



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2025-0105558
(43) 공개일자 2025년07월08일

(51) 국제특허분류(Int. Cl.)
G06Q 20/36 (2012.01) G06Q 20/38 (2012.01)
H04L 9/00 (2022.01) H04L 9/32 (2006.01)
(52) CPC특허분류
G06Q 20/3674 (2013.01)
G06Q 20/3678 (2013.01)
(21) 출원번호 10-2023-0198158
(22) 출원일자 2023년12월31일
심사청구일자 2023년12월31일

(71) 출원인
충남대학교산학협력단
대전광역시 유성구 대학로 99 (궁동, 충남대학교)
(72) 발명자
류재철
대전광역시 유성구 어은로 57, 132동 801호(어은동, 한빛아파트)
김근영
대전광역시 유성구 문화원로47번길 38, A동 403호(궁동, 스위트빌)
(74) 대리인
이은철, 김재문

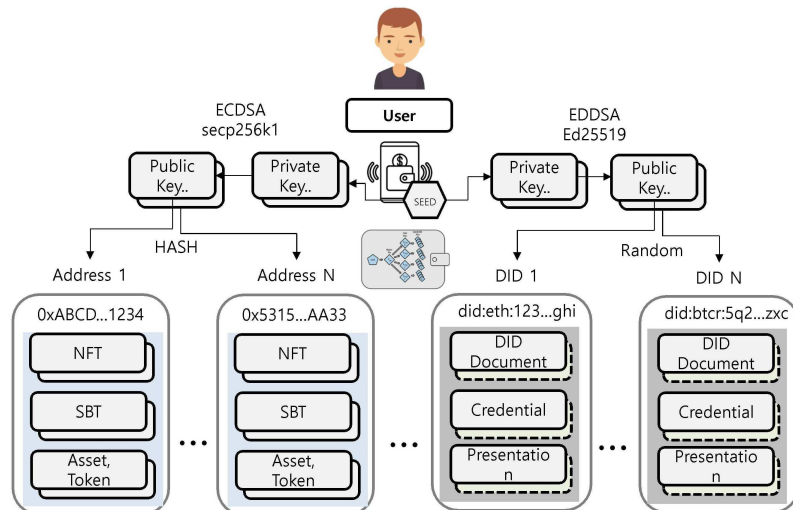
전체 청구항 수 : 총 7 항

(54) 발명의 명칭 DID와 Crypto를 함께 관리할 수 있는 디지털 지갑

(57) 요약

분산 아이디(DID)와 암호자산(Crypto)을 함께 관리할 수 있는 디지털 지갑에 관한 기술이 개시된다. DID와 Crypto를 함께 관리할 수 있는 디지털 지갑을 생성하는 방법은, 사용자의 시드(Seed)를 이용하여 암호화폐 지갑 주소에 사용되는 사용자의 개인키를 생성하고, 개인키에 곡선 알고리즘을 적용하여 공개키를 생성하고, 개인키와 공개키에 해시 알고리즘을 적용하여 암호화폐의 지갑주소를 생성하고, 동일한 시드(Seed)를 이용하여 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키를 생성하고, 개인키에 곡선 알고리즘을 적용하여 공개키를 생성하여 분산 아이디(DID)의 식별자를 생성한다. DID 크리덴셜과 SBT를 통합 시드 값으로 관리하는 디지털 지갑을 이용하여 사용자 편의성을 높일 수 있다.

대표도



(52) CPC특허분류

G06Q 20/3827 (2013.01)

G06Q 20/3829 (2013.01)

H04L 9/32 (2022.08)

H04L 9/50 (2022.05)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711195736
과제번호	00229400
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	정보보호핵심원천기술개발(R&D)
연구과제명	안전한 메타버스 환경을 위한 사용자 인증 및 프라이버시 보호 기술개발
기 여 율	1/1
과제수행기관명	한국전자통신연구원
연구기간	2023.04.01 ~ 2023.12.31

명세서

청구범위

청구항 1

분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑을 생성하는 방법에 있어서,
 사용자의 시드(Seed)를 이용하여 암호화폐 지갑 주소에 사용되는 사용자의 개인키를 생성하는 단계;
 암호화폐 지갑 주소에 사용되는 사용자의 개인키에 암호화폐에 사용되는 곡선 알고리즘을 적용하여 하나 이상의 암호화폐 지갑 주소에 사용되는 공개키를 생성하는 단계;
 암호화폐의 지갑주소에 사용되는 사용자의 개인키와 공개키에 기초하여 해시 알고리즘을 적용하여 암호화폐의 지갑 주소를 생성하는 단계;
 상기 사용자의 시드(Seed)와 동일한 시드(Seed)를 이용하여 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키를 생성하는 단계;
 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키에 분산 아이디(DID) 식별자에 사용되는 곡선 알고리즘을 적용하여 하나 이상의 분산 아이디(DID) 식별자에 사용되는 공개키를 생성하는 단계; 및
 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키와 공개키를 이용하여 분산 아이디(DID)의 식별자를 생성하는 단계;를 포함하는,
 분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑을 생성하는 방법.

청구항 2

제 1 항에 있어서,
 상기 암호화폐에 사용되는 곡선 알고리즘과 분산 아이디(DID) 식별자에 사용되는 곡선 알고리즘은 서로 다른 것을 특징으로 하는,
 분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑을 생성하는 방법.

청구항 3

제 1 항에 있어서,
 상기 암호화폐 지갑 주소에 사용되는 공개키를 생성하는 단계는,
 ECDSA의 secp256k1 타원 곡선 알고리즘을 사용하여 암호화폐 지갑 주소에 사용되는 공개키를 생성하는 것을 특징으로 하는,
 분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑을 생성하는 방법.

청구항 4

제 1 항에 있어서,
 상기 분산 아이디(DID) 식별자에 사용되는 공개키를 생성하는 단계는,
 EDDSA의 Ed25519 타원 곡선 알고리즘을 사용하여 분산 아이디(DID) 식별자에 사용되는 공개키를 생성하는 것을 특징으로 하는,

분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑을 생성하는 방법.

청구항 5

분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑에 있어서,

사용자의 시드(Seed);

상기 사용자의 시드(Seed)를 이용하여 생성된 암호화폐 지갑 주소에 사용되는 사용자의 개인키;

상기 암호화폐 지갑 주소에 사용되는 사용자의 개인키에 암호화폐에 사용되는 곡선 알고리즘을 적용하여 생성된 하나 이상의 암호화폐 지갑 주소에 사용되는 공개키;

상기 암호화폐 지갑 주소에 사용되는 사용자의 개인키와, 상기 암호화폐 지갑 주소에 사용되는 공개키에 기초하여 해시 알고리즘을 이용하여 생성된 하나 이상의 암호화폐 지갑 주소;

상기 사용자의 시드(Seed)를 생성된 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키;

상기 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키에 분산 아이디(DID) 식별자에 사용되는 곡선 알고리즘을 적용하여 생성된 하나 이상의 분산 아이디(DID) 식별자에 사용되는 공개키; 및

상기 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키와 상기 분산 아이디(DID) 식별자에 사용되는 공개키를 이용하여 생성된 분산 아이디(DID)의 식별자;를 포함하는,

분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑.

청구항 6

제 1 항에 있어서,

상기 사용자의 시드(Seed), 암호화폐 지갑 주소에 사용되는 사용자의 개인키와 공개키, 및 암호화폐 지갑 주소는, 계층 구조를 이루고,

상기 사용자의 시드(Seed), 분산 아이디(DID)의 식별자에 사용되는 사용자의 개인키와 공개키, 및 분산 아이디(DID)의 식별자는, 계층 구조를 이루는 것을 특징으로 하는,

분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑.

청구항 7

제 1 항에 있어서,

상기 암호화폐 지갑 주소에 사용되는 사용자의 개인키와, 상기 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키는, 서로 동일한 값을 가지는 것을 특징으로 하는,

분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑.

발명의 설명

기술 분야

[0001] 본 발명은 분산 아이디(DID)와 암호자산(Crypto)을 함께 관리할 수 있는 디지털 지갑에 관한 것으로, 특히 분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑을 생성하고 관리하는 방법에 관하여 개시한다.

배경 기술

[0002] 블록체인 기술을 활용한 암호화폐의 등장은 금융시스템에 탈중앙화와 혁신의 기회를 제공하지만 동시에 많은 문

제를 야기한다. 특히 금융시스템의 안전과 규제에 대한 우려가 커지고 있다. 가장 심각한 문제는 자금세탁방지 와 테러자금조달이다. 또한, 암호화폐는 국경이 없기 때문에 무제한적인 외환거래가 가능하며, 이는 자본유출로 이어져 국가경제의 지속가능성에 영향을 미칠 수 있다.

[0003] 암호화폐가 제공하는 익명성으로 인해 암호화폐 사용자의 실제 신원을 확인하는 것은 쉽지 않다. 이에 따라 암호화폐를 이용한 금융거래의 신뢰성과 안전성에 대한 우려가 제기되고 있으며, 이를 해결하기 위해서는 사용자 인증과 고객 확인(Know Your Customer; KYC) 프로세스가 필수적이다.

[0004] 특히, 블록체인 기술을 통해 운영되는 Web 3.0을 대표하는 메타버스 환경에서도 사용자 인증은 점점 더 중요한 요소로 자리잡고 있다. 이 경우 게임 아이템 획득 조작 등의 문제를 포함해 모든 이용자의 개인정보를 서버에서 관리해야 하는 단점이 있다.

[0005] 그러나 블록체인을 활용하는 탈중앙화 메타버스 시스템(예: Decentraland 및 Sandbox)의 경우 사용자는 암호화폐 지갑 소프트웨어(예: Metamask)를 사용하여 로그인하여 메타버스의 아바타와 매핑하고 지갑에 있는 NFT의 소유권을 통해 서비스를 제공한다. 다만, 암호화폐 지갑 소프트웨어가 사용자의 신원을 기반으로 NFT를 발행하는 대신 특정 항목의 소지 여부를 확인하는 데 의존하기 때문에 신원 인증에 적합하지 않다. 또한, NFT가 사용자를 인증하더라도 해당 정보는 여전히 블록체인에서 공개적으로 표시될 수 있으므로 잠재적으로 개인 정보 보호 문제가 발생할 수 있다.

[0006] 사용자 인증은 메타버스 환경뿐만 아니라 대부분의 Web 3.0 애플리케이션에서도 필수적이다. 금융 분야에서 중앙화된 거래소는 이미 사용자의 신원을 확인하기 위해 KYC 절차를 구현하고 있다. 또한 자금세탁 방지 노력에 대한 트래블 룰(Travel rule)과 같은 규정에서는 가상자산 서비스 제공업체(Virtual Asset Service Providers; VASP)가 보낸 사람과 받는 사람의 이름, 계좌 번호, 주소 등과 같은 정보를 수집하고 전송하도록 요구한다. 이러한 의무적 보고는 자금 세탁 및 테러 자금 조달과 관련된 위험을 완화하는 데 도움이 된다.

[0007] 마찬가지로, 탈중앙화 거래소 역시 사용자의 신원을 확인하기 위해 다양한 노력을 기울이고 있다. 예를 들어, 일부 거래소에서는 사용자의 신원을 인증하기 위해 분산 아이디(Decentralized Identity; DID) 자격 증명을 활용하고 있다. 신원인증에서는 지갑 주소와 연관된 사용자의 신원을 확인하는 것이 중요하다.

[0008] 사용자의 신원을 나타내기 위해 양도할 수 없는 NFT를 발행하는 소울바운드 토큰(Soul Bound Token; SBT)를 이용하여, 탈중앙화 사회 내에서 신원 확인 및 검증을 할 수 있다. 예를 들어, 세계 최대 암호화폐 거래소인 바이낸스는 SBT의 한 형태인 BABT(Binance Account Bound Token) 라는 신원 토큰을 발행했다. 이 토큰은 바이낸스의 KYC 프로세스를 완료한 사용자에게 발행되어 BNB 체인을 기반으로 다양한 프로젝트에 참여하고 그에 따른 보상을 받을 수 있다. 한편, SBT를 활용한 사용자 신원 관리를 진행하기 위해서는 실제 지갑 사용자가 SBT 발급을 원하는지 확인하는 것이 필수적이다.

[0009] 블록체인 생태계에서는 DID 또는 SBT를 이용해 사용자의 신원을 검증할 수 있다. 그러나, DID 검증은 블록체인 외부의 오프체인 환경에서 이루어져야 하고, 사용자가 자신의 신원을 증명할 때마다 자격증명을 제출해야 한다는 점이 단점이다.

[0010] 반면에, SBT를 사용하면 SBT 보유를 기반으로 블록체인 환경 내에서 온체인으로 신원 확인이 이루어질 수 있다. 그러나 SBT를 발행하려면 사용자가 SBT 발행자에게 개인정보를 제공해야 하므로 개인정보 보호에 대한 우려가 있다.

[0011] 2020년 07월 30일 공고된 등록특허 제10-2139645호는 "블록체인 기반의 신원증명 시스템 및 그 구동방법"에 관한 것이고, 2023년 11월 09일 공고된 등록특허 제10-2600516호는 "탈중앙화 신원증명 인증서를 이용한 웹사이트 자동 로그인 방법 및 장치"에 관한 것이다.

[0012] 다만, DID를 활용하여 스마트 계약을 통해 사용자를 검증한 후 SBT를 발행하여 Web 3.0 환경, 특히 메타버스에서 원활한 통합을 가능하게 하는 방식 및 DID와 SBT를 통합하여 관리하는 지갑에 관한 기술은 아직 구체적으로 개시되지 않고 있다.

선행기술문헌

특허문헌

[0013] (특허문헌 0001) KR 10-2139645 B1 (2020.07.30.)

(특허문헌 0002) KR 10-2600516 B1 (2023.11.09.)

발명의 내용

해결하려는 과제

- [0014] 본 발명의 일 목적은, 개인정보를 보호할 수 있으면서도 아바타 사용자의 신원을 인증하는 방법 및 시스템을 제공하는 것이다.
- [0015] 본 발명의 다른 일 목적은, 암호학적 증명을 직접 생성하지 않고도 신원 인증을 위한 암호화 증명을 생성할 수 있는 영지식 증명(ZKP)을 위한 도구를 제공하는 것이다.
- [0016] 본 발명의 또 다른 일 목적은, DID 크리덴셜과 SBT를 통합 시드 값으로 관리하는 지갑을 제공하는 것이다.
- [0017] 본 발명이 해결하고자 하는 과제들은 이상에서 언급한 과제들로 제한되지 않으며, 언급되지 않은 또 다른 과제들은 아래의 기재로부터 통상의 기술자에게 명확하게 이해될 수 있을 것이다.

과제의 해결 수단

- [0018] 제안된 발명의 일 양상에 따르면, 분산 아이디(DID)와 암호화패를 함께 관리할 수 있는 디지털 지갑을 생성하는 방법은, 사용자의 시드(Seed)를 이용하여 암호화패 지갑 주소에 사용되는 사용자의 개인키를 생성하는 단계와, 암호화패 지갑 주소에 사용되는 사용자의 개인키에 암호화패에 사용되는 곡선 알고리즘을 적용하여 하나 이상의 암호화패 지갑 주소에 사용되는 공개키를 생성하는 단계와, 암호화패의 지갑주소에 사용되는 사용자의 개인키와 공개키에 기초하여 해시 알고리즘을 적용하여 암호화패의 지갑 주소를 생성하는 단계와, 상기 사용자의 시드(Seed)와 동일한 시드(Seed)를 이용하여 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키를 생성하는 단계와, 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키에 분산 아이디(DID) 식별자에 사용되는 곡선 알고리즘을 적용하여 하나 이상의 분산 아이디(DID) 식별자에 사용되는 공개키를 생성하는 단계와, 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키와 공개키를 이용하여 분산 아이디(DID)의 식별자를 생성하는 단계를 포함한다.
- [0019] 추가적인 양상에 따르면, 상기 암호화패에 사용되는 곡선 알고리즘과 분산 아이디(DID) 식별자에 사용되는 곡선 알고리즘은 서로 다르다.
- [0020] 추가적인 양상에 따르면, 상기 암호화패 지갑 주소에 사용되는 공개키를 생성하는 단계는, ECDSA의 secp256k1 타원 곡선 알고리즘을 사용하여 암호화패 지갑 주소에 사용되는 공개키를 생성한다.
- [0021] 추가적인 양상에 따르면, 상기 분산 아이디(DID) 식별자에 사용되는 공개키를 생성하는 단계는, EDDSA의 Ed25519 타원 곡선 알고리즘을 사용하여 분산 아이디(DID) 식별자에 사용되는 공개키를 생성한다.
- [0022] 제안된 발명의 다른 일 양상에 따르면, 분산 아이디(DID)와 암호화패를 함께 관리할 수 있는 디지털 지갑은, 사용자의 시드(Seed)와, 상기 사용자의 시드(Seed)를 이용하여 생성된 암호화패 지갑 주소에 사용되는 사용자의 개인키와, 상기 암호화패 지갑 주소에 사용되는 사용자의 개인키에 암호화패에 사용되는 곡선 알고리즘을 적용하여 생성된 하나 이상의 암호화패 지갑 주소에 사용되는 공개키와, 상기 암호화패 지갑 주소에 사용되는 사용자의 개인키와, 상기 암호화패 지갑 주소에 사용되는 공개키에 기초하여 해시 알고리즘을 이용하여 생성된 하나 이상의 암호화패 지갑 주소와, 상기 사용자의 시드(Seed)를 생성된 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키와, 상기 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키에 분산 아이디(DID) 식별자에 사용되는 곡선 알고리즘을 적용하여 생성된 하나 이상의 분산 아이디(DID) 식별자에 사용되는 공개키와, 상기 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키와 상기 분산 아이디(DID) 식별자에 사용되는 공개키를 이용하여 생성된 분산 아이디(DID)의 식별자를 포함한다.
- [0023] 추가적인 양상에 따르면, 상기 사용자의 시드(Seed), 암호화패 지갑 주소에 사용되는 사용자의 개인키와 공개키, 및 암호화패 지갑 주소는, 계층 구조를 이룬다. 상기 사용자의 시드(Seed), 분산 아이디(DID)의 식별자에 사용되는 사용자의 개인키와 공개키, 및 분산 아이디(DID)의 식별자도, 계층 구조를 이룬다.
- [0024] 추가적인 양상에 따르면, 상기 암호화패 지갑 주소에 사용되는 사용자의 개인키와, 상기 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키는, 서로 동일한 값을 가진다.

