(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2018-0112061
(43) 공개일자 2018년10월11일

(51) 국제특허분류(Int. Cl.)
*G06Q 20/36* (2012.01) *G06Q 20/38* (2012.01)
*G06Q 30/06* (2012.01) *H04L 9/06* (2006.01)
(52) CPC특허분류
*G06Q 20/367* (2013.01)
*G06Q 20/3827* (2013.01)
(21) 출원번호 10-2018-7027478
(22) 출원일자(국제) 2017년02월16일
심사청구일자 2018년09월28일
(85) 번역문제출일자 2018년09월20일
(86) 국제출원번호 PCT/IB2017/050865
(87) 국제공개번호 WO 2017/145019
국제공개일자 2017년08월31일
(30) 우선권주장
1603114.8 2016년02월23일 영국(GB)
*(뒷면에 계속)*

(71) 출원인
**엔체인 홀딩스 리미티드**
안티구아바부다 세인트존스, 처치 스트리트 44, 피츠제럴드 하우스
(72) 발명자
**라이트, 크레이그 스티븐**
영국, 씨에프10 2에이치에이치 카디프, 처칠 웨이, 처칠 하우스 7층, 어커트-디키스 앤 로드 엘엘피
**사바나, 스테판**
영국, 씨에프10 2에이치에이치 카디프, 처칠 웨이, 처칠 하우스 7층, 어커트-디키스 앤 로드 엘엘피
(74) 대리인
**특허법인다나**

전체 청구항 수 : 총 27 항

(54) 발명의 명칭 **블록체인 집행의 스마트 계약을 위한 레지스트리 및 자동화 관리 방법**

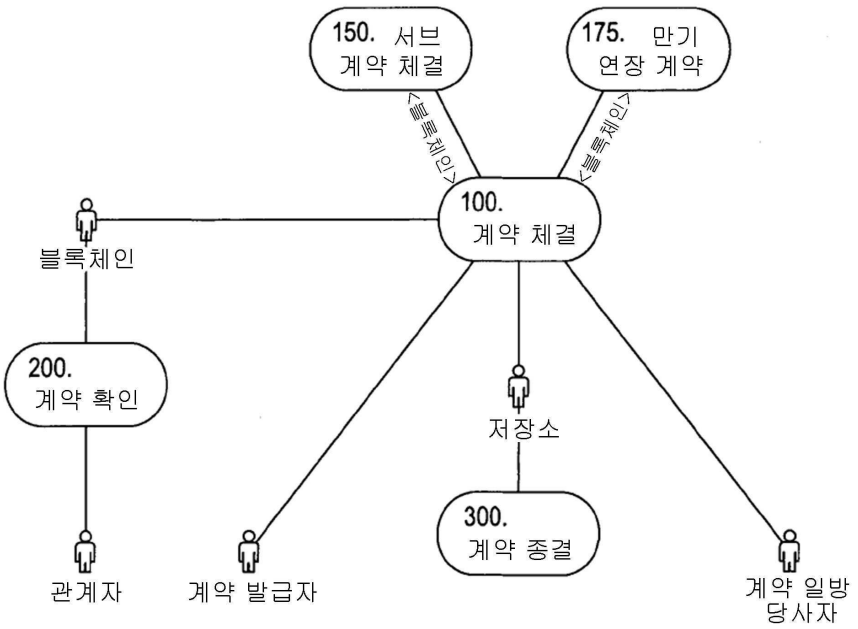
## *(57) 요 약*

블록체인 집행의 스마트 계약을 위한 레지스트리 및 자동화 관리 방법.

본 발명은 토큰화, 블록체인 및 스마트 계약 기술 분야에 관련된다. 계약의 자동화 관리를 단순화 하는 기술 구성을 제공한다. 본 발명은 계약의 저장을 위해 컴퓨터 기반 저장소를 이용하는 시스템 및 방법을 포함한다. 계약

(뒷면에 계속)

*대 표 도* - 도1



은 블록체인 상에 거래에 의해 표현된다. 거래 스크립트 내 메타데이터는 계약의 해시 및 저장소 내 그 위치를 식별하는 수단을 포함한다. 거래는 또한 열린(즉 종결되지 않은) 계약으로서 그 상태를 나타내는 미사용 출력(UTXO)을 포함한다. 계약은 예를 들어, nLockTime + CheckLockTimeVerify (CLTV)를 이용하여, 나중에 출력을 소비함으로써 종결된다. 다른 기술과 컴퓨팅 요소를 가지는 이 컨셉을 결합함으로써, 본 발명은 계약 만기 연장 및 갱신과 같은 다양한 작업을 수행하거나 이를 서브 계약이나 조건으로 분할하기 위한 강력한 메커니즘을 제공할 수 있다. 더욱이, 계약의 상태 및 존재가 블록체인을 통해 증명되므로, 계약의 영구적, 공개 가시적 및 변경할 수 없는 기록을 제공한다.

(52) CPC특허분류
***G06Q 30/06*** (2013.01)
***H04L 9/0643*** (2013.01)
*G06Q 2220/12* (2013.01)
*H04L 2209/56* (2013.01)

(30) 우선권주장
1603117.1 2016년02월23일 영국(GB)
1603123.9 2016년02월23일 영국(GB)
1603125.4 2016년02월23일 영국(GB)
1605571.7 2016년04월01일 영국(GB)
1619301.3 2016년11월15일 영국(GB)

# *명 세 서*

## *청구범위*

### 청구항 1

계약의 이행 및/또는 가시성을 제어하는 컴퓨터 구현 방법에 있어서,

(a) 컴퓨터 기반 저장소에 계약을 저장하는 단계,

(b) 블록체인에 거래를 브로드캐스팅하는 단계,

상기 거래는,

i) 적어도 하나의 UTXO, 그리고

ii) 상기 계약이 저장된 위치를 나타내는 식별자를 포함하는 메타데이터를 포함하고,

(c) 아래에 의해 계약을 갱신 또는 연장하는 단계를 포함하는 방법:

상기 계약에 연관된 이전 키에 관한 데이터를 사용하여 새로운 키를 생성하는 것,

상기 새로운 키, 상기 계약의 위치 및 상기 계약의 해시를 포함하는 스크립트를 생성하는 것, 그리고

상기 스크립트에 통화량을 지불하는 것.

### 청구항 2

제1항에 있어서,

상기 거래는, 결정성 리딤 스크립트(redeem script) 주소를 더 포함하고,

여기서, 상기 리딤 스크립트 주소는 페이 투 스크립트 해시(P2SH) 주소인 방법.

### 청구항 3

제2항에 있어서,

상기 출력(UTXO)을 소비하기 위해 블록체인에 추가 거래를 브로드캐스트하여 상기 계약을 종결하는 단계를 더 포함하는 방법.

### 청구항 4

제1항 내지 제3항 중 어느 하나에 있어서,

상기 추가 거래는,

상기 출력(UTXO)인 입력, 그리고

서명, 상기 메타데이터 및 공개키를 포함하는 해제 스크립트를 포함하는 방법.

### 청구항 5

제1항 내지 제4항 중 어느 하나에 있어서,

상기 계약은,

i) 적어도 하나의 조건, 그리고

ii) 상기 조건의 평가에 따라 수행이 달라지는 적어도 하나의 실행(action)을 포함하는 방법.

### 청구항 6

제1항 내지 제5항 중 어느 하나에 있어서,

상기 메타데이터는,

i) 컴퓨터 기반 저장소에 저장된 상기 계약의 주소 표현 또는 주소, 및/또는

ii) 상기 계약의 해시를 포함하는 방법.

**청구항 7**

제1항 내지 제6항 중 어느 하나에 있어서,

상기 미사용 거래 UTXO가 상기 블록체인에 대한 미사용 거래 출력 목록에 있는지 결정하는 것에 의해 상기 계약이 종결되었는지 여부를 확인하는 단계를 포함하는 방법.

**청구항 8**

제1항 내지 제7항 중 어느 하나에 있어서,

상기 계약은,

분산 해시 테이블(DHT)에 저장되는 방법.

**청구항 9**

제1항 내지 제8항 중 어느 하나에 있어서,

명시된 날짜 및/또는 시간에서 상기 출력을 소비하는 명령어를 포함하는 상기 블록체인에 거래를 브로드캐스팅하고,

상기 명령어는,

CheckLockTimeVerify 명령어인 방법.

**청구항 10**

제1항 내지 제9항 중 어느 하나에 있어서,

상기 계약의 내용의 일부 또는 전부에 대한 접근은 적어도 하나의 지정된 권한 당사자로 제한되는 방법.

**청구항 11**

제1항 내지 제10항 중 어느 하나에 있어서,

상기 계약은,

상기 계약을 이행하기 위한 결정성 유한 오토머턴(DFA)을 포함하는 방법.

**청구항 12**

제11항에 있어서,

상기 결정성 유한 오토머턴은,

부호화 스키마(codification scheme)를 이용하여 정의되는 방법.

**청구항 13**

제11항 또는 제12항에 있어서,

상기 결정성 유한 오토머턴은,

i) 적어도 하나의 블록체인 거래, 바람직하게는 스크립팅 언어를 이용하여,

ii) 상기 블록체인의 상태를 모니터링하도록 구성된 컴퓨팅 에이전트(agent), 및/또는

디지털 지갑에 대한 명령어 세트를 이용하여 구현되는 방법.

**청구항 14**

계약의 이행 및/또는 가시성을 제어하는 컴퓨터 구현 방법에 있어서,

(a) 컴퓨터 기반 저장소에 계약을 저장하는 단계,

(b) 블록체인에 거래를 브로드캐스팅하는 단계,

상기 거래는,

i) 적어도 하나의 UTXO, 그리고

ii) 상기 계약이 저장된 위치를 나타내는 식별자를 포함하는 메타데이터를 포함하고,

(c) 상기 계약으로부터 파생된 서브-계약을 생성하는 단계를 포함하고,

여기서, 상기 서브 계약은, 결정성 주소(deterministic address)와 연관되거나,

iii) 시드를 이용하여 파생된 새로운 공개키를 이용하는 것,

iv) 상기 계약에 대한 참조로 상기 저장소에 서브-계약을 저장하고, 상기 참조를 포함하는 스크립트를 포함하는 상기 블록체인에 거래를 브로드캐스팅하는 것, 및/또는

v) 상기 기존 계약의 상기 메타데이터에 상기 서브-계약에 대한 참조를 추가하는 것에 의해 생성되는 방법.

**청구항 15**

제14항에 있어서,

상기 거래는, 결정성 리딤 스크립트(redeem script) 주소를 더 포함하고,

여기서, 상기 리딤 스크립트 주소는, 페이 투 스크립트 해시(P2SH) 주소인 방법.

**청구항 16**

제15항에 있어서,

상기 출력(UTXO)을 소비하기 위해 상기 블록체인에 추가 거래를 브로드캐스팅함으로써 상기 계약을 완료하는 단계를 더 포함하는 방법.

**청구항 17**

제14항 내지 제16항 중 어느 하나에 있어서,

상기 추가 거래는,

상기 출력(UTXO)인 입력, 그리고

서명, 상기 메타데이터 및 공개키를 포함하는 해제 스크립트를 포함하는 방법.

**청구항 18**

제14항 내지 제17항 중 어느 하나에 있어서,

상기 계약은,

i) 적어도 하나의 조건, 그리고

ii) 상기 조건의 평가에 따라 수행이 달라지는 적어도 하나의 실행(action)을 포함하는 방법.

**청구항 19**

제14 내지 제18항 중 어느 하나에 있어서,

상기 메타데이터는,

i) 컴퓨터 기반 저장소에 저장된 상기 계약의 주소 표현 또는 주소, 및/또는

ii) 상기 계약의 해시를 포함하는 방법.

**청구항 20**

제14항 내지 제19항 중 어느 하나에 있어서,

상기 미사용 거래 UTXO가 상기 블록체인에 대한 미사용 거래 출력 목록에 있는지 결정하는 것에 의해 상기 계약이 종결되었는지 여부를 확인하는 단계를 포함하는 방법.

**청구항 21**

제14항 내지 제20항 중 어느 하나에 있어서,

상기 계약은,

분산 해시 테이블(DHT)에 저장되는 방법.

**청구항 22**

제14항 내지 제21항 중 어느 하나에 있어서,

명시된 날짜 및/또는 시간에서 상기 출력을 소비하는 명령어를 포함하는 상기 블록체인에 거래를 브로드캐스팅하고,

여기서, 상기 명령어는,

CheckLockTimeVerify 명령어인 방법.

**청구항 23**

제14항 내지 제22항 중 어느 하나에 있어서,

상기 계약의 내용의 일부 또는 전부에 대한 접근은 적어도 하나의 지정된 권한 당사자로 제한되는 방법.

**청구항 24**

제14항 내지 제23항 중 어느 하나에 있어서,

상기 계약은,

상기 계약을 이행하기 위한 결정성 유한 오토머턴(DFA)을 포함하는 방법.

**청구항 25**

제24항에 있어서,

상기 결정성 유한 오토머턴은,

부호화 스키마(codification scheme)를 이용하여 정의되는 방법.

**청구항 26**

제24항 또는 제25항에 있어서,

상기 결정성 유한 오토머턴은,

i) 적어도 하나의 블록체인 거래, 바람직하게는 스크립팅 언어를 이용하여,

ii) 상기 블록체인의 상태를 모니터링하도록 구성된 컴퓨팅 에이전트(agent), 및/또는

디지털 지갑에 대한 명령어 세트를 이용하여 구현되는 방법.

**청구항 27**

제1항 내지 제26항 중 어느 하나의 방법을 수행하도록 구성되는 시스템.

## *발명의 설명*

## 기 술 분 야

[0001] 본 발명은 일반적인 컴퓨터 프로토콜에 관한 것으로, 더 구체적으로, 예를 들어, 계약과 관련한 것들과 같은 조건 제어 프로세스의 집행, 강화 및/또는 수행에 관한 것이다. 본 발명은 블록체인 네트워크를 이용하기에 특히 적합하며, 스마트 계약에 유리하게 이용될 수 있다.

## 배 경 기 술

[0002] 블록체인은 결과적으로 거래로 이루어지는 변경이 불가능한 블록으로 형성된 분산, 분배 컴퓨터 시스템이다. 각 블록은 이전 블록의 해시를 포함하고, 따라서 블록은 그 시작부터 블록체인에 기록된 모든 거래의 기록을 생성하기 위해 서로 연결된다. 거래는 입력 및 출력에 포함된 스크립트로 알려진 작은 프로그램을 포함하며, 이는 어떻게 또는 누구에 의해 거래의 출력이 접근될 수 있는지를 특정한다. 각 미사용 거래(UTXO로 지칭되는)는 새로운 거래에 입력으로서 소비될 수 있다.

[0003] 다른 블록체인 구현이 제안되고 개발되었지만, 가장 널리 알려진 블록체인 기술의 애플리케이션은 비트코인 원장이다. 비트코인은 편의와 설명의 목적으로 여기서 언급되지만, 본 발명은 비트코인 블록체인을 이용하는 것에 제한되지 않으며 다른 블록체인 구현은 본 발명의 범위 내에 있다는 점에 유의해야 한다.

[0004] 블록체인 기술은 암호화폐 구현의 용도로 알려져 있다. 그러나, 더 최근에는, 디지털 기업이 새로운 시스템을 구현하기 위해 비트코인이 기반하는 암호화 보안 시스템의 용도 및 블록체인 상에 저장될 수 있는 데이터를 탐구하기 시작했다. 이는 다음을 포함하지만 이에 제한되지 않는다.

[0005] • 메타데이터 저장하기

[0006] • 디지털 토큰 구현하기

[0007] • 계약 이행 및 관리하기

[0008] 현대 계약 관리의 주요 문제점 중 하나는 수동으로 유지 관리되는 계약 사본 및 지역 상점을 통한 임시방편인 경향이 있다는 것이다. 그 결과, "스마트 계약"으로 알려진 컴퓨터 프로토콜은 부분적으로 또는 전체적으로 계약의 자동 집행 또는 수행을 가능하게 할 수 있어 주목을 끌기 시작했다. 스마트 계약은 보안 강화 및 거래 비용 감소와 같은 이점을 제공할 수 있다. 그러나, 이러한 계약이 일단 저장되면 수정할 수 없도록 하는 것을 목표로 하는 알려진 기술 솔루션이 있지만, 일반적으로 계약의 정당성(예를 들어, 계약이 여전히 유효한지 종결되었는지)을 확인하기 위해 공개 레지스트리를 채택하지 않는다.

[0009] 따라서, 계약의 존재에 대한 대중적 가시성을 제어할 수 있고, 자동화된 방식(즉, 인간 관리보다는 기계에 의한)의 계약과 같은 이행 기반 프로세스를 관리, 집행 및 유지하기 위한 관련 당사자들의 능력을 촉진할 수 있는 컴퓨터 구현 메커니즘을 제공하는 것이 바람직하다. 중요한 것은, 이러한 메커니즘은 계약에서 정의된 행동에 대한 트리거(trigger) 및 제어 조건을 명시하는 기술 능력을 제공할 수 있다.

## 발명의 내용

### 해결하려는 과제

### 과제의 해결 수단

[0010] 여기서 정의되고 설명된 본 발명은 단어의 법적 의미에서의 계약으로 사용이 제한되지 않는다는 점을 알아야 한다. 계약은 문서, 파일 또는 명시된 조건 하에서 트리거될 수 있는 일련의 행동을 정의하는 다른 메커니즘일 수 있다. 제어 조건은 공개적으로 충족될 수 있다. 본 발명은 법적 또는 상업상의 상황 내에서의 사용으로 제한되는 것으로 간주되어서는 안되며, '계약'이라는 단어는 제한적 의미로 해석되어서는 안된다.

[0011] 따라서, 본 발명은 첨부된 청구항에 정의된 바와 같이 제공된다.

[0012] 본 발명의 일 측면에 따르면, 계약의 이행 및/또는 가시성을 제어하는 컴퓨터 구현 방법에 있어서, 상기 방법은

[0013] (a) 컴퓨터 기반 저장소에 계약을 저장하는 단계,

[0014] (b) 블록체인에 거래를 브로드캐스팅하는 단계,

[0015] 상기 거래는,

[0016] i) 적어도 하나의 UTXO, 그리고

[0017] ii) 상기 계약이 저장된 위치를 나타내는 식별자를 포함하는 메타데이터를 포함하고,

[0018] (c) 아래에 의해 계약을 갱신 또는 연장하는 단계를 포함하는 방법:

[0019] 상기 계약에 연관된 이전 키에 관한 데이터를 사용하여 새로운 키를 생성하는 것,

[0020] 상기 새로운 키, 상기 계약의 위치 및 상기 계약의 해시를 포함하는 스크립트를 생성하는 것, 그리고

[0021] 상기 스크립트에 통화량을 지불하는 것이다.

[0022] 계약과 관련된 이전 키에 관한 데이터를 이용하여 새로운 키를 생성하고, 새로운 키, 계약의 위치 및 계약의 해시를 포함하는 스크립트를 생성하고, 스크립트에 통화량을 지불하는 것에 의해 계약을 갱신 또는 연장함으로써, 이것은 새로운 키가 이전 키와 연관되기 때문에 승인된 당사자는 갱신 또는 연장된 계약과의 연결에 의하여 원천 계약을 볼 수 있고, 이에 따라 확실성이나 프라이버시의 손실 없이 연장된 계약을 확인할 수 있다는 이점을 제공한다. 추가 이점은 갱신 또는 연장된 계약에 관한 키는 원천 계약의 키에 연관되기 때문에 확실성이나 프라이버시의 손실 없이 컴퓨터 기반 오프-체인 저장소에 계약을 저장함으로써(즉, 블록체인의 일부를 형성하지 않음) 메모리 및 프로세서 용량을 줄일 수 있다는 것이다.

[0023] 만기 연장 상황에서, UTXO는 새로이 연장된 계약에 이를 보낼 수 있다. 그러나, 잠금 시간 이전에 출력을 소비하여 전체 계약을 취소함으로써 기존 계약을 취소할 수 있다.

[0024] 본 발명은 계약의 수행 및/또는 가시성을 제어하기 위한 컴퓨터 구현 방법 및 시스템을 제공할 수 있다. 가시성은 어떻게 그리고 누구에 의해 계약의 내용 및/또는 존재가 이용가능하거나 접근가능한지를 의미할 수 있다. 계약은 '스마트 계약'일 수 있다. 방법은 자동화된 스마트 계약 방법일 수 있다. 계약의 수행, 검증 및/또는 존재를 모니터링하는 프로세스를 자동화하기 위한 방법일 수 있다. 계약은 블록체인 거래의 적어도 일부의 형태로 표현될 수 있으므로, 본 발명은 토큰화 방법/시스템으로 언급될 수 있다. 거래의 메타데이터는 계약을 표현 및/또는 접근하는데 이용되는 블록체인 구현 토큰을 제공할 수 있다.

[0025] 본 발명은 저장소(레지스트리)에 계약의 저장을 가능하게 하는 방법/시스템을 제공할 수 있고, 여기서 계약의 해시는 계약을 찾기 위한 룩업키로서 이용될 수 있다.

[0026] 방법은 컴퓨터 기반 저장소에 계약을 저장하는 단계, 블록체인에 거래를 브로드캐스팅하는 단계를 포함할 수 있고,

[0027] 상기 거래는 적어도 하나의 UTXO, 그리고 상기 계약이 저장된 위치를 나타내는 식별자를 포함하는 메타데이터를 포함한다.

[0028] 계약은 블록체인에서 UTXO가 소비될 때까지 열린 또는 유효한 것으로 해석될 수 있다. 블록체인은 비트코인 블록체인이거나 아닐 수 있다. UTXO에 의해 표현되는 블록체인 상의 계약 유효성 또는 상태를 표현하기 위한 새로운 메커니즘의 이점을 제공한다.

[0029] 방법은 블록체인의 상태를 관찰하고 출력이 현재 미사용인지 아닌지에 기초하여 특정 방법으로 행동하기 위하여 오프-체인 컴퓨터 구현 프로세스, 에이전트 또는 다른 독립체를 이용하는 단계를 포함할 수 있다. 프로세스는 계약 상태의 지표로서 미사용 출력을 해석하도록 구성될 수 있다. 다시 말해, 블록체인 상의 UTXO 목록 내에 출력이 남아 있는 동안(즉, 거래가 아직 미사용), 메타데이터에 의해 언급되거나 지시된 계약의 '열린' 상태 또는 유효성을 나타내기 위해 사용될 수 있다. 계약은 UTXO가 소비되면 완료(종결)된 것으로 간주될 수 있다. 이 조건은 계약 내 언급될 수 있다. 그러나 UTXO가 소비되면, 메타데이터는 계약에 대한 포인터 또는 참조 및 계약의 해시를 계속 포함할 수 있고 따라서 계약은 그 기능을 유지할 수 있다.

[0030] 방법은 계약의 존재를 공개하는 단계를 더 포함할 수 있다. 이는 아래의 단계에 의해 수행될 수 있다.

[0031] • 계약 발급자는 새로운 계약 문서를 생성할 수 있고 저장소에 이를 공개할 수 있다. 해당 문서의 보안 해시 및 저장 위치는 이후 사용을 위해 저장될 수 있다.

[0032] • n개의 다중 서명 구조 중 m개에서 보안된 계약 문서를 다루는 리딤 스크립트를 생성한다. 여기서,

[0033] o m은 적어도 하나이고,

[0034] o n은 m에 메타데이터 블록의 수를 더한 것이다.

[0035] • 스크립트에 적어도 하나의 공개키를 포함한다. 이는 계약 발급자의 공개키일 수 있다. 그러나, 다른 서명이 필요할 수 있다.

