이 올바르게 수행되었음을 검증하기 위해 Pairing, Multi-Party Computation, Hash 알고리즘 등의 수학적 알고리즘을 이용해 비밀 값을 가릴 수 있다.

[0047] 검증부(50)는 증명부(30)에서 생성된 증명 값을 검증하여 증명자가 보유한 비밀 값의 신뢰 여부를 판단할 수 있다. 검증부(50)는 검증 셋업부(510)와 판단부(530)를 포함할 수 있다. 검증 셋업부(510)는 증명부(30)에서 생성된 증명 값을 입력받고, 소스 코드로부터 증명 값에 따른 검증 회로를 생성할 수 있다. 이는 도 1의 종래 영지식 증명 알고리즘이 신뢰 기관을 통해 생성된 circuit을 그대로 이용하여 증명 값을 검증하는 것과 달리, 자체적으로 검증 회로를 생성하는 점에서 차이가 있으며, 특히 검증 셋업부(510)는 for 문의 반복 횟수, 변수의 가변 길이에 따라 조건 또는 증명 값에 적합하도록 검증 회로를 생성 또는 변경할 수 있어 보다 유연하다.

[0048] 판단부(530)는 검증 회로를 통해 증명 값의 참/거짓 여부를 판단할 수 있다.

[0049] 도 6은 본 발명에 따른 다른 실시예로서, 비밀 값에 대한 공개 없이 비밀 값에 대한 신뢰 가능 여부를 검증하는 블록체인에 적용되는 영지식 증명 방법을 나타낸다. (a)소스 코드를 생성하는 단계, (b)오프-체인(off-chain)에서 소스 코드를 입력받아, 소스 코드를 증명 회로로 변환하는 단계, (c)오프-체인(off-chain)에서 비밀 값을 입력으로 하여 증명 회로에 대한 비밀 값의 출력인 증명 값을 생성하는 단계, (d)오프-체인(off-chain)에서 생성된 증명 값과 소스 코드를 전달받아, 소스 코드로부터 증명 값에 따른 검증 회로를 온-체인(on-chain)에서 생성하는 단계, (e)검증 회로를 통해 증명 값의 참/거짓 여부를 판단하는 단계를 포함할 수 있다.

[0050] (a) 내지 (e)단계는 전술한 영지식 증명 시스템(1)의 실시예에서 수행되는 과정을 나타낸 것으로 각 단계의 의미와 의의를 전술한 바, 중복 설명은 생략한다.

[0051] 이상에서 대표적인 실시예를 통하여 본 발명을 상세하게 설명하였으나, 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자는 상술한 실시예에 대하여 본 발명의 범주에서 벗어나지 않는 한도 내에서 다양한 변형이 가능함을 이해할 것이다. 그러므로 본 발명의 권리 범위는 설명한 실시예에 국한되어 정해져서는 안 되며, 후술하는 특허청구범위뿐만 아니라 특허청구범위와 균등 개념으로부터 도출되는 모든 변경 또는 변형된 형태에 의하여 정해져야 한다.