발명의 효과

- [0025] 본 발명에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템은, 검증에 영지식 증명(ZKP) 기반 증명을 사용하여 서비스를 제공하는 VASP에게 중요한 사용자 정보가 알려지지 않으므로 개인정보를 보호할 수 있다.
- [0026] 나아가 본 발명에 의하면, 사용자는 영지식 증명(ZKP)을 위해 설계된 도구를 통해 암호학적 증명을 직접 생성하지 않고도 신원 인증을 위한 암호화 증명을 생성할 수 있어 사용자 편의성이 높아진다.
- [0027] 또한, DID 크리덴셜과 SBT를 통합 시드 값으로 관리하는 지갑을 제시하여 사용자 편의성을 높일 수 있다.

도면의 간단한 설명

- [0028] 도 1은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에 사용되는 크리덴셜의 주요 구성을 나타내는 예시도이다.
- 도 2는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에 사용할 수 있는 NFT의 구성을 나타내는 예시도이다.
- 도 3a는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 디지털 지갑을 이용한 메타버스 서비스의 로그인 절차를 개략적으로 나타내는 예시도이고, 도 3b는 보다 구체적인 절차를 나타내는 예시도이다.
- 도 4a는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템 중 Off-Chain에서 동작하는 방법 및 시스템을 나타내는 개념도이고, 도 4b는 On-Chain에서 동작하는 방법 및 시스템을 나타내는 개념도이고, 도 4c는 사용자의 디지털 지갑과 매핑된 아바타를 나타내는 개념도이다.
- 도 5는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 DID를 통한 크리덴셜 및 프리젠테이션 생성 과정을 나타내는 흐름도이다.
- 도 6은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 검증가능한 프리젠테이션과 그로부터 증명 및 검증 계약을 생성하는 것을 나타내는 개념도이다.
- 도 7은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 나이에 따라 생성된 서명 검증과 증명을 나타내는 예시도이다.
- 도 8은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 "greater than" 함수를 사용하여 증명을 생성하기 위한 Solidity 코드를 나타내는 예시도이다.
- 도 9는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 스마트 계약에 따라 SBT가 발행되는 과정을 나타내는 흐름도이다.
- 도 10은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 SBT 생성 계약 코드를 나타내는 예시도이다.
- 도 11a는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 디지털 지갑내 SBT를 나타내는 예시도이고, 도 11b는 SBT와 연결된 크리덴셜을 나타내는 JSON 코드를 나타내는 예시도이다.
- 도 12는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 생성된 SBT의 사용자 정보를 나타내는 예시도이다.
- 도 13은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 암호화페 지갑을 나타내는 개념도이다.
- 도 14는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 DID 지갑을 나타내는 개념도이다.
- 도 15a는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 HD 지갑을 통한 메타버스 지갑과 DID의 통합 관리 지갑을 나타내는 개념도이고, 도 15b는 Goerli Test에서 확인한 지갑 리스트를 나타내는 도면이다.

도 16은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 디지털 집합을 통해 메타버스의 아바타에 대한 권한을 관리하는 방법을 나타내는 흐름도이다.

도 17은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템을 다른 블록체인 신원인증 방법들과 비교한 결과를 나타내는 도면이다.

도 18은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 서비스 제공 서버의 구성을 나타내는 구성도이다.

도 19는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 사용자 단말의 구성을 나타내는 구성도이다.

발명을 실시하기 위한 구체적인 내용

- [0029] 전술한, 그리고 추가적인 양상들은 첨부된 도면을 참조하여 설명하는 실시예들을 통해 구체화된다. 각 실시예들의 구성 요소들은 다른 언급이나 상호간에 모순이 없는 한 실시예 내에서 또는 타 실시예의 구성 요소들과 다양한 조합이 가능한 것으로 이해된다. 발명자는 그 자신의 발명을 가장 최선의 방법으로 설명하기 위해 용어의 개념을 적절하게 정의할 수 있다는 원칙에 입각하여 본 명세서 및 청구범위에 사용된 용어는 기재 내용 혹은 제안된 기술적 사상에 부합하는 의미와 개념으로 해석되어야만 한다. 본 명세서에서 모듈 또는 부분은, 컴퓨터 또는 프로세서에서 실행할 수 있도록 메모리에 저장된 프로그램 명령어의 집합이거나, 이러한 명령들을 수행할 수 있도록 ASIC, FPGA 등의 전자 부품 또는 회로의 집합을 이용하여 구현할 수 있다. 또한, 각 모듈 또는 부분의 동작은 하나 또는 복수의 프로세서 또는 장치에 의해 수행될 수 있다. 동일·유사한 부호가 표시된 구성요소는 동일·유사한 기능을 수행하므로, 설명을 생략할 수 있다. 설명이 생략된 도면부호를 가진 구성요소에 대해서는, 동일·유사한 부호를 가진 구성요소에 대해 앞에서 설명한 내용을 참조할 수 있다.
- [0030] 이하 첨부된 도면을 참조하여 본 발명의 바람직한 실시예를 상세히 설명하기로 한다.
- [0031] 도 1은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에 사용되는 크리덴셜의 주요 구성을 나타내는 예시도이다.
- [0032] 분산 아이디(Decentralized Identity; DID)는 중앙 집중식 등록 메커니즘 없이 신뢰 분산 프레임워크 내에서 사용되는 신원 정보 시스템이다. 개인이 신원증명을 자율적으로 관리하고, 신원정보 제출 범위와 수신자를 통제할 수 있는 분산원장 기반 기술이다. 도 1과 같이 사용자 정보를 나타내는 부분을 클레임(110)이라고 한다. 이러한 클레임(110)은 수집되어 발급자가 서명한 후 자격 증명 형식으로 보유자에게 전송된다.
- [0033] 그런 다음 보유자는 크리덴셜(Credential; 자격 증명)(130)을 저장하고 특정 정보를 공개하고 싶을 때 정보와 자신의 서명을 추가하여 검증자에게 보내 사용자의 정보를 확인할 수 있다. 예를 들어, 숙박 시설에서 연령 확인을 요구하는 경우 직원에게 제시된 주민등록증에는 생년월일뿐만 아니라 주소, 주민등록번호, 이름 등 모든 개인 정보가 공개된다.
- [0034] 하지만 DID를 활용하면 정확한 나이를 밝히지 않고도 '19세 이상' 등을 증명하는 것이 가능하다. DID 기술은 사용자가 신원 확인에 필요한 정보만을 프리젠테이션(150)하면서 인증 데이터를 저장하고 관리할 수 있는 자기주권신원(Self-Sovereign Identity; SSI)의 원리를 사용한다는 점에서 주목을 받고 있다. 데이터 경제 시대에 DID 기술은 데이터 주권, 개인정보 보호 등의 이유로 다양한 인증 서비스에 널리 활용되고 있다. 다만, Web 3.0 시스템에서 DID를 사용하는 데에는 단점이 있다.
- [0035] 도 2는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에 사용할 수 있는 NFT의 구성을 나타내는 예시도이다.
- [0036] NFT는 "Non-Fungible Token"의 약자로 고유하고 대체할 수 없는 토큰을 나타낸다. 각 NFT에는 고유한 속성과 소유권 정보를 캡슐화하는 고유한 식별자가 있다. 이 데이터는 블록체인에 업로드되므로 위조나 변조가 어렵다. NFT는 Ethereum의 ERC-721 표준을 기반으로 성장했다. 이더리움은 스마트 계약 기능으로 가장 잘 알려진 블록체인 기술을 기반으로 구축된 분산형 플랫폼이다.
- [0037] 스마트 계약은 Turing-complete 프로그래밍 언어인 EVM(Ethereum Virtual Machine) 바이트코드의 인코딩된 세그먼트이다. 블록체인의 온체인 환경에는 소유자의 주소를 나타내는 "ownerOf" 함수와 NFT 이미지를 포함한 다양한 속성을 가진 JSON 파일에 대한 URL을 포함하는 "tokenURI" 함수가 있다.

- [0038] NFT의 실제 정보는 IPFS 또는 서버에 오프체인으로 저장된다. 이 기능을 활용하면 블록체인에서 특정 NFT의 ID(tokenID)를 이용하여 해당 NFT의 소유권과 데이터 정보를 검증하는 것이 가능하다. 도 2에서 볼 수 있듯이 NFT 마켓플레이스는 블록체인에 업로드된 데이터를 기반으로 NFT 정보를 쉽게 볼 수 있는 사용자 인터페이스를 제공한다. NFT는 주로 디지털 아트, 음악, 비디오, 게임 아이템과 같은 디지털 콘텐츠의 소유권을 나타내는 데 사용된다. 따라서 저작물로부터 소유자의 저작권을 보호하고, 디지털 자산의 권리를 투명하게 관리할 수 있는 장점을 제공한다.
- [0039] 도 3a는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 디지털 지갑을 이용한 메타버스 서비스의 로그인 절차를 개략적으로 나타내는 예시도이고, 도 3b는 보다 구체적인 절차를 나타내는 예시도이다.
- [0040] NFT 거래를 지원하는 블록체인 기반 메타버스 서비스는 도 3과 같이 운영할 수 있다. 먼저, 사용자 단말(User A)에서 서비스를 요청하면(S311), 서비스 제공자 서버(Decentraland)는 사용자의 지갑 주소를 요청한다(S312). 사용자 단말에서 메타마스크와 같은 디지털 지갑 프로그램에 로그인하고(S313), 지갑 주소와 서명을 서비스 제공자 서버로 전송하면(S320), 서비스 제공자 서버는 사용자의 지갑 주소를 확인하기 위해 서명값을 확인하는 과정을 거친다. 그 후 지갑 주소와 해당 아바타가 1:1로 매핑된다. 이로써 메타버스가 지원하는 블록체인 지갑 주소는 아바타가 로그인할 수 있는 ID 개념으로 확장될 수 있다. 그리고 사용자 지갑의 NFT목록을 체크하여(S330) 아바타의 권한을 설정한다. 예를 들어, 디센트럴랜드에서는 만 19세 이상 이용자가 홀덤 포커에 참여하기 위해서는 NFT로 발행된 아이스브레이커 캐릭터 의상이 필요하다. 사용자 인증은 특정 NFT 보유 여부와 상관없이 진행된다. 그러나 NFT는 소유자가 소유권을 이전할 수 있기 때문에 NFT로서 사용자 ID를 생성하는 데는 적합하지 않다.
- [0041] 따라서, 전통적인 NFT와 달리 양도나 거래가 불가능한 소울바운드 토큰(Soul Bound Token; SBT)을 도입하여, 메타버스와 같은 탈중앙화 사회(DeSoc)의 신원 증명이 가능해진다. 소울바운드 토큰(SBT)은 분산형 디지털 세계에서 개인의 신원을 나타내도록 설계되었다. SBT를 활용하면 양도 불가능한 NFT 배지 양식을 생성하여 고용, 봉사 활동, 학력, 자격, 의료 기록, 범죄 기록, 회원 상태, 소속 등 개인의 다양한 신원을 안전하게 확인할 수 있다. 그런 다음 이러한 배지는 사용자의 디지털 지갑 주소에 매핑되어 신원이 보존된다.
- [0042] SBT는 다양한 ERC 표준(ERC-5192, ERC-5727 및 ERC-6454)을 이용해 만들 수 있으며, 일반적으로 전송 불가능한 특성을 지닌 최소한의 기능과 인터페이스가 정의된다. 많은 서비스에서 SBT를 사용하여 신원 확인 시스템을 구현하려고 시도하고 있다.
- [0043] 한편, 영지식 증명(Zero Knowledge Proof; ZKP)은 증명자가 진술의 유효성을 제외하고 진술에 대한 세부 사항을 공개하지 않고 진술의 진실성을 검증자에게 입증할 수 있는 수단을 제공하는 중요한 암호화 개념이다. 이 암호화 프로세스에는 증명자와 검증자 간의 일련의 대화형 교환이 수반된다. 이러한 상호 작용 중에 증명자는 진술 자체의 기밀성과 프라이버시를 유지하면서 특정 정보를 공유함으로써 진술을 인증하려고 한다. 이러한 교환을 통해 검증자는 기본 정보를 손상시키지 않으면서 증명자의 주장의 진실성을 안전하게 확인할 수 있다.
- [0044] 일반적인 영지식 증명(ZKP) 프레임워크는 세 가지 단계로 구성된다. (1) 증인 단계(Witness Phase)에서는, 초기 단계에서 증명자는 진술을 바탕으로 증명을 구성하여 검증자에게 전송한다. (2) 챌린지 단계(Challenge Phase)에서는, 이후 검증자는 받은 증거를 기반으로 챌린지를 공식화하고 이를 증명자에게 전달한다. (3) 응답 단계(Response Phase)에서는, 마지막으로 증명자는 문제를 해결하고 응답을 검증자에게 다시 보낸다.
- [0045] 영지식 증명(ZKP)은 현대 암호화의 기본 도구로, 중요한 보안 및 개인 정보 보호 문제를 해결하는 능력으로 인해 상당한 주목을 받고 있다. 디지털 신원, 블라인드 서명, 속성 확인을 포함한 다양한 도메인에서 적용이 검토된다. ZKP는 블록체인 애플리케이션의 개인 정보 보호, 보안 및 효율성을 향상시키는 강력한 암호화 기술을 제공한다. 익명 거래 활성화, 보안 투표 시스템 촉진, 공급망 관리 개선 등 블록체인 컨텍스트에서 ZKP를 구현하기 위한 다양한 애플리케이션이 있다.
- [0046] 메타버스 시스템에서는 고객 확인(KYC) 절차를 위해 메타버스 서비스 제공자가 사용자 정보를 별도로 처리하여 신원을 확인할 수 있다. 확인된 ID를 기반으로 특정 권한을 부여할 때 중앙 집중식 방법을 사용하여 사용자 권한을 관리한다. 이 방식에서는 메타버스 서비스 제공자는 사용자 정보를 처리해야 하고 사용자는 자신의 정보를 각 메타버스 서비스 제공자에게 전송해야 하므로 중앙 집중식 데이터베이스와 관련된 해킹 위험이 발생할 수 있다.
- [0047] 시스템이 사용자 신원을 인증하기 위해 중앙화된 데이터베이스에 의존하지 않고 SBT를 발행하는 경우, 검증은

서비스 제공자가 사용자의 신원을 확인하는 것이 아닌 블록체인을 통해 투명하게 이루어져야 한다. 이용자의 정보는 ZKP나 암호화를 통해 보호될 수 있지만, 이용자가 정확한 정보를 제공했는지 여부를 확인하는 것은 어렵다. 이 문제를 해결하기 위해 DID를 사용하여 제공된 사용자 정보가 신뢰할 수 있는 발급자로부터 제공되었음을 증명한다.