[0036] • 바람직하게는 P2SH 거래를 통해, 스크립트에 통화량(예를 들어, 비트코인)을 지불한다.

[0037] • 거래가 블록체인에 공개될때까지 기다리고 공개 거래에 대한 거래 ID를 추출한다.

[0038] • 계약 만료 시간 설정된 잠금 시간을 통해 다시 공개키 해시에 거래의 출력을 지불하는 새로운 거래를 생성한다. 또는, 연장 기간 계약의 경우. 블록체인 상의 거래를 검출하고 새로운 계약으로 연장하기 위해 코드를 트리거하기 전 계약 만료 시간까지 기다리기 위해 자동화된 컴퓨팅 에이전트를 이용한다.. 또는, 완료 기반 계약(여기서 y개 독립체 중 x개는 계약이 충족되었다는 것에 동의함)의 경우, n개의 다중 서명 거래 중 m개를 생성하고 완료시 공동 서명하도록 독립체에 이를 발행한다.

[0039] 저장소는 오프-블록 저장소 자원일 수 있다. 다시 말해, 저장소는 블록체인 자체의 일부를 형성하지 않을 수 있다. 컴퓨터 기반 저장소는 서버이거나 이를 포함할 수 있다. 저장소는 데이터베이스 또는 컴퓨터 기반 자원상에 제공되는 다른 저장 시설일 수 있다. 저장소는 검색되도록 하기 위해 인덱스될 수 있다. 저장소는 분산 해시 테이블을 포함할 수 있다. 계약은 분산 해시 테이블(DHT)과 연관되거나 분산 해시 테이블에 저장될 수 있다.

[0040] 거래는 결정성 리딤 또는 잠금 스크립트 주소를 더 포함할 수 있다. 주소는 페이 투 스크립트 해시(P2SH) 주소일 수 있다. 따라서 계약의 존재(또는 계약 내 정의된 요소)는 계약의 발급자 및/또는 계약의 메타데이터에 의해 제공되거나 결정될 수 있는 페이 투 스크립트 해시 주소를 이용하여 블록체인에 공개된 거래를 이용하여 공개될 수 있다.

[0041] 방법은 출력(UTXO)을 소비하기 위해 블록체인에 (추가) 거래를 브로드캐스팅함으로써 계약을 종결시키는 단계를 더 포함할 수 있다. 추가 거래는 출력(UTXO)인 입력, 그리고 서명, 메타데이터 및 공개키를 포함하는 해제 스크립트를 포함할 수 있다.

[0042] 이는 출력을 소비하기 위한 블록체인 거래의 이용을 통해 계약의 자동 종결 이점을 제공할 수 있다.

[0043] 계약은, i)적어도 하나의 조건, 그리고 ii) 조건의 평가에 따라 수행이 달라지는 적어도 하나의 실행을 정의할 수 있다. 조건은 참 또는 거짓으로 평가될 수 있는 테스트일 수 있다. 조건은 계약의 일부(예를 들어, 조항)일 수 있다. 조건의 수행 또는 완료는 계약의 실현을 위해 요구될 수 있다. 계약은 참으로 평가되면 완료될 수 있다.

[0044] 메타데이터는 i) 컴퓨터 기반 저장소에 저장된 계약의 주소 표현 또는 주소, 및/또는 ii) 계약의 해시를 포함할 수 있다.

[0045] 방법은 블록체인 상태를 관찰하는 단계를 포함할 수 있다. UTXO를 포함하는 거래를 찾기 위해 블록체인을 검색하는 단계를 포함할 수 있다. 미사용 거래 UTXO가 블록체인에 대한 미사용 거래 출력 목록에 있는지 결정하는 것에 의해 계약이 종결되었는지 여부를 확인하는 단계를 포함할 수 있다. 이 모니터링 또는 확인 단계는 자동화될 수 있다. 니는 적합하게 프로그램된 컴퓨팅 에이전트 또는 자원에 의해 수행될 수 있다. “본 발명과 함께 사용하기 위한 예시적인 컴퓨팅 에이전트”라는 제목의 섹션에서 실질적으로 설명될 수 있다. 에이전트는 UTXO의 사용 또는 미사용 상태에 기초하여 실행을 수행할 수 있다. 따라서, UTXO의 상태는 오프-블록 컴퓨팅 에이전트의 동작에 영향을 미치거나 이를 제어할 수 있다.

[0046] 방법은 명시된 날짜 및/또는 시간에서 상기 출력을 소비하는 명령어를 포함하는 블록체인에 거래를 브로드캐스팅하는 단계를 포함할 수 있다. 명령어는, CheckLockTimeVerify 명령어일 수 있다.

[0047] 계약의 내용의 일부 또는 전부에 대한 접근은 적어도 하나의 지정된 권한 당사자로 제한될 수 있다. 다시 말해, 계약의 일부 또는 전부에 접근하거나 보기 위하여 권한이 요구될 수 있다. 몇몇 예에서, 보호 메커니즘은 계약 자체에 적용될 수 있다. 예를 들어, 파일의 하나 이상의 부분은 보호될 수 있지만 전체 내용을 공개될 수 있다.

이 부분적 보호는 계약 내 정보의 암호화 뿐만 아니라 그 내용의 변경을 감지하는 해시 모두에 적용될 수 있다.

[0048] 계약은 계약을 이행하기 위한 결정성 유한 오토머턴(DFA)을 포함할 수 있다. 결정성 유한 오토머턴은 부호화 스키마를 이용하여 정의될 수 있다. 결정성 유한 오토머턴은, i) 적어도 하나의 블록체인 거래, 바람직하게는 스크립팅 언어를 이용하여, ii) 블록체인의 상태를 모니터링하도록 구성된 컴퓨팅 에이전트, 및/또는디지털 지갑에 대한 명령어 세트를 이용하여 구현될 수 있다.

[0049] 본 발명의 다른 관점에 따르면, 계약의 이행 및/또는 가시성을 제어하는 컴퓨터 구현 방법에 있어서, (a) 컴퓨터 기반 저장소에 계약을 저장하는 단계, (b) 블록체인에 거래를 브로드캐스팅하는 단계, 거래는, i) 적어도 하나의 UTXO, 그리고 ii) 계약이 저장된 위치를 나타내는 식별자를 포함하는 메타데이터를 포함하고, (c) 계약으로부터 파생된 서브-계약을 생성하는 단계를 포함하고, 여기서, 서브 계약은, 결정성 주소(deterministic address)와 연관되거나, iii) 시드를 이용하여 파생된 새로운 공개키를 이용하는 것, iv) 상기 계약에 대한 참조로 상기 저장소에 서브-계약을 저장하고, 참조를 포함하는 스크립트를 포함하는 블록체인에 거래를 브로드캐스팅하는 것, 및/또는 v) 기존 계약의 메타데이터에 서브-계약에 대한 참조를 추가하는 것에 의해 생성되는 방법을 제공한다.

[0050] 계약으로부터 파생된 서브 계약을 생성하고, 여기서 서브 계약은 결정성 주소와 관련되고 시드를 이용하여 파생된 새로운 공개키를 이용하여 생성되며, 계약과 관련 저장소에 서브 계약을 저장하고, 참조 및/또는 기존 계약의 메타데이터에 서브 계약에 대한 참조를 추가하는 것를 포함하는 스크립트를 포함하는 블록체인에 거래를 브로드캐스팅함으로써, 서브 계약이 원천 계약에 암호로 묶여 있기 때문에 확실성이나 프라이버시의 손실 없이 독립적으로 관리될 수 있는 이점을 제공한다. 또한, 메모리 및 프로세싱 자원은 오프-블록 저장소에 서브 계약을 저장함으로써 최소화될 수 있다.

[0051] 방법은 블록체인을 모니터링 및/또는 계약 내용에 기초하여 실행을 수행하기 위한 컴퓨터 기반 에이전트의 이용을 포함할 수 있다. 에이전트는 아래의 "본 발명과 함께 이용되는 예시적인 컴퓨팅 에이전트"로 명명된 섹션에서 대체로 설명될 수 있다.

[0052] 또한, 본 발명은 상기에서 언급된 모든 방법 단계 또는 여기서 설명된 방법의 모든 실시예를 수행하도록 구성된 컴퓨터 구현 시스템을 제공할 수 있다. 본 발명은 계약의 이행 및/또는 가시성을 제어하는 컴퓨터 구현 시스템을 제공하며, 시스템은 계약을 저정하도록 구성된 컴퓨터 기반 저장소, 거래를 포함하는 블록체인을 포함하고, 거래는 i) 적어도 하나의 미사용 출력(UTXO), 그리고 ii) 계약이 저장된 위치를 나타내는 식별자를 포함하는 메타데이터를 포함한다.

[0053] 메타데이터는 또한 계약의 해시를 저장할 수 있다. 계약은 스마트 계약일 수 있다.

[0054] 저장소는 데이터베이스를 포함할 수 있다. 이는 DHT를 포함할 수 있다. 이는 검색할 수 있도록 인덱스될 수 있다. 이는 계약에의 접근을 제어하기 위하여 적어도 하나의 보안 메커니즘을 포함할 수 있다.

[0055] 시스템은 또한 적합하게 구성된 컴퓨팅 기반 독립체 또는 에이전트를 포함할 수 있다. 에이전트는 블록체인을 모니터링 및/또는 검색하도록 구성될 수 있다. 블록체인의 상태에 기초하여 적어도 하나의 실행을 수행하도록 구성될 수 있다. UTXO가 소비되었는지 아닌지를 결정하도록 구성될 수 있다. UTXO가 소비되었는지 아닌지에 기초하여 하나 이상의 실행을 수행하도록 구성될 수 있다.

[0056] 하나의 실시예 또는 개념과 관련하여 여기서 설명된 어떤 특징은 또한 다른 실시예 또는 개념에 관련되어 이용될 수 있다. 예를 들어, 방법에 관련하여 설명된 어떤 특징은 또한 시스템에 관련하여 이용될 수 있고 그 반대도 가능하다.

[0057] 본 발명에 의 해 제공될 수 있는 몇가지 이점의 비-한정적 목록이 제공된다.

[0058] 본 발명은 여기서 계약으로 언급될 수 있는 구조 제어 조건의 자동 관리를 단순화하는 기술 구성을 제공할 수 있다. 이는 결과적으로 분쟁 발생시 계약 상태에 동의하는 것을 쉽게 만든다. 본 발명은 또한 컴퓨터에 의한 유효성의 자동 결정을 허용하는 방식으로 계약의 공개 기록을 안전하게 하고 유효성 확인시 승인된 독립체에 세부 정보를 공개하는 메커니즘을 제공할 수 있다. 따라서, 본 발명은 지능적인 방식으로 자원에 접근을 허용하거나 금지하는 보안 강화 제어 메커니즘을 제공할 수 있다.

[0059] 본 발명은 계약의 세부 사항이 승인된 독립체에만 제한될 수 있어 컴퓨터 시스템을 통해 청중에게 계약을 공개할 수 있는 능력을 제공하며, 계약의 존재에 대한 지식은 공개적으로 알려진다. 다시 말해, A와 B 사이의 계약이 있다는 것은 공개적으로 알려질 수 있고, 공개적으로 검증될 수 있다. 그러나, 그것의 존재 이외에 다른 것

은 승인된 당사자들에 제한된다(일반적으로 A 및 B일 수 있다).

[0060] 또한, 계약이 시간부(즉, 주어진 날짜 또는 특적 시간 후에 만료됨), 조건부(즉, 계약 내 특정된 상품이 충족되면 만료됨), 또는 제한 없음(즉, 종결시키는 통지 기간을 통해 계속 연장됨)이 되도록 하는 컴퓨터 구현 메커니즘을 제공한다.

[0061] 공개적인 방식으로 해당 계약을 종결시키는 통지를 제공할 수 있는 메커니즘을 제공할 수 있다. 예를 들어, 만기를 제정하기 위해 소비 거래에 nLockTime + CheckLockTimeVerify(CLTV)를 이용하는 것.

[0062] 분할되는 계약의 다른 측면을 제어하기 위하여, 결정성 방식에서 서브 계약의 계층 구조를 구조화하는 메커니즘을 제공할 수 있다. 예를 들어, 기술 개발 프로세스에서, 요구 사항 단계는 개발 단계와는 다른 제어 트리거 세트를 가질 수 있다.

[0063] 본 발명은 블록체인 플랫폼 상에서 구현될 수 있고, 기술적으로 상이한 방법에서 이용될 수 있어 블록체인의 기능을 확장할 수 있으므로, 본 발명은 개선된 블록체인 시스템 또는 플랫폼을 제공할 수 있다.

[0064] 본 발명은 어떤 미사용 거래(UTXO)를 디지털 접근과 같은 스마트 계약으로 전환하는데 이용될 수 있다. 예를 들어, 소비자가 일정 시간 동안 거래에 접근하기 위해 판매자에게 비용을 지불하는 시나리오를 고려한다. 만약 판매자의 지불 주소가 스마트 계약으로 구현되면, 본 발명은 서비스에 대한 접근 제어 메커니즘을 구현하는데 이용될 수 있다. 돈이 지불되었는지를 확인하기 위해 점검이 될 수 있고, 기간 만료시 판매자의 계좌에 가치를 스윕(sweep)하는데 이용되는 자동화 프로세스를 수행할 수 있다.

## *도면의 간단한 설명*

[0065] 본 발명의 이들 및 다른 양태는 본원에 기재된 실시예를 참조하여 명백하게 설명될 것이다. 본 발명의 실시 예는 첨부된 도면을 참조하여 예로서 만 설명될 것이다.

도 1은 다양한 계약 관련 작업을 수행하기 위해 본 발명의 실시예에 의해 어떻게 블록체인 거래가 사용될 수 있는지에 대한 개요를 보여준다.

도 2a는 두 개의 상태를 가지는 간단한 상태 머신을 보여준다 : (i) 계약이 개시된다 (ii) 계약이 종료된다

도 2b는 도 2a의 시나리오에 대한 메타데이터 정의를 보여준다. 메타데이터는 (비트코인) 거래 출력에 전달되며 (해시를 통해) 계약 위치 및 유효성 증명을 특정한다.

도 2c는 도2a 및 도 2b의 시나리오에 관한 '발행' 거래를 보여주며 처음에는 블록체인 상의 계약(의 해시)을 저장한다.

도 2d는 비트코인을 소비함으로써 도 2a 및 도 2c의 계약을 취소한다.

도 3a는 시나리오에 대한 예시적인 메타데이터를 보여주며, 여기서 숨겨진 소유권을 가진 자산이 블록체인에 생성되고 게시된다.

도 3b는 도 3a의 자산을 '운용'하기 위한 예시적인 거래를 보여준다. 즉, 자산의 공개키 내에 일부 비트코인을 넣어 자산이 그 거래(도 3c에 도시된 공개 거래와 같은)를 운용할 수 있도록 한다.

도 3c는 도 3a 및 도 3b의 자산의 발생에 대한 예시적인 블록체인 거래를 보여준다.

도 3d는 도 3a, 도 3b 및 도 3c에 관한 계약 종료에 대한 예시적인 거래를 보여준다. 계약 취소가 요청될 때, UTXO는 소비된다. 이 시나리오에서, 요청은 자산 및 서명할 자산의 숨은 소유자에 대한 것이다.

도 4a는 임대 계약을 포함하는 시나리오에 대한 예시적인 상태 머신 모델을 보여준다.

도 4b는 도 4a의 시나리오에 대한 예시적인 메타데이터를 보여준다.

도 4c는 블록체인 상에 도 4a 및 도 4b의 자산의 소유권을 공개하기 위한 예시적인 거래를 보여준다.

도 5a는 시나리오에 대한 예시적인 상태 머신 모델을 보여주며, 여기서 계약은 체결된다.

도 5b는 도 5a의 시나리오에 대한 예시적인 메타데이터를 보여준다.

도 5c는 블록체인 상에 계약의 최초 연장 및 도 5a 및 도 5b의 최초 계약을 공개하는데 이용될 수 있는 예시적인 거래를 보여준다.

도 5d는 도 5a 내지 도 5d의 계약 종결에 대한 예시적인 거래를 보여준다.

도 6a는 계약 조건을 포함하는 시나리오에 대한 예시적인 상태 머신 모델을 보여준다.

도 6b는 도 6a의 시나리오에 대한 예시적인 메타데이터를 보여준다.

도 6c는 최초 계약 및 두 개의 서브 계약을 생성하고 이들을 공개하기 위해 이용될 수 있는 예시적인 거래를 보여준다.

도 6d는 도 6a 내지 도 6c의 시나리오에 관련하여 사용하기 위한 예시적인 거래를 보여준다.

도 7 내지 도 13은 부모키로부터 서브키를 도출하기 위한 기술의 다양한 양상을 도시하며, 이 기술은 본 발명의 양상과 관련하여 사용하기에 적합하다.

## *발명을 실시하기 위한 구체적인 내용*

[0066] 블록체인에 내장된 스마트 계약은 외부 컴퓨터 기반 애플리케이션을 통해 및/또는 비트코인 거래 내(즉, 잠금/해제 스크립트 내)에 직접 내장된 로직을 통해 집행될 수 있다. 이러한 외부 컴퓨터 기반 애플리케이션은 '에이전트(agent)', '오라클(oracle)' 또는 '봇(bot)'으로 불릴 수 있다. 또한, 몇몇 계약 조건은 nLockTime 필드와 같은 다른 비트코인 거래 요소를 통해 집행될 수 있다.

[0067] 본 발명은 여기서 계약을 나타내는 블록체인 상에 유효한 미사용 거래 출력 UTXO가 존재하는 한 계약이 유효한 것으로 해석되는 것으로 설명된다. 이러한 미사용 상태는 행동이 계약 자체의 약정 또는 조건에 의해 제어되는 다양한 메커니즘(예를 들어, 프로그램된 컴퓨팅 에이전트)의 결과로서 변경되고 영향을 받을 수 있음을 이해해야 한다. 예를 들어, 계약은 특정 날짜에 만료되거나 특정값이 특정된 임계값에 도달하여 만료될 것을 규정한다.

[0068] 계약을 나타내기 위해 미사용 출력을 이용하는 원리는 암화화 기술과 같은, 다른 특징과 결합하여 사용될 수 있다. 이는 복잡한 시나리오 및 활동을 구현하도록 한다. 실제로, 그것이 사용될 수 있도록 하는 스크립트 내의 관련 메타데이터 및 서명되지 않은 거래 출력 UTXO 주변의 상황은 거래가 계약의 정식 세부 사항을 포함하는 오프-체인(off-chain) 저장소에 대한 참조 또는 포인터로서 실행하도록 한다. 여기서, '오프-체인'은 블록체인 자체의 일부가 아니라는 의미이다. 이는 누구나 블록체인을 검사함으로써 계약이 여전히 유효/개방인지 종결되었는지를 결정하기 위한 도구 또는 소프트웨어 기반 구성 요소를 이용할 수 있는 메커니즘을 제공한다. 계약이 종결되면, 이는 거래의 사용 출력으로서 블록체인 상에 기록될 수 있고, 이는 공개 검사가 가능하다. 블록체인 거래는 계약의 존재 및 현재 상태의 영구적, 불변적, 그리고 공개적 기록이 된다.

[0069] ('레지스트리' 또는 '레지스터'라고 하는) 저장소는 예를 들어, 분산 해시 테이블(DHT)을 비롯한 다양한 방식으로 구현될 수 있다. 계약의 해시는 블록체인 거래 내 메타데이터로서 저장되고 생성될 수 있으며, 블록체인으로부터 계약을 참조하기 위한 룩업키 역할을 할 수 있다. 또한, 계약의 위치에 대한 참조는 거래 메타데이터 내에 제공된다. 예를 들어, 저장소에 대한 URL이 제공된다. 메타데이터가 공개적으로 볼 수 있도록 열려 있지만, 계약 자체는 부분적으로 보호되거나 그렇지 않을 수 있다.

[0070] CheckLockTimeVerify (CLTV)와 같은 표준 비트코인 특징은 계약이 향후 시점에서 공식적인 자동 만료가 되도록 할 수 있다. 블록체인의 사용은 사용 만료일이 안전한 (불변의) 공개 기록으로 간주될 수 있도록 한다. 아래에 설명된 다중 암호화 키의 사용과 결합되는 이 개념은 CLTV 모델이 명시적으로 취소하지 않는 한 계약을 자동적으로 체결 또는 갱신하도록 한다.

[0071] 여기서 설명된 토큰화 메커니즘과 결합하는 결정성 서브키의 사용은 계약에 대한 일정 또는 서브 계약이 생성되도록 한다.

[0072] 더욱이, 오프-블록 컴퓨팅 에이전트(오라클)의 사용은 계약 조건이 신뢰할 수 있는 제3자에 의해 수정되고 내장되도록 한다. 에이전트의 실행은 계약 정의 내에 제공된 조건(예를 들어, "IF"문)에 영향을 받을 수 있다는 것을 의미한다.

[0073] 키 용어

[0074] 다음 용어는 여기에 사용될 수 있다.

[0075] • 계약 발급자

[0076] 이 독립체는 블록체인 상 계약의 공개에 대한 책임이 있는 관여자를 나타낸다.

[0077] • 이해 관계자

[0078] 이 독립체는 특정 계약이 여전히 존재하는지 아닌지를 결정할 필요가 있을 수 있거나 계약의 특정을 결정할 필요가 있을 수 있는 관여자를 나타난다.

[0079] • 저장소

[0080] 이 독립체는 블록체인 스마트 계약이 참조하는 계약의 구조화된 표현을 저장/보호하는 위치를 나타낸다.

[0081] • 계약 상대방

[0082] 이 독립체는 특정 계약의 상대방을 나타낸다. 많은 경우에서, 이 독립체는 존재하지 않을 수 있음을 알아야 한다.

[0083] • 계약

[0084] 이는 저장소 내에 저장된 구조화된 문서 또는 파일이고, 이는 블록체인에서 참조된다. 계약은 모든 유형의 계약 또는 합의일 수 있다. 예를 들어, 금융 계약, 부동산 권리증서, 서비스 계약 등이 포함될 수 있다. 계약은 내용면에서 공개적 또는 개인적일 수 있다. 계약은 부호화 스키마를 이용하여 구조화된 방식으로 표현된다는 점에서 공식화된다.

[0085] 계약 모델

[0086] • 계약 모델의 기본 요소는 아래와 같다.

[0087] 계약의 어떠한 형태를 완전히 설명할 수 있는 부호화 스키마. 스키마는 XBRL, XML, JSON 등과 같은 기존 설비를 사용할 수 있거나 새로운 구조일 수 있다.

[0088] • 부호화 스키마 내에 전체로 정의될 수 있는 계약을 수행하기 위한DFA(결정성 유한 오토머턴). 이는 아래와 같이 구성된다.

[0089] o 파라미터의 세트, 그리고 그 파라미터를 공급하는 장소

[0090] o 상태 정의 세트

[0091] o 전환을 위한 트리거와 전환에 따른 규칙을 포함하는 상태 사이의 전환 세트

[0092] o 규칙 정의 테이블

[0093] • 계약의 인스턴스에 대한 특정 파라미터의 정의

[0094] • 계약을 보호하고 안전하게 하는 메커니즘

[0095] • 컨택(contact)을 공식 법률 언어로 사람이 읽을 수 있게 하는 '브라우저'

[0096] • 비트코인 스크립트와 같은 스크립트 및/또는 오라클 코드 내 부호화 스키마를 변환하는 '컴파일러'

[0097] 계약 이행하기

[0098] 계약이 저장소에 등록되면, 관련 주소(예를 들어, URL) 및 해시는 계약 자체의 제어와 함께 체인상 거래에 관련하여 블록체인 거래 내 메타데이터로서 이용될 수 있다. 이는 다양한 형태로 수행될 수 있지만, 적절한 부호화 스키마는 "부호화 스키마"로 명명된 섹션에서 완전성을 위해 아래에 제공된다.