부호의 설명

[0053] 1 : 영지식 증명 시스템

10 : 소스 코드 생성부

30 : 증명부

310 : 증명 셋업부

330 : 증명값 생성부

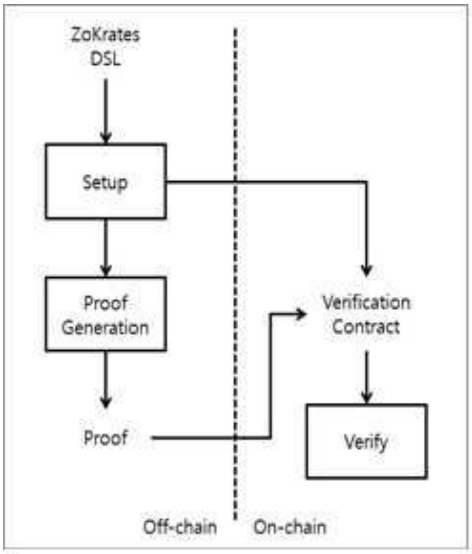
50 : 검증부

510 : 검증 셋업부

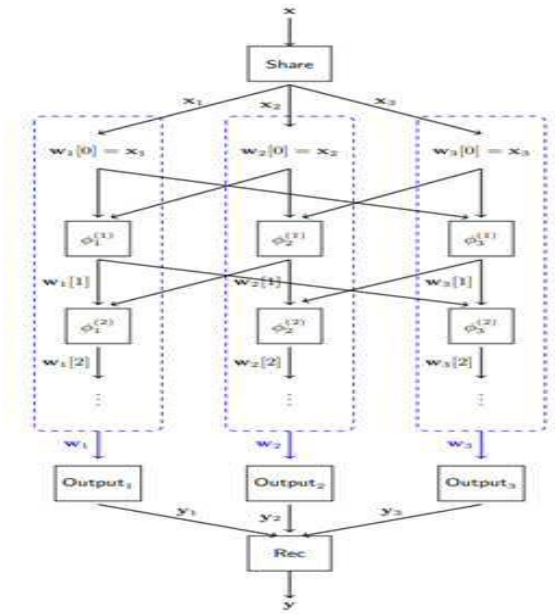
530 : 판단부

도면

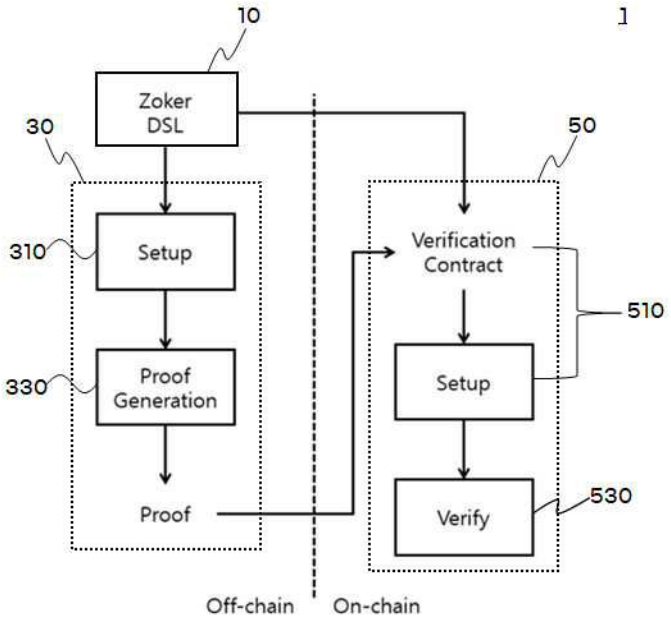
도면1



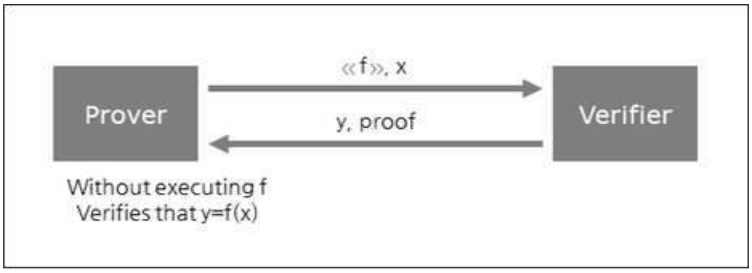
도면2



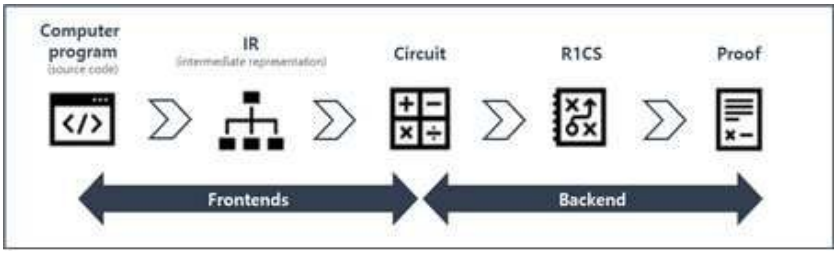
도면3


10
1
Zoker DSL
30
50
310
Setup
Verification Contract
510
330
Proof Generation
Setup
Proof
Verify
530
Off-chain
On-chain

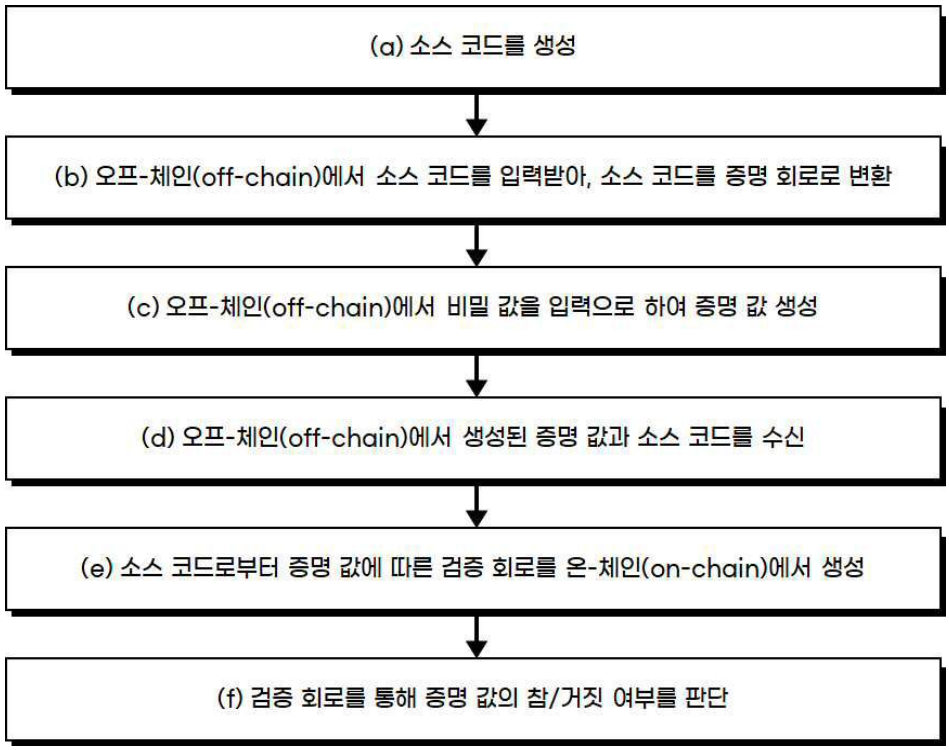
도면4


«f», x
Prover
y, proof
Verifier
Without executing f
Verifies that y=f(x)

도면5


Computer program
(source code)
IR
(intermediate representation)
Circuit
R1CS
Proof
Frontends
Backend

도면6


(a) 소스 코드를 생성
(b) 오프-체인(off-chain)에서 소스 코드를 입력받아, 소스 코드를 증명 회로로 변환
(c) 오프-체인(off-chain)에서 비밀 값을 입력으로 하여 증명 값 생성
(d) 오프-체인(off-chain)에서 생성된 증명 값과 소스 코드를 수신
(e) 소스 코드로부터 증명 값에 따른 검증 회로를 온-체인(on-chain)에서 생성
(f) 검증 회로를 통해 증명 값의 참/거짓 여부를 판단