- [0048] 또한 사용자가 신원 관련 SBT를 보유하고 나면 다양한 메타버스 서비스에 액세스할 때 지갑에 SBT가 있음을 증명하기만 하면 된다. 결과적으로 각각의 고유한 메타버스 서비스 공급자에게 자격 증명이나 ID 정보를 전송할 필요가 없다. 또한, SBT에 관한 세부 정보에는 사용자의 개인 데이터가 포함되지 않고, DID 자격증명을 관리하는 발급자만이 이 정보에 접근할 수 있다. 향후 문제가 발생할 경우 서비스 제공자는 사용자의 권한을 효과적으로 관리하기 위해 DID 정보를 요청할 수 있다. 이 접근 방식은 메타버스 시스템의 개인 정보 보호 및 보안을 강화한다.
- [0049] 도 4a는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템 중 Off-Chain 에서 동작하는 방법 및 시스템을 나타내는 개념도이고, 도 4b는 On-Chain 에서 동작하는 방법 및 시스템을 나타내는 개념도이고, 도 4c는 사용자의 디지털 지갑과 매핑된 아바타를 나타내는 개념도이다.
- [0050] 제안된 체계는 도 4에 표시된 대로 사용자 단말(420), 크리덴셜 발급자 서버(410), SBT 발급자 서버(450) 및 검증자 서버의 네 가지 개체로 구성될 수 있다. 검증자 서버는 SBT 발급자 서버 또는 블록체인 상의 다른 공인된 서버일 수 있다. 한편 서비스 제공자 서버는 SBT 발급자 서버 또는 검증자 서버의 역할을 할 수 있다.
- [0051] 사용자 단말(420)은 DID 자격증명을 관리하는 지갑(430)과 동일한 시드값을 갖는 지갑 주소(435)를 통해 메타버스 아바타(480)와 매핑된다. SBT 발행자(450)는 사용자의 신원을 관리하고 개인 자격 증명을 처리하며 메타버스, DeFi 또는 DEX에서 SBT 발행을 담당하는 정부 부서일 수 있다. 검증자(서비스 제공자)는 특정 SBT를 보유한 사용자에게 서비스 권한을 부여하기 위해 스마트 계약을 활용한다.
- [0052] 구체적인 단계를 살펴보면 다음과 같다.
- [0053] (1) 사용자는 신분증 발급자에게 자신의 DID를 제공하고, 신분증 정보 제공과 유사하게 KYC 절차에 필요한 정보를 제출한다(S410). 사용자 정보가 확인되면 발급사는 신분증과 유사하게 자격과 신원을 입증할 수 있는 크리덴셜을 전달한다(S420).
- [0054] (2) 사용자는 자격 증명을 지갑에 저장한다.
- [0055] (3) 서비스 제공자 서버(455)는 특정 권한으로 SBT를 확인하고 생성하기 위해 스마트 계약(460)을 생성한 다음 이 계약을 이더리움 네트워크(470)에 배포한다(S440). 이때, 서비스 제공자(455)는 영지식 증명 도구(440)를 이용하여 영지식 증명(ZKP)에 기반하는 스마트 계약을 생성할 수 있다.
- [0056] (4) 사용자(420)는 크리덴셜에 서명을 추가하고 (3) 단계에서 설정된 조건을 충족하는 증거를 생성하여 프리젠테이션을 생성한다(S430). 이때 사용자 단말(420)에서 영지식 증명 도구(440)를 이용하여 영지식 증명(ZKP)에 기반하는 증명(Proof)을 생성할 수 있다.
- [0057] (5) 스마트 계약은 사용자가 제공한 증명을 검증하고(S450), 조건이 충족되면 SBT를 사용자의 지갑 주소로 발행한다(S460).
- [0058] (6) 사용자는 SBT가 저장된 지갑 주소를 사용하여 아바타(480)에 로그인한다(S470).
- [0059] (7) 특정 서비스에서는 SBT 존재 여부에 따라 이용 권한을 부여한다.
- [0060] 제안된 발명의 일 양상에 따르면, 분산 아이디(DID)와 소울바운드 토큰(SBT)을 결합한 메타버스에서의 아바타 신원인증 방법은, 서비스 제공자 서버가 특정 권한에 대한 증명이 검증되는 경우 소울바운드 토큰(SBT)을 발행하는 스마트 계약을 생성하는 단계(S440)와, 서비스 제공자 서버가 생성된 스마트 계약을 블록체인 네트워크에 배포하는 단계와, 사용자 단말이 분산 아이디(DID)를 포함하는 크리덴셜에 기초하여 증명을 생성하고 사용자의 서명을 추가하여 프리젠테이션을 생성하는 단계(S430)와, 사용자 단말이 블록체인 네트워크에서 발행된 소울바운드 토큰(SBT)이 저장된 지갑 주소를 아바타와 매핑하는 단계(S470)를 포함한다.
- [0061] 추가적인 양상에 따르면, DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법은, 사용자 단말이 사용자의 분산 아이디(DID) 및 고객확인(KYC)에 필요한 정보를 크리덴셜 발행자 서버로 전달하는 단계(S410)와, 사용자 단말이 크리덴셜 발행자 서버로부터 전달받은 크리덴셜을 지갑에 저장하는 단계(S420)를 더 포함한다.

[0062] 추가적인 양상에 따르면, DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법은, SBT 발행자 서버가 사용자 단말로부터 전달된 증명에 기초하여 스마트 계약의 특정 권한에 대한 증명을 검증하는 단계(S450)와, SBT 발행자 서버가 특정 권한에 대한 증명이 검증된 경우 소울바운드 토큰(SBT)을 사용자 지갑으로 발행하는 단계(S460)를 더 포함한다.

[0063] 추가적인 양상에 따르면, DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법은, 서비스 제공자 서버가 아바타와 매핑된 지갑 주소의 소울바운드 토큰(SBT)을 확인하는 단계와, 서비스 제공자 서버가 소울바운드 토큰(SBT)의 존재 여부에 기초하여 아바타의 이용 권한을 설정하는 단계를 더 포함한다.

[0064] 도 5는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 DID를 통한 크리덴셜 및 프리젠테이션 생성 과정을 나타내는 흐름도이다.

[0065] DID는 그림 5와 같이 사용자의 신원을 확인하는 데 사용된다. 사용자의 분산 아이디(DID)를 기반으로 발급자는 자격 증명을 생성한다. 사용자(420)는 DID를 자체 생성하고 DID 문서를 블록체인(470)으로 업로드 한다(S510). 사용자(420)가 크리덴셜 발급자(410)에게 DID를 전달하고 본인 인증을 요청하면(S520), 크리덴셜 발급자(410)는 검증가능한 크리덴셜을 생성하여 사용자에게 전달하고(S530), 사용자(420)는 검증가능한 프리젠테이션을 자체 생성한다(S540). 이 과정에서 인증에 사용된 이용자의 개인정보는 자격증명(크리덴셜) 내의 자격증명(크리덴셜)번호를 포함하여 데이터베이스에 저장되어 이용자에게 제공된다.

[0066] 다음 표 1은 검증가능한 크리덴셜의 구조를 나타낸다.

표 1

[0067]

Verifiable credential { credential DID, issuer DID, issuance date, (claim 1, claim 2, ?, claim n), proof(issuer's signature) }

[0068] 발급자가 발급한 자격 증명은 메타버스 내에서 사용자의 신원을 나타내야 한다. 따라서 표 1과 같이 주민등록증, 운전면허증, 졸업장, 기타 관련 서류 등 이용자를 식별하는 다양한 항목이 포함될 수 있다. 사용자(420)는 사용자(크리덴셜 보유자)의 서명을 Credential에 포함시켜 검증 가능한 프레젠테이션을 생성한다.

[0069] 다음 표 2는 검증 가능한 프리젠테이션의 구조를 나타낸다.

표 2

[0070]

Verifiable presentation { presentation DID, presentation metadata, verifiable credential, proof(holder's signature) }

[0071] DID를 사용하여 사용자의 자격 증명이나 개인 정보를 자격 증명 및 프레젠테이션에 매핑할 때 불필요한 세부 정보를 포함한 모든 정보가 공개되는 개인 정보 보호 문제가 있다. 이를 해결하기 위해서는 이용자가 꼭 필요한 정보만 공개하는 선택적 공개 방식을 채택하는 것이 필수적이다.

[0072] 이를 위해 CL 서명(CL signature)을 사용할 수 있다. CL 서명은 강력한 RSA 가정 하에서 확률적으로 안전한 것으로 간주될 수 있다. 서명 검증 중에 숨기려는 정보에 대해 시그마 프로토콜(Sigma protocol)을 사용하여 그 값을 계산할 수 있다.

[0073] 도 6은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 검증가능한 프리젠테이션과 그로부터 증명 및 검증 계약을 생성하는 것을 나타내는 개념도이다.

[0074] 영지식 도구(440)를 사용하면 사용자는 프레젠테이션을 사용하여 ZKP를 통해 증거를 생성하고 이러한 증거를 확인하기 위한 스마트 계약을 자동으로 생성할 수 있다. 이 도구는 피노키오(Pinocchio) 알고리즘을 활용하여 ZKP

구현을 위한 ZK-SNARK를 기반으로 스마트 계약을 생성한다. ZKP를 처리하기 위해 ZoKrates 도구를 사용하여 사용자 증명 및 스마트 계약을 생성한다.

- [0075] 도 6을 참조하면, 해당 도구를 통해 증명을 생성하는 과정은 다음과 같다.
- [0076] (1) ZKP 도구는 사용자의 프리젠테이션(610)을 입력으로 받아 분석한 후 추가 처리를 위해 필드와 기호 구성 요소를 분리한다. 프리젠테이션(610)은 프리젠테이션 DID(620)와, 프리젠테이션 메타데이터(630)와, 검증가능한 크리덴셜(640) 및 증명(Proof)을 포함한다.
- [0077] (2) 검증인은 DID 서명(CL 서명) 검증, 연령 검증(19세 이상)을 위한 "보다 큼" 연산 구현, "같음" 연산 실행 등의 작업을 수행한다.
- [0078] (3) ZoKrates 라이브러리는 증명 및 검증 기능을 구현하는 데 사용된다.
- [0079] (4) 생성자는 사용자에게 프레젠테이션의 증명 가치를 제공한다. 검증자는 스마트 계약에서 제공하는 검증 기능을 사용하여 사용자 증명의 참/거짓 값을 확인한다.
- [0080] 추가적인 양상에 따르면, 상기 스마트 계약을 생성하는 단계는, 영지식 증명(ZKP)에 기초하여 사용자의 증명을 검증하는 계약(690)을 생성하는 단계를 포함한다.
- [0081] 추가적인 양상에 따르면, DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법은, 사용자 단말이 프리젠테이션으로부터 영지식 증명(ZKP)에 기초하여 사용자의 증명(670)을 생성하는 단계를 더 포함한다.
- [0082] 제안된 발명의 또 다른 양상에 따르면, 분산 아이디(DID)와 소울바운드 토큰(SBT)을 결합한 메타버스에서의 아바타 신원인증 방법을 컴퓨터에서 실행시키기 위해 기록매체에 저장된 프로그램은, 분산 아이디(DID)를 포함하는 크리덴셜에 기초하여 생성된 프리젠테이션을 입력받아 파싱하여 필드부분과 서명부분을 분리하고, 분산 아이디(DID)의 서명 검증 연산 및 스마트 계약의 특정 권한을 검증하는 연산을 생성하고, 영지식 증명(ZKP)에 기초하여 사용자의 권한에 대한 증명을 생성하는 함수(660)와 사용자의 권한에 대한 증명을 검증하는 함수(680)를 생성하고, 사용자의 권한에 대한 증명이 검증되는 경우 소울바운드 토큰(SBT)을 발행하는 함수를 생성한다.
- [0083] 도 7은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 나이에 따라 생성된 서명 검증과 증명을 나타내는 예시도이다.
- [0084] 서명검증과 연령을 통해 생성된 증명은 도 7과 같다. 유효한 증명이 제공되면 이는 참 값을 반영하여 사용자의 값을 검증할 수 있다. 스마트 컨트랙트에서 값을 검증하기 위한 함수 호출 시 "_"로 시작하는 함수가 호출되며, 회로(circuit) 생성 시 호출되는 함수의 회로가 추가된다.
- [0085] 검증 기능이 실행되면 증명의 진위 여부를 판단하게 된다. 증명이 참이면 'out' 값에서 수신된 결과를 바로 반환하므로 일반 함수를 호출하는 것과 유사한 흐름이 가능하다. 이를 통해 원활한 활용이 가능하다. 증명이 거짓이면 함수는 실행을 취소하기 위해 Assert 문을 실행한다.
- [0086] 도 8은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 "greater than" 함수를 사용하여 증명을 생성하기 위한 Solidity 코드를 나타내는 예시도이다.
- [0087] 도 8의 (a)는 구조를 나타내고, 도 8의 (b)는 계약을 체크하는 함수를 나타낸다. 이러한 함수는 영지식 도구(440)에 의해 자동으로 생성될 수 있다. 증명 생성 함수는 증명 값을 기반으로 한 "보다 큼" 작업에 대한 계산과 관련된다. 이 도구를 사용하면 사용자는 지갑에 저장된 자격 증명(JSON 형식)을 입력할 수 있다. 그 후, 사용자는 계약 확인 절차를 호출하여 생성하려는 증명 유형을 선택할 수 있다.
- [0088] 나중에 사용자가 특정 ZKP(예: 19세 이상)를 요청하면 도구는 이 정보를 구문 분석하고 해당 증명을 생성한다. 생성된 증명을 통해 사용자는 아래 verifyProof 엔티티로부터 검증을 받고 SBT 발행을 받을 수 있다. 따라서 사용자는 ZKP 암호화 알고리즘의 세부 사항을 알 필요 없이 지갑과 계약서에 있는 자격 증명만을 사용하여 원하는 값에 대한 증명을 생성할 수 있어 편의성을 높일 수 있다.
- [0089] 도 9는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 스마트 계약에 따라 SBT가 발행되는 과정을 나타내는 흐름도이다.
- [0090] 계약의 운영 프로세스는 도 9에 나와 있다. zok-verify 계약은 사용자의 프레젠테이션 증명을 확인한 다음 계약을 통해 SBT를 발행한다. 지정된 SBT 발행자만이 SBT를 발행할 수 있으며, 사용자의 증명 값이 true인 경우에만 작동한다. SBT 계약의 경우 OpenZeppelin의 ERC721 표준을 사용하여 SBT를 생성하고, 다른 사람에게 양도할 수

없도록 "양도할 수 없음" 조건을 포함한다.

[0091] 도 9를 참조하면, 서비스 제공자 서버(455)는 블록체인으로 증명 검증 계약을 업로드하고, SBT계약을 생성한다(S910). 크리덴셜 보유자인 사용자 단말(425)은 증명 검증 계약을 읽고(S920), 검증가능한 프리젠테이션을 생성하고(S930), 영지식 도구를 이용하여 증명을 생성한다(S940). 생성된 증명을 이용하여 블록체인 상의 스마트 계약을 증명한다(S960). 증명이 완료되면, 블록체인의 스마트 계약에 의해 SBT가 발행된다(S960). 서비스 제공자 서버(455)는 사용자 지갑에서 SBT를 확인할 수 있다(S970).

[0092] 도 10은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 SBT 생성 계약 코드를 나타내는 예시도이다.

[0093] 스마트 계약의 코드는 도 10에 도시되어 있다. 컨트랙트는 사용자의 증명을 검증하는 `zok-verify`와 SBT를 발행하는 `SBTCreator`의 두 컨트랙트로 구성된다. `zok-verify` 컨트랙트에서는 `verifyProof_sendTX()`를 먼저 호출해야 하며, 증명이 `true`인 경우에만 `SBTCreator` 컨트랙트의 `verifyProof()`를 호출할 수 있다. 이러한 요건을 충족해야만 SBT 발행이 가능하다.

[0094] 표 3은 검증 계약 절차의 `verifyProof_sendTX()` 함수가 수행하는 제1 알고리즘을 나타낸다.

표 3

[0095]	Algorithm 1. <code>verifyProof_sendTX()</code> [Verify.Contract] Input: calldata proof, calldata input, calldata issuer, calldata user, uint numOfCred If (<code>verify(proof, input) == true</code>) <code>SBTContract.verifyProof(msg.sender, issuer, holder, numOfCred)</code> ▷ ZoKrates를 사용하여 프리젠테이션 증명 확인 프리젠테이션 증명이 참인 경우 <code>SBTCreatContract</code>를 호출 <code>duplicatedProof[numOfProof] ← proof</code> ▷ 중복확인을 위한 List 저장
--------	---

[0096] 제1 알고리즘에서는 계약 확인 절차 내의 `verifyProof_sendTX()` 함수가 사용자의 제시 증명을 확인한다. 이 기능은 섹션 3_4에서 생성된 증명과 프리젠테이션에 있는 발급자와 사용자의 DID, 자격 증명 번호를 입력으로 제공하여 호출된다. 확인 기능은 ZoKrates 플러그인을 사용하여 생성된다. 이러한 맥락에서 사용자가 제출한 증명 값이 `true`인 경우 `SBTContract` 내의 `verifyProof()` 함수는 프리젠테이션 증명이 이미 성공적으로 검증되었음을 확인한다. 다른 사용자가 동일한 증명 값을 사용할 가능성을 방지하기 위해 `DuplicateProof` 배열을 사용하여 증명 값을 저장한다.

[0097] 이 계약에서는 ZoKrates에서 사용되는 ZK-SNARK가 사용된다. 증명을 생성하는 데에는 상대적으로 비용이 많이 들지만, 검증 과정에서는 $O(1)$ 복잡도를 유지한다. 따라서 스마트 계약 내에서 중요한 계산이 이루어지더라도 솔루션은 항상 일정한 시간에 해결되므로 GAS 오버헤드가 최소화된다. Solidity 버전 ^0.8.0을 사용하면 `verifyProof_sendTX()` 함수 배포 비용은 GAS 1,336,311이고, 검증 함수 호출에는 GAS 227,578이 소비된다. 현재 3개의 증명을 검증하고 있으며, 증명 수를 늘려도 GAS 비용은 GAS 227,578로 유지된다. `SBTContract`에서는 이전 계약 확인 절차에서 호출된 `verifyProof`를 사용하여 인증Addr에 사용자 주소를 등록한다.