[0099] 계약 정의 내에 포함된 DFA가 어떻게 수행될 수 있는지에 대한 많은 상이한 방법이 있다.

[0100] • 블록체인 거래 또는 거래 시퀀스. DFA의 다양한 형태는 비트코인 스크립트 언어 내에서 직접 구현될 수 있다. 기술 분야의 당업자는 이를 이해할 것이며, 본 발명은 DFA가 블록체인 거래(s)를 통해 구현되는 방식과 관련하여 제한되지 않는다.

[0101] • 에이전트 기반(예를 들어, 오라클) 프로세스 또는 프로세스의 시퀀스. "본 발명과 함께 이용하기 위한 예시적인 컴퓨팅 에이전트"라고 아래에서 명명된 섹션은 적절한 에이전트가 블록체인 및 가능한 다른 외부 소스를 모니터링하도록 정의하고 실행하는 기본 프로세스를 설명한다.

[0102] • 스마트 지갑에 대한 명령어 세트. 이 콘텐츠에서, 스마트 지갑은 사실상 단순히 블록체인에 대한 거래 입력 할당과 같은 특정 계약 조건을 처리할 수 있는 로컬 오라클 프로세스이다.

[0103] 주어진 계약 정의가 위의 3개 메커니즘의 혼합으로 구현될 수 있으며, 여기서 각 계약 상태 전환은 사실상 별도 구현이다. 관련 거래/코드를 수작업하는 것을 포함하여 계약 정의로부터 구현을 생성하는 여러가지 방법이 있다.

[0104] 계약 존재 공개하기

[0105] 계약 존재(또는 계약 내 정의된 요소)를 공개하기 위해, 거래 Tx는 페이 투 스크립트 해시 주소(P2SH)를 이용하여 블록체인에 공개된다. P2SH 거래는 거래가 소비되기 위해서, 스크립트 해시를 매칭하는 스크립트 및 스크립트를 참(true)으로 평가하는 데이터를 수령인이 제공해야 한다. 본 발명의 실시예와 관련하여, 페이 투 스트립트 해시(P2SH)는 다음으로부터 쉽게 결정될 수 있다.

[0106] - 계약의 발급자, 그리고

[0107] - 계약의 메타데이터

[0108] 본 발명의 몇몇 실시예에 따르면, 미사용 거래는 계약 상태의 지표로서 해설될 수 있다. 오프-체인 프로세스는 블록체인을 모니터링하고 출력이 미사용인지 아닌지에 기초한 특정 방식으로 동작하도록 구성될 수 있다. 다시 말하면, 블록체인 상의 UTXO 목록 내에 출력이 남아있는 동안(즉, 거래가 여전히 미사용), 이것은 메타데이터에 의해 지시되거나 참조된 계약의 유효성을 나타낸다. 이 출력이 소비되면 계약은 완료된 것으로 간주된다. 이 조건(UTXO가 존재하는 한 계약이 유효(valid/open)한 상태로 유지됨)은 계약 자체의 조건일 수 있다. 그러나, 다른 예에서 대체 종결 조건이 준비되어 있는 것처럼 프로토콜의 필수 조건은 아니다. 거래가 소비된 후(그러므로 더 이상 UTXO 목록에 존재하지 않는)라도, 블록체인 상의 영구적으로 상주하고 계약의 해시 및 계약에 참조 또는 포인트를 여전히 보유하므로, 소비된 후에도 그 기능을 유지할 수 있다.

[0109] 서브-계약/조건

[0110] 서브 계약은 기존 계약에 직접 관련된 계약이다. 조건은 그 계약의 조건을 만족시키기 위해 성취되어야 할 기존 계약 내의 조항이다.

[0111] 본 발명의 일 실시예에 따르면, 서브-계약 및 조건 모두 동일한 방식으로 구현될 수 있다(즉, 결정성 리딤 스크립트 주소를 가지는 UTXO로서 이행되는 계약으로서). 두 경우 모두, 독립체는 UTXO가 소비될 때 완료된 것으로 해석될 수 있다(조건의 경우에서, 이는 조건이 충족되었음을 나타낸다). 위에서 언급했듯이, 메타데이터는 저장소 및 그 해시 내에서 독립체의 위치에 대한 포인터 또는 참조를 여전히 포함할 것이다. 그러므로, 다른 실시예에서, 계약된 특정 조건에 따라, 서브-계약 또는 조건은 존재를 유지할 수 있고, 출력이 소비된 후에도 기능을 유지할 수 있다.

[0112] 조건 또는 서브 계약에 대한 결정성 주소를 생성하는데 사용될 수 있는 여러 메커니즘이 있다.

[0113] - 시드 정보를 이용하여 새로운 공개키를 도출하는 것

[0114] - 주 계약을 참조하여 저장소 내에 서브 계약을 생성 및 공개하는 것 그리고 메타데이터 참조로서 이를 이용하는 것

[0115] - 기존 계약의 메타데이터에 대한 조건/서브 계약 참조를 추가하는 것

[0116] 계약 보호하기

[0117] 계약의 공식적인 표현(즉, 계약의 내용을 특정하는 파일 또는 문서)은 특정 계약의 공식적인 필요에 따라 다양한 방식으로 보호될 수 있으며, 모든 경우에 계약 존재의 공개 기록은 메타데이터 기록 내에 포함된 블록체인 상에 공개된다(특정 메타데이터 구조의 세부 사항에 "부호화 스키마"로 명명된 섹션을 참조).

[0118] 블록체인 기록으로부터, 승인된 독립체는 거래가 공개된 이후에 공식 표현이 수정되지 않았다는 것을 결정하기 위해 해시와 함께 공식 표현의 위치를 학습할 수 있다.

[0119] 그러나, 다양한 방법을 통해 공식 표현 자체를 더 안전하게 할 수 있다.

[0120] - 문서 저장소 자체는 접근 제어 메커니즘을 나타낼 수 있다. 그리고

[0121] - 계약 자체는 관련 암호해독키에 대한 접근 권한을 가진 독립체에 대한 접근을 제한하는 표준 암호화 기술을 통해 보호될 수 있다.

[0122] 많은 경우에서, 계약 자체는 부분적으로 보호될 수 있다. 예를 들어, 전체 콘텐츠가 공개되는 반면 파일 내 몇몇 섹션은 보호될 수 있다(예를 들어, 어떻게 고정 금리 대출을 어떻게 구현하는지에 대한 세부 사항은 공개되나 누가 대출을 받는지, 얼마나 많이 그리고 이율이 어떤지는 계약 당사자에게만 알려진다.)

[0123] 이러한 부분적인 보호는 그 콘텐츠 변경을 감지하는 해시 뿐만 아니라 계약 내 정보의 암호화 모두에 적용된다.

[0124] 많은 계약에 대해, 계약의 세부 사항은 그 수명 동안 수정될 수 있으며 계약 자체의 재발급을 요구해서는 안된다. 계약의 서브 세트를 통해 해시의 범위를 결정함으로써 달성될 수 있다. 이것이 유용할 수 있는 예는 단위형 투자신탁의 구현에 있다. 단위형 투자신탁을 지지하는 계약은 변하지 않으나, 단위에 대한 수혜자는 계약의 제3자 매도를 통해 수정될 수 있다. 일 실시예에서, 변화를 기록하는 것은 서브 계약을 이용하여 달성될 수 있다.

[0125] 계약 종료하기

[0126] 블록체인은 거래의 영구적이고 불변의 기록을 제공하므로, 계약은 단순히 관련 계약 문서를 제거함으로써 종결될 수 없다. 이는 안전한 계약 저장소가 많은 표준 메커니즘을 통해 지원되는 블록체인 자체와 동일한 저장소 및 보존 규칙을 가져야 한다는 것을 의미한다. 이는 솔루션이 직접 블록체인 기록을 통해 계약 종료를 감지하는 메커니즘을 제시해야 한다는 것을 의미한다.

[0127] 종료 방법은 계약에 조건으로서 정의되고 다양한 방법으로 수행될 수 있고 이들 모두는 본 발명에 의해 개념적으로 포함된다. 본 발명의 바람직한 실시예에서, 종료는 계약을 나타내는 UTXO의 소리를 통해 처리된다.

[0128] 여러 계약 형태의 경우, 계약 종료는 계약 자체의 공개와 동시에 공개될 수 있따. 효과적으로 두 개의 거래는 생성되며, 하나는 계약을 공개하고 계약을 나타내는 거래 출력을 얻는 것이고, 둘은 그 출력을 소비하는 것이다. 이 두번째 거래는 주어진 미래 날짜(계약 종료를 나타냄)에 출력을 소비하기 위하여 CheckLockTimeVerify 세트를 가진다. 이전 내용에 따르면, 이것은 표준 방법이지만 유일한 것은 아니다. 이러한 자동 소비는 계약의 계속을 지원하기 위해 확장될 수 있다(예를 들어, 취소되지 않으면 12개월 더 자동 연장되는 계약). 이 상황에서, UTXO는 이를 '새로운' 계속 계약에 전송하는데 사용된다. 그러나, 잠금 시간 전에 출력을 소비하고 전체 계약을 취소하여 오래된 것을 취소할 수 있다.

[0129] 사용 케이스 모델

[0130] 도 1은 본 발명의 실시예에 따른 사용 케이스 모델의 개략을 보여준다. 이러한 예시적인 사용 케이스 모델은 어떻게 표준 비트코인 거래가 비트코인 스크립트 내 직접적으로 DFA의 요소를 구현하는데 이용될 수 있는지를 보여준다. 키 사용 케이스의 예는 설명의 목적을 위해 제공된다.

[0131] 계약 생성하기

[0132] 계약 발급자(본 예시의 주 관여자)는 공개 가시성을 위해 블록체인 상에 계약을 공개할 수 있다. 이 프로세스는 표 1에 요약된다.

*표 1*

[0133]

| 단계 | 상세한 설명 |
|---|---|
| 100.10 | 계약 발급자는 새로운 계약 문서를 생성하고, 나중에 사용할 수 있도록 해당 문서의 보안 해시 및 저장 장소를 저장하는 저장소에 이를 공개한다. 이 저장소는 계약 문서 자체의 성질에 따라 공개적, 개인적 또는 반 개인적(semi-private)일 수 있다는 것에 유의해야 한다. 저장소는 다양한 속성에 의해 검색될 수 있도록 인덱스된다. |
| 100.20 | 계약 발급자는 n개의 다중 서명 구조 중 m개에서 보안된 계약 문서를 다루는 리딤 스크립트를 생성한다. 여기서,<br>- m은 적어도 하나이고,<br>- n은 m에 메타데이터 블록의 수를 더한 것이다(적어도 2개가 됨).<br>이 스크립트에 항상 제공되어야 하는 하나의 공개키는 계약 발급자의 것이다. 그러나 계약의 조건에 따라 다른 서명이 필요할 수도 있다. |

| | |
|---|---|
| 100.30 | 계약 발급자는 명목 통화량(예를 들어, 비트코인)을 표준 P2SH 거래를 통해 100.20 단계에서 계산되는 리딤 스크립트에 지불한다. |
| 100.40 | 계약 발급자는 거래가 블록체인 상에 공개되고 공개 거래에 대한 거래 ID를 추출할때까지 기다린다. |
| 100.50 | 고정 기간 계약의 경우, 계약 발급자는 100.40 단계로부터의 결과물을 계약의 만료 시간으로 설정된 잠금 시간을 통해 다시 계약 발급자의 공개키 해시에 지불하는 새로운 거래를 생성한다.<br>롤링(연장ing) 기간 계약의 경우, 컴퓨터 기반 에이전트는 표 3의 '연장(연장over)' 사용 케이스를 트리거 하기 전에 계약에서 이를 굴리기 위해 거래를 선택하고 계약 만료 시간까지 기다릴 수 있다.<br>완료 기반 계약의 경우(여기서, y개의 독립체 중 x개는 계약이 이행되었음에 동의함), n개의 다중 서명 거래 중 m개는 완료 후 공동 서명하기 위해 이러한 독립체에 발생되고 생성된다. |

[0134] 아래에서 상세하게 설명된 이 시나리오의 두 개의 주요한 구현예 및 변형예가 있다

[0135] • 기존 계약에서 서브 계약 생성

[0136] • 기존 계약을 새로운 계약으로 롤오버(갱신)

[0137] 서브 계약 생성하기

[0138] 이 상황에서, 계약 발급자는 기존 계약으로부터 서브 계약을 생성하기 원한다. 이러한 프로세스는 표 2에 요약된다.

표 2

[0139]

| 단계 | 상세한 설명 |
|---|---|
| 150.10 | 계약 발급자는 부모 계약으로부터 서브키 정보를 도출하는데 시드값을 이용하여 부모 계약을 생성하는데 이용되는 그들의 공개키로부터 새로운 서브키를 생성한다. 이는 계약 발급자가 원하는(그리고 약속한) 어떠한 도출이 될 수 있지만 적절한 시드의 예는 다음을 포함할 수 있다.<br>- 거래 ID / 100.40 단계에서 생성된 계약 UTXO의 인덱스, 또는<br>- 100.20 단계에서 생성된 리딤 스크립트 해시<br>본 예에서는 상기에서 언급된 공개키가 계약 발급자의 공개키임을 가정한다. 그러나 당업자는 파생된 서브키(즉, 서브 계약의 서브 계약)인 이것을 막을 수 있는 방법이 없다는 것을 인식할 것이다. |
| 150.20 | 생성되는 서브 계약의 성격에 따라 계약 발급자는 다음중 하나를 수행한다.<br>- 주 계약 문서의 해시 및 위치를 사용한다. 또는,<br>- 내장된 주 계약 문서에 대한 링크를 통해 새로운 계약 문서를 생성하고, 이후 사용을 위해 해당 문서의 보안 해시 및 문서의 위치를 저장한다.<br>- 내장된 주 계약 문서에 대한 링크를 통해 새로운 계약 문서를 생성하고, 다뤄진 원본 계약 문서로부터 필드 목록을 추가한다. 실제로, 이는 서브 계약이 원본 정보를 복제하는 것이 아니라 또 다른 문서의 특정 섹션을 다루는 것을 지정한 문서이다.<br>이 저장소는 계약 문서 자체의 성격에 따라 공개적, 개인적 또는 반 공개적일 수 있음에 유의해야 한다. |
| 150.30 | 계약 발급자는 n개의 다중 서명 구조 중 m개에서 보안된 계약 문서를 다루는 리딤 스크립트를 생성한다. 여기서,<br>- m은 적어도 하나이고,<br>- n은 m에 메타데이터 블록의 수를 더한 것이다(적어도 2개가 됨).<br>이 스크립트에 항상 제공되어야 하는 하나의 공개키는 계약 발급자의 것이다. 그러나 계약의 조건에 따라 다른 서명이 필요할 수도 있다. |
| 150.40 | 계약 발급자는 명목 통화량(예를 들어, 비트코인)을 표준 P2SH 거래를 통해 150.30 단계에서 계산되는 리딤 스크립트에 지불한다. |
| 150.50 | 계약 발급자는 거래가 블록체인 상에 공개되고 공개 거래에 대한 거래 ID를 추출할때까지 기다린다. |
| 150.60 | 고정 기간 계약의 경우, 계약 발급자는 150.50 단계로부터의 결과물을 계약의 만료 시간으로 설정된 잠금 시간을 통해 다시 계약 발급자의 공개키 해시에 지불하는 새로운 거래를 생성한다. |

[0140] 하나 이상의 실시예에 따르면, 서브 계약은 독립적으로 모니터링 될 수 있다. 예를 들어, 감정인으로부터 승인이 요청되고 계약이 '< x>의 승인 대상'을 명시하는 부동산 건설 계약을 고려한다. 이를 구현하기 위하여, 150.60 단계는 서명을 위해 <x>에게 배포되고 생성된다. 재지불 스크립트는 잠긴 시간이 아니지만 요청된 서명

자가 <x>인 n개의 다중 서명 요소 중 m개로 생성된다. 일부 실시예에서, 거래는 두 개의 출력을 가질 것이다:<x>에 대한 수수료와 150.50 단계에서 생성된 UTXO의 지불금.

[0141] 예시 사용 케이스 : 기존 계약 만기 연장

[0142] 이 사용 케이스에서, 계약 발급자는 새로운 계약에서 기존 계약의 만기 연장을 원한다. 표 3에서 제공된 예시적인 프로세스:

표 3

[0143]

| 단계 | 상세한 설명 |
|---|---|
| 175.10 | 계약 발급자는 이전 UTXO가 소비되었는 아닌지를 검증하는 것에 의해 계약이 취소되었는지 아닌지를 결정하기 위하여 블록체인을 체크할 것이다. 만약 소비되었다면, 프로세스는 종료된다. |
| 175.20 | 계약 발급자는 부모 계약 시퀀스로부터 서브키 정보를 도출하는데 시드값으로 이를 이용하여, 부모 계약을 생성하는데 이용되는 그들의 공개키로부터 새로운 서브키를 생성한다. 이는 계약 발급자가 원하는(그리고 약속한) 어떠한 결정성 도출이 될 수 있지만 다음일 수 있다.<br>- 시퀀스 수(예를 들어, 연장된 인스턴스 '1'), 또는<br>- 연장된 계약에 대한 날짜 범위<br>위는 위에서 언급된 공개키가 계약 발급자의 공개키 일 수 있음을 가정하나, 실제로 파생된 서브키(즉, 서브 계약의 서브 계약)인 이것을 막을 수 있는 방법은 없다. 서브키가 어떻게 생성될 수 있는지에 대한 예시에 대해서는 “서브키 생성 방법”으로 명명된 섹션을 참조한다. |
| 175.30 | 계약 발급자는 기존 계약 문서의 해시 및 위치를 가진다.<br>이 저장소는 계약 문서 자체의 성격에 따라 공개적, 개인적 또는 반 공개적일 수 있음에 유의해야 한다. |
| 175.40 | 계약 발급자는 n개의 다중 서명 구조 중 m개에서 보안된 계약 문서를 다루는 리딤 스크립트를 생성한다. 여기서,<br>- m은 적어도 하나이고,<br>- n은 m에 메타데이터 블록의 수를 더한 것이다(적어도 2개가 됨).<br>이 스크립트에 항상 제공되어야 하는 하나의 공개키는 계약 발급자의 것이다. 그러나 계약의 조건에 따라 다른 서명이 필요할 수도 있다. |
| 175.50 | 계약 발급자는 명목 통화량(예를 들어, 비트코인)을 표준 P2SH 거래를 통해 175.40 단계에서 계산되는 리딤 스크립트에 지불한다 |
| 175.60 | 계약 발급자는 거래가 블록체인 상에 공개되고 공개 거래에 대한 거래 ID를 추출할때까지 기다린다. |
| 175.70 | 프로세스(봇 또는 오라클 기반 구현과 같은)는 취소되지 않은 경우 다시 시작하기 위해 표 3의 '연장' 프로세스를 재트리거 하기 전에 거래를 선택하고 계약 만료 시간까지 기다릴 것이다. |

[0144] 예 : 계약 확인하기

[0145] 이 사용 케이스에서, 이해 당사자는 자신이 탐구하고 있는 활동을 다루기 위한 계약이 있음을 확인하기를 원한다. 이러한 프로세스는 표 4에 나와 있다.

표 4

[0146]

| 단계 | 상세한 설명 |
|---|---|
| 200.10 | 관심 당사자는 그들이 관심을 가지고 있는 계약에 관한 UTXO가 소비되었는지 아닌지를 확인하기 위해 블록체인을 확인할 것이다. 여기서, UTXO가 여전히 소비되지 않은 경우, 계약은 유효하다. 여기서, UTXO가 여전히 소비되지 않은 경우, 그러나 보류 중인 잠금 시간이 있으면, 계약에 대한 만료 시간을 결정할 수 있다. 여기서 UTXO가 소비된 경우, 계약은 어떤 면에서 완료된다. |

[0147] 상기의 주요 변수는 이해 당사자가 몇몇 다른 루트를 통해 계약을 지배하는 거래를 알고 있다고 가정한다(일반적으로 계약 발급자 또는 계약 상대자 중 하나임). 그러나, 계약 발급자에 대한 지식 및 계약 문서에 접근할 수 있는 어떤 독립체는 다음에 의해 확인할 수 있을 것이다.

[0148] - UTXO 거래에 대한 리딤 스크립트를 도출하는 것, 그리고

[0149] - 리딤 스크립트 해시를 매칭하는 것을 통해 UTXO를 찾기 위하여 블록체인을 스캔하는 것.

[0150] 예 : 계약 종료하기

[0151] 이 사용 케이스에서, 계약 발급자 또는 계약 당사자는 기존 계약을 종료하길 원한다. 이 프로세스는 표 5에 도

시된다.

표 5

[0152]

| 단계 | 상세한 설명 |
|---|---|
| 300.10 | 종료의 주동자는 이전 UTXO가 소비되었는지 아닌지를 검증하는 것에 의해 계약이 취소되었는지 아닌지를 결정하기 위하여 블록체인을 확인할 것이다. 만약 소비되었다면, 프로세스는 계약이 이미 종료되었으므로 종료된다. |
| 300.20 | 기존 종료 거래가 있다면, 주동자는 이 거래에 서명하고 블록체인 상에 제출할 것이다. |
| 300.30 | 기존 종료 거래가 없다면, 주동자는 마지막 계약의 UTXO인 거래 입력, 그들의 서명인 해제 스크립트, 계약에 관한 메타데이터 그리고 그들의 공개키를 통해 거래를 생성할 것이다. |
| 300.40 | 거래가 플록체인 상에서 수락되는 시점에서 계약이 종료되었음을 공개적으로 알 수 있다(비록 참가자만이 특정 이유를 알 수 있음). |

[0153] 계약 조건

[0154] 위에서 설명된 동일한 메커니즘은 체크 포인트와 같은 주어진 계약 내 조건을 모니터링하는데 이용될 수 있다. 예를 들어, 계약이 100BTC의 가치가 있는 것으로 결정되고, 20 BTC가 1에서 5까지의 체크 포인트에서 지급되면, 위에서 설명된 서브 계약 모델은 하나의 주 계약과 5개의 서브 계약을 도출하는데 이용될 수 있다. 이들 서브 계약 각각은 동일 또는 상이한 서명자(예를 들어, 공증인 또는 이와 유사한 것)를 이용하여 완전한 것으로 표시할 수 있다. 이 방식에서, 공개 기록은 유지되고 계약에 첨부된 조건이 충족되었음을 보여줄 수 있다. 그러면 이 개념을 일단 계약이 완료로 표시되면 20 BTC 지불을 트리거하는데 사용할 수 있는 프로세스 또는 애플리케이션('봇')과 결합할 수 있다.

[0155] 예시의 목적을 위해, 본 발명이 사용될 수 있는 몇몇 애플리케이션을 보여주는 몇몇 예시 시나리오가 아래에서 제공된다. 이러한 시나리오 모두에서, 계약 자체의 내용은 관련이 없으며 비제한적으로 간주된다.

[0156] 예시 시나리오 1 : 자산의 공개 레지스트리

[0157] 이 시나리오에서, 밥은 블록체인 상에 자산(예를 들어, 그의 집)의 소유권을 공개하기로 결정한다. 이 단계에서는 완료된 것이 없다. 단지 후속 거래에서 이용될 수 있는 자산이다. 이 상황에서, 계약에 대한 종료 날짜는 없다. 도 2a는 두 개 상태를 통해 단순 상태 머신을 보여준다. (i) 계약이 개시되고 (ii)계약이 종료된다. 도 2b는 비트코인 거래 출력에 전달된 메타데이터 정의를 보여주며 해시를 통해 유효성 증명 및 계약 위치를 지정한다. 도 2c는 블록체인 상 계약을 최초 저장하는 '발행' 거래를 보여준다(비록, 실제로는 실제 계약이 아닌 해시만 저장한다). 도 2d는 비트코인을 소비함으로써 계약을 취소한다.