[0098] 추가적인 양상에 따르면, 상기 사용자의 증명을 검증하는 계약을 생성하는 단계는, 사용자 제시 증명이 검증된 경우에 소울바운드 토큰(SBT) 발행 계약의 증명 검증 함수를 호출하는 함수를 생성하는 단계를 포함한다.

[0099] 표 4는 SBT 계약의 `verifyProof` 함수가 수행하는 제2 알고리즘을 나타낸다.

표 4

[0100]

Algorithm 2. verifyProof [SBTContract]
Input: string tokenurl, address user, string issuer, string user, uint numOfCred onlyVerify.Contract ▷ Verify Contract에서만 이 함수를 실행할 수 있음. authenticationAddr[user] ← True ▷ Verify Contract로부터 Proof 인증 받은 사용자임을 나타냄. Vp ← push(VerifiedPresentation(addr, issuer, user, numOfCred) ▷ 감사를 위해 vpList에 정보 추가. authCount[addr] ← vp.length ▷ 몇 번째로 인증된 사용자임을 관리.

[0101]

제2 알고리즘 2에서 verifyProof 함수는 계약 확인 절차 내에서만 호출될 수 있다. 향후 감사를 위해 자격 증명
의 확인된 정보는 관리 목적으로 vp 배열에 저장된다.

[0102]

추가적인 양상에 따르면, 상기 스마트 계약을 생성하는 단계는, 사용자의 증명을 검증하는 계약의 함수에서만
호출 가능한, 검증된 사용자의 크리덴셜 정보를 저장하는 함수를 생성하는 단계를 포함한다.

[0103]

표 5는 createSBT 함수가 수행하는 제3 알고리즘을 나타낸다.

표 5

[0104]

Algorithm 3. createSBT()
Input: string tokenurl, address user onlyOwner() ▷ 운영자만 SBT를 발급할 수 있음. require(authenticationAddr[user], "Not Authentication person"); ▷ Proof 검증 받은 사람만 SBT 발급 가능. require(sbtToPerson[user], "already exist"); ▷ 한 아이디당 SBT는 하나만 발급 가능. tokenIds.increment(); uint256 newItemId = tokenIds.current(); _mint(user , newItemId); ▷ user에게 SBT번호를 발급하여 minting _setTokenURI(newItemId , tokenURI) ; ▷ SBT와 Token URI 설정 sbtToPerson[user] = true; ▷ SBT 발급 address 추가 sbturlToPerson[user] = tokenURI; ▷ 빠른 검색을 위한 tokenURI 추가 sbtNumber[user] = newItemId; ▷ 토큰 번호 검색 Output: newItemId;

[0105]

createSBT() 함수는 SBT 발행을 담당한다. 이 기능은 발급자만이 접근할 수 있으며, 이전에 authenticationAddr에
등록된 증명을 검증받은 사람에게만 발급이 가능하다. 또한 주소당 하나의 SBT만 발급할 수 있다. SBT는
ERC721_sol을 통해 mint 함수를 호출하여 생성된다. 발급자는 tokenURI를 생성하고 이를 사용하여 SBT를 생성한
다.

[0106]

추가적인 양상에 따르면, 상기 스마트 계약을 생성하는 단계는, 소울바운드 토큰(SBT)의 발급자만 호출 가능한,
검증된 사용자의 지갑 주소당 하나의 소울바운드 토큰(SBT)을 발행하는 함수를 생성하는 단계를 포함한다.

[0107]

도 11a는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 디지털 지
갑내 SBT를 나타내는 예시도이고, 도 11b는 SBT와 연결된 크리덴셜을 나타내는 JSON 코드를 나타내는 예시도이
다.

[0108]

제3 알고리즘의 createSBT() 함수에 의해 생성된 SBT는 사용자 단말에서 도 11a와 같이 표시될 수 있다. 도 11a
의 SBT는 도 11b의 JSON 구조로 표시될 수 있다.

[0109]

이후에는 SBT 관리를 용이하게 하는 기능이 구현된다. 여기에는 SBT 발급 상태를 확인하기 위한 주소 검색, 관
련 tokenURI 정보 및 토큰 번호가 포함된다.

[0110] 표 6은 SBT 계약의 transferFrom 함수가 수행하는 제4 알고리즘을 나타낸다.

표 6

[0111]

Algorithm 4. transferFrom(_form, _to, _tokenId)
Input: address _form, address _to, uint256 _tokenId
require(cannotTransfer == false, "You cannot transfer this SBT")
▷ 발급된 SBT는 다른사람에게 전송 불가.

[0112] ERC-1138 및 ERC-5192와 같은 전송 불가능한 SBT에 대한 사양이 개발 중이지만 알고리즘 4에 설명된 대로 transferFrom 작업을 근본적으로 불가능하게 만드는 방식으로 코드를 구현한다. 결과적으로 SBTContract를 통해 생성된 SBT에 대해서는 거래가 불가능하다.

[0113] 추가적인 양상에 따르면, 상기 스마트 계약을 생성하는 단계는, 발행되는 소울바운드 토큰(SBT)의 전송을 불가능하게 설정하는 함수를 생성하는 단계를 포함한다.

[0114] 표 7은 SBT 계약의 updateURL 함수가 수행하는 제5 알고리즘을 나타낸다.

표 7

[0115]

Algorithm 5. updateURL()
Input: string tokenurl, string tokenURI
onlyOwner() ▷ 운영자만 SBT를 업데이트 할 수 있음
_setTokenURI(sbtNumber[user], tokenURI)
▷ sbt의 tokenURI를 변경함으로써 안의 내용을 수정.

[0116] updateURL() 함수는 발급자만 호출할 수 있으며 사용자 요청에 따라 이전에 생성된 SBT에 대한 tokenURI를 수정할 수 있도록 하기 위한 것이다. 이 기능을 사용하면 사용자 지갑 주소에 있는 SBT가 취소와 유사한 기능을 부여받을 수 있다.

[0117] 추가적인 양상에 따르면, 상기 스마트 계약을 생성하는 단계는, 소울바운드 토큰(SBT)의 발급자만 호출 가능한, 발행된 소울바운드 토큰(SBT)의 주소를 변경하는 함수를 생성하는 단계를 포함한다.

[0118] 도 12는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 생성된 SBT의 사용자 정보를 나타내는 예시도이다.

[0119] 블록체인은 소유자, 발급자 및 TokenURL 정보를 저장하는 반면, External_url의 메타데이터는 발급자의 서버에 저장된다. 개인정보를 제외한 사용자 정보가 노출되어 개인정보 보호 문제를 해결한다. 앞으로는 특정 SBT를 보유한 개인이 범죄 행위에 가담할 경우 해당 크리덴셜을 발급한 발급자로부터 정보를 얻는 것이 가능해진다. 도 12에 도시된 것처럼 "attributes" 중 "kycNumber"를 사용하면 발급된 자격 증명의 세부 정보에 액세스하고 검토할 수 있다.

[0120] 도 13은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 암호화페 지갑을 나타내는 개념도이다.

[0121] 도 13은 암호화페와 NFT를 보관하는 지갑인 크립토 지갑(Crypto Wallet)을 나타낸다. Crypto Wallet은 메타마스크에서 사용하는 API를 활용하여 생성할 수 있다. 키 생성 알고리즘으로는 BIP-44를 사용할 수 있다. BIP-44의 경우에는 HD(Hierarchical Deterministic) 기술을 활용하여 시드(SEED) 값에서 다양한 하위 개인키(Private key)를 유도해 낼 수 있다. 이때 ECDSA의 secp256k1 타원 곡선을 활용하여 개인키와 대응되는 공개키(Public key)를 생성할 수 있다. 개인키와 공개키에 기초하여 해시(HASH)를 통해 크립토 지갑의 주소(Address)를 생성할 수 있다. 지갑 주소를 검색하여, 사용자가 보관하고 있는 비트코인, 이더리움, NFT 등의 보유 여부를 확인할 수 있다. 다만, 익명성을 보장하기 때문에 지갑 주소를 통해 사용자를 특정할 수 없는 문제가 있다. 즉, 크립토 지갑 주소 만으로는 사용자 인증을 할 수 없다.

[0122] 도 14는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 DID 지갑을 나타내는 개념도이다.

- [0123] 도 14는 DID 지갑을 나타낸다. DID 지갑의 경우에도 개인키와 공개키가 존재하고, 각각의 공개키를 통해 DID를 생성할 수 있다. 이후 DID와 매핑되는 DID 문서(DID Document), 사용자의 정보를 나타내는 크리덴셜(Credential), 사용자가 검증자(Verifier)에게 제출하는 프리젠테이션(Presentation) 등이 담길 수 있다. DID 주소만으로 사용자를 특정하기는 어렵지만, 크리덴셜 안에 사용자의 정보가 들어있기 때문에, 크리덴셜과 프리젠테이션을 통해 사용자 인증을 할 수 있다. 다만, DID 지갑에서도 각각의 개인키를 관리하기 어려운 문제가 있다. 탈중앙화 생태계에서 사용하는 ID이지만, 블록체인 내의 스마트 계약과 호환하기에는 어려움이 존재한다. 예를 들어, DID 또는 크리덴셜을 블록체인 내에서 검증하기 어렵고, 블록체인에서는 모든 정보가 공개되기 때문에 프라이버시 문제가 있으며, 사용자를 인증할 때마다 크리덴셜에서 프리젠테이션을 만들어 제출해야 한다. 따라서 DID는 탈중앙화 생태계보다는 중앙화된 생태계에서 주로 사용되고 있다.
- [0124] 따라서, 사용자의 신원을 블록체인에서 나타낼 수 있고, 사용자 입장에서는 크립토 지갑과 DID 지갑을 따로 관리할 필요 없이 하나의 시드(Seed) 값을 통해 DID와 Crypto를 모두 관리하는 통합 지갑의 필요성이 증대되고 있다.
- [0125] 도 15a는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 HD 지갑을 통한 메타버스 지갑과 DID의 통합 관리 지갑을 나타내는 개념도이고, 도 15b는 Goerli Test에서 확인한 지갑 리스트를 나타내는 도면이다.
- [0126] 사용자가 증명을 확인한 후 받은 SBT를 보관하는 지갑은 도 15a와 같다. 기존 암호화폐 지갑을 사용하여 SBT를 저장하는 것이 가능하지만, SBT를 획득하는 데 사용된 프레젠테이션 및 자격 증명은 사용자에게 의해 다른 지갑에 안전하게 저장된다. 본 발명에서는 SBT의 보관을 용이하게 할 뿐만 아니라 암호화폐 지갑을 모두 한곳에서 통합하고 관리할 수 있는 지갑을 제안한다.
- [0127] 암호화폐 지갑의 경우 Metamask API를 사용하여 구현할 수 있다. DID 지갑의 경우 W3C의 다가오는 Universal Wallet 2020 표준에 설명된 요구 사항을 따른다. 지갑은 BIP-44를 사용하는 HD(Hierarchical Deterministic) 지갑 구조를 사용한다. 키 파생 알고리즘을 사용하면 단일 시드 값에서 다양한 하위 수준 개인 키를 파생할 수 있다. 이 과정에서 ECDSA 알고리즘의 secp256k1 곡선을 활용하여 개인키에 대응하는 공개키를 생성한다. 그리고, 해싱을 통해 암호화폐 지갑 주소를 생성한다.
- [0128] 마찬가지로 DID의 경우 시드값에서 개인키를 도출하고, 해당 공개키 생성에는 EdDSA 알고리즘의 Ed25519 곡선을 사용한다. 이 프로세스를 통해 DID별 식별자가 생성되고, 이는 DID 자체를 생성하는 데 사용된다.
- [0129] 시드값을 보유하는 단일 지갑을 사용하여 메타버스 지갑과 DID 지갑을 모두 통합하여 관리한다.
- [0130] HD 지갑 구조는 이러한 작업을 위해 BIP-44 알고리즘을 따른다. 사용자는 메타마스크 내에서 블록체인 네트워크를 선택할 수 있다. 메타버스 지갑 주소에는 ERC-721 NFT, ERC-20 토큰, 자산, SBT 등 다양한 아이템을 저장할 수 있다.
- [0131] DID 지갑에서 사용자는 DID 문서에 접근할 수 있을 뿐만 아니라 해당 DID로 발급된 자격 증명도 확인할 수 있다. 또한 사용자 지갑은 프레젠테이션을 생성하는 기능을 제공한다. 원하는 클레임을 선택하면 사용자는 자신의 서명이 포함된 프레젠테이션을 만들 수 있다. 이러한 프리젠테이션은 증거를 생성하고 이 증거를 검증하는 기능을 포함하는 데 사용될 수 있다.
- [0132] 실제로 지갑 주소 및 DID 생성에는 ECDSA 알고리즘에 비해 상대적으로 간단하고 효율적인 서명 생성 및 검증 프로세스를 포함하는 EdDSA 알고리즘을 활용하는 것이 유리한 접근 방식이 될 것이다. 그러나 비트코인 지갑은 ECDSA 알고리즘을 기반으로 주소와 거래 서명을 생성한다. 따라서, 다양한 다중 블록체인을 지원하기 위해 ECDSA 및 EdDSA 알고리즘이 모두 사용한다.
- [0133] 즉, 동일한 Seed를 생성하고, 각 Seed 값을 기반으로 ECDSA, EdDSA 개인키 쌍을 생성함으로써 계층적 구조를 이용한 다중 주소 지원이 가능하다. 이 접근 방식은 단일 시드에서 여러 주소를 파생시키는 이점을 제공한다. 또한 Metamask의 오픈 소스 코드를 활용하면 네트워크를 추가하는 것이 본질적으로 가능하다.
- [0134] 네트워크에는 이더리움이 기본 기반으로 포함되어 있으며 Goerli와 같은 이더리움 테스트 네트워크가 본질적으로 등록되어 있다. DID는 W3C 표준을 기반으로 하므로 호환성 문제가 없다. 주소 지갑의 경우 이더리움, 폴리곤, 옵티미즘 등의 네트워크에 존재하는 SBT가 이더리움에서 포크된다. 따라서, 다른 플랫폼을 사용하여 생성된 SBT를 처리할 때 추가적인 개발이 필요하다.
- [0135] 제안된 발명의 일 양상에 따르면, 분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑을 생성하는

방법은, 사용자의 시드(Seed)를 이용하여 암호화폐 지갑 주소에 사용되는 사용자의 개인키를 생성하는 단계와, 암호화폐 지갑 주소에 사용되는 사용자의 개인키에 암호화폐에 사용되는 곡선 알고리즘을 적용하여 하나 이상의 암호화폐 지갑 주소에 사용되는 공개키를 생성하는 단계와, 암호화폐의 지갑주소에 사용되는 사용자의 개인키와 공개키에 기초하여 해시 알고리즘을 적용하여 암호화폐의 지갑 주소를 생성하는 단계와, 상기 사용자의 시드(Seed)와 동일한 시드(Seed)를 이용하여 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키를 생성하는 단계와, 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키에 분산 아이디(DID) 식별자에 사용되는 곡선 알고리즘을 적용하여 하나 이상의 분산 아이디(DID) 식별자에 사용되는 공개키를 생성하는 단계와, 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키와 공개키를 이용하여 분산 아이디(DID)의 식별자를 생성하는 단계를 포함한다.