[0158] 예시 시나리오 2 : 숨겨진 소유권을 가진 자산의 레지스트리 및 생성

[0159] 이는 시나리오 1의 약간 강화된 버전으로, 여기서 밥은 블록체인 상에 자산을 공개하길 원하나 그의 소유권을 직접 밝히길 원하진 않는다.

[0160] 이 상황에서, 밥은 우선 자산을 나타내기 위해 그의 공개키로부터 서브키를 생성한다. 그러면, 이 서브키는 블록체인 상에 자산의 세부 정보의 일부로서 공개된다. 다시 한번, 이 상황에서, 자산에 대한 종료일은 없다(상세한 예는 서브키가 생성되는 하나의 방법에 대해 아래에서 제공된다. 아래에서 "서브키 생성 방법"으로 명명된 섹션을 참조한다).

[0161] 이 시나리오의 상태 머신은 도 2a에 도시된 것처럼, 시나리오 1에 대한 것과 동일하다. 도 3a는 이 시나리오에 대한 메타데이터 정의를 보여준다. 메타데이터는 비트코인 거래 출력에 실리고, 계약 위치를 특정하고, 유효성을 증명한다(해시를 통해). 도 3b는 자산을 '운용'하기 위한 거래를 보여준다. 이는, 자산이 거래를 운용할 수 있도록 자산의 공개키에 일부 비트코인을 넣는 것이다(도 3c에서 공개 거래와 같은). 도 3b는 비트코인 거래가 아니므로 밥의 자산 서브키 생성을 보여주진 않는다.

[0162] 도 3c는 자산의 공개에 대한 블록체인 거래를 보여준다. 도 3d는 계약의 종료에 대한 거래를 보여준다. 계약의 취소가 요청되면, UTXO는 소비된다. 이 상황에서, 요청은 자산과 자산의 숨겨진 소유자 모두가 서명하도록 요청된다.

[0163] 예시 시나리오 3 : 임대 계약

[0164] 이 예시적인 상황에서, 밥은 3년의 고정된 기간 동안 이브와 임대 계약을 체결한다. 계약 조건은 지불 횟수를

명시한다. 지불의 세부 사항은 본 발명과 관련이 없다. 그러나 계약은 중단 조항이 없는 고정된 기간을 가진다.

[0165] 이것은 도 4a에 도시된 것처럼 단순한 상태 머신 모델이다. 도 4b는 이 시나리오에 대한 메타데이터를 보여준다. 도 4c는 블록체인 상에 자산의 소유권을 공개하기 위한 거래를 보여준다. 우선, 밥이 자산에 몇몇 자금을 제공하면, 자산은 스스로 공개한다.

[0166] 예시 시나리오 4 : 연장 계약

[0167] 이 예시적인 상황에서, 밥은 연장(rolling) 연간 기준으로 이브로부터 주택을 임대하고로 결정한다. 여기서 그는 갱신 날짜에 임대를 취소하기 위해 두 달의 통지를 제공할 필요가 있고 그렇지 않으면 자동적으로 연장(rolled-on)된다. 이는 도 5a에 도시된 단순한 상태 머신 모델을 가진다. 도 5b는 이 시나리오에 대한 메타데이터를 보여준다. 도 5c는 블록체인 상에 계약의 최초 만기 연장(rollover) 및 최초 계약을 공개하기 위한 거래를 보여준다.

[0168] 첫해 이후, 밥은 임대를 계속했고 종료하지 않았다. EVE-S3-T2이 공개된 후 이는 바로 자동화된 컴퓨팅 에이전트에 의해 선택되고 다른 해에 연장된다. 그녀 자신의 내부 로직을 이용하는 EVE에 의해 완료될 가능성 또한 있음에 유의해야 한다.

[0169] 두번째 해 후, 밥은 임대를 종료하기로 선택하고, EVE-S3-T3와 동일한 입력을 이용하여 거래를 제출한다. 그러나 이 거래는 아직 제출되지 않았기 때문에, 입력은 사용되지 않고, 밥의 거래가 블록체인에 먼저 공개되면 EVE-S3-T3를 무효로 할 것이다. 관련 금액은 사소하지만, 봇은 출력이 이브의 공개키 해시(또는 계약에 실제로 명시된 것)로 향하지 않는 한 거래를 부서하지 않을 것이다. 밥의 계약의 종료에 대한 거래는 도 5d에 도시된다.

[0170] 예시 시나리오 5 : 계약 조건

[0171] 이 예시적인 상황에서, 밥은 새로운 부동산을 인도하기 위해 건설업자 집단과 계약을 체결한다. 그리고 독립적인 승인(sign-off)을 요구하는 계약 내에 다수의 조건을 지정한다(첫번째는 지방 계획 당국으로부터의 계획 승인이다). 이는 도 6a에 도시된 것처럼 단순 상태 머신 모델을 가진다. 도 6b는 이 시나리오에 대한 메타데이터를 보여준다. 도 6c는 거래를 보여주며, 여기서 밥은 최초 계약 및 두 개의 서브 계약을 생성하고(아래에 설명된 서브키 생성 기술을 이용하여 관련 서브키를 파생한 후), 이들을 공개한다. 도 6d는 계획 허가가 언제 승인되었는지에 대한 거래를 보여준다.

[0172] **부호화 스키마**

[0173] 계약을 참조하는데 이용되는 메타데이터는 다양한 방식으로 형식을 지정할 수 있다. 그러나 적합한 부호화 스키마는 아래에 설명된다.

[0174] 계약은 정의한 권리가 계약의 소유자 또는 보유자에게 부여되면 양도가 가능하다. 비양도성 계약의 예시는 참가자가 지명된 계약, 즉 계약의 보유자가 아닌 특정 지명된 독립체에 권리가 부여되는 계약이다. 오직 양도가 가능한 계약이 이 부호화 스키마에서 논의된다.

[0175] 토큰은 계약에 의해 부여된 권리를 상세히 설명하거나 정의하는 특정 계약을 나타낸다. 본 발명에 따르면, 토큰은 비트코인 거래의 형식에서 계약의 표현이다.

[0176] 이 체계화 방법은 세 개의 파라미터 또는 데이터 아이템을 포함하는 메타데이터를 이용한다. 이 데이터는 다음으로 나타낼 수 있다.

[0177] i) 계약하에서 가능한 공유의 양

[0178] (이는 여기서 ' NumShares'일 수 있다.)

[0179] ii) 송신자로부터 적어도 하나의 수령인에게 전송되는 전송 유닛의 양

[0180] (이는 여기서 'ShareVal'일 수 있다.)

[0181] iii) 전송 유닛의 양에 대한 값을 계산하기 위한 요소

[0182] (이는 여기서 'pegging rate'일 수 있다.)

[0183] 이 부호화 스키마의 이점은 위에서 설명된 세 개의 파라미터만 이용하여 블록체인 상에 토큰으로 계약을 표현하거나 캡슐화하는데 이용될 수 있다는 것이다. 결과적으로, 계약은 이러한 세 개의 데이터 아이템의 최소를 이용

하여 특정될 수 있다. 부호화 스키마가 양도가능한 계약의 모든 형태에 대해 시용될 수 있으므로, 일반적인 알고리즘이 고안되고 적용될 수 있다. 이러한 메타데이터 아이템의 더 상세한 사항은 아래에서 제공된다.

[0184] 분할 가능 토큰은 거래 출력 상의 값이 다중 토큰을 통해 할당된 더 작은 양으로 세분될 수 있다(즉, 다중 거래를 통해 할당됨). 원형은 신용 화폐로 토큰화된다. 분할가능한 계약은 0이 아닌 고정율(pegging rate)을 특정하는 것으로 정의된다. 분할가능한 계약의 경우, 거래 출력에 전송된 토큰화된 값은 고정율을 통해 기반 비트코인(BTC) 값에 연결된다. 이는 계약이 고정율에 관하여 보유자의 권리를 특정하는 것이다. 분할할 수 없는 토큰의 경우, 고정율이 없고 계약은 고정된 값에 관하여 보유자의 권리를 특정한다(예를 들어, 무기명 채권 : 정확히 1000달러의 상황이 가능하다 또는 '이 계약은 한번의 이발에 대한 상환이 가능하다'는 바우처). 분할할 수 없는 계약의 경우, 기반 거래 BTC 값은 계약값과 관련이 없다.

[0185] "기반 BTC 값"이라는 구절은 거래 출력에 첨부된 비트코인 양(BTC)을 나타낸다. 비트코인 프로토콜에서, 모든 거래 출력은 가치를 고려하여 0이 아닌 BTC 양을 가져야 한다. 사실, BTC 양은 작성 시점에서 현재 546 사토시로 설정된 세트 최소값('먼지(dust)'로 알려진)보다 커야 한다. 1 비트코인은 1억 사토시와 같은 것으로 정의된다. 비트코인 거래가 여기서 오직 소유권의 교환을 용이하게 하기 위한 수단으로서 이용되기 때문에, 실제 기반 BTC 양은 임의적이다. 실제 값은 계약 명세서에 있다. 이론상 모든 토큰은 먼지에 의해 운반될 수 있다.

[0186] 본 부호화 스키마에 따르면, 특히 분할 가능한 토큰의 경우, 기반 BTC 값은 수단을 가진다: 고정율을 통해 계약값과의 관계를 유지. 고정율은 그 자체로 임의적이고 기반 BTC 양을 작게 유지하도록 선택된다. 먼지를 통한 모든 토큰 거래를 단순화하는 대신 고정율을 이용하는 이유는 본 발명의 프로토콜이 가분성을 용이하게 하기 때문이다. 토큰이 더 작은 수량의 몇몇 거래 출력에 나누어지면, 원본 계약을 조정할 필요가 없다. 각 세분화된 토큰의 계약값은 고정율과 세분화된 기반 BTC 값의 양에 기초하여 단순히 계산된다.

[0187] 제한된 토큰은 NumShares로 불리는 양에 의해 정의된 고정된 0이 아닌 수에 의해 총 발행 수가 고정(제한)된 것이다. 그러므로, 제한된 계약 하에서 추가적인 주식(Shares)은 발생되지 않는다. 예를 들어, 경주마의 부분 소유권에 대한 계약이 경주마의 100%로 제한된다(예를 들어, 1% 당 100 주 또는 10%당 10 주 등). 무제한 계약은 발급자가 예를 들어, 그들의 지급 준비 계정(Reserve Account)에 신용 화폐의 요청량을 추가함으로써, 주식의 추가 발행을 기명할 수 있다는 것을 의미한다.  NumShares는 모든 계약에 명시되어야 한다. 제한된 계약은 NumShares>0이어야 하고, 무제한 계약은 NumShares=0으로 설정하여 나타낸다.

[0188] 전형예는 예금 은행 계좌가 보유한 총액이 현존하는 약속 어음의 총액과 일치시키는 것과 같은 통화 준비금(currency reserve)이다(금 보유와 유사). 기 개념은 통화 준비금을 넘어서 재고 목록을 포함한다. 예를 들어, 허가된 인쇄 티셔츠 토큰의 발급자는 재고가 있는 10000장의 티셔츠 목록을 가지고 시작할 수 있고, 10000장의 티셔츠를 나타내기 위해 분배가능한 토큰을 발행할 수 있다(여기서, 각 주식 = 1개의 티셔츠이다). 원본 토큰은 세분화될 수 있고 각 세분화된 토큰은 페깅률에 의해 정의된 출력의 기반 BTC 값에 따라 얼마간의 티셔츠에 대해 상환될 수 있다. 그러나 수요가 증가하면, 발급자는 추가 주식을 발행하기로 결정할 수 있다(즉, 또 다른 10000개에 의해 유통되는 주식의 수를 늘린다). 그러한 경우 추가 발생을 기명하기 위해서 발급자는 그의 지급 준비 계정(즉, 재고 창고)에 10000개의 티셔츠를 더 입금해야 한다. 따라서, 한번에 재고가 있는(여기서 재고는 '지급 준비 계정'으로서 역할한다) 티셔츠의 총 수 = 상황되지 않은 주식의 총 수 이다.

[0189] 고정율은 오직 분할가능한 계약에만 적용되며, 여기서, 주식의 가치(ShareVal로 불리는 수량으로 표시된)는 기반 BTC 양에 고정된다. 예를 들어, 계약은 발급자가 매 기반 1 BTC에 대해 10000달러의 비율로 토큰을 상환하도록 명시될 수 있다. 예를 들어, 15400 사토시의 토큰화된 기반 출력값을 가진 거래는 1.54 달러로 상황될 수 있다. 고정율에 대한 0의 값은 계약이 분할되지 않는다는 것을 나타낸다(즉, 무기명 채권처럼 전체를 양도할 수 있음). 고정율이 0(토큰 분할이 안된다는 의미)이면, 기반 BTC 값은 계약 값에 관련이 없고 어떤 금액이든 설정될 수 있다. 일반적으로 이러한 경우에서, 기반 BTC 양은 운영 비용을 최소화하기 위해 가능한 작게(즉, 먼지로 설정된) 유지되는 것이 바람직하다.

[0190] NumShares는 (제한된) 계약 하에서 가능한 주식의 전체 (고정) 수이다. 제한된 계약의 경우, NumShares는 0보다 큰 전체 수를 가져야 한다. 무제한 계약의 경우, NumShares는 어떤 시점에 발행될 수 있는 주식보다 더 고정되지 않고(그들에 의해 기명되어 제공된), 이는 0의 값으로 설정되어 표시된다.

[0191] 주식은 전송 단위로 정의되고, ShareVal은 해당 단위의 값이다. 예를 들어, 신용화폐에 대해, 전송 단위는 1 센트로 설정될 수 있다. 또는 예를 들어, 50센트로 설정할 수 있다. 이 경우 전송은 50센트의 'lots'로만 허용된다. ShareVal은 또한 백분율로 표시될 수 있다. 예를 들어, 만약 브리더가 10과 같은 주식으로 경주마를 팔기

원한다면 ShareVal =10%이다. ShareVal은 >0이어야 하고 계약상 정의되어야 한다.

[0192] TotalIssuance 은 발행된 주식의 전체 수이다. 이 값은 무제한 계약과 같이 제한된 계약에서만 관련되고, 발행이 고정되지 않고 더 많은 주식이 발행될 수 있다. 주식이 백분율로 표현되면 TotalIssuance = 100%로 정의된다.

[0193] 제한 계약의 경우, NumShares, ShareVal, 및 TotalIssuance는 아래의 방법과 연관된다.

[0194] NumShares x ShareVal = TotalIssuance.

[0195] TotalIssuance에 대한 0의 값은 무제한 계약을 암시한다. 무제한 계약의 예는 신용 화폐이다(그래서 TotalIssuance는 0으로 설정된다). 제한 계약의 예는 다음과 같다. (i) 한정판 기념 동전(1000 주, 여기서 1주 = 1코인): TotalIssuance = 1000 x 1 = 1000 코인 (ii)티켓 발매소의 좌석, 여기서 TotalIssuance = 이용 가능한 전체 좌석 수

[0196] 유통은 미사용 토큰의 전체 값으로 정의된다(즉, UTXO의 거래에 의해 결정됨 - 미사용 거래 출력). 모든 미사용 거래의 전체 세트는 모든 비트코인 노드에서 사용할 수 있는 목록에 보관된다. 예를 들어, 발급자가 신용 화폐 형태 토큰으로 10000달러를 발행하고 시간이 지남에 따라 5500달러 상당의 토큰을 상환하면, 유통 = 4500달러가 된다(상환된 토큰의 가치). 이 가치는 연관된 예비 계정의 잔액과 조정되어야 한다.

[0197] **서브키 생성 방법**

[0198] 위에서, 표 3 및 예시 시나리오는 원본 (마스터)키로부터 서브키를 생성하는 것이 유리한 상황을 참조한다. 이를 획득하기 위한 방법은 이것이 수행되는 하나의 방법의 예시에 대해 제공된다.

[0199] 도 7은 통신 네트워크(5)를 통해 제2 노드(7)과 통신하는 제1 노드(3)를 포함하는 시스템(1)을 도시한다. 제1 노드(3)는 제1 처리 장치(23)와 연관되고, 제2 노드(5)는 제2 처리 장치(27)와 연관된다. 제1 및 제2 노드(3, 7)는 컴퓨터, 폰, 태블릿 컴퓨터, 모바일 통신 장치, 컴퓨터 서버 등과 같은 전자 장치를 포함한다. 일 예에서, 제1 노드(3)는 클라이언트(사용자) 장치일 수 있고, 제2 노드(7)는 서버일 수 있다. 서버는 디지털 지갑 제공자의 서버일 수 있다.

[0200] 제1 노드(3)는 제1 노드 마스터 개인키($V_{1C}$) 및 제1 노드 마스터 공개키 ($P_{1C}$)를 가지는 제1 비대칭 암호쌍과 연관된다. 제2 노드 (7)는 제2 노드 마스터 개인키($V_{1S}$) 및 제2 노드 마스터 공개키($P_{1S}$)를 가지는 제2 비대칭 암호쌍과 연관된다. 다시 말해, 제1 및 제2 노드는 각각 각각의 공개-개인키 쌍을 소유한다.

[0201] 제1 및 제2 노드(3, 7) 각각에 대한 제1 및 제2 비대칭 암호쌍은 지갑 등록과 같은 등록 절차 동안 생성된다. 각 노드에 대한 공개키는 통신 네트워크(5)를 통해 공개적으로 공유된다.

[0202] 제1 노드(3) 및 제2 노드(7) 모두에 공통 비밀(C)을 결정하기 위해, 노드(3, 7)는 통신 네트워크(5)를 통해 개인키를 통신하지 않고 각 방법(300, 400)의 단계를 수행한다.

[0203] 제1 노드(3)에 의해 수행되는 방법(300)은 적어도 제1 노드 마스터 개인키($V_{1C}$) 및 생성기 값(GV)에 기초하여 제1 노드 제2 개인키($V_{2C}$)를 결정하는 단계(330)를 포함한다. 생성기 값은 제1 및 제2 노드 사이에 공유된 메시지(M)에 기초할 수 있다. 이는 아래에서 추가 설명된 것처럼 통신 네트워크(5)를 통해 메시지를 공유하는 단계를 포함한다. 방법(300)은 또한 적어도 제2 노드 마스터 공개키 ($P_{1S}$) 및 생성기 값(GV)에 기초하여 제2 노드 제2 공개키($P_{2S}$)를 결정하는 단계(370)를 포함한다. 방법(300)은 제1 노드 제2 개인키($V_{2C}$) 및 제2 노드 제2 공개키 ($P_{2S}$)에 기초하여 공통 비밀(CS)을 생성하는 단계(380)를 포함한다.

[0204] 중요한 것은, 동일한 공통 비밀(CS)은 또한 방법(400)에 의해 제2 노드(7)에서 결정될 수 있다. 방법(400)은 제1 노드 마스터 공개키($P_{1C}$) 및 생성기 값(GV)에 기초하여 제1 노드 제2 공개키($P_{2C}$)를 결정하는 단계(430)를 포함한다. 방법(400)은 제2 노드 마스터 개인키($V_{1S}$) 및 생성기 값(GV)에 기초하여 제2 노드 제2 개인키($V_{2S}$)를 결정하는 단계(470)를 더 포함한다. 방법(400)은 제2 노드 제2 개인키($V_{2S}$) 및 제1 노드 제2 공개키($P_{2C}$)에 기초하여 공통 비밀(CS)을 결정하는 단계(480)를 포함한다.

[0205] 통신 네트워크(5)는 근거리 네트워크, 광역 네트워크, 셀룰러 네트워크, 라디오 통신 네트워크, 인터넷 등을 포함한다. 데이터가 전선, 광섬유 또는 무선과 같은 통신 매체를 통해 전송될 수 있는 이러한 네트워크는 도청자

(11)에 의한 것과 같이 도청에 취약할 수 있다. 방법(300, 400)은 제1 노드(3) 및 제2 노드(7) 둘다 통신 네트워크(5)를 통해 공통 비밀을 전송하지 않고 공통 비밀을 독립적으로 결정할 수 있도록 한다.

[0206] 따라서, 공통 비밀(CS)이 잠재적 비보안 통신 네트워크(5)를 통해 개인키를 전송하지 않아도 각 노드에 의해 안전하고 독립적으로 결정될 수 있는 이점이 있다. 결과적으로, 공통 비밀은 비밀키(또는 비밀키의 기초)로서 이용될 수 있다.

[0207] 방법(300, 400)은 추가적인 단계를 포함할 수 있다. 도 11을 참조한다. 방법(300)은 제1 노드(3)에서 메시지(M) 및 제1 노드 제2 개인키($V_{2C}$)에 기초하여 서명 메시지(SM1)을 생성하는 단계를 포함한다. 방법(300)은 통신 네트워크를 통해 제1 서명 메시지(SM1)를 제2 노드(7)로 전송하는 단계(360)를 더 포함한다. 결과적으로, 제2 노드(7)는 제1 서명 메시지(SM1)를 수신하는 단계(440)를 수행할 수 있다. 방법(400)은 또한 제1 노드 제2 공개키($P_{2C}$)를 통해 제1 서명 메시지(SM2)를 검증하는 단계(450) 및 제1 서명 메시지(SM1)을 검증한 결과에 기초하여 제1 노드(3)를 인증하는 단계(460)를 포함한다. 이점으로, 이는 제2 노드(7)가 지칭된 제1 노드(여기서 제1 서명 메시지는 생성되었음)는 제1 노드(3)인지 인증하도록 한다. 이것은 제1 노드(3)만이 제1 노드 마스터 개인키($V_{1C}$)에 접근할 수 있다는 가정에 기반하고, 따라서 오직 제1 노드(3)만이 제1 서명 메시지(SM1)를 생성하기 위해 제1 노드 제2 개인키($V_{2C}$)를 결정할 수 있다. 피어 투 피어 시나리오와 같이 제2 노드(3)가 제2 노드(7)를 인증할 수 있도록 제2 서명 메시지(SM2)가 제2 노드(7)에서 생성되고 제1 노드(3)에 전송될 수 있도록 하는 것과 유사할 수 있다.

[0208] 제1 및 제2 노드 사이에 메시지를 공유하는 것은 다양한 방법에 의해 구현될 수 있다. 일 예로, 메시지는 제1 노드(3)에서 생성될 수 있고, 통신 네트워크(5)를 통해 제2 노드(7)로 전송될 수 있다. 대안적으로 메시지는 제2 노드(7)에서 생성될 수 있고, 그러면 통신 네트워크(5)를 통해 제2 노드(7)로 전송될 수 있다. 일부 예에서, 메시지(M)는 공개될 수 있고, 따라서 비보안 네트워크(5)를 통해 전송될 수 있다. 하나 이상의 메시지(M)는 데이터 저장소(13, 17, 19)에 저장될 수 있다. 당업자는 메시지 저장이 다양한 방법으로 구현될 수 있음을 이해할 것이다.

[0209] 이점으로는, 공통 비밀(CS)의 재생성을 허용하기 위한 기록은 개인적으로 저장되거나 보안 전송되어야 하는 기록 없이 유지될 수 있다.

[0210] 등록 방법(100, 200)

[0211] 등록 방법(100,200)의 예는 도 3을 참조하여 설명되며, 여기서 방법(100)은 제1 노드(3)에 의해 수행되고, 방법(200)은 제2 노드(7)에 의해 수행된다. 이는 제1 및 제2 노드(3, 7) 각각에 대한 제1 및 제2 비대칭 암호쌍을 설정하는 것을 포함한다.

[0212] 비대칭 암호쌍은 공개치 암호화에 이용되는 것과 같은 연관된 개인키 및 공개키를 포함한다. 이 예에서, 비대칭 암호쌍은 타원 곡선 암호화(ECC) 및 타원 곡선 연산의 속성을 이용하여 생성된다.

[0213] 방법(100, 200)에서, 이는 공통의 ECC 시스템을 결정하고 베이스 포인트(G)를 사용하는(110, 210) 제1 및 제2 노드를 포함한다. (주의 : 베이스 포인트는 공통 생성기로 불릴 수 있으나, '베이스 포인트'라는 용어는 생성기 값(GV)과의 혼란을 피하기 위해 사용된다). 하나의 예에서, 공통의 ECC 시스템은 비트코인(Bitcoin)에 의해 사용된 ECC 시스템인 secp256K1에 기반할 수 있다. 베이스 포인트(G)는 선택되거나, 무작위로 생성되거나, 할당될 수 있다.

[0214] 제1 노도(3)으로 돌아가면, 방법(100)은 공통의 ECC 시스템과 베이스 포인트(G)를 결정하는 단계를 포함한다. 이는 제2 노드(7) 또는 제3 노드(9)로부터 공통의 ECC 시스템 및 공통 생성기를 검색하는 단계를 포함할 수 있다. 대안적으로, 사용자 인터페이스(15)는 제1 노드(3)와 연관될 수 있고, 이에 의해 사용자는 공통의 ECC 시스템 및/또는 베이스 포인트(G)를 선택적으로 제공할 수 있다. 또 다른 대안에서, 공통의 ECC 시스템 및/또는 공통 생성기(G) 중 적어도 하나는 제1 노드(3)에 의해 무작위로 선택될 수 있다. 제1 노드(3)는 통신 네트워크(5)를 통해, 제2 노드(7)로 베이스 포인트(G)를 갖는 공통의 ECC 시스템을 사용하는 것을 나타내는 통지를 전송할 수 있다. 결과적으로, 제2 노드(7)는 공통의 ECC 시스템 및 베이스 포인트(G)를 사용하는 것에 대한 승인을 나타내는 통지를 전송함으로써 결정할 수 있다(210).

[0215] 방법(100)은 또한 제1 노드 마스터 개인키($V_{1C}$) 및 제1 노드 마스터 공개키($P_{1C}$)를 포함하는 제1 비대칭 암호쌍을 생성하는(120) 제1 노드(3)를 포함한다. 이는 적어도 부분적으로, 공통 ECC 시스템에 규정된 허용 범위 내의

임의 정수에 기초한 제1 노드 마스터 개인키($V_{1C}$)를 생성하는 단계를 포함한다. 이는 또한 아래의 수학식에 따른 제1 노드 마스터 개인키($V_{1C}$) 및 공통 생성기(G)의 타원 곡선 점 합산에 기초한 제1 노드 마스터 공개키($P_{1C}$)를 결정하는 단계를 포함한다.

[0216] [수학식 1]

[0217] $P_{1C} = V_{1C} \times G$

[0218] 따라서 제1 비대칭 암호쌍은,

[0219] $V_{1C}$ : 제1 노드에 의해 비밀이 유지된 제1 노드 마스터 개인키

[0220] $P_{1C}$: 공개적으로 알려진 제1 노드 마스터 공개키를 포함한다.

[0221] 제1 노드(3)는 제1 노드(3)와 연관된 제1 데이터 저장소(13)에 제1 노드 마스터 개인키($V_{1C}$) 및 제1 노드 마스터 공개키($P_{1C}$)를 저장할 수 있다. 보안을 위해, 제1 노드 마스터 개인키($V_{1C}$)는 키를 개인적으로 유지하기 위해 제1 데이터 저장소(13)의 보안 부분에 저장될 수 있다.

[0222] 방법(100)은 통신 네트워크(5)를 통해, 제2 노드(7)에 제1 노드 마스터 공개키($P_{1C}$)를 전송하는 단계(130)를 포함한다. 제2 노드(7)는 제1 노드 마스터 공개키($P_{1C}$)를 검색하는 단계에서, 제2 노드(7)와 연관된 제2 데이터 저장소(17)에 제1 노드 마스터 공개키($P_{1C}$)를 저장할 수 있다.

[0223] 제1 노드(3)와 유사하게, 제2 노드(7)의 방법(200)은 제2 노드 마스터 개인키($V_{1S}$) 및 제2 노드 마스터 공개키($P_{1S}$)를 포함하는 제2 비대칭 암호쌍을 생성하는 단계(240)를 포함한다. 제2 노드 마스터 개인키($V_{1S}$)는 또한 허용 범위 내의 임의 정수이다. 결과적으로, 제2 노드 마스터 공개키($P_{1S}$)는 아래의 수학식에 의해 결정된다.

[0224] [수학식 2]

[0225] $P_{1S} = V_{1S} \times G$

[0226] 따라서 제2 비대칭 암호쌍은,

[0227] $V_{1S}$ : 제2 노드에 의해 비밀로 유지되는 제2 노드 마스터 개인키

[0228] $P_{1S}$: 공개적으로 알려진 제2 노드 마스터 공개키을 포함한다.

[0229] 제2 노드(7)는 제2 데이터 저장소(17)에 제2 비대칭 암호쌍을 저장할 수 있다. 방법(200)은 제1 노드(3)에 제2 노드 마스터 공개키($P_{1S}$)를 전송하는 단계(250)를 더 포함한다. 결과적으로, 제1 노드(3)는 제2 노드 마스터 공개키($P_{1S}$)를 수신하고(140) 저장할(150) 수 있다.

[0230] 일부 대안에서, 각각의 공개 마스터키는 제3 노드(9, 신뢰할 수 있는 제3자와 같은)와 연관된 제3 데이터 저장소(19)에 수신되고 저장될 수 있다. 이는 인증 기관과 같은 공개 디렉터리(public directory) 역할을 하는 제3자를 포함할 수 있다. 따라서, 일부 예에서, 제1 노드 마스터 공개키($P_{1C}$)는 공통 비밀(CS)이 필요하다고 결정할 때만(그리고 그 반대의 경우) 제2 노드(7)에 의해 요청되고 수신될 수 있다.

[0231] 등록 단계는 최초 설정으로 한 번만 발생할 필요가 있을 것이다.

[0232] 세션 시작 및 제1 노드(3)에 의한 공통 비밀 결정

[0233] 공통 비밀 (CS)을 결정하는 예시는 도 4를 참조하여 설명될 것이다. 공통 비밀(CS)은 특정 세션, 시간, 거래 또는 제1 노드(3)와 제2 노드(7) 사이의 다른 목적으로 이용될 수 있으며, 동일한 공통 비밀(CS)을 사용하는 것이 바람직하지 않거나 안전하지 않을 수 있다. 따라서, 공통 비밀(CS)은 다른 세션, 시간, 거래 등 사이에서 변경될 수 있다.

[0234] 아래는 위에서 설명된 보안 전송 기술의 예시가 제공된다.

[0235] 메시지 (M)를 생성하기(310)

[0236] 본 예에서, 제1 노드(30)에 의해 수행되는 방법(300)은 메시지(M)를 생성하는 단계(310)를 포함한다. 메시지(M)는 랜덤, 의사 랜덤 또는 사용자 정의일 수 있다. 일 예에서, 메시지 (M)는 유닉스 타임(Unix time) 및 논스(nonce) (그리고 임의의 값)에 기초할 수 있다. 예를 들어, 메시지(M)는 다음과 같이 제공될 수 있다.

[0237] [수학식 3]

[0238] Message (M) = UnixTime + nonce

[0239] 일부 예에서, 메시지(M)는 임의적이다. 그러나 메시지 (M)는 일부 어플리케이션에서 유용할 수 있는 선택 값(예를 들어 유닉스 타임 등)을 가질 수 있음을 이해해야 한다.

[0240] 방법(300)은 통신 네트워크(3)를 통해, 제2 노드(7)로 메시지(M)를 전송하는 단계(315)를 포함한다. 메시지(M)는 개인키에 대한 정보를 포함하지 않으므로 메시지(M)는 비보안 네트워크를 통해 전송될 수 있다.

[0241] 생성기 값(GV) 결정하기(320)

[0242] 방법(300)은 메시지(M)에 기초하여 생성기 값(GV)을 결정하는 단계(320)를 더 포함한다. 본 예에서, 이는 메시지의 암호화 해시를 결정하는 단계를 포함한다. 암호화 해시 알고리즘의 예는 256-bit 생성기 값(GV)을 생성하기 위한 SHA-256를 포함한다. 그건 아래와 같다.

[0243] [수학식 4]

[0244] GV = SHA-256(M)

[0245] 다른 해시 알고리즘이 이용될 수 있음을 이해해야 한다. 이는 보안 해시 알고리즘(Secure Hash Algorithm, SHA) 제품군의 다른 알고리즘이 포함될 수 있다. 몇몇 특정 예는 SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256을 포함한 SHA-3 하위 집합의 인스턴스를 포함한다. 다른 해시 알고리즘은 RIPEMD(RACE Integrity Primitives Evaluation Message Digest) 제품군의 것들을 포함한다. 특정 예는 RIPEMD-160을 포함한다. 다른 해시 함수는 세모르-틸리히(Z ιmor-Tillich) 해시 함수 및 냅색 기반(knapsack-based) 해시 함수을 포함한다.

[0246] 제1 노드 제2 개인키 결정하기(330)

[0247] 방법(300)은 다음으로 제2 노드 마스터 개인키($V_{1C}$) 및 생성기 값(GV)에 기초하여 제1 노드 제2 개인키($V_{2C}$)를 결정하는 단계(330)를 포함한다. 이는 아래의 수학식에 따라, 제1 노드 마스터 개인키($V_{1C}$) 및 생성기 값(GV)의 스칼라 합산에 기초할 수 있다.

[0248] [수학식 5]

[0249] $V_{2C} = V_{1C} + GV$

[0250] 따라서 제1 노드 제2 개인키($V_{2C}$)는 무작위 값이 아니라 제1 노드 마스터 개인키로부터 결정론적으로 파생된 것이다. 암호쌍에 대응하는 공개키, 즉, 제1 노드 제2 공개키($P_{2C}$)는 다음과 같은 관계를 가진다.

[0251] [수학식 6]

[0252] $P_{2C} = V_{2C} \times G$

[0253] 수학식 5로부터 수학식 6에 $V_{2C}$ 을 대입하면 다음과 같다.

[0254] [수학식 7]

[0255] $P_{2C} = (V_{1C} + GV) \times G$

[0256] 여기서, ' + ' 연산자는 스칼라 합산을 나타내고, 'x' 연산자는 타원 곡선 점 합산을 나타낸다. 타원 곡선 암호화 대수학이 분산적이라는 것을 주목하면, 수학식 7은 다음과 같이 표현될 수 있다.

[0257] [수학식 8]

[0258] $P_{2C} = V_{1C} \times G + GV \times G$

[0259] 결론적으로, 수학식 1은 수학식 7에 대입되어 다음과 같이 나타날 수 있다.

[0260] [수학식 9.1]

[0261] $P_{2C} = P_{1C} + GV \times G$

[0262] [수학식 9.2]

[0263] $P_{2C} = P_{1C} + SHA\text{-}256(M) \times G$

[0264] 수학식 8 내지 9.2에서, ' + ' 연산자는 타원 곡선 점 합산을 나타낸다. 그래서 대응하는 제1 노드 제2 공개키($P_{2C}$)는 제1 노드 마스터 공개키($P_{1C}$) 및 메시지(M)의 주어진 지식을 파생시킬 수 있다. 제2 노드(7)는 방법(400)과 관련하여 아래에서 더 상세히 설명되는 바와 같이, 제1 노드 제2 공개키($P_{2C}$)를 독립적으로 결정할 수 있는 그러한 지식을 가질 수 있다.

[0265] 메시지 및 제1 노드 제2 개인키에 기초하여 제1 서명 메시지(SM1)을 생성하기(350)

[0266] 방법(300)은 메시지(M) 및 결정된 제1 노드 제2 개인키($V_{2C}$)에 기초하여 제1 서명 메시지(SM1)를 생성하는 단계(350)를 더 포함한다. 서명 메시지를 생성하는 것은 메시지(M)에 디지털 서명을 하기 위해 디지털 서명 알고리즘을 적용하는 것을 포함한다. 일 예에서, 이는 제1 서명 메시지(SM1)를 획득하기 위해 타원 곡선 전자 서명 알고리즘(Elliptic Curve Digital Signature Algorithm, ECDSA)의 메시지에 제1 노드 제2 개인키($V_{2C}$)를 적용하는 것을 포함한다. ECDSA의 예는 secp256k1, secp256r1, secp384r1, se3cp521r1를 갖는 ECC 시스템에 기초하는 것들을 포함한다.

[0267] 제1 서명 메시지(SM1)는 제2 노드(7)에서 대응하는 제1 노드 제2 공개키($P_{2C}$)로 확인될 수 있다. 이러한 제1 서명 메시지(SM1)의 확인은 제1 노드(3)를 인증하기 위해 제2 노드(7)에 의해 이용될 수 있으며, 이는 아래의 방법(400)에서 논의 될 것이다.

[0268] 제2 노드 제2 공개키 결정하기(370')

[0269] 그러면, 제1 노드(3)는 제2 노드 제2 공개키($P_{2S}$)를 결정할 수 있다(370). 위에서 논의된 것처럼, 제2 노드 제2 공개키($P_{2S}$)는 적어도 제2 노드 마스터 공개키($P_{1S}$) 및 생성기 값(GV)에 기초할 수 있다. 본 예에서, 공개키는 생성기(G)와 타원 곡선 점 곱셈을 갖는 개인키로서 결정되기 때문에(370`), 제2 노드 제2 공개키($P_{2S}$)는 수학식 6과 유사한 방식으로 표현될 수 있다.

[0270] [수학식 10.1]

[0271] $P_{2S} = V_{2S} \times G$

[0272] [수학식 10.2]

[0273] $P_{2S} = P_{1S} + DK \times G$

[0274] 수학식 10.2에 대한 수학적 증명은 제1 노드 제2 공개키($P_{2C}$)에 대한 수학식 9.1을 유도하기 위해 위에서 설명된 것과 동일하다. 제1 노드(3)는 제2 노드(7)와 독립적으로 제2 노드 제2 공개키를 결정할 수 있음이(370) 인정되어야 한다.

[0275] 제1 노드(3)에서 공통 비밀 결정하기(380)

[0276] 그러면, 제1 노드(3)는 결정된 제1 노드 제2 개인키($V_{2C}$) 및 결정된 제2 노드 제2 공개키($P_{2S}$)에 기초하여 공통 비밀(CS)을 결정할 수 있다(380). 공통 비밀(CS)은 아래의 수학식에 따라, 제1 노드(3)에 의해 결정될 수 있다.

[0277] [수학식 11]

[0278] $S = V_{2C} \times P_{2S}$

[0279] 제2 노드(7)에서 수행되는 방법(400)

[0280] 제2 노드(7)에서 수행되는 대응하는 방법(400)이 설명될 것이다. 이러한 단계들 중 몇몇은 제1 노드(3)에 의해

수행된 전술한 단계들과 유사하다는 것을 이해해야 한다.

[0281] 방법(400)은 통신 네트워크(5)를 통해 제1 노드(3)로부터 메시지(M)를 수신하는 단계(410)를 포함한다. 이는 315 단계에서 제1 노드(3)에 의해 전송된 메시지(M)를 포함할 수 있다. 그러면, 제2 노드(7)는 메시지(M)에 기초하여 생성기 값(GV)를 결정한다(420). 제2 노드(7)에 의해 생성기 값(GV)를 결정하는 단계(420)는 위에서 설명된 제1 노드에 의해 수행되는 320 단계와 유사하다. 본 예에서, 제2 노드(7)는 제1 노드(3)와 독립적으로 이러한 결정 단계(420)를 수행한다.

[0282] 다음 단계는 제1 노드 마스터 공개키($P_{1C}$) 및 생성기 값(GV)에 기초하여 제1 노드 제2 공개키($P_{2C}$)를 결정하는 단계(430)를 포함한다. 본 예에서, 공개키는 베이스 포인트(G)와 타원 곡선점 곱셈을 갖는 개인키로서 결정(430`)되기 때문에, 제1 노드 제2 공개키($P_{2C}$)는 수학식 9와 유사한 형태로 표현될 수 있다.

[0283] [수학식 12.1]

[0284] $P_{2C} = V_{2C} \times G$

[0285] [수학식 12.2]

[0286] $P_{2C} = P_{1C} + GV \times G$

[0287] 수학식 12.1 및 12.2의 수학적 증명은 수학식 10.1 및 10.2를 위해 위에서 논의된 것과 동일하다.

[0288] 제1 노드(3)를 인증하는 제2 노드(7)

[0289] 방법(400)은 주장된 제1 노드(3)가 제1 노드(3)임을 인증하기 위해 제2 노드(7)에 의해 수행되는 단계를 포함할 수 있다. 이전에 논의된 것처럼, 제1 노드(3)로부터 제1 서명 메시지(SM1)를 수신하는 것(440)을 포함한다. 그러면, 제2 노드(7)는 430 단계에서 결정된 제1 노드 제2 공개키($P_{2C}$)를 통해 제1 서명 메시지(SM1)의 서명을 검증할 수 있다(450).

[0290] 디지털 서명을 검증하는 것은 위에서 논의된 타원 곡선 전자 서명 알고리즘(ECDSA)에 따라 행해질 수 있다. 중요한 것은, 제1 노드 제2 개인키($V_{2C}$)로 서명된 제1 서명 메시지(SM1)는 $V_{2C}$ 및 $P_{2C}$가 암호쌍을 형성하기 때문에, 대응하는 제1 노드 제2 공개키($P_{2C}$)와 정확하게 일치해야 한다. 이러한 키들은 제1 노드(3)의 등록시 생성되는 제1 노드 마스터 개인키($V_{1C}$) 및 제1 노드 마스터 공개키($P_{1C}$)에 대해 결정적으므로, 제1 서명 메시지(SM1)을 결정하는 것은 제1 서명 메시지(SM1)를 전송하는 주장된 제1 노드가 등록시 동일한 제1 노드(3)임을 인증하는 근거로 이용될 수 있다. 그래서 제2 노드(7)는 제1 서명 메시지를 검증하는 단계(450)의 결과에 기초하여 제1 노드(3)를 인증하는 단계(460)를 더 수행할 수 있다.

[0291] 공통 비밀을 결정하는 제2 노드(7)

[0292] 방법(400)은 제2 노드(7)가 제2 노드 마스터 개인키($V_{1S}$) 및 생성기 값(GV)에 기초하여 제2 노드 제2 개인키($V_{2S}$)를 결정(470)하는 노드를 더 포함할 수 있다. 제1 노드(3)에 의해 수행되는 330 단계와 유사하게, 제2 노드 제2 개인키($V_{2S}$)는 아래의 수학식에 따라, 제2 노드 마스터 개인키($V_{1S}$) 및 생성기 값(GV)의 스칼라 합산에 기초할 수 있다.

[0293] [수학식 13.1]

[0294] $V_{2S} = V_{1S} + DK$

[0295] [수학식 13.2]

[0296] $V_{2S} = V_{1S} + SHA\text{-}256(M)$

[0297] 그러면, 제2 노드(7)는 아래의 수학식에 기초하여, 제1 노드(3)와 독립적으로, 제2 노드 제2 개인키($V_{2S}$) 및 제1 노드 제2 공개키($P_{2C}$)에 기초한 공통 비밀(CS)을 결정(480)할 수 있다.

[0298] [수학식 14]

[0299] $S = V_{2S} \times P_{2C}$

[0300] 제1 노드(3) 및 제2 노드(7)에 의해 결정된 공통 비밀 (CS)의 증명

[0301] 제1 노드(3)에 의해 결정된 공통 비밀(CS)은 제2 노드(7)에서 결정된 공통 비밀(CS)과 동일하다. 수학식 11 및 수학식 14가 동일한 공통 비밀(CS)을 제공하는 것에 대한 수학적 증명을 이제 설명할 것이다.

[0302] 제1 노드(3)에 의해 결정된 공통 비밀(CS)로 돌아가면, 수학식 10.1은 아래의 수학식 11에 대입될 수 있다.

[0303] [수학식 11]

[0304] $S = V_{2C} \times P_{2S}$

[0305] $S = V_{2C} \times (V_{2S} \times G)$

[0306] [수학식 15]

[0307] $S = (V_{2C} \times V_{2S}) \times G$

[0308] 제2 노드(7)에 의해 결정된 공통 비밀(CS)로 돌아가면, 수학식 12.1은 아래의 수학식 14에 대입될 수 있다.

[0309] [수학식 14]

[0310] $S = V_{2S} \times P_{2C}$

[0311] $S = V_{2S} \times (V_{2C} \times G)$

[0312] [수학식 16]

[0313] $S = (V_{2S} \times V_{2C}) \times G$

[0314] ECC 대수학은 교환 가능하기 때문에, 수학식 15 및 수학식 16은 동일하다. 그러므로 아래와 같다.

[0315] [수학식 17]

[0316] $S = (V_{2C} \times V_{2S}) \times G = (V_{2S} \times V_{2C}) \times G$

[0317] 공통 비밀 (CS) 및 비밀키

[0318] 공통 비밀(CS)은 비밀키로서 또는 제1 노드(3) 및 제2 노드(7) 사이의 보안 통신을 위한 대칭키 알고리즘에서 비밀키의 근거로서 이용될 수 있다.

[0319] 공통 비밀(CS)은 타원 곡선 점 ($x_S$, $y_S$)의 형태일 수 있다. 이는 노드(3, 7)에 의해 합의된 표준 공개 연산을 이용하여 표준키 포멧으로 변환될 수 있다. 예를 들어, $x_S$ 값은 $AES_{256}$ 암호화를 위한 키로서 이용될 수 있는 256-bit 정수일 수 있다. 또한, 이 길이키를 요구하는 어떤 어플리케이션을 위해 RIPEMD160을 사용하여 160-bit 정수로 변환될 수 있다.

[0320] 공통 비밀(CS)은 필요에 따라 결정될 수 있다. 중요하게도, 제1 노드(3)는 메시지(M)에 기초하여 재결정될 수 있으므로 공통 비밀(CS)을 저장할 필요가 없다. 몇몇 예에서, 사용된 메시지(M)는 마스터 개인키에 대해 요구되는 것과 동일한 보안 레벨 없이 데이터 저장소(13, 17, 19)(또는 다른 데이터 저장소)에 저장될 수 있다. 몇몇 예에서, 메시지(M)는 공개적으로 이용 가능할 수 있다.

[0321] 그러나 몇몇 어플리케이션에 따르면, 공통 비밀(CS)은 공통 비밀(CS)이 제1 노드 마스터 개인키($V_{1C}$)만큼 보안이 유지되는 것을 제공하는 제1 노드와 연관된 제1 데이터 저장소(X)에 저장될 수 있다.

[0322] 이점으로는, 이 기법은 하나의 마스터키 암호화 쌍에 기초하여 다중 보안 비밀키에 대응할 수 있는 다중 공통 비밀을 결정하는데 이용될 수 있다.