- [0136] 추가적인 양상에 따르면, 상기 암호화폐에 사용되는 곡선 알고리즘과 분산 아이디(DID) 식별자에 사용되는 곡선 알고리즘은 서로 다르다.
- [0137] 추가적인 양상에 따르면, 상기 암호화폐 지갑 주소에 사용되는 공개키를 생성하는 단계는, ECDSA의 secp256k1 타원 곡선 알고리즘을 사용하여 암호화폐 지갑 주소에 사용되는 공개키를 생성한다.
- [0138] 추가적인 양상에 따르면, 상기 분산 아이디(DID) 식별자에 사용되는 공개키를 생성하는 단계는, EDDSA의 Ed25519 타원 곡선 알고리즘을 사용하여 분산 아이디(DID) 식별자에 사용되는 공개키를 생성한다.
- [0139] 제안된 발명의 다른 일 양상에 따르면, 분산 아이디(DID)와 암호화폐를 함께 관리할 수 있는 디지털 지갑은, 사용자의 시드(Seed)와, 상기 사용자의 시드(Seed)를 이용하여 생성된 암호화폐 지갑 주소에 사용되는 사용자의 개인키와, 상기 암호화폐 지갑 주소에 사용되는 사용자의 개인키에 암호화폐에 사용되는 곡선 알고리즘을 적용하여 생성된 하나 이상의 암호화폐 지갑 주소에 사용되는 공개키와, 상기 암호화폐 지갑 주소에 사용되는 사용자의 개인키와, 상기 암호화폐 지갑 주소에 사용되는 공개키에 기초하여 해시 알고리즘을 이용하여 생성된 하나 이상의 암호화폐 지갑 주소와, 상기 사용자의 시드(Seed)를 생성된 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키와, 상기 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키에 분산 아이디(DID) 식별자에 사용되는 곡선 알고리즘을 적용하여 생성된 하나 이상의 분산 아이디(DID) 식별자에 사용되는 공개키와, 상기 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키와 상기 분산 아이디(DID) 식별자에 사용되는 공개키를 이용하여 생성된 분산 아이디(DID)의 식별자를 포함한다.
- [0140] 추가적인 양상에 따르면, 상기 사용자의 시드(Seed), 암호화폐 지갑 주소에 사용되는 사용자의 개인키와 공개키, 및 암호화폐 지갑 주소는, 계층 구조를 이룬다. 상기 사용자의 시드(Seed), 분산 아이디(DID)의 식별자에 사용되는 사용자의 개인키와 공개키, 및 분산 아이디(DID)의 식별자도, 계층 구조를 이룬다.
- [0141] 추가적인 양상에 따르면, 상기 암호화폐 지갑 주소에 사용되는 사용자의 개인키와, 상기 분산 아이디(DID) 식별자에 사용되는 사용자의 개인키는, 서로 동일한 값을 가진다.
- [0142] 도 16은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 디지털 지갑을 통해 메타버스의 아바타에 대한 권한을 관리하는 방법을 나타내는 흐름도이다.
- [0143] SBT가 생성되면 메타버스 아바타 및 지갑 주소에 연결할 수 있다. 특정 서비스에 접근하기 위해서는 사용자의 지갑에 연결된 아바타 내에서 해당 SBT의 존재 여부를 확인한다. 이 확인 프로세스는 사용자가 메타버스 내에서 서비스를 사용할 수 있는지 여부를 결정한다. 이러한 서비스가 구현되기 위해서는 도 14와 같이 디지털 지갑 주소 시스템을 개발해야 한다.
- [0144] 현재 디지털 지갑은 NFT, 배지, 자산과 같은 토큰만 저장할 수 있으며 DID, 자격 증명 및 프레젠테이션에 대한 지원이 부족하다. 이를 해결하려면 사용자가 자신의 DID를 디지털 지갑에 연결할 수 있는 지갑 프로그램을 개발해야 하며, 발행자의 자격증명을 사용해 NFT를 발행할 수 있는 서비스를 만들 수 있어야 한다.
- [0145] 현재 이더리움 기반 네트워크에서 운영되는 메타버스 시스템과의 통합이 가능하다. 증명 검증을 위해 생성된 스마트 계약은 EVM(Ethereum Virtual Machine) 기반 네트워크 내에서만 검증이 가능하다. 결과적으로 이러한 증명은 주로 이더리움 플랫폼에서 작동하는 네트워크 내에서만 확인할 수 있다.
- [0146] 도 17은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템을 다른 블록체인 신원인증 방법들과 비교한 결과를 나타내는 도면이다.
- [0147] 메타버스나 DeFi 등 기존 시스템에서는 서비스 제공자가 모든 사용자 정보를 보유하고 있어 해킹 시 상당한 위험이 따른다. 그러나 제안 시스템에서는 DID 발급자(추후 감사도 수행할 수 있음)만이 사용자 정보를 가지고 있

다. 서비스 제공자는 프레젠테이션에서 파생된 증거만 확인한다. 이 설정은 서비스 제공자가 개인 데이터에 액세스할 수 없지만 제시된 증거를 기반으로 누군가가 특정 기준을 충족하는지 확인할 수 있으므로 사용자의 개인 정보가 보호되도록 보장한다.

[0148] 서비스 제공자로부터 SBT를 받은 사람이 자금세탁이나 불법행위에 연루된 경우, SBT 외부링크에 저장된 자격증명번호를 이용해 DID 발급자에게 해당 자격증명에 대한 정보를 요청함으로써 의심스러운 행위를 추적할 수 있다. 또한 시스템에는 SBT 철회 기능이 포함되어 있다. 사용자의 지갑 주소에 존재하는 다양한 SBT에서 특정 개인의 신원을 추론할 수 있는 잠재적 위험을 고려하여 시스템은 사용자 요청에 응답하여 ERC-4906 및 ERC-5185를 사용한다. 이를 통해 사용자 SBT의 메타데이터를 업데이트하고 효과적으로 링크를 끊고 개인정보를 보호할 수 있다.

[0149] 제안된 프로토콜에서는 자격 증명이 사용자의 지갑에 직접 매핑되지 않기 때문에 증명이 복제되어 사용될 수 있는 잠재적인 문제가 존재한다. 생성된 증명에는 해당 DID의 소유자가 진짜임을 주장하는 서명만 포함되어 있다. 따라서 사용자가 제시 증명을 생성하는 데 사용된 것과 동일한 증명을 제공하는 경우 관련 자격 증명을 소유하고 있다고 가정할 수 있다. 이를 해결하기 위해 현재 프로토콜에서는 서비스 제공자가 SBT를 발행할 때 해당 증거가 스마트 계약에 저장된다. 동일한 증명이 다시 제출되면 프로토콜은 SBT 발행을 방지한다.

[0150] 개발된 스마트 계약은 이더리움 표준 ERC-721을 따르고 OpenZeppelin 라이브러리를 사용하여 전송 불가능한 기능을 구현하여 증명이 전송되지 않도록 보장한다. 증명 검증은 ZoKrates의 ZK-SNARK를 사용하여 이루어진다. OpenZeppelin과 ZoKrates는 모두 보안 검증을 거쳐 스마트 계약 내에서 안전하게 활용되도록 보장한다.

[0151] Rust 언어를 사용하면 범위 증명이나 산술 회로를 사용하여 연립 및 서명과 같은 사용자 생성 프레젠테이션 내의 내용을 확인할 수 있다. Remix에서 제공하는 ZoKrates 도구는 이 프로세스를 용이하게 한다. 회로의 설정 단계는 오프체인으로 수행되며 결과는 검증 계약에서 공유된다. 사용자가 자신의 비밀 값을 사용하여 진술을 증명하려는 경우 ZoKrates 언어와 설정에서 생성된 증명 키를 비밀 값과 함께 사용하여 증명을 만든다.

[0152] 설정에서 검증 키로 생성된 검증 계약을 배포하여 블록체인에서 증명의 정확성을 검증함으로써 블록체인에서 증명 검증이 가능하다. 이 도구를 사용하면 Remix에서 제공하는 ZoKrates 도구를 사용자를 위한 증거를 생성하는 언어로 변환할 수 있으며 스마트 계약은 검증 계약을 제공한다. 이러한 방식으로 ZKP에 대한 심층적인 지식이 없는 개발자라도 사용자의 신원을 확인할 수 있어 프로세스에 접근하고 유리하게 만들 수 있다.

[0153] 도 17에서 각각의 강점과 약점을 강조하면서 이러한 정체성을 비교했다. 본 논문에서 언급한 바와 같이 제안하는 기법은 메타버스 아바타와의 연결에 적합하다.

[0154] 이하에서는 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법을 수행하는 시스템의 구성에 대하여 간단히 설명한다. 구체적인 시스템 구성 및 동작에 대해서는 앞에서 설명한 내용을 참조할 수 있다.

[0155] 도 18은 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 서비스 제공 서버의 구성을 나타내는 구성도이다.

[0156] 제안된 발명의 다른 양상에 따르면, 분산 아이디(DID)와 소울바운드 토큰(SBT)을 결합한 메타버스에서의 아바타 신원인증 시스템의 서비스 제공 서버(455)는, 사용자의 분산 아이디(DID)를 포함하는 크리덴셜에 기초하여 특정 권한에 대한 증명이 검증되는 경우 소울바운드 토큰(SBT)을 발행하는 스마트 계약을 생성하는 스마트 계약 생성부(1810)와, 생성된 스마트 계약을 블록체인 네트워크에 배포하는 스마트 계약 배포부(1830)와, 아바타와 매핑된 지갑 주소의 소울바운드 토큰(SBT)을 확인하는 SBT 확인부(1850)와, 소울바운드 토큰(SBT)의 존재 여부에 기초하여 아바타의 이용 권한을 설정하는 아바타 권한 설정부(1870)를 포함한다.

[0157] 도 19는 일 실시예에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템에서 사용자 단말의 구성을 나타내는 구성도이다.

[0158] 제안된 발명의 또 다른 양상에 따르면, 분산 아이디(DID)와 소울바운드 토큰(SBT)을 결합한 메타버스에서의 아바타 신원인증 시스템의 사용자 단말(420)은, 사용자의 분산 아이디(DID) 및 고객확인(KYC)에 필요한 정보를 크리덴셜 발행자 서버로 전달하는 크리덴셜 요청부(1910)와, 크리덴셜 발행자 서버로부터 전달받은 크리덴셜을 지갑에 저장하는 크리덴셜 저장부(1930)와, 분산 아이디(DID)를 포함하는 크리덴셜에 기초하여 증명을 생성하고 사용자의 서명을 추가하여 프리젠테이션을 생성하는 프리젠테이션 생성부(1950)와, 프리젠테이션으로부터 영지식 증명(ZKP)에 기초하여 사용자의 증명을 생성하는 사용자 증명 생성부(1970)와, 블록체인 네트워크에서 발행된 소울바운드 토큰(SBT)이 저장된 지갑 주소를 아바타와 매핑하는 아바타 매핑부(1990)를 포함한다.

- [0159] 본 발명에 따른 DID와 SBT를 결합한 메타버스에서의 아바타 신원인증 방법 및 시스템은, 사용자 개인 정보 보호를 위해 DID를 그대로 전송하는 대신에, 검증에 영지식 증명(ZKP) 기반 증명을 사용한다. 따라서 서비스를 제공하는 VASP에게 중요한 사용자 정보가 알려지지 않으므로 개인정보를 보호할 수 있다.
- [0160] 나아가 본 발명에 의하면, 사용자는 영지식 증명(ZKP)을 위해 설계된 도구를 통해 암호학적 증명을 직접 생성하지 않고도 암호화 증명을 생성할 수 있어 사용자 편의성이 높아진다. 또한, DID 크리덴셜과 SBT를 통합 시드값으로 관리하는 지갑을 제시하여 사용자 편의성을 높일 수 있다.
- [0161] 그 결과, 사용자의 Web 3.0 활용 장벽을 낮출 수 있고, 범죄 예방 및 범죄 근절에 기여할 수 있다.
- [0162] 이상에서 본 발명을 첨부된 도면을 참조하는 실시예들을 통해 설명하였지만 이에 한정되는 것은 아니며, 이들로부터 당업자라면 자명하게 도출할 수 있는 다양한 변형예들을 포괄하도록 해석되어야 한다. 특허청구범위는 이러한 변형예들을 포괄하도록 의도되었다.

부호의 설명

- [0163] 410 : 크리덴셜 발행자 서버
420 : 사용자 단말
430 : 디지털 지갑
440 : 영지식 증명(ZKP) 도구
450 : SBT 발행자 서버 (서비스 공급자 서버)
460 : 스마트 계약 프로그램
470 : 블록체인 네트워크
480 : 아바타

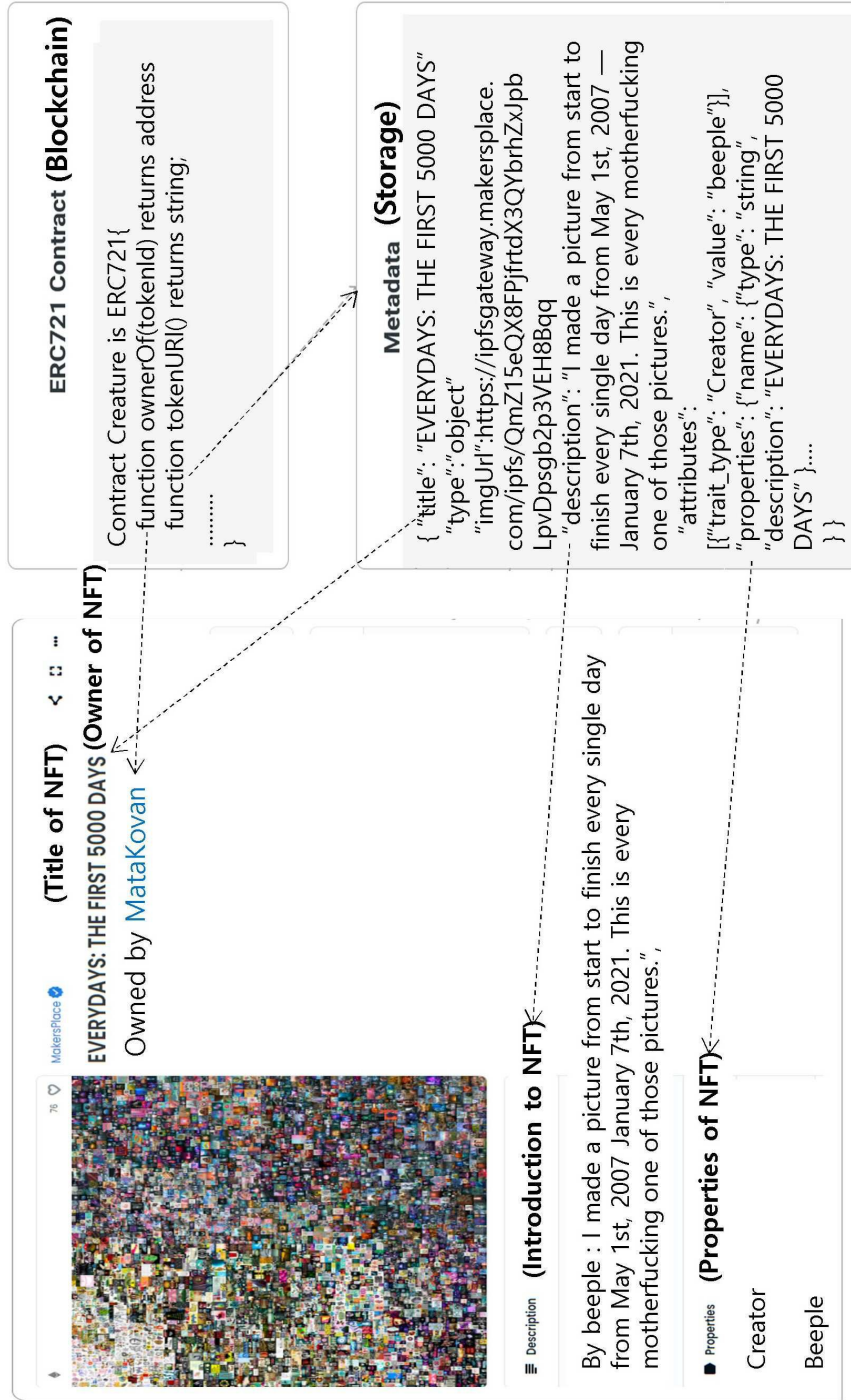
도면

도면1



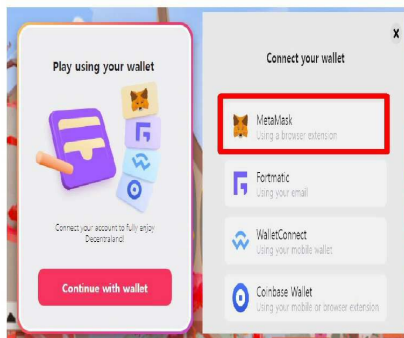
도면2

The actual storage space for NFTs

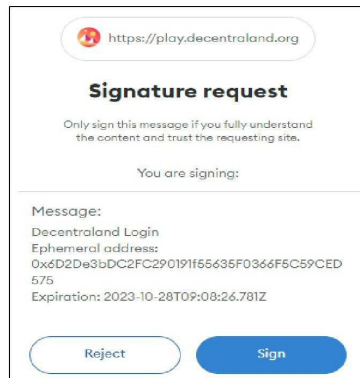


도면3a

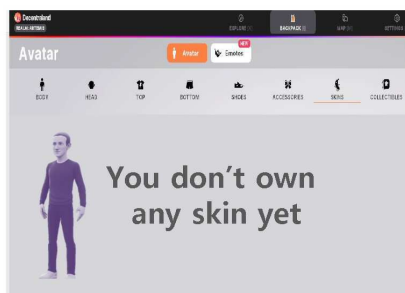
1. Access the decentraland
(Connected Wallet)



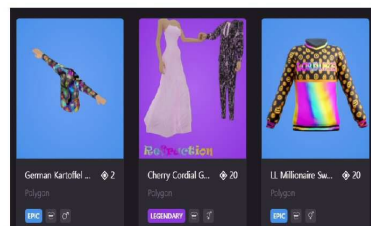
2. Submit signature to
verify user wallet address