[0323] 생성기 값(키)의 계층 구조

[0324] 일 예에서, 일련의 연속적인 생성기 값(GVs)이 결정될 수 있고, 여기서, 각각의 연속적인 GV는 이전 생성기 값(GV)에 기초하여 결정될 수 있다. 예를 들어, 연속적인 단일 목적 키를 생성하기 위해 310 내지 370 단계 및

410 내지 470 단계를 반복하는 대신, 노드 사이의 사전 합의에 의해, 이전에 사용된 생성기 값(GV)은 생성기 값의 계층 구조를 정하기 위해 양 당사자에 의해 반복적으로 재순환될 수 있다. 결과적으로, 메시지(M)의 해시에 기초한 결정키는 다음 생성 생성기 값(GV')를 위한 다음 생성 메시지(M')일 수 있다. 이를 통해 프로토콜 설정 전송, 특히 공통 비밀의 각 생성을 위한 다수 메시지의 전송을 더 필요로 하지 않고 계산되기 위하여 공유 비밀의 연속적인 생성을 허용할 수 있다. 다음 생성 공통 비밀(CS')은 아래와 같이 연산될 수 있다.

[0325] 첫째로, 제1 노드(3) 및 제2 노드(7) 모두 생성기 값(GV')의 다음 생성을 독립적으로 결정한다. 이는 320 및 420 단계와 유사하지만 아래의 수학식으로 수정된다.

[0326] [수학식 18]

[0327] M' = SHA-256(M)

[0328] [수학식 19.1]

[0329] GV' = SHA-256(M')

[0330] [수학식 19.2]

[0331] GV' = SHA-256(SHA-256(M))

[0332] 그러면, 제1 노드(3)는 위에서 설명된 370 및 330 단계와 유사하게 제2 노드 제2 공개키($P_{2S}$') 및 제1 노드 제2 개인키($V_{2C}$')의 다음 생성을 결정할 수 있지만, 아래의 수학식으로 수정된다.

[0333] [수학식 20.1]

[0334] $P_{2S}' = P_{1S} + DK' \times G$

[0335] [수학식 20.2]

[0336] $V_{2C}' = V_{1C} + DK'$

[0337] 그러면, 제2 노드(7)는 위에서 설명된 430 및 470 단계와 유사하게 제1 노드 제2 공개키($P_{2C}$') 및 제2 노드 제2 개인키($V_{2S}$')의 다음 생성을 결정할 수 있지만, 아래의 수학식으로 수정된다.

[0338] [수학식 21.1]

[0339] $P_{2C}' = P_{1C} + DK' \times G$

[0340] [수학식 21.2]

[0341] $V_{2S}' = V_{1S} + DK'$

[0342] 그러면, 제1 노드(3) 및 제2 노드(7)는 다음 생성 공통 비밀(CS')을 각각 결정할 수 있다. 특히, 제1 노드(3)는 수학식을 통해 다음 생성 공통 비밀(CS')를 결정한다.

[0343] [수학식 22]

[0344] $CS' = V_{2C}' \times P_{2S}'$

[0345] 제2 노드(7)는 수학식을 통해 다음 생성 공통 비밀(CS')을 결정한다.

[0346] [수학식 23]

[0347] $CS' = V_{2S}' \times P_{2C}'$

[0348] 추가적인 생성 (CS'', CS''', 등)은 체인 계층을 생성하기 위해 동일한 방법으로 계산될 수 있다. 이 기법은 제1 노드(3) 및 제2 노드(7) 모두가 원본 메시지(M) 또는 원본의 계산된 결정키(DK), 그리고 그것이 관련된 노드의 추적을 유지할 것을 요구한다. 이것은 공개적으로 알려진 정보이므로 이 정보의 보존에 관련된 보안 문제는 없다. 따라서, 이 정보는 '해시 테이블(hash table)'에 보관될 수 있으며(해시 값을 공개키에 연결), 네트워크(5)를 통해 자유롭게 배포될 수 있다(예를 들어, 토렌트를 사용). 또한, 계층의 어떤 개인적인 공통 비밀(CS)이

손상된다면, 개인키 $V_{1C}$, $V_{1S}$ 가 안전하다는 전제하에 계층 내의 다른 공통 비밀의 보안에 영향을 미치지 않는다.

[0349] 키의 트리 구조

[0350] 위에서 설명된 체인(선형) 계층은 물론, 트리 구조의 형태에서 계층이 생성될 수 있다. 트리 구조를 통해, 인증키, 암호화키, 서명키, 지불키 등과 같은 상이한 목적을 위한 다양한 키들이 결정될 수 있으며, 이러한 키들은 하나의 안전하게 유지된 마스터키에 모두 연결된다. 이는 다양한 상이한 키들을 통한 트리 구조(901)를 나타내는 도 9에 가장 잘 도시되어 있다. 이들 각각은 다른 당사자와 공유된 비밀을 생성하는데 사용될 수 있다. 트리 분기는 몇 가지 방법으로 수행될 수 있으며, 그 중 3개가 아래에서 설명된다.

[0351] (i) 마스터키 생성

[0352] 체인 계층에서, 각각의 새로운 '연결'(공개/개인키 쌍)은 원본 마스터키에 다수의 재해시된(rehashed) 메시지를 추가하여 생성된다. 예를 들어, (명료성을 위해 제1 노드(3)의 개인키만 표시하는 것).

[0353] [수학식 24]

[0354] $V_{2C} = V_{1C} + SHA\text{-}256(M)$

[0355] [수학식 25]

[0356] $V_{2C}' = V_{1C} + SHA\text{-}256(SHA\text{-}256(M))$

[0357] [수학식 26]

[0358] $V_{2C}'' = V_{1C} + SHA\text{-}256(SHA\text{-}256(SHA\text{-}256(M)))$

[0359] … 등.

[0360] 분기를 생성하기 위해, 어떤 키는 서브 마스터키로서 이용될 수 있다. 예를 들어, $V_{2C}'$는 일반 마스터키에 대해 수행되는 해시를 추가함으로써 서브 마스터키 ($V_{3C}$)로서 사용될 수 있다.

[0361] [수학식 27]

[0362] $V_{3C} = V_{2C}' + SHA\text{-}256(M)$

[0363] 서브 마스터키($V_{3C}$)는 자체로 다음 생성키($V_{3C}'$)를 가질 수 있다. 예를 들어,

[0364] [수학식 28]

[0365] $V_{3C}' = V_{2C}' + SHA\text{-}256(SHA\text{-}256(M))$

[0366] 이는 도 10에 도시된 마스터키 생성 방법을 이용하는 트리 구조(903)를 제공한다.

[0367] (ii) 논리적 연합

[0368] 본 방법에서, 트리(공개/개인 키 쌍)의 모든 노드들은 체인으로(또한 다른 방법으로) 생성되고, 트리에서 노드 사이의 논리적 관계는 트리의 각 노드가 포인터(pointer)를 사용하여 트리의 부모 노드와 단순히 연관된 테이블에 의해 유지된다. 그래서, 포인터는 세션에 대한 공통 비밀키(CS)를 결정하기 위한 관련 공개/개인키 쌍을 결정하는데 이용될 수 있다.

[0369] (iii) 메시지 다중성

[0370] 새로운 개인/공개키 쌍은 체인 또는 트리의 어떤 포인트에서 새로운 메시지를 소개하는 것에 의해 생성될 수 있다. 메시지 자체는 임의적이거나 어떤 의미 또는 기능을 포함할 수 있다(예, '실제' 은행 계좌 번호 등과 연관될 수 있음). 새로운 개인/공개키 쌍을 형성하기 위한 각각의 새로운 메시지는 안전하게 유지되는 것이 바람직할 수 있다.

[0371] **본 발명과 함께 이용하기 위한 예시적인 컴퓨팅 에이전트**

[0372] 본 발명은 계약 프로세스의 자동화 개념을 수행하기 위해 컴퓨팅 자원 또는 에이전트를 활용할 수 있다. 적합한 에이전트의 예시는 아래에 제공되나, 다른 구현도 이용될 수 있다.

[0373] 에이전트는 블록체인과 결합하여 작동할 수 있고, 튜링 머신의 구현에서 삭제할 수 없는 테이프로서 이를 이용한다. 에이전트는 블록체인에 병렬적으로 실행되고, (루프) 프로세스의 실행을 처리하고 감시한다. 루프 프로세스는 장치 또는 시스템의 제어 또는 처리의 자동화와 같은 주어진 작업을 수행하기 위해 설계된다. 병렬 자원은 블록체인의 상태를 모니터링하고 블록체인에 기록된 거래를 야기할 수 있다. 어떤 의미에서, 다음의 정의와 특징을 갖는 튜링 머신의 삭제할 수 없는 테이프로서 블록체인을 활용할 수 있다.

[0374] 1. 블록체인은 튜링 머신의 테이프로서 동작한다. 블록체인의 각 거래는 테이프 상에 셀(cell)을 나타낸다. 이 셀은 유한 알파벳의 기호를 포함할 수 있다.

[0375] 2. 테이프 헤드(tape head)는 블록체인상에 이미 기록된 블록으로부터 정보를 읽을 수 있다.

[0376] 3. 테이프 헤드는 블록체인의 끝에 많은 거래를 포함하는 새로운 블록을 기록할 수 있다. 그러나, 이미 존재하는 블록 상에는 기록할 수 없다. 따라서, 블록체인 테이프는 삭제되지 않는다.

[0377] 4. 각 거래의 메타데이터는 다중 서명 페이 투 스크립트 해시(signature pay-to-script-hash, P2SH) 거래 의 부분으로서 저장될 수 있다.

[0378] 에이전트의 중요한 기능은 블록체인의 현재 상태를 모니터링하는 자동화된 독립체로서 동작하는 것이다. 또한 어떠한 오프 블록 소스로부터 신호 또는 입력을 수신할 수 있다. 블록체인 상태 및/또는 수신된 입력에 따라, 에이전트는 특정 작업을 수행할 수 있다. 에이전트는 어떠한 작업이 수행될 것인지를 결정한다. 이는 '실제 세상'(즉, 오프 블록)에서 작업 및/또는 블록체인 상의 작업(새로운 거래를 생성하고 브로드캐스팅하는 것과 같은)을 수반하거나 수반하지 않을 수 있다. 에이전트가 취하는 작업은 블록체인 상태 또는 어떠한 오프-블록 입력에 의해 작동될 수 있다. 에이전트는 또한 거래의 다음 세트가 비트코인 네트워크에 브로드캐스트되고 다음으로 블록체인에 기록될 수 있도록 결정할 수 있다.

[0379] 에이전트의 작업은 블록체인(예를 들어, 비트코인)에 병렬적 및 동시적으로 실행한다. 어떤 의미에서, 블록체인(예를 들어, 비트코인) 스크립트의 기능을

[0380] 튜링 머신은 두 개의 스택을 포함한다.

[0381] • 데이터 스택 : 이는 상기에서 설명한 것처럼 블록체인으로 표현된다.

[0382] • 제어 스택 : 이는 관리자 기능으로 표현된다. 이는 반복 제어 흐름 기능에 관련한 정보를 저장한다.

[0383] 데이터 스택으로부터의 제어 스택 분리는 비트코인 코어 내에서 무한 루프가 발생하는 것을 방지할 수 있는 이점을 제공하며 서비스 거부 공격을 완화한다.

[0384] 이에전트는 루프 구조의 어떠한 형태(예를 들어, FOR-NEXT; REPEAT UNTIL 등)를 통해 루프할 수 있는 서브 루틴을 관리하고 수행한다. 여기서 설명된 실시예는 '반복' 구조의 일 예를 사용하는 포로세스를 포함한다(도 2 참조). 사용자는 인덱스($i$) 및 제한($J$)을 특정한다. 이는 현재 반복 횟수(일반적으로 0으로부터 시작하여 카운트된) 및 반복 루프의 반복 총 횟수를 각각 나타낸다.

[0385] 각 반복은 아래와 같다.

[0386] 1. 인덱스는 1씩 증가한다. 종료 조건을 위해, 반복은 인덱스가 제한에 도달할 때 멈춘다.

[0387] 2. ICTA 문(statement)을 포함하는 코드 블록은 실행된다; 실행은 블록체인 상 또는 밖의 어떠한 실행일 수 있다.

[0388] 3. 이러한 서브루틴의 암호화 해시는 계산된다. 이는 거래(Tx)의 일부로서 블록체인에 저장될 수 있다. 각 코드 블록마다 해시는 고유하기 때문에, 해당 코드가 사용되었는지 검증할 수 있다.

[0389] 따라서, 루프의 본체는 코드 블록을 포함한다. 각 코드 블록은 ICTA 문을 포함한다(도 3 참조). 이는 아래를 매칭하여 거래를 위한 블록체인의 현재 상태를 모니터링한다.

[0390] • 시작 또는 트리거 조건(즉, 특정 비트코인 주소가 10BTC에 도달한 경우)

[0391] • 반복 조건(즉, 이전 반복과 관련된 메타데이터 또는 해시)

[0392] • 중지 조건(즉, 루프의 마지막 반복)

[0393] ICTA 문은 블록체인의 현재 상태에 기초하여, 관리자가 다음 거래에서 생성하도록 결정될 수 있다. 다음 거래를 생성하는 것은 비트코인 네트워크상에 거래를 브로드캐스팅하는 것과 블록체인 상에 새로운 거래를 기록하는 것을 수반한다. 이는 이러한 반복이 실행되는 기록으로서 동작한다. 거래가 블록체인상에 기록되면, 관리자는 이전 반복이 블록체인 상에서 실행되고 기록된 것을 발견하고 다음 반복을 실행할 것이다. 후자는 인덱스(i)가 코드 블록에 지정된 제한(J)에 도달하여 반복 로프가 종료될 때까지 계속된다.

[0394] 각 거래는 재사용될 수 있는 방법으로 블록체인에 저장된다. 비트코인 구현에서, 거래의 각 서명에는 SIGHASH 플래그가 추가된다. 이 플래그는 다른 값을 가질 수 있으며, 거래의 다른 부분인지를 나타내는 각각은 이 서명 소유자의 개입 없이 수정될 수 있다. 재사용 가능한 거래는 거래 입력의 하나에 SIGHASH 플래그 'SigHash_AnyoneCanPay'를 가진다. 이는 누구나 거래의 입력에 기여하도록 허용한다. 이 파라미터는 관리자의 ICTA 기능이 다른 입력과 함께 여러 번 실행되고 반복될 수 있도록 한다. 기능의 사용은 승인된 당사자로 제한될 수 있다(예를 들어, 재사용 가능한 거래의 저작권을 통해).

[0395] ICTA 코드 블록의 'If condition' 섹션은 조건의 어떤 형태를 모니터링할 수 있다. 이는 다른 프로그래밍 언어(예를 들어, C, C++, Java)와 유사하고 블록체인에 저장된 정보에 제한되지 않는다. 예를 들어, 날짜 및 시간을 모니터링 할 수 있거나(즉, 언제 특정 날짜 및 시간에 도달하는지) 날씨를 모니터링 할 수 있고(즉, 언제 온도가 10도 이하이고 비가 오는지), 신뢰 또는 계약의 조건을 모니터링 할 수 있다(즉, 언제 회사 A가 회사 B를 인수하는지).

[0396] ICTA 코드 블록의 'Then *action*' 섹션은 여러가지 실행을 수행할 수 있다. 본 발명은 취할 수 있는 실행의 횟수 또는 유형에 관련하여 제한되지 않는다. 비록 실행과 관련된 메타데이터를 포함하는 거래가 블록체인 상에 기록될 수 있지만, 실행은 블록체인 상의 거래에만 국한되지 않는다.

[0397] 메타데이터는 어떠한 형태일 수 있다. 그러나, 본 발명의 실시예에 따르면, 메타데이터는 실행에 관련한 더 많은 데이터 또는 명령어를 포함하는 파일에 대한 하이퍼링크(hyperlink)를 저장할 수 있다. 메타데이터는 해시 테이블에 대한 룩업(look-up)키로서 동작하는 실행의 해시와 함께 실행에 관련한 더 많은 데이터 또는 명령어를 포함하는 해시 테이블에 대한 하이퍼링크를 저장할 수 있다.

[0398] 에이전트의 제어 스택은 각 사용자의 요구에 따라 특정된 다양한 방법으로 구현될 수 있다. 예를 들어, 제어 스택의 반복 루프는 어떤 튜링 완전 언어에 기초할 수 있다. 한가지 가능한 언어 선택은 Forth 스타일 스택 기반 언어이다. 이 언어를 사용하는 이점은 이미 알려져 있고 널리 쓰이는 비트코인 스크립트를 통해 제어 스택을 프로그래밍 스타일에 일관되게 유지할 수 있다는 것이다.

[0399] 데이터 저장 공간으로서 비트코인 스크립트의 대체 스택 이용

[0400] 비트코인 스크립트는 연산 코드(op code)라고 불리는 명령을 포함하며, 이는 사용자가 알트 스택(alt stack)이라고 알려진 대체 스택에 데이터를 이동시킬 수 있도록 한다.

[0401] 연산 코드는 다음과 같다.

[0402] • OP_TOALTSTACK - 이는 메인 스택의 상부에서 알트 스택의 상부로 데이터를 이동시킨다.

[0403] • OP_FROMALTSTACK - 이는 알트 스택의 상부에서 매인 스택의 상부로 데이터를 이동시킨다

[0404] 이는 데이터가 계산기 상에 저장될 수 있도록 하는 '메모리' 기능과 유사하게, 계산 중간 단계의 데이터를 알트 스택에 저장할 수 있다. 일 실시예에서, 알트 스택은 비트코인 스크립트가 작은 계산 작업을 해결하도록 구성되고 계산 결과를 반환하는데 이용될 수 있다.

[0405] 에이전트를 관리하기 위하여 코드 레지스터를 사용하기

[0406] 에이전트는 또한 소유하고 수행하는 모든 코드의 레지스트리를 관리한다. 이러한 레지스트리는 특정키를 특정값에 매핑하는 룩업 테이블 또는 사전과 같이 구성된다. 키 및 값 쌍은 코드 블록 해시($H_1$) 및 코드가 각각 저장되는 IPv6 주소로 표시된다. 키 $H_1$을 사용하여 코드 블록을 검색하기 위해, 룩업 테이블은 관련된 값(이는 코드가 저장된 위치이다)을 검색하는데 이용되고 그에 따라 소스 코드를 검색한다. 코드 레지스트리의 구현은 다양할 수 있다.

[0407] 관리자 코드의 거래 메타데이터, 그리고 루프의 재생성(Re-Spawning)

[0408] 특정 반복에서 관리자 루프를 재생성하기 위해 요구되는 정보는 블록체인 상에 기록된 거래에 메타데이터로 저장된다.

[0409] 이러한 방법에서, 블록체인 상의 거래는 에이전트에서 실행중인 루프의 주어진 반복에 대한 정보를 저장하거나 정보로의 접근을 제공한다. 이 정보는 인덱스 i와 같은 루프에 관련한 어떤 변수값, 그리고 코드 블록 또는 더 요청된 정보에 접근할 수 있는 곳을 특정하는 위치 관련 데이터에 이용된 파라미터 값과 같은 다른 필수 정보를 포함한다.

[0410] 메타데이터 자체는 거래의 다중 서명 P2SH의 부분으로서 저장된다. 거래와 함께 기록된 메타데이터는 또한 과거에 어떻게 코드가 실행되었는지에 대한 감사 추적을 기록할 수 있는 기능을 제공한다.

[0411] 에이전트가 각 반복에서 반복 루프 코드 블록을 재생성할 수 있는 몇몇 방법이 있다. 코드 블록은 에이전트 자체에서 하드 코드(hard-code)되거나, 개인적 또는 공개적으로 사용가능한 파일에 저장되거나, 개인 또는 공개 해시 테이블 파일상에 항목으로서 저장되거나, 상기의 조합일 수 있다. 코드 블록은 하드 코드된 변수로 고정될 수 있고, 또는 고정될 수 있지만 덧붙일 수 있는 파라미터를 포함할 수 있다. 파라미터는 어떤 데이터 포멧의 단일 값이거나, 작은 코드 덩어리이거나, 상기의 조합일 수 있다. 파라미터는 거래(예를 들어, 비트코인 거래)의 메타데이터로부터 또는 내부 데이터 베이스 또는 개인/공개 파일 또는 해시 테이블 또는 상기의 조합과 같은 외부 소스로부터 직접 그것들을 검색함으로써 덧붙여질 수 있다. 파라미터 값의 외부 소스에 대한 포인터(pointer)는 거래의 메타데이터에 저장될 수 있다.

[0412] 아래의 단계는 어떻게 에이전트가 i번째 반복에서 반복 루프 코드 블록을 재생성할 수 있는지에 대한 일례를 제공한다. 예를 들어, 코드 레지스트리는 해시 테이블이고 이로써 해시 값은 테이블에 대해 룩업키로서 동작하고 거래 상 메타데이터에 저장된다.

[0413] 1. 에이전트는 코드 레지스트리의 항목과 매칭되는 코드 블록의 해시를 포함하는 거래에 대한 블록체인을 모니터링한다.

[0414] 2. 에이전트는 대응하는 해시($H_1$)를 포함하는 거래를 찾는다.

[0415] 3. 에이전트는 '메타데이터-코드해시(Metadata-CodeHash)를 읽고, $H_1$을 획득하기 위하여 코드해시를 획득하고, 코드($C_1$)를 검색하기 위해 이를 사용한다. 만약 RIPEMD-160(SHA256($C_1$))과 $H_1$이 동일하면, 코드는 변경되지 않고 다음 단계로 진행하는 것이 안전하다.

[0416] 4. 에이전트는 인덱스 I를 저장하는 '메타데이터-코드해시'를 읽고, i번째 반복에서 코드를 재생성한다. 다시 말해, 루프는 적절한 반복에서 '리로드(reload)'된다.

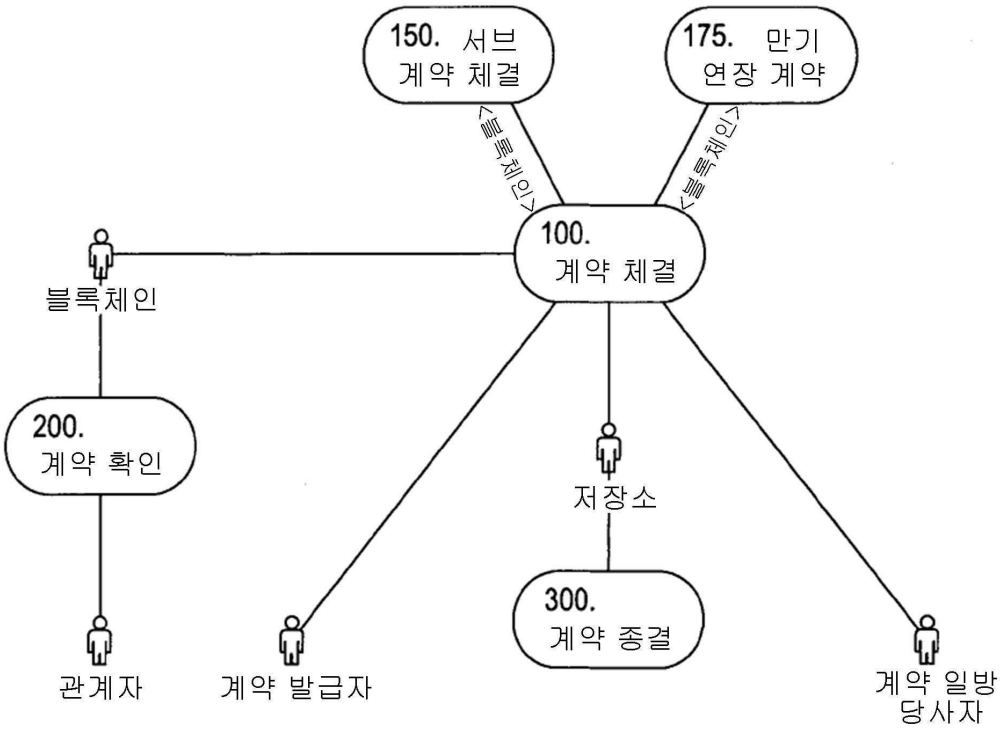
[0417] 5. 사용자 서명은 메타데이터의 출처를 확인하기 위해 P2SH 명령에 포함된다.

[0418] 6. 에이전트는 루프의 반복에 이러한 데이터가 필요하다면, 이전 단계의 출력을 검색하기 위해 '메타데이터-출력해시(Metadata-OutputHash)' 및 '메타데이터-출력포인터(Metadata-OutputPointer)'(도 6 참조)를 읽는다.

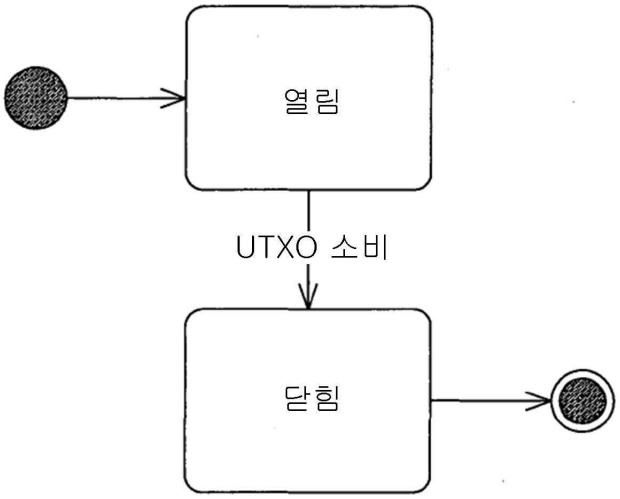
[0419] 전술한 실시예들은 본 발명을 제한하기보다는 예시하고, 당업자는 첨부된 청구 범위에 의해 정의된 본 발명의 범위를 벗어나지 않고 많은 대안적인 실시예를 설계 할 수 있음을 알아야 한다. 청구 범위에서, 괄호 안의 임의의 참조 부호는 청구 범위를 제한하는 것으로 해석되어서는 안된다. 용어 "포함하는" 및 "포함하다" 등은 청구 범위 또는 명세서 전체에 열거 된 요소 또는 단계 이외의 요소의 존재를 배제하지 않는다. 본 명세서에서, "포함한다"는 "포함하거나 구성한다"를 의미하고 "포함하는"는 "포함하거나 구성하는"을 의미한다. 요소의 단일 참조는 이러한 요소의 복수 참조를 배제하지 않으며 그 반대도 마찬가지이다. 본 발명은 몇몇 별개의 요소들을 포함하는 하드웨어에 의해 그리고 적절하게 프로그래밍 된 컴퓨터에 의해 구현될 수 있다. 여러 수단들을 열거하는 장치 청구항에서, 이들 수단들 중 몇몇은 하나의 동일한 하드웨어 아이템에 의해 구현될 수 있다. 특정 측정값이 서로 다른 종속 항에서 인용된다는 단순한 사실만으로 이 측정 값의 조합을 활용할 수 없다는 것을 의미하지는 않는다.

## 도면

### 도면1


150. 서브 계약 체결
175. 만기 연장 계약
<블록체인>
<블록체인>
100. 계약 체결
블록체인
200. 계약 확인
저장소
300. 계약 종결
관계자
계약 발급자
계약 일방 당사자

### 도면2a


열림
UTXO 소비
닫힘

도면2b

시나리오 정의
밥: 블록체인 상에 집을 등록하기 원함

집 기반 메타데이터

| 필드 | 서브 필드 | 바이트 | 값 | 설명 |
|---|---|---|---|---|
| 자산 메타데이터 A | 계약 유형 | 4 | 0x0000FF04 | 단위 표시 |
| | 계약 포인터 | 16 | xxxx.xxxx.xxxx.xxxx (...) .xxxx | 자산 정의 파일의 주소 |
| | 패딩 | 12 | | 예비 |
| 자산 메타데이터 B | 계약 해시 | 20 | ##################... | 자산 정의 파일의 해시(토큰화 아님) |
| | 관할 | 2 | EN | 자산이 영국법에 의해 다루어짐을 특정 |
| | 옵션 | 2 | 0x0000 | 특정된 옵션 없음 |
| | 패딩 | 8 | | 예비 |

도면2c

| 블록체인 상에 자산의 소유권을 공개하기 위한 거래이다. 이는 매우 간단한 거래이다. | | |
|---|---|---|
| 밥의 공개 | | |
| BOB-S1-T1 | 거래 ID | |
| 버전 번호 | 버전 번호 | |
| 1 | 입력 번호 | |
| <밥의 이전 미사용 BTC 출력 – 500,000 사토시를 가정> | 이전 거래 출력 | |
| IDX-00 | 이전 거래 출력 인덱스 | |
| 스크립트 길이 | 스크립트 길이 | |
| Sig-Bob PubK-Bob | ScriptSig | |
| 시퀀스 번호 | 시퀀스 번호 | |
| 1 | 출력 번호 | |
| 2,000 | 출력 값 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 | 리딤 스크립트; 언제든 밥이 계약을 취소할 수 있도록 함 |
| OP_HASH160 <redeem script hash> OP_EQUAL | 출력 스크립트 | <<<< OP_1AssetMetaDataA AssetMetadataB PubK-Bob OP_3 OP_CHECKMULTISIG |
| 498,000 | 출력 값 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 | |
| OP_DUP OP_HASH160 <PubK-Bob Hash> OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 | |
| 잠금 시간 | 잠금 시간 | |

도면2d

| 밥이 자산을 없애거나 공개(준 공개) 지식을 더 이상 요구하지 않으면, 거래 출력을 단순히 소비한다. | |
|---|---|
| 밥의 거래 취소 | |
| BOB-S1-T2 | 거래 ID |
| 버전 번호 | 버전 번호 |
| 1 | 입력 번호 |
| BOB-S1-T1 | 이전 거래 출력 |
| IDX-00 | 이전 거래 출력 인덱스 |
| 스크립트 길이 | 스크립트 길이 |
| Sig-Bob OP_1AssetMetadataA AssetMetadataB PubK-Bob OP_3 OP_CHECKMULTISIG | ScriptSig |
| 시퀀스 번호 | 시퀀스 번호 |
| 1 | 출력 번호 |
| 1,000 | 출력 값 |
| 출력 스크립트 길이 | 출력 스크립트 길이 |
| OP_DUP OP_HASH160<PubK-Bob Hash>OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 |
| 잠금 시간 | 잠금 시간 |

도면3a

시나리오 정의
밥 : 숨겨진 소유권을 가진 자산을 생성하고 이를 블록체인에 공개하기 원함

집 기반 메타데이터

| 필드 | 서브 필드 | 바이트 | 값 | 설명 |
|---|---|---|---|---|
| 자산 메타데이터 A | 계약 유형 | 4 | 0x0000FF04 | 단위 표시 |
| | 계약 포인터 | 16 | xxxx.xxxx.xxxx.xxxx (...) .xxxx | 자산 정의 파일의 주소 |
| | 패딩 | 12 | | 예비 |
| 자산 메타데이터 B | 계약 해시 | 20 | ################... | 자산 정의 파일의 해시(토큰화 아님) |
| | 관할 | 2 | EN | 자산이 영국법에 의해 다루어짐을 특정 |
| | 옵션 | 2 | 0x0000 | 특정된 옵션 없음 |
| | 패딩 | 8 | | 예비 |

도면3b

| 밥의 자산 펀딩 | |
|---|---|
| BOB-S2-T1 | |
| | |
| 1 | |
| <밥의 이전 미사용 BTC 출력 – 500,000 사토시를 가정> | |
| IDX-00 | |
| | |
| Sig-Bob PubK-Bob | |
| | |
| 2 | |
| 4,000 | 출력값 |
| | |
| OP_DUP OP_HASH160<PubK-Asset Hash>OP_EQUALVERIFY OP_CHECKSIG | |
| 496,000 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 |
| OP_DUP OP_HASH160 <PubK-Bob Hash> OP_EQUALVERIFY OP_CHECKSIG | |
| | |

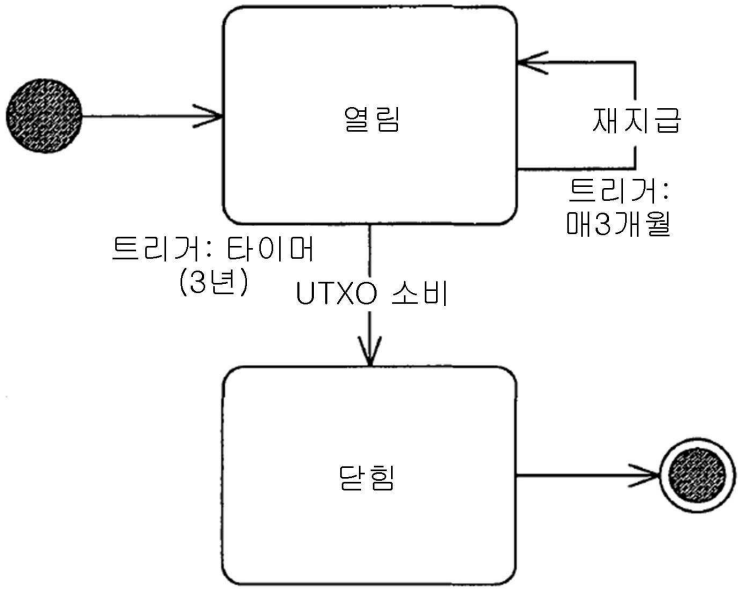
도면3c

| | 자산 공개 | |
|---|---|---|
| | ASSET-S2-T1 | |
| | 버전 넘버 | 버전 넘버 |
| | 1 | |
| | BOB-S2-T1 | |
| | IDX-00 | |
| | | |
| | Sig-Asset PubK-Asset | |
| | | |
| | 2 | |
| | 1,000 | |
| 리딤 스크립트, 밥과 자산에 그것을 최소하라고 요청 | | |
| OP_2AssetMetaDataA AssetMetadataB PubK-Asset PubK-Bob OP_4 OP_CHECKMULTISIG >>>> | OP_HASH160<Redeem script hash>OP_EQUAL | |
| | 2,000 | |
| | | |
| | OP_DUPOP HASH160 <PubK-AssetHash> OP EQUALVERIFY OP_CHECKSIG | |
| | | |

도면3d

| 계약 마감 | |
|---|---|
| ASSET-S2-T2 | 거래 ID |
| 버전 번호 | 버전 번호 |
| 1 | 입력 번호 |
| ASSET-S2-T1 | 이전 거래 출력 |
| IDX-00 | 이전 거래 출력 인덱스 |
| 스크립트 길이 | 스크립트 길이 |
| Sig-Asset Sig-Bob OP_2 AssetMetaDataA AssetMetadataB PubK-Asset PubK-Bob OP_4 OP_CHECKMULTISIG | ScriptSig |
| 시퀀스 번호 | 시퀀스 번호 |
| 1 | 출력 번호 |
| 1,000 | 출력 값 |
| 출력 스크립트 길이 | 출력 스크립트 길이 |
| OP_DUP OP_HASH160<PubK-Bob Hash>OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 |
| 잠금 시간 | 잠금 시간 |

도면4a




도면4b

시나리오 정의
밥: 나는 3년 동안 이브로부터 차를 임대하고 있다.

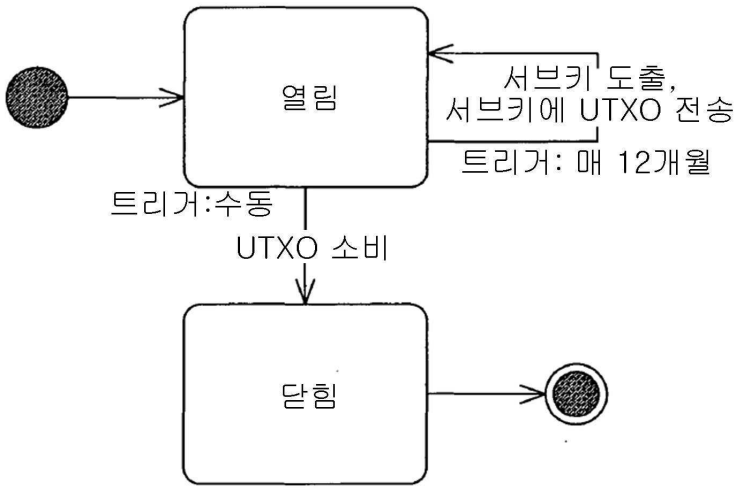
임대 기반 메타데이터

| 필드 | 서브 필드 | 바이트 | 값 | 설명 |
|---|---|---|---|---|
| Asset Metadata A | 계약 유형 | 4 | 0x0000FF04 | 단위 표시 |
| | 계약 포인터 | 16 | xxxx.xxxx.xxxx.xxxx (...) .xxxx | 자산 정의 파일의 주소 |
| | 패딩 | 12 | | 예비 |
| Asset Metadata B | 계약 해시 | 20 | ################... | 자산 정의 파일의 해시(토큰화 아님) |
| | 관할 | 2 | EN | 자산이 영국법에 의해 다루어짐을 특정 |
| | 옵션 | 2 | 0x0000 | 특정된 옵션 없음 |
| | 패딩 | 8 | | 예비 |

도면4c

| | | | |
|---|---|---|---|
| | | 이브의 임대 계약 생성 | |
| | | EVE-S3-T1 | 거래 ID |
| | | 버전 번호 | 버전 번호 |
| | | 1 | 입력 번호 |
| | | <Eve's previous unspent BTC output - assume 500,000 satoshi> | 이전 거래 출력 |
| | | IDX-00 | 이전 거래 출력 인덱스 |
| | | 스크립트 길이 | 스크립트 길이 |
| | | Sig-Eve PubK-Eve | ScriptSig |
| | | 시퀀스 번호 | 시퀀스 번호 |
| | | 2 | 출력 번호 |
| | | 2,000 | 출력 값 |
| 리딩 스크립트: 단순히 이브가 그것을 닫을 것을 요구함 (밥과 이브 모두에게 요구하는 솔루션이 또한 가능할 수 있음에 유의) | | 출력 스크립트 길이 | 출력 스크립트 길이 |
| OP_1AssetMetaDataA AssetMetadataB PubK-Eve OP_3 OP_CHECKMULTISIG | >>>> | OP_HASH160<Redeem script hash>OP_EQUAL | 출력 스크립트 |
| | | 498,000 | 출력 값 |
| | | 출력 스크립트 길이 | 출력 스크립트 길이 |
| | | OP_DUP OP_HASH160 <PubK-Eve Hash> OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 |
| | | 잠금 시간 | 잠금 시간 |
| | | 이브의 타임락된 계약 종결 | |
| | | EVE-3-T2 | |
| | | 버전 번호 | 버전 번호 |
| | | 1 | 입력 번호 |
| | | EVE-3-T1 | 이전 거래 출력 |
| | | IDX-00 | 이전 거래 출력 인덱스 |
| | | Script length | 스크립트 길이 |
| | | Sig-Eve OP_1AssetMetaDataA AssetMetadataB PubK-Eve OP_3 OP_CHECKMULTISIG | ScriptSig |
| | | Sequence number | 시퀀스 번호 |
| | | 1 | 출력 번호 |
| | | 2,000 | 출력 값 |
| | | 출력 스크립트 길이 | 출력 스크립트 길이 |
| | | OP_DUP OP_HASH160<PubK-Eve Hash>OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 |
| 주의 : 거래는 3년 기간의 끝보다 미리 제출된다. | >>>> | EVE-S3-T1이 전송된 날짜 + 3년 | 잠금 시간 |

도면5a




도면5b

시나리오 정의
밥 : 나는 연간 기준으로 이브로부터 집을 임대하고 있다. 그러나 연간의 2개월 내에 취소할 수 있다.

임대 기반 메타데이터

| 필드 | 서브 필드 | 바이트 | 값 | 설명 |
|---|---|---|---|---|
| 자산 메타데이터 A | 계약 유형 | 4 | 0x0000FF04 | 단위 표시 |
| | 계약 포인터 | 16 | xxxx.xxxx.xxxx.xxxx (...) .xxxx | 자산 정의 파일의 주소 |
| | 패딩 | 12 | | 예비 |
| 자산 메타데이터 B | 계약 해시 | 20 | ################... | 자산 정의 파일의 해시(토큰화 아님) |
| | 관할 | 2 | EN | 자산이 영국법에 의해 다루어짐을 특정 |
| | 옵션 | 2 | 0x0000 | 특정된 옵션 없음 |
| | 패딩 | 8 | | 예비 |

도면5ca

| | | 이브의 임대 계약 생성 | |
|---|---|---|---|
| | | EVE-S4-T1 | 거래 ID |
| | | 버전 번호 | 버전 번호 |
| | | 1 | 입력 번호 |
| | | <이브의 이전 미사용 BTC 출력 - 500,000 사토시 추정> | 이전 거래 출력 |
| | | IDX-00 | 이전 거래 출력 인덱스 |
| | | 스크립트 길이 | 스크립트 길이 |
| | | Sig-Eve PubK-Eve | ScriptSig |
| | | 시퀀스 번호 | 시퀀스 번호 |
| | | 2 | 출력 번호 |
| | | 1,000 | 출력 값 |
| 리딤 스크립트: 단순히 이브가 그것을 달을 것을 요구함 (밥과 이브 모두에게 요구하는 솔루션이 또한 가능할 수 있음에 유의) | | 출력 스크립트 길이 | 출력 스크립트 길이 |
| OP_2AssetMetaDataA AssetMetadataB PubK-Bob PubK-Eve PubK-Oracle OP_5 OP_CHECKMULTISIG | >>>> | OP_HASH160<Redeem script hash>OP_EQUAL | 출력 스크립트 |
| | | 499,000 | 출력 값 |
| | | 출력 스크립트 길이 | 출력 스크립트 길이 |
| | | OP_DUP OP_HASH160 <PubK-Eve Hash> OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 |
| | | 잠금 시간 | 잠금 시간 |
| | | 이브의 타임락된 계약 만기 연장 | |
| | | EVE-S4-T2 | 거래 ID |
| | | 버전 번호 | 버전 번호 |
| | | 2 | 입력 번호 |
| | | <1000 사토시의 이브의 채광료(타임락 효과 때문에 이 거래로부터 변화 없음에 주의) 이는 정확한 값의 입력을 얻기 위해이전 거래를 생성할 것을 의미한다> | |
| | | IDX-00 | 이전 거래 출력 |
| | | 스크립트 길이 | 스크립트 길이 |
| | | Sig-Eve Sig-Bob OP_2AssetMetaDataA AssetMetadataB PubK-Bob PubK-Eve OP_5 OP_CHECKMULTISIG | ScriptSig |
| | | 시퀀스 번호 | 시퀀스 번호 |
| | | 1 | 출력 번호 |
| | | 1,000 | 출력 값 |
| 리딤 스크립트: 밥 외 2개를 요청, 이브의 계약 연장 서브키 및 독립적인 에이전트(오라클)가 언락한다. | | 출력 스크립트 길이 | 출력 스크립트 길이 |
| OP_2AssetMetaDataA AssetMetadataB PubK-Bob PubK-Eve SK1 PubK-Oracle OP_5 OP_CHECKMULTISIG | | OP_HASH160 <redeem script hash> OP_EQUAL | 출력 스크립트 |
| 주의 : 거래는 미리 제출된다. | >>>> | EVE-S4-T1이 전송된 날짜 + 1년 | 잠금 시간 |

도면5cb

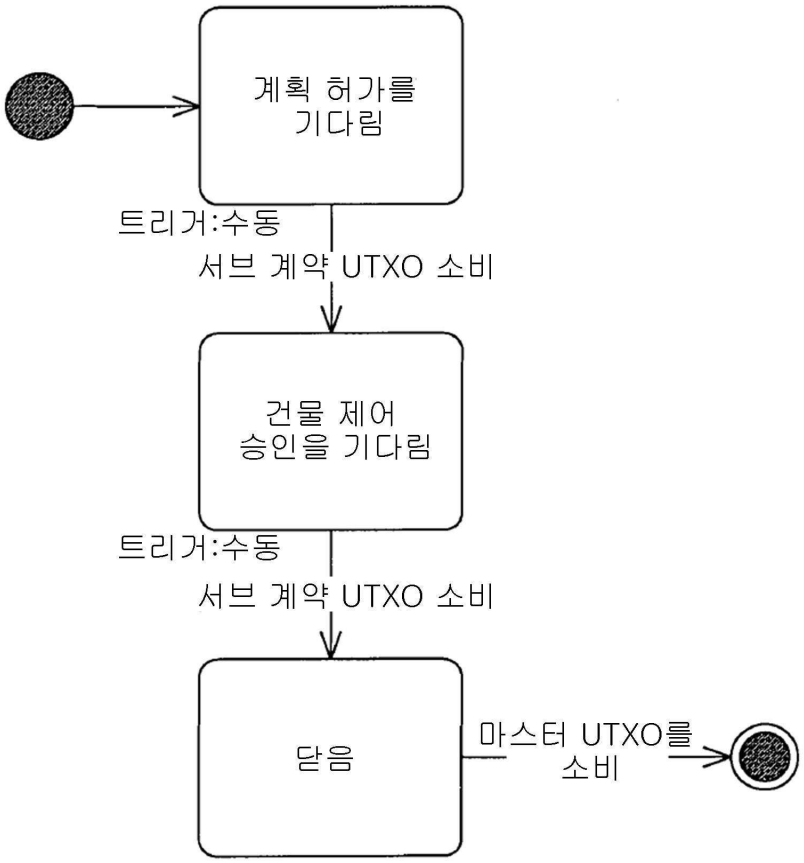
1년 후, 밥은 임대를 계속하며 종결하지 않는다. 바로 후 EVE-S3-T2가 공개되면 에이전트(오라클)에 의해 선택되고 또 다른 해로 연장된다 (그녀 자신의 내부 로직을 이용하여 이브에 의해 완료될 수 있음에 유의)

| | | 이브의 타임락된 계약 연장 | |
|---|---|---|---|
| | | EVE-S4-T3 | 거래-ID |
| | | 버전 번호 | 버전 번호 |
| | | 2 | 입력 번호 |
| | | <1000 사토시의 이브의 채광료(타임락 효과 때문에 이 거래로부터 변화 없음에 주의) 이는 정확한 값의 입력을 얻기 위해 이전 거래를 생성할 것을 의미한다> | 이전 거래 출력 |
| | | IDX-00 | 이전 거래 출력 인덱스 |
| | | 스크립트 길이 | 스크립트 길이 |
| | | Sig-Eve PubK-Eve | ScriptSig |
| | | EVE-S4-T2 | 시퀀스 번호 |
| | | IDX-00 | 이전 거래 출력 인덱스 |
| | | 스크립트 길이 | 스크립트 길이 |
| | | Sig-EveSK1 Sig-Oracle OP_2AssetMetaDataA AssetMetadataB PubK-Bob PubK-EveSK1 PubK-Eve OP_5 OP_CHECKMULTISIG | ScriptSig |
| | | 시퀀스 번호 | 시퀀스 번호 |
| | | 1 | 출력 번호 |
| | | 1,000 | 출력 값 |
| 리딤 스크립트: 밥 외 2개를 요청, 이브의 계약 연장 서브키 및 독립적인 에이전트(오라클)가 언락한다. | | 출력 스크립트 길이 | 출력 스크립트 길이 |
| OP_2AssetMetaDataA AssetMetadataB PubK-Bob PubK-EveSK2 PubK-Oracle OP_5 OP_CHECKMULTISIG | | OP_HASH160 <redeem script hash> OP_EQUAL | 출력 스크립트 |
| 주의 : 거래는 미리 제출된다 | >>>> | Date when EVE-S4-T2 was transmitted plus 1 year | 잠금 시간 |

도면5d

| 밥의 계약 종결 | |
|---|---|
| BOB-S4-T1 | 거래 ID |
| 버전 번호 | 버전 번호 |
| 2 | 입력 번호 |
| <1000 사토시의 밥의 채광료(타임락 효과 때문에 이 거래로부터 변화 없음에 주의) 이는 정확한 값의 입력을 얻기 위해 이전 거래를 생성할 것을 의미한다> | 이전 거래 출력 |
| IDX-00 | 이전 거래 출력 인덱스 |
| 스크립트 길이 | 스크립트 길이 |
| Sig-Bob PubK-Bob | ScriptSig |
| EVE-S4-T2 | 시퀀스 번호 |
| IDX-00 | 이전 거래 출력 인덱스 |
| 스크립트 길이 | 스크립트 길이 |
| Sig-Bob Sig-Oracle OP_2AssetMetaDataA AssetMetadataB PubK-Bob PubK-EveSK1 PubK-Eve OP_3 OP_CHECKMULTISIG | ScriptSig |
| 시퀀스 번호 | 시퀀스 번호 |
| 1 | 출력 번호 |
| 1,000 | 출력 값 |
| 스크립트 길이 | 출력 스크립트 길이 |
| OP_DUP OP_HASH160<PubK-Eve Hash>OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 |
| 잠금 시간 | 잠금 시간 |

도면6a



도면6b

시나리오 정의

밥 : 나는 건물을 짓고 있고, 계약이 만족되기 전에 그 과정에서 서로 다른 시간에 두 가지 독립적인 평가가 필요합니다(계획 허가 및 건물 승인).