3. Check the user's NFT &
Avatar mapping
(NFT linked to wallet address)

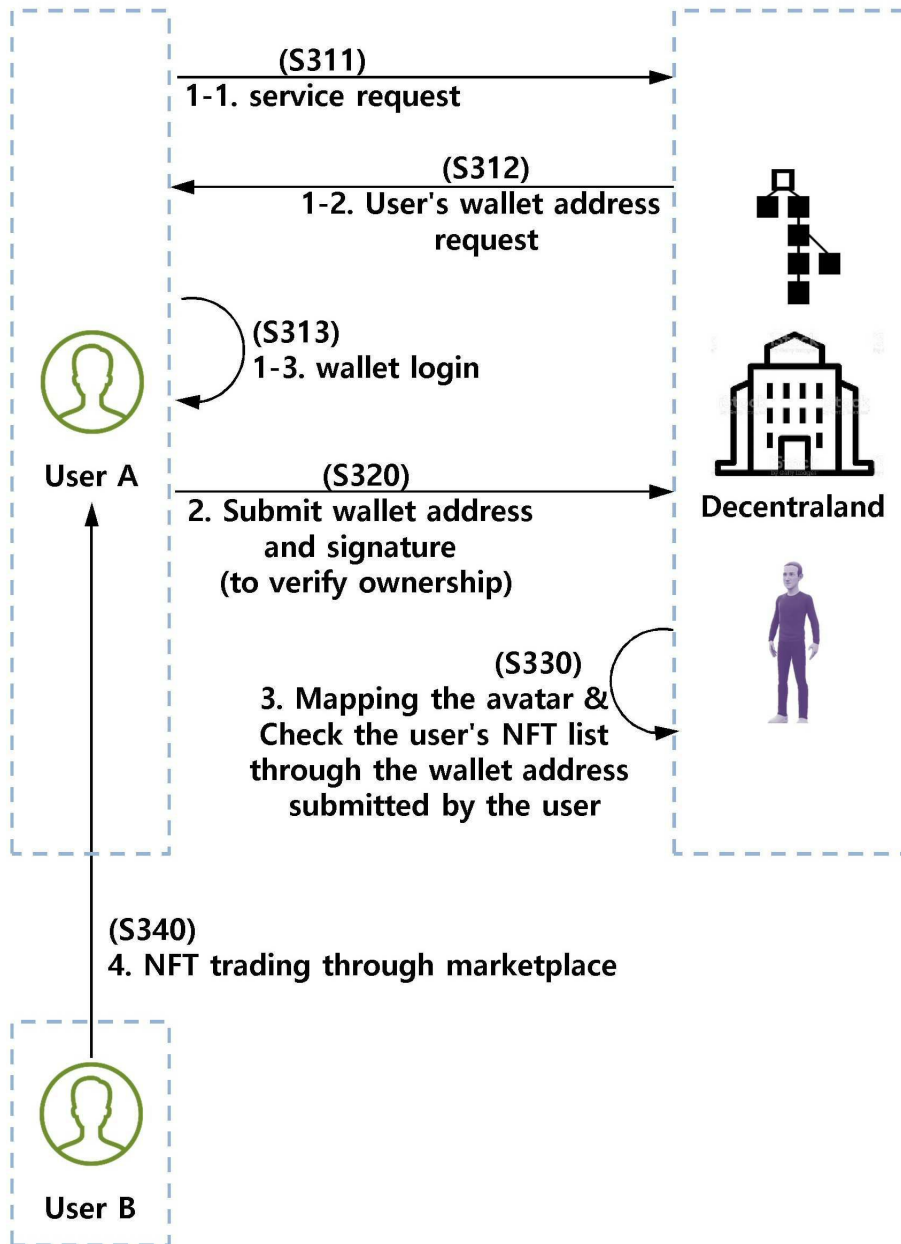


4. NFT trading
in the marketplace

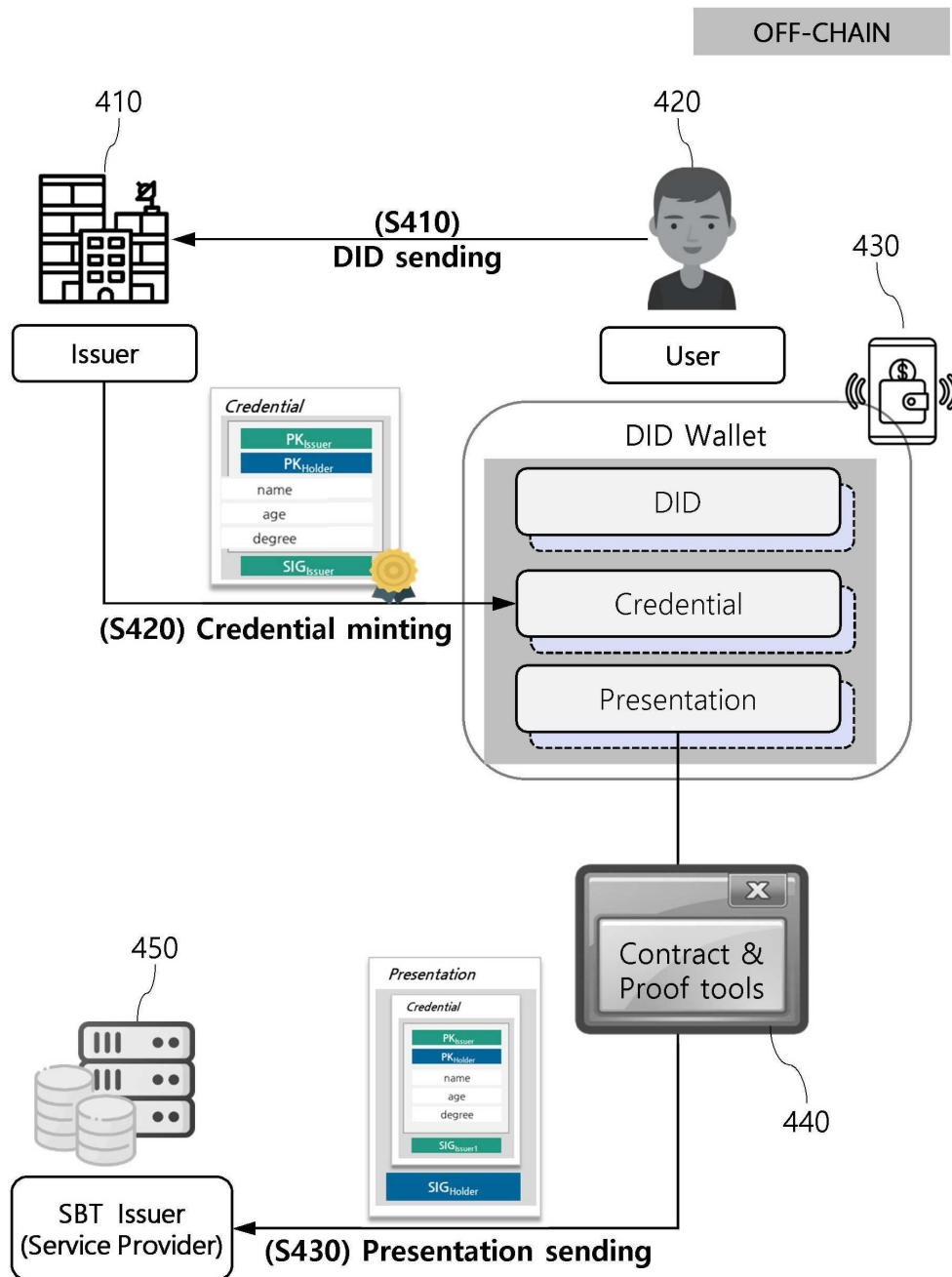


(Item List)

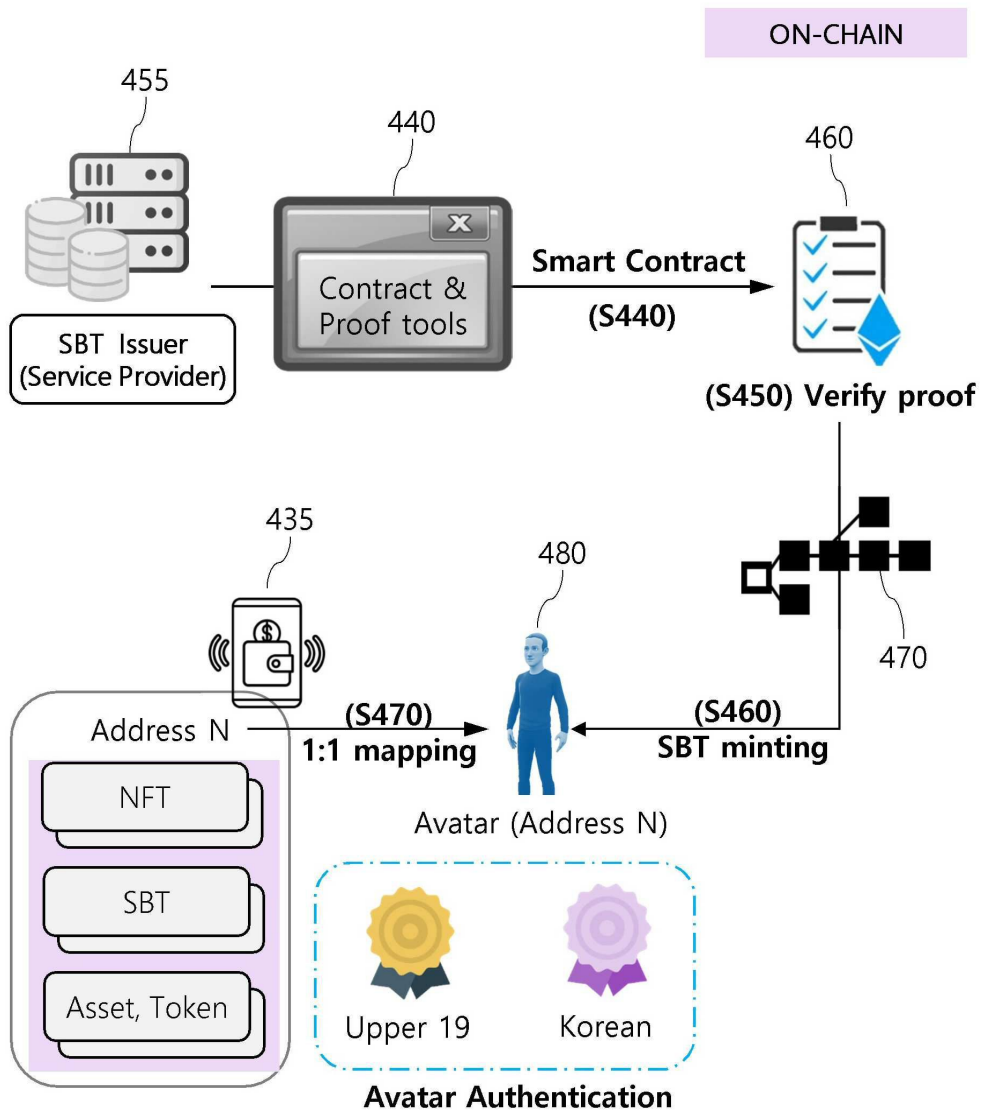
도면3b



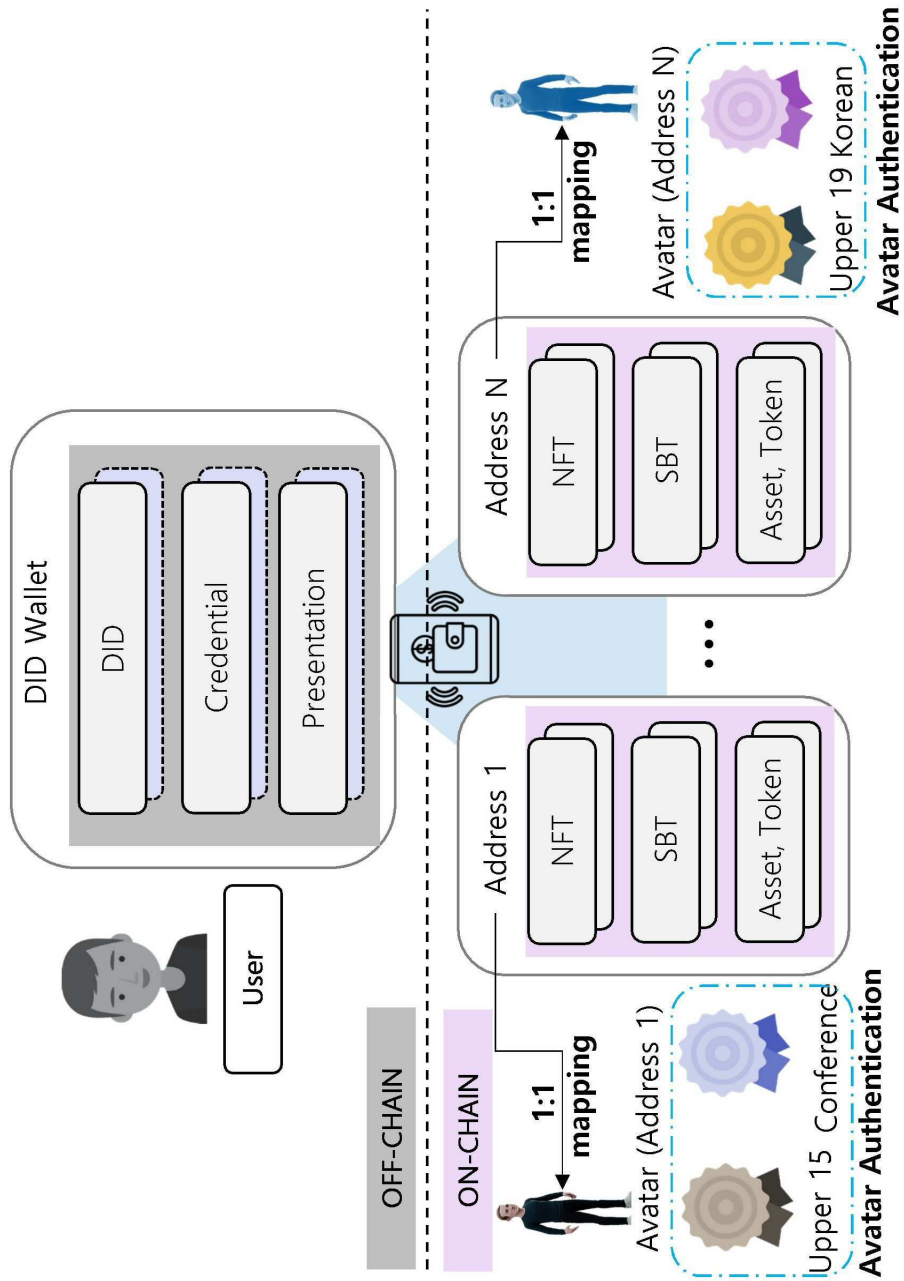
도면4a



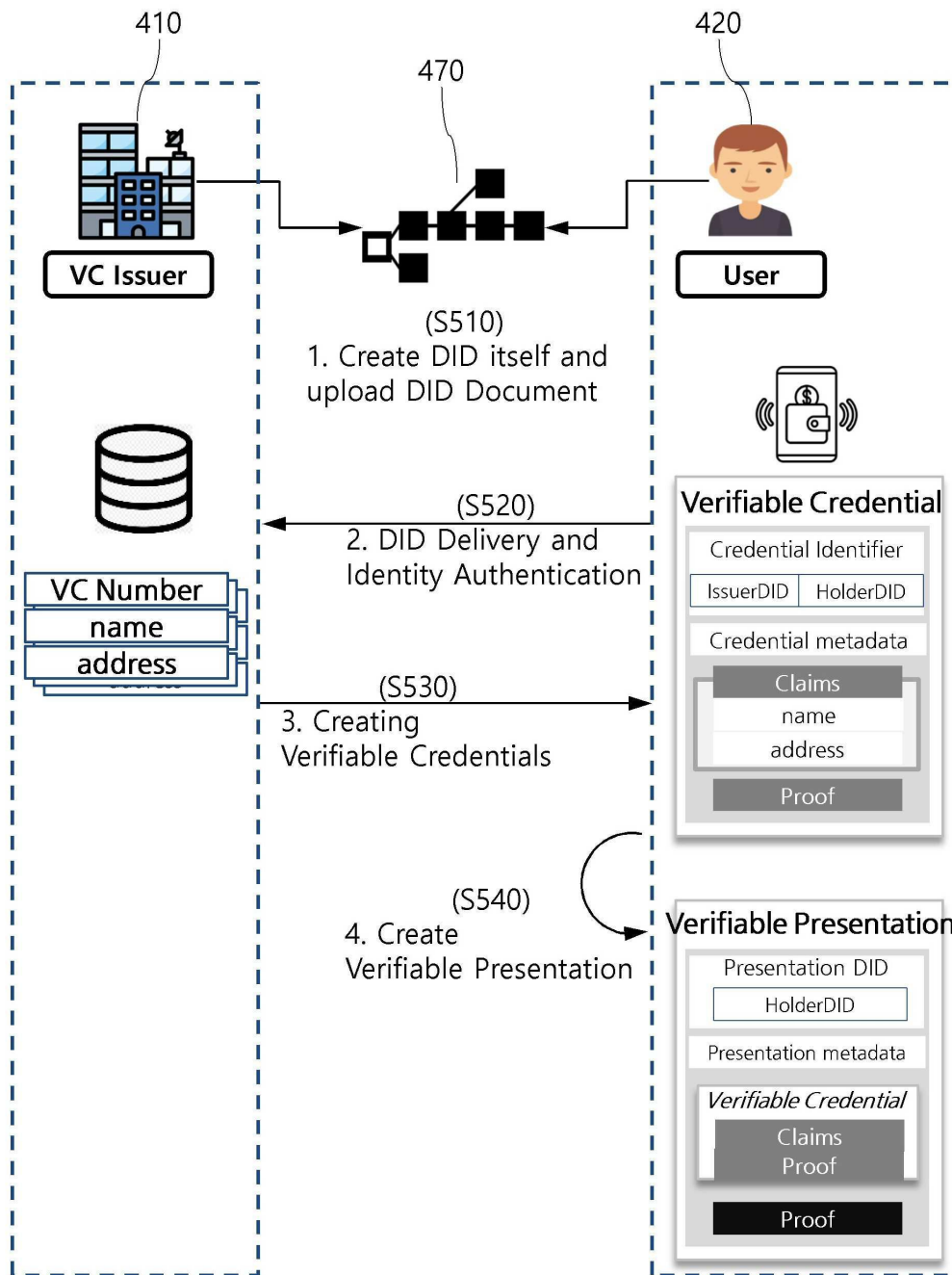
도면4b



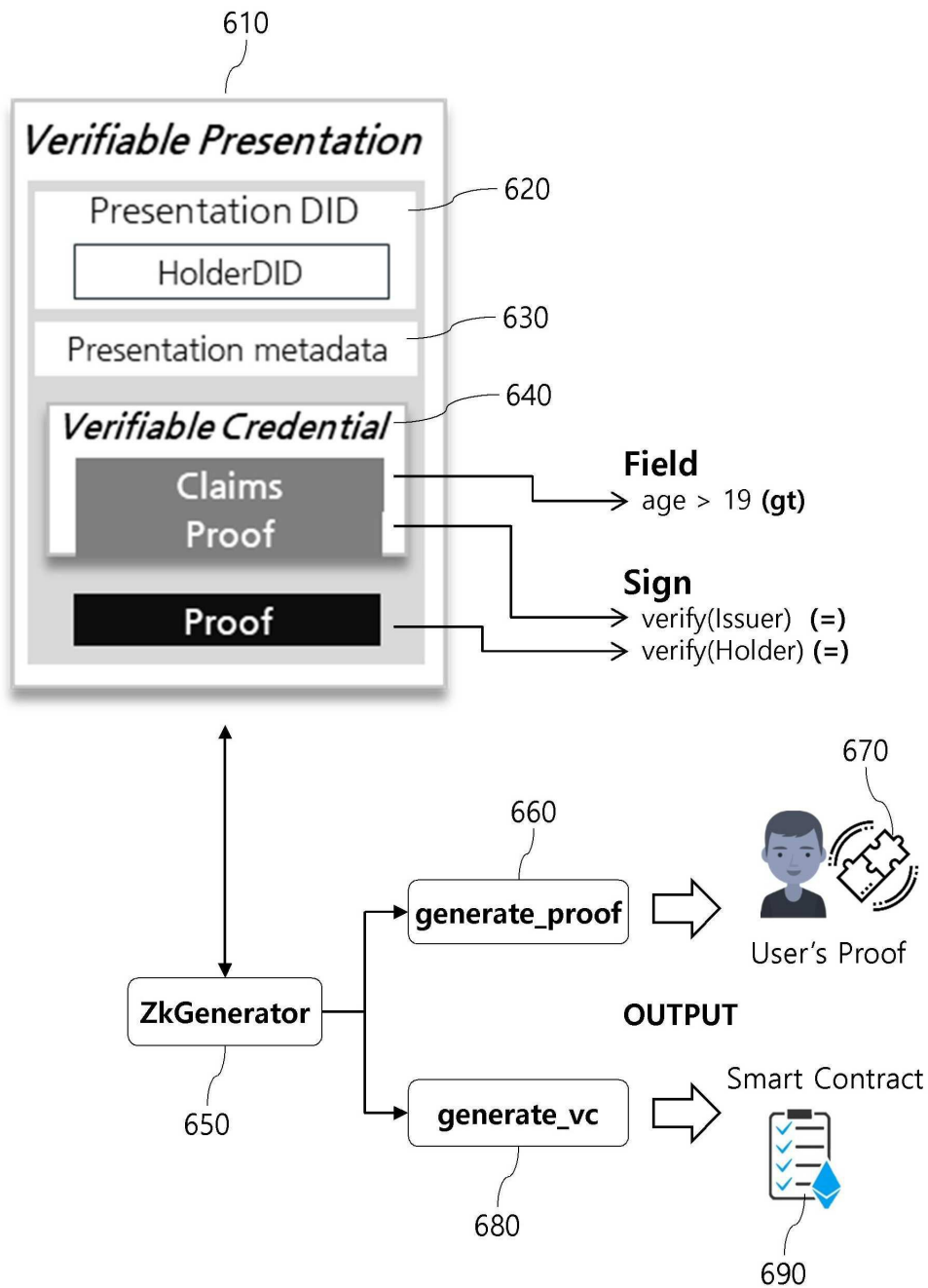
도면4c



도면5



도면6



도면7

[illegible]

도면8

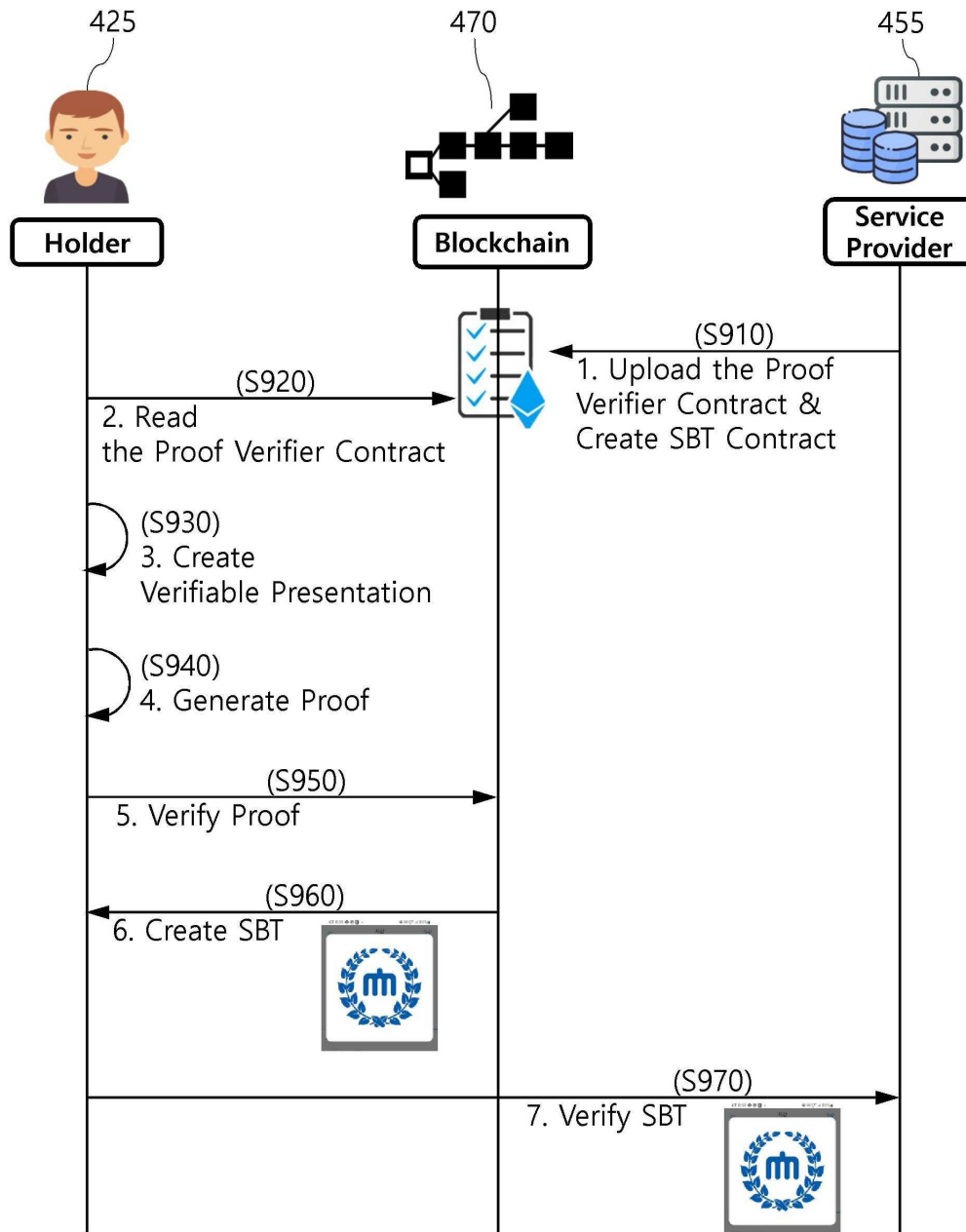
(a)

```
struct Proof {
    ...
}
struct IkosVariable {
    ...
}
```

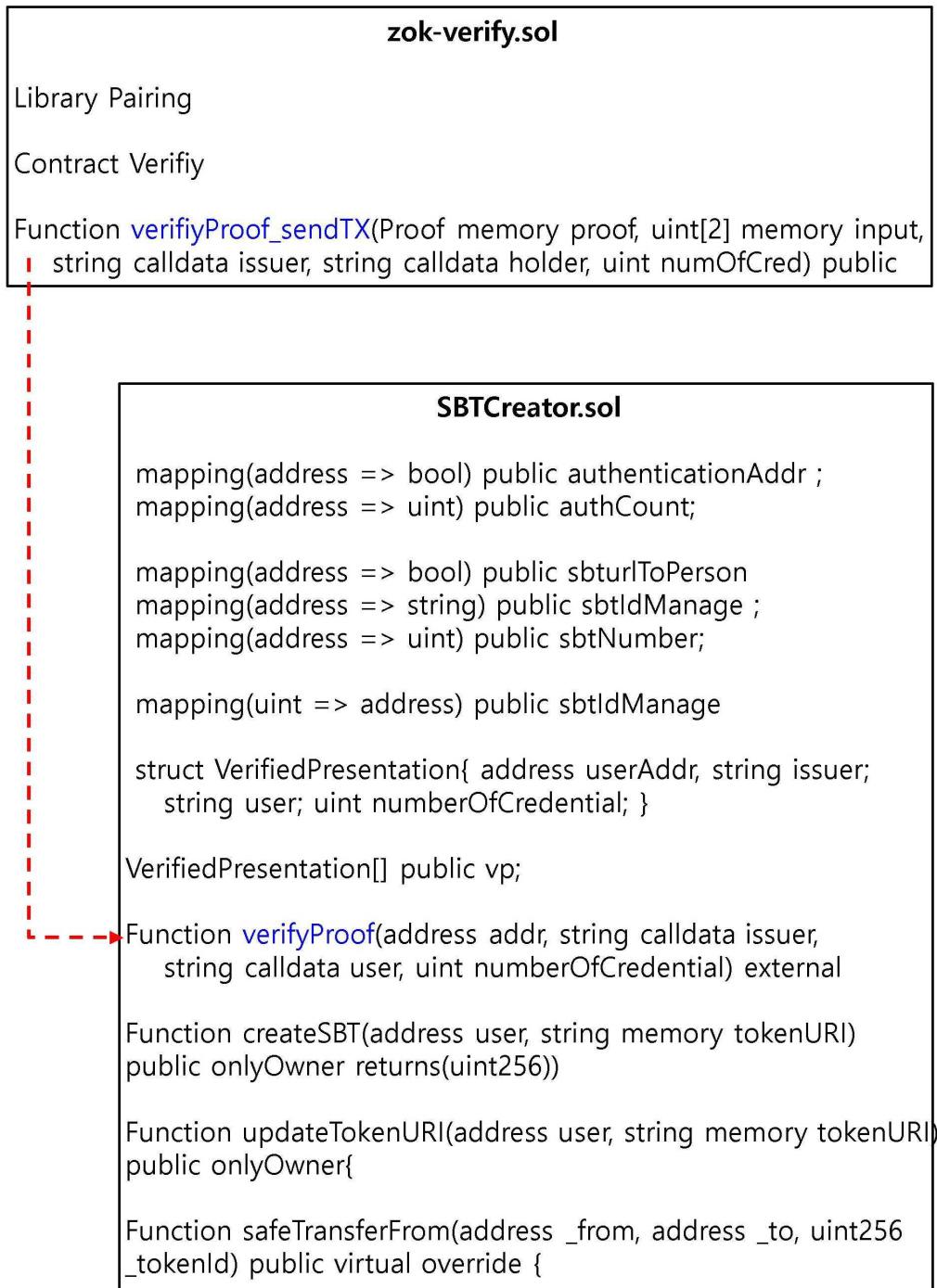
(b)

```
Contract Checker {
    function _greaterthan(
        Proof proof
    ) internal pure returns (IkosVariable memory) {
        ...
        return gt(ikosCtx[0], proof.pub[0]);
    }
    function greaterthan(
        Proof proof
    ) public pure returns (bool) {
        IkosVariable res = _greaterthan(proof);
        ...
        return proof.out[0];
    }
    ...
}
```

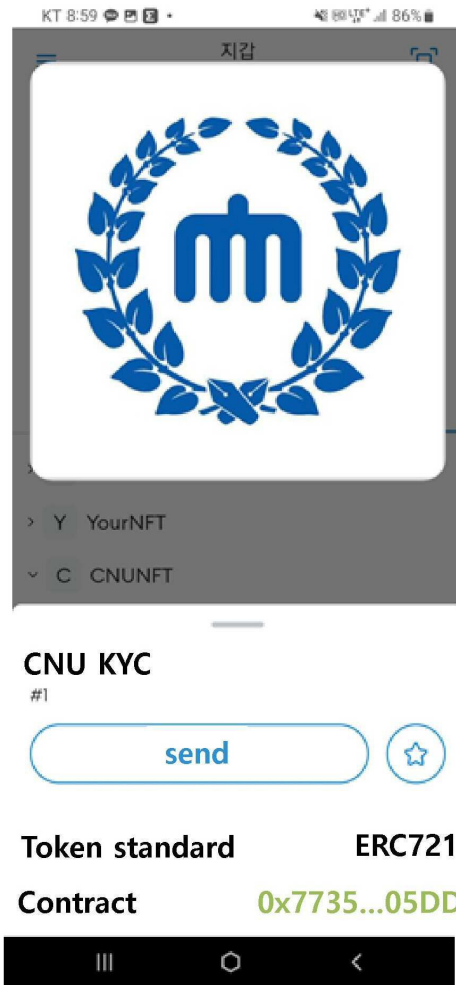
도면9



도면10



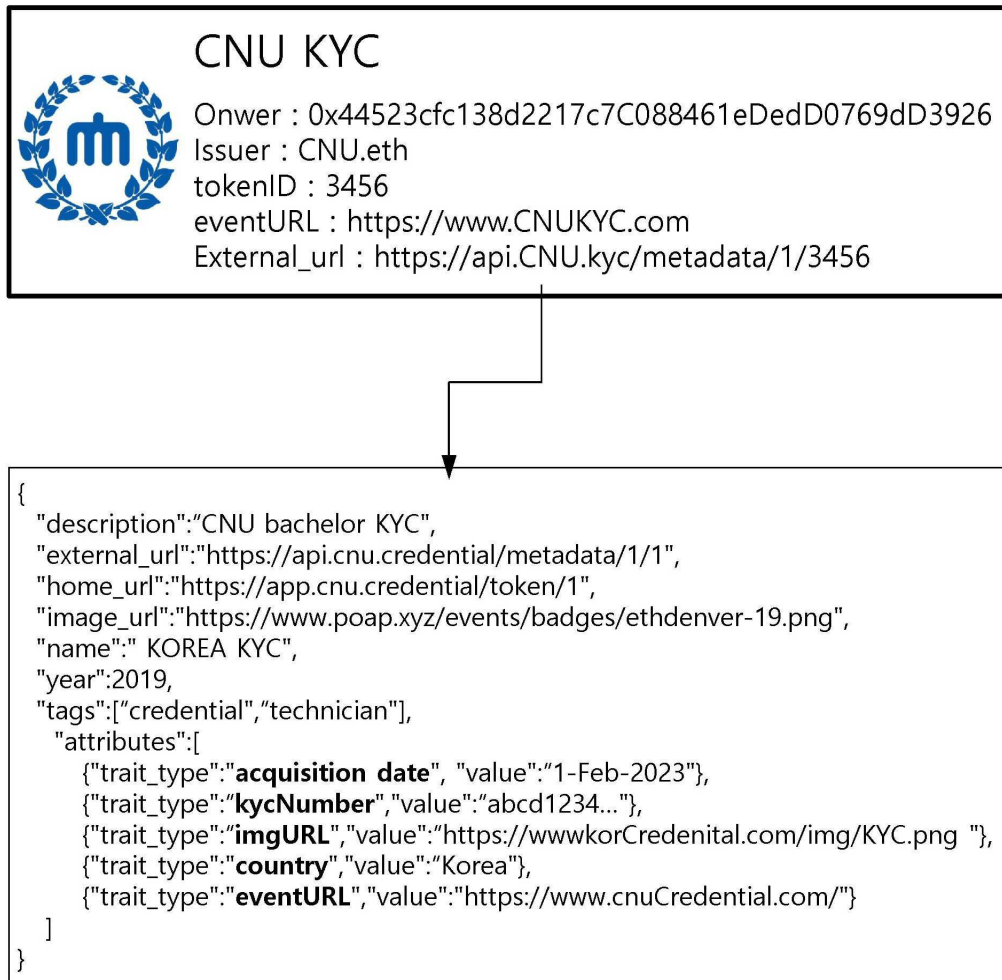
도면11a



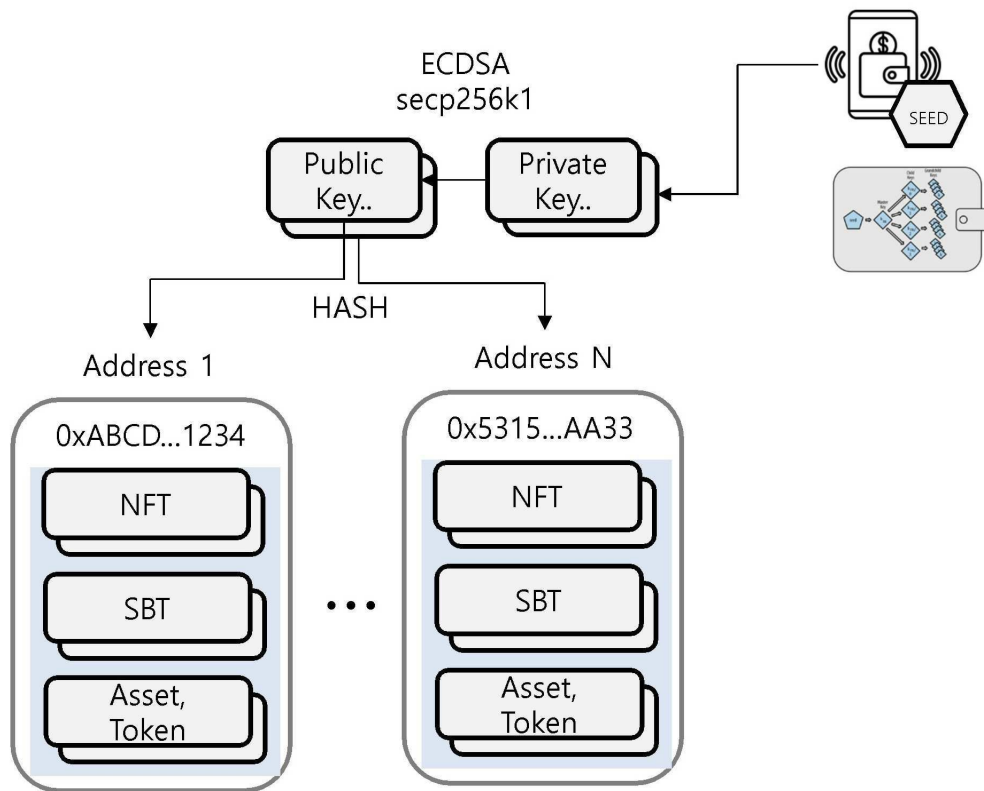
도면11b

```
{
  "@context": [
    "https://www.w3.org/2023/credential/v1",
    "https://www.w3.org/2023/credentials/examples/v1"
  ],
  "type": ["VerifiableCredential", "UniversityDegreeCredential"],
  "CredentialsSchema": {
    "id": "did:example:cdcf:3Gl87w9UeWbagPL94T9bMltyXDj9pX5o",
    "type": "did:example:schema:22KpXgecyx9k7N6XN1QoN3gXwBkSU8FyYyqG"
  },
  "issuer": "did:example:Wz4eUg7SetGfaUVCn8U9d62oDyUJLuUty619",
  "credentialNumber": "abcd1234",
  "credentialSubject": {
    "givenName": "Gildong",
    "familyName": "Hong",
    "degree": {
      "type": "CNU KYC",
      "name": "Bachelor of computer Science",
      "university": "University of Chungnam National"
    }
  },
  "proof": {
    "type": "CLSignature2023",
    "issuerData": "5NQ4Tasndkjfqwhui23Ajd fhgdfkjs...Bxjkhasjdk4517289qwdhjs",
    "attributes": "pPYmajmqwwwweusadbnnm2huaasf...ajkkjqkwe1q2Osjdwhje",
    "signature": "8eGujqweyhu2QWebnjh23237ensdnjwEjsadhwdkqwebhkf1u"
  }
}
```

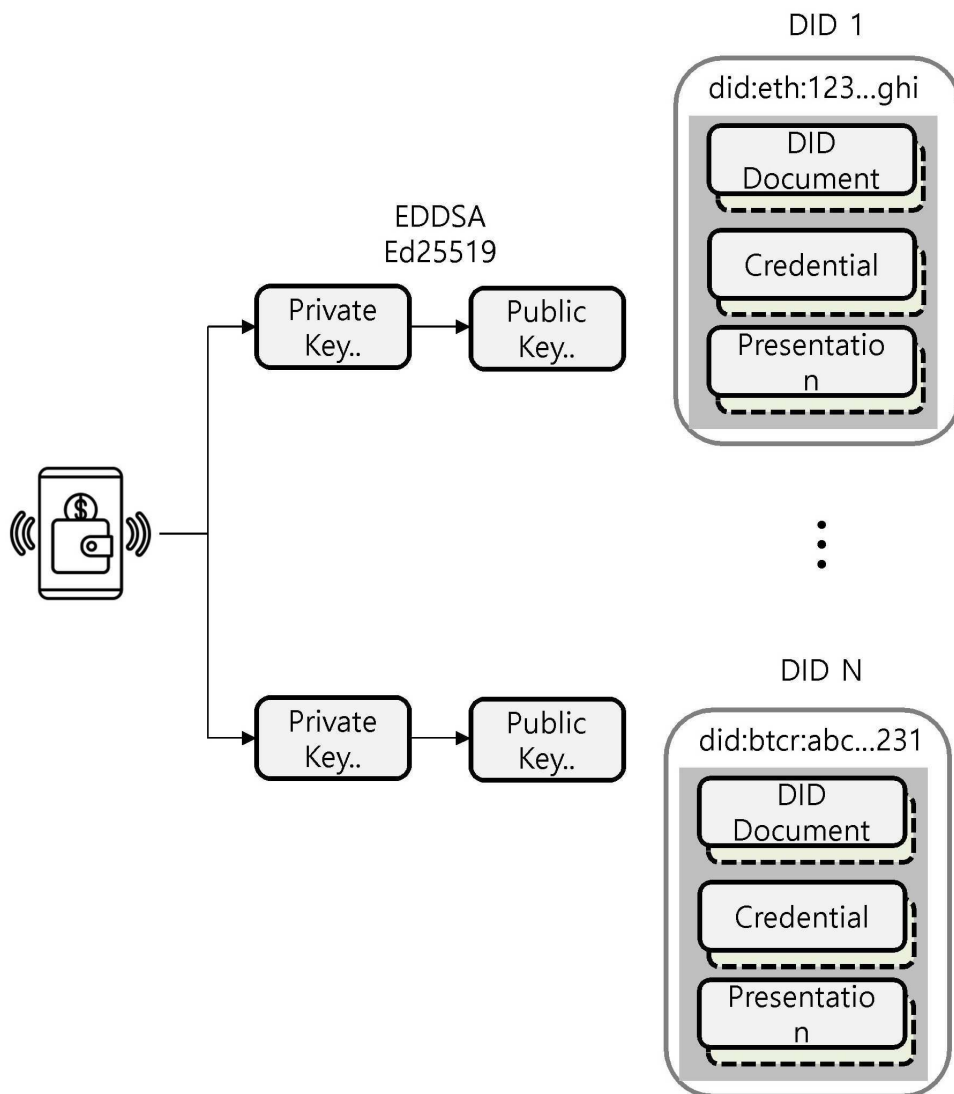
도면12



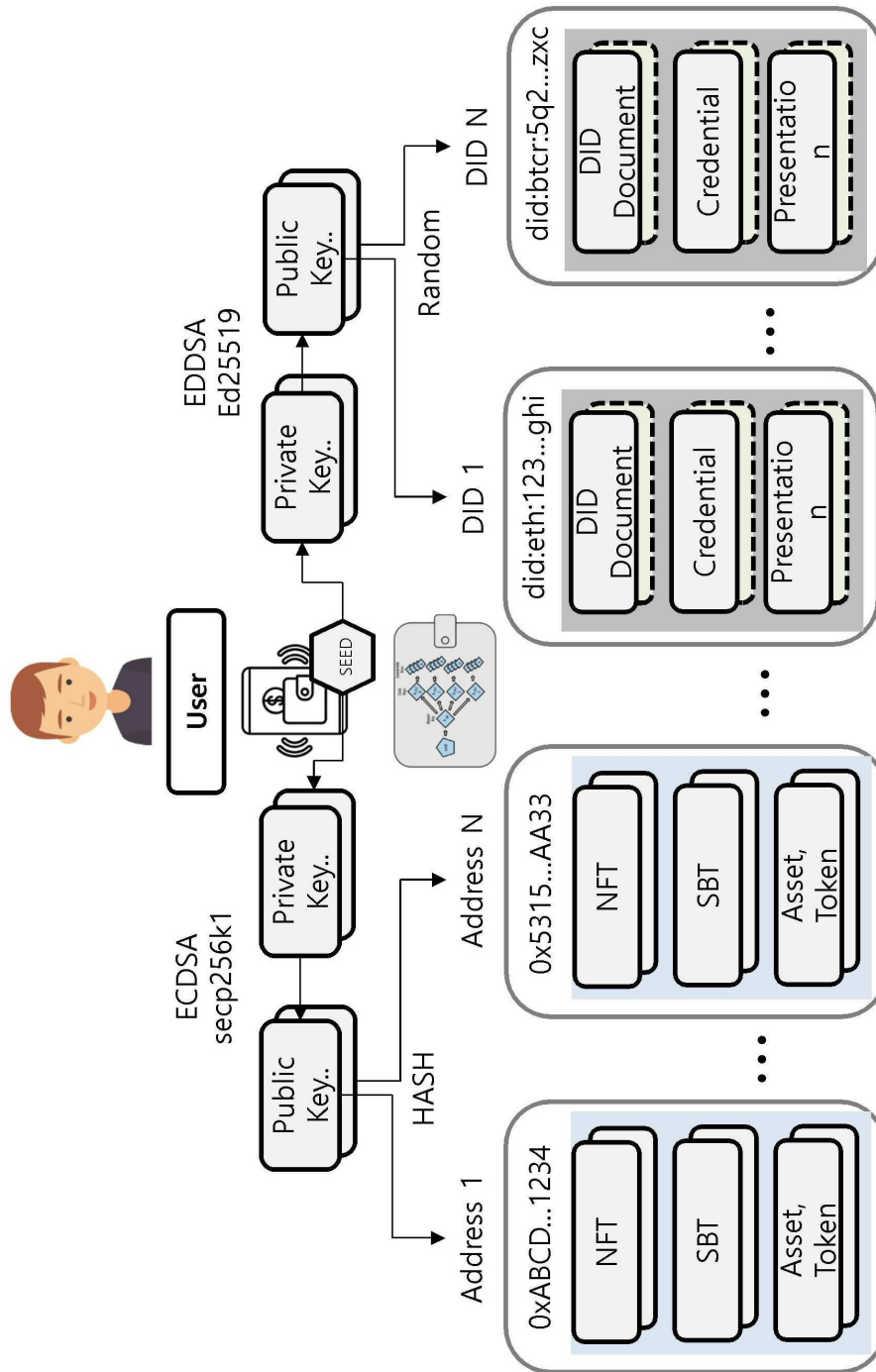
도면13



도면14



도면15a



도면 15b

Goerli Test

Wallet List

0xABCD...1234
1 ETH, 10 NFT, 10 SBT

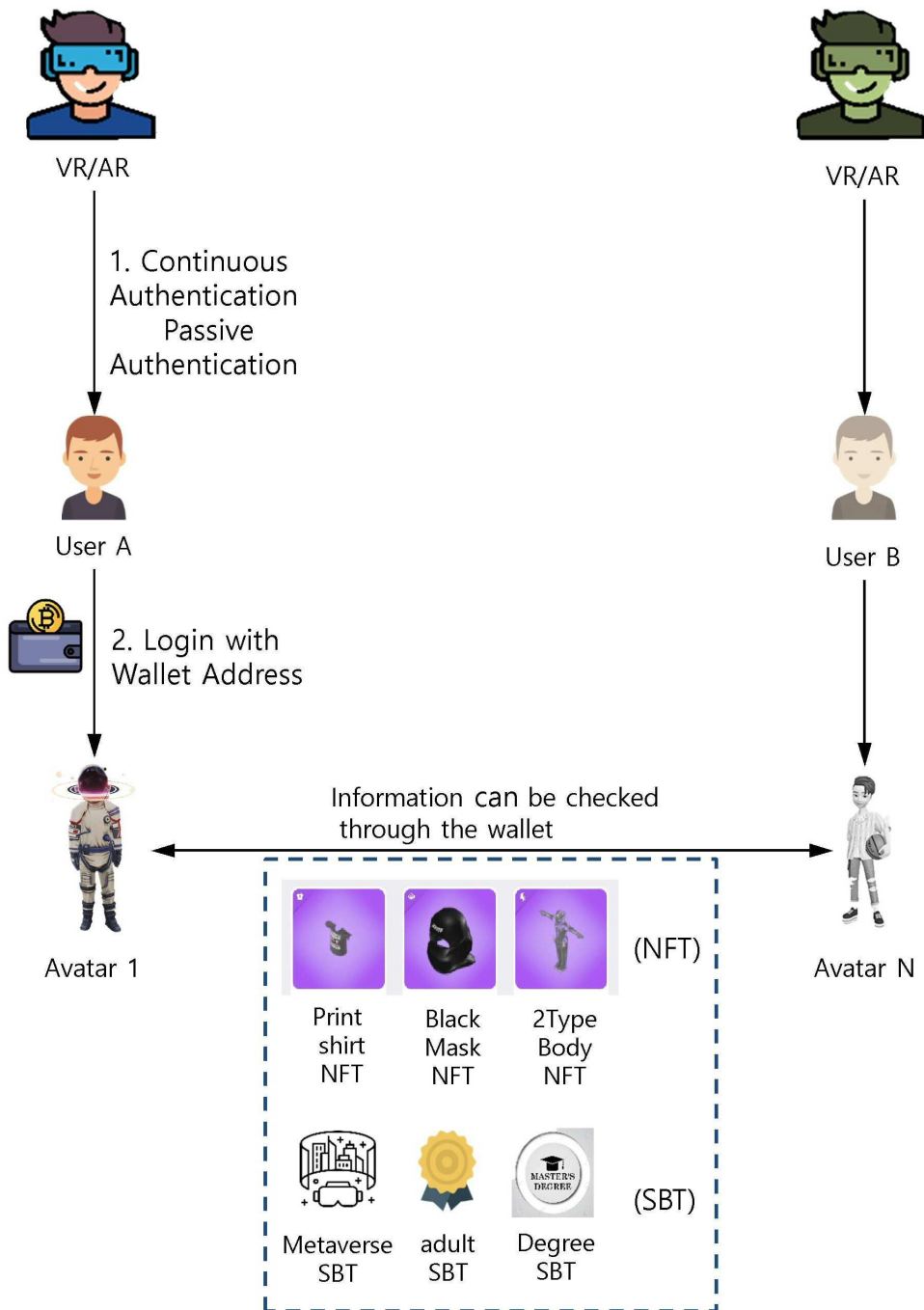
0x5315...AA33
3 ETH, 5 NFT, 1 SBT

... ..

 did:eth:123...ghi
2 Credential

 did:btc:5q2...zxc
4 Credential

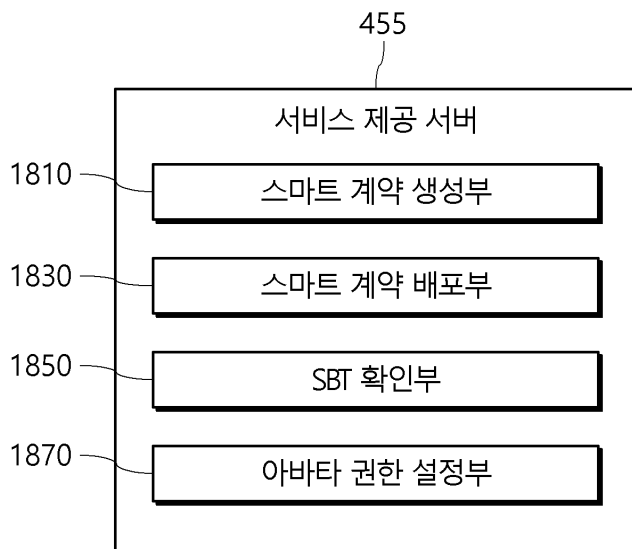
도면16



도면17

시스템	1. Credential Identity System	2. Personality Identity System	3. Reputation Identity System	4. DID System	5. The Scheme
구분	POAP/ENS	BrightID	Project Galaxy	Sovrign	SBT
On-chain Verification	Yes	No	Yes	No	Yes
신원 보장 토큰	No	Yes	Yes	No	Yes
Is the wallet account anonymous to the Verifier ?	No	No	No	Yes (Selective Disclosure)	Yes (Selective Disclosure)
Off-chain 신원과 Wallet의 상호 호환	Yes	Yes	No	No	Yes

도면18



도면19