임대 기반 메타데이터

| 필드 | 서브 필드 | 바이트 | 값 | 설명 |
|---|---|---|---|---|
| 자산 메타데이터 A | 계약 유형 | 4 | 0x0000FF04 | 단위 표시 |
| | 계약 포인터 | 16 | xxxx.xxxx.xxxx.xxxx (...) .xxxx | 자산 정의 파일의 주소 |
| | 패딩 | 12 | | 예비 |
| 자산 메타데이터 B | 계약 해시 | 20 | ################... | 자산 정의 파일의 해시(토큰화 아님) |
| | 관할 | 2 | EN | 자산이 영국법에 의해 다루어짐을 특정 |
| | 옵션 | 2 | 0x0000 | 특정된 옵션 없음 |
| | 패딩 | 8 | | 예비 |

도면6ca

| 밥의 부동산 건축 계약 생성 | | |
|---|---|---|
| BOB-S5-T1 | 거래 ID | |
| 버전 번호 | 버전 번호 | |
| 1 | 입력 번호 | |
| <Bob's previous unspent BTC output - assume 500,000 satoshi> | 이전 거래 출력 | |
| IDX-00 | 이전 거래 출력 인덱스 | |
| 스크립트 길이 | 스크립트 길이 | |
| Sig-Bob PubK Bob | ScriptSig | |
| 시퀀스 번호 | 시퀀스 번호 | |
| 2 | 출력 번호 | |
| 2,000 | 출력 값 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 | 리딩 스크립트; 밥 외 2명을 요청하고, 오라클이 포함한다. |
| OP_HASH160 <redeem script hash> OP_EQUAL | 출력 스크립트 | >>>> OP_1AssetMetaDataA AssetMetadataB PubK-Bob PubK-Oracle OP_4 OP_CHECKMULTISIG |
| 497,000 | 출력 값 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 | |
| OP_DUP OP_HASH160 <PubK-Bob Hash> OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 | |
| 잠금 시간 | 잠금 시간 | |

도면6cb

| 계획 승인을 확정하기 위하여그의 도출키를 이용하여 밥의 부동산 건축 계약 생성 | | |
|---|---|---|
| BOB-S5-T2 | 거래 ID | |
| 버전 번호 | 버전 번호 | |
| 1 | 입력 번호 | |
| BOB-S5-T1 | 이전 거래 출력 | |
| IDX-01 | 이전 거래 출력 인덱스 | |
| 스크립트 길이 | 스크립트 길이 | |
| Sig-Bob PubK Bob | ScriptSig | |
| 시퀀스 번호 | 시퀀스 번호 | |
| 2 | 출력 번호 | |
| 2,000 | 출력 값 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 | 리딩 스크립트: 승인을 위한 계획 승인 및 승인을 위한 오라클을 요청, 그러나 밥 및 둘 중 하나를 위하여 대기 |
| OP_HASH160 <redeem script hash> OP_EQUAL | 출력 스크립트 | >>>> OP_2AssetMetaDataA AssetMetadataB PubK-BobSK1 PubK-PlanningAuthority PubK-Oracle OP_5 |
| 494,000 | 출력 값 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 | |
| OP_DUP OP_HASH160 <PubK-Bob Hash> OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 | |
| 잠금 시간 | 잠금 시간 | |

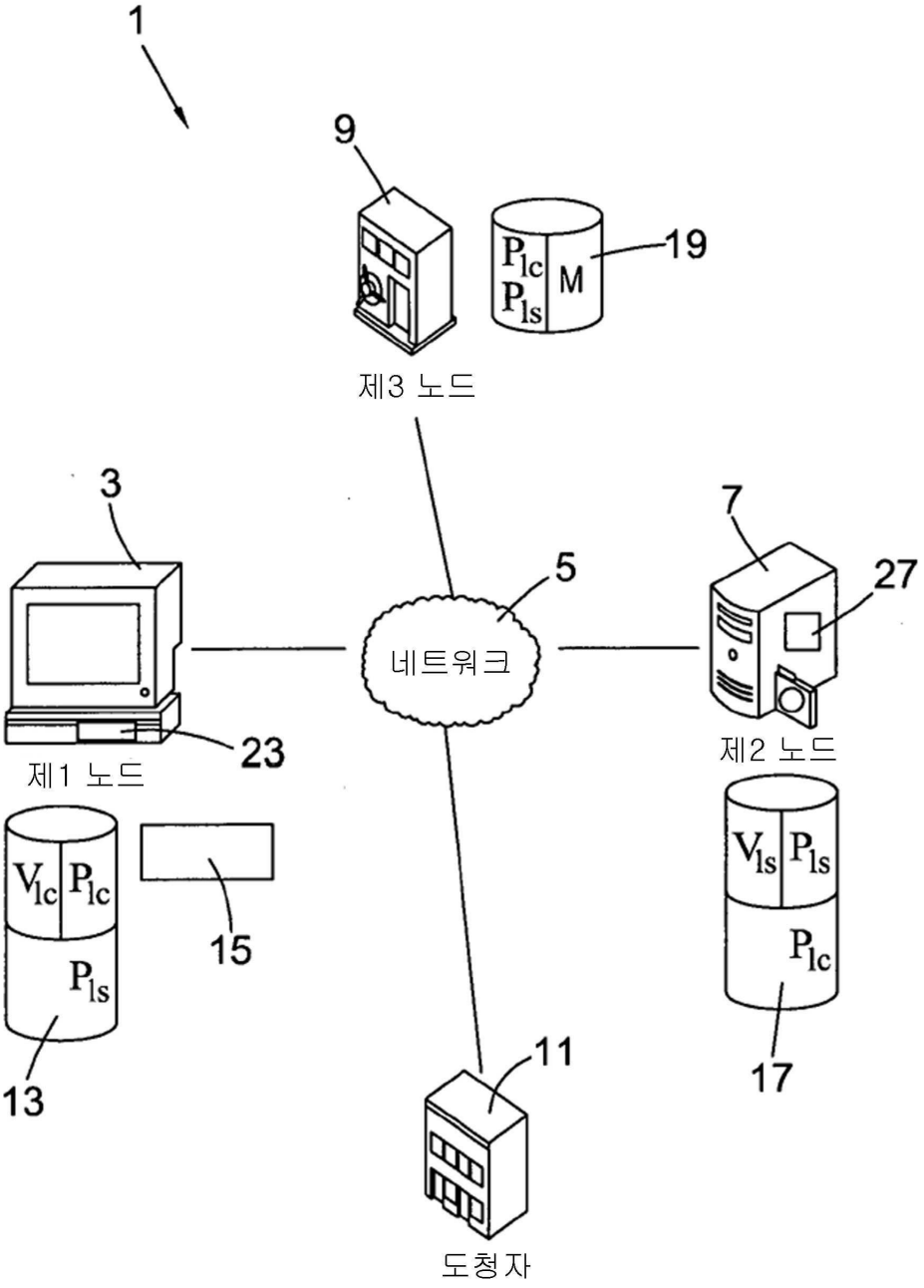
도면6cc

| 계획 승인을 확정하기 위하여그의 도출키를 이용하여 밥의 부동산 건축 계약 생성 | | |
|---|---|---|
| BOB-S5-T3 | 거래 ID | |
| 버전 번호 | 버전 번호 | |
| 1 | 입력 번호 | |
| BOB-S5-T2 | 이전 거래 출력 | |
| IDX-01 | 이전 거래 출력 인덱스 | |
| 스크립트 길이 | 스크립트 길이 | |
| Sig-Bob PubK Bob | ScriptSig | |
| 시퀀스 번호 | 시퀀스 번호 | |
| 2 | 출력 번호 | |
| 2,000 | 출력 값 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 | 리딤 스크립트: 승인을 위한 계획 승인 및 승인을 위한 오리클을 요청, 그러나 밥 및 톰 하나를 위하여 대기 |
| OP_HASH160 <redeem script hash> OP_EQUAL | 출력 스크립트 | >>>> OP_2AssetMetaDataA AssetMetadataB PubK-BobSK2 PubK-BuildingStandard PubK-Oracle OP_5 |
| 491,000 | 출력 값 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 | |
| OP_DUP OP_HASH160 <PubK-Bob Hash> OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 | |
| 잠금 시간 | 잠금 시간 | |

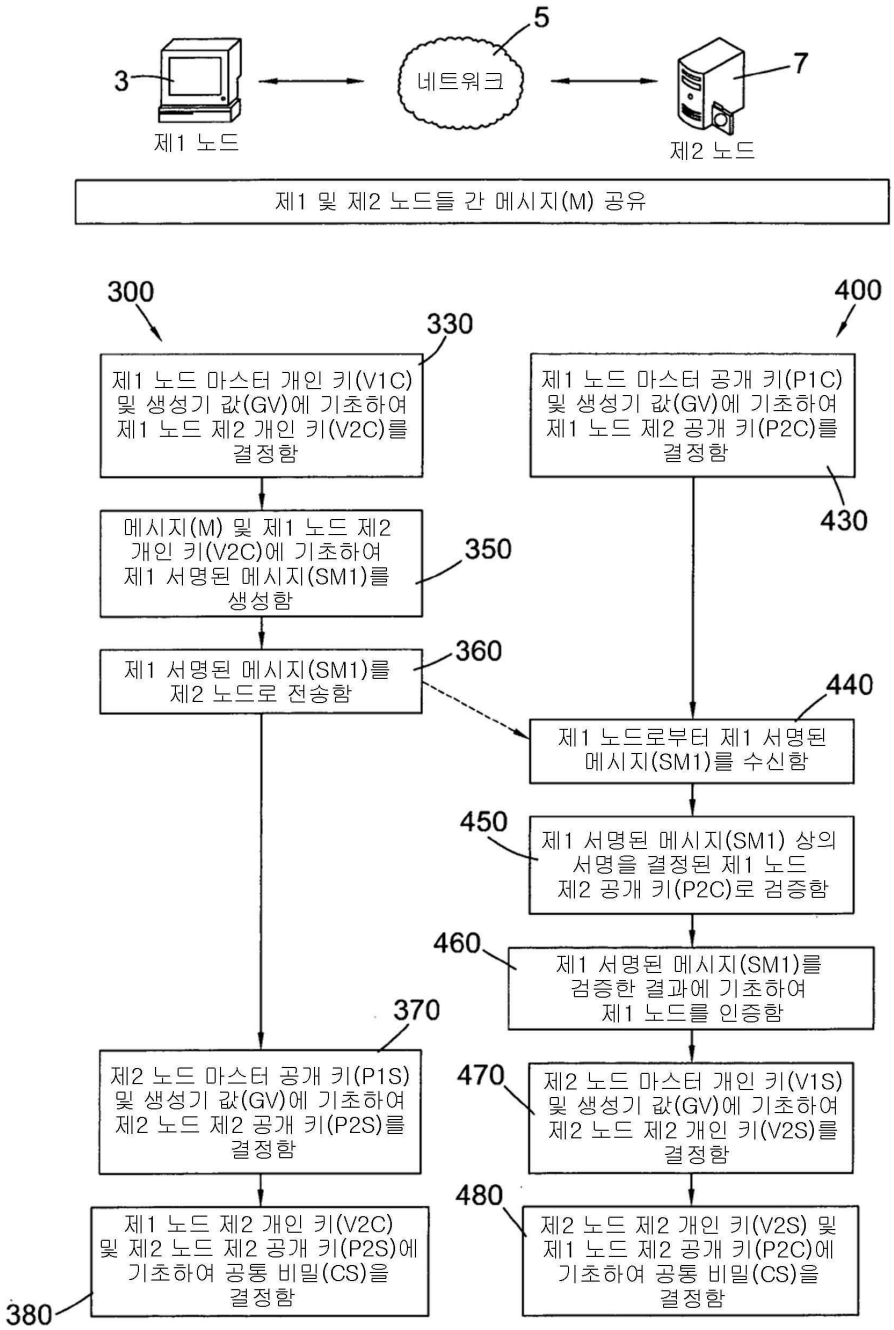
도면6d

| 계획 권한 사인오프 | | |
|---|---|---|
| BOB-S5-T4 | 거래 ID | |
| 버전 번호 | 버전 번호 | |
| 1 | 입력 번호 | |
| BOB-S5-T2 | 이전 거래 출력 | |
| IDX-00 | 이전 거래 출력 인덱스 | |
| 스크립트 길이 | 스크립트 길이 | |
| Sig-PlanningAuthority Sig-Orade OP_2AssetMetaDataA AssetMetadateB PubK-BobSK1 PubK-PlanningAuthority PubK-Orade OP_5 OP_CHECKMULTISIG | ScriptSig | |
| 시퀀스 번호 | 시퀀스 번호 | |
| 1 | 출력 번호 | |
| 1,000 | 출력 값 | |
| 출력 스크립트 길이 | 출력 스크립트 길이 | 주의 |
| OP_DUP OP_HASH160<PubK-PlanningAuthority Hash>OP_EQUALVERIFY OP_CHECKSIG | 출력 스크립트 | 사토시는 계획 권한 요금을 지불한다(오라클 및 계획 권한 모두에 대해 2개의 출력 모델이 있을 수 있다는 점에 유의) <<<< model for both the orade and planning authority) |
| 잠금 시간 | 잠금 시간 | |

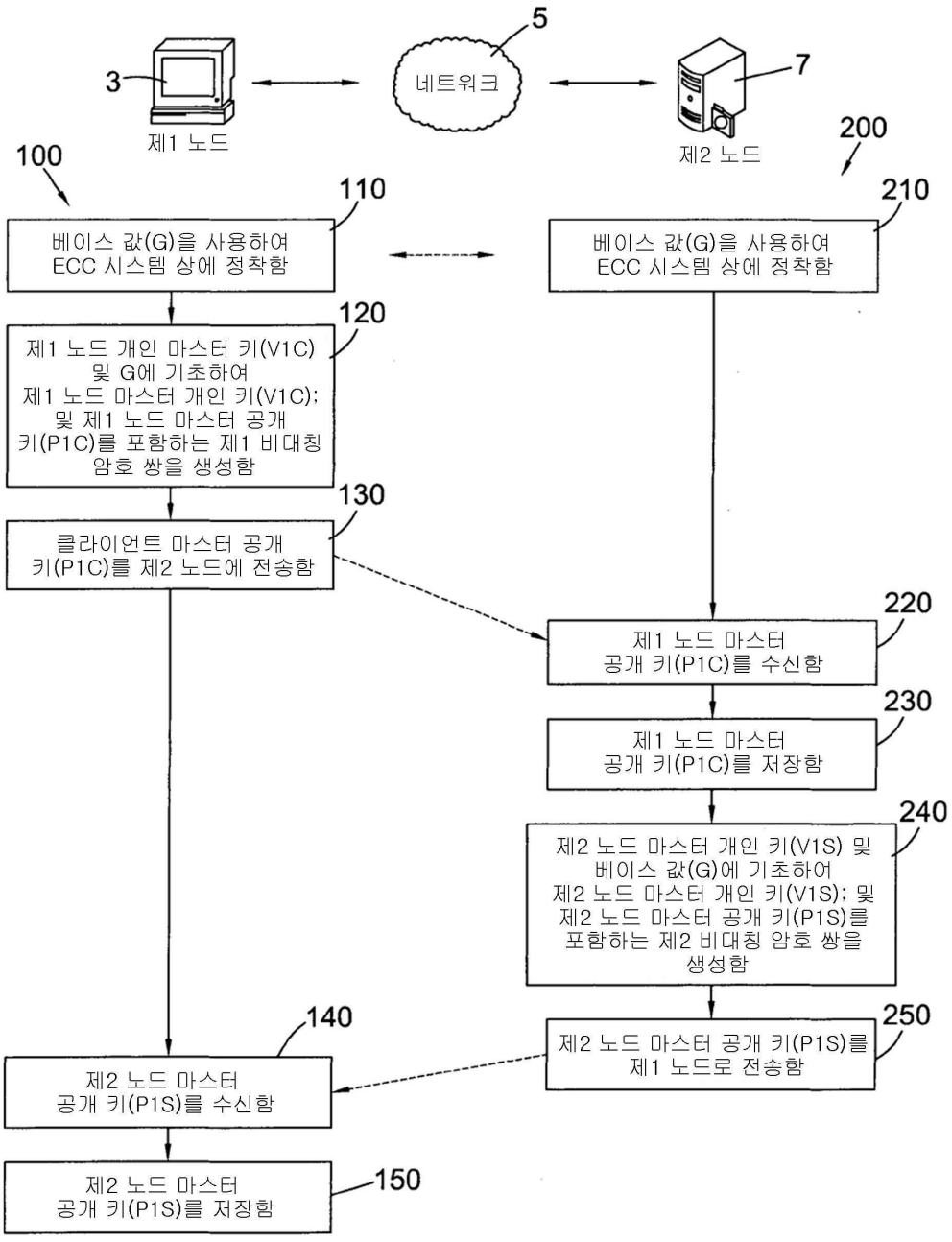
도면7



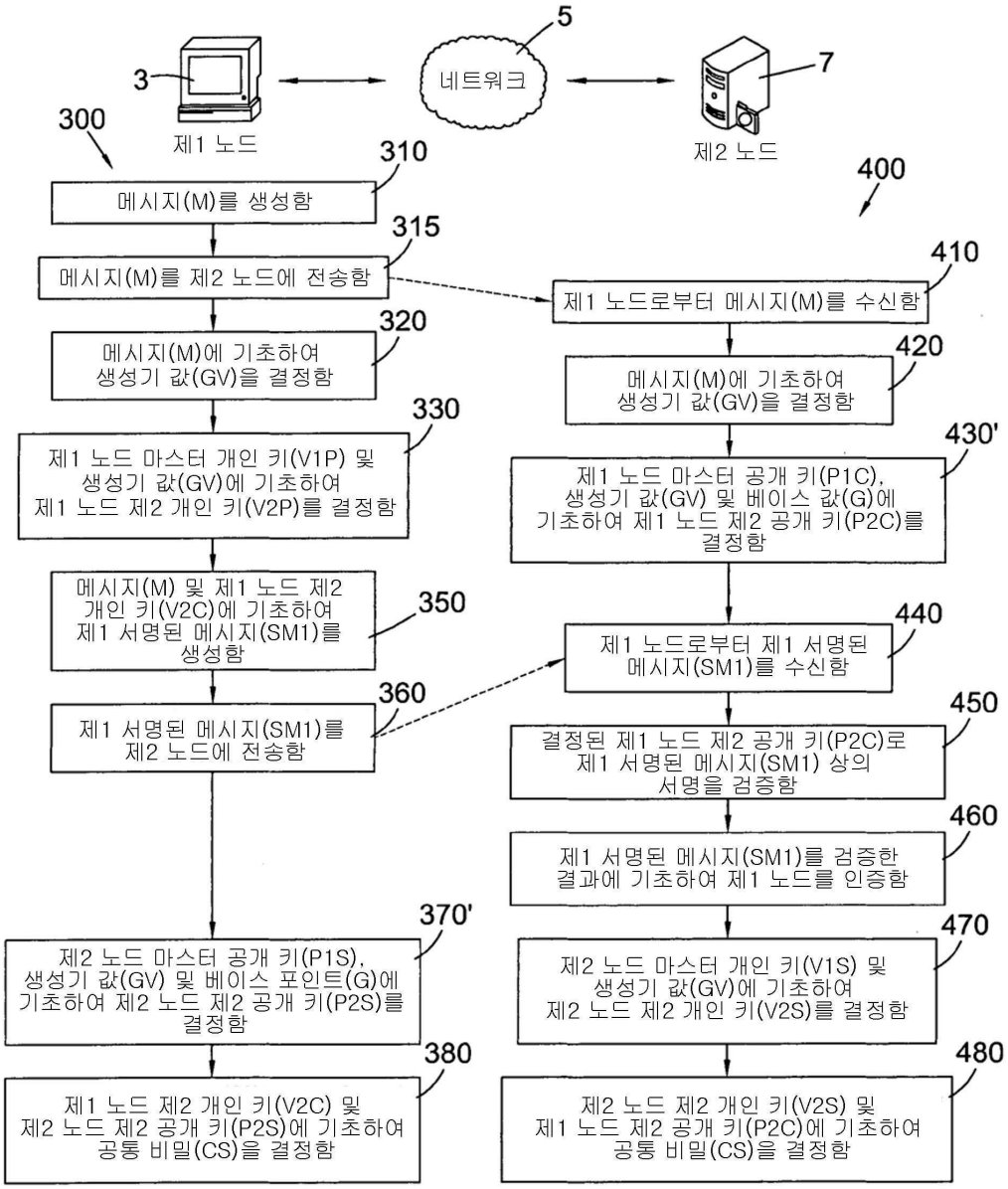
도면8


3
제1 노드
5
네트워크
7
제2 노드
제1 및 제2 노드들 간 메시지(M) 공유
300
330
제1 노드 마스터 개인 키(V1C) 및 생성기 값(GV)에 기초하여 제1 노드 제2 개인 키(V2C)를 결정함
400
제1 노드 마스터 공개 키(P1C) 및 생성기 값(GV)에 기초하여 제1 노드 제2 공개 키(P2C)를 결정함
430
메시지(M) 및 제1 노드 제2 개인 키(V2C)에 기초하여 제1 서명된 메시지(SM1)를 생성함
350
제1 서명된 메시지(SM1)를 제2 노드로 전송함
360
440
제1 노드로부터 제1 서명된 메시지(SM1)를 수신함
450
제1 서명된 메시지(SM1) 상의 서명을 결정된 제1 노드 제2 공개 키(P2C)로 검증함
460
제1 서명된 메시지(SM1)를 검증한 결과에 기초하여 제1 노드를 인증함
370
제2 노드 마스터 공개 키(P1S) 및 생성기 값(GV)에 기초하여 제2 노드 제2 공개 키(P2S)를 결정함
470
제2 노드 마스터 개인 키(V1S) 및 생성기 값(GV)에 기초하여 제2 노드 제2 개인 키(V2S)를 결정함
480
제1 노드 제2 개인 키(V2C) 및 제2 노드 제2 공개 키(P2S)에 기초하여 공통 비밀(CS)을 결정함
380
제2 노드 제2 개인 키(V2S) 및 제1 노드 제2 공개 키(P2C)에 기초하여 공통 비밀(CS)을 결정함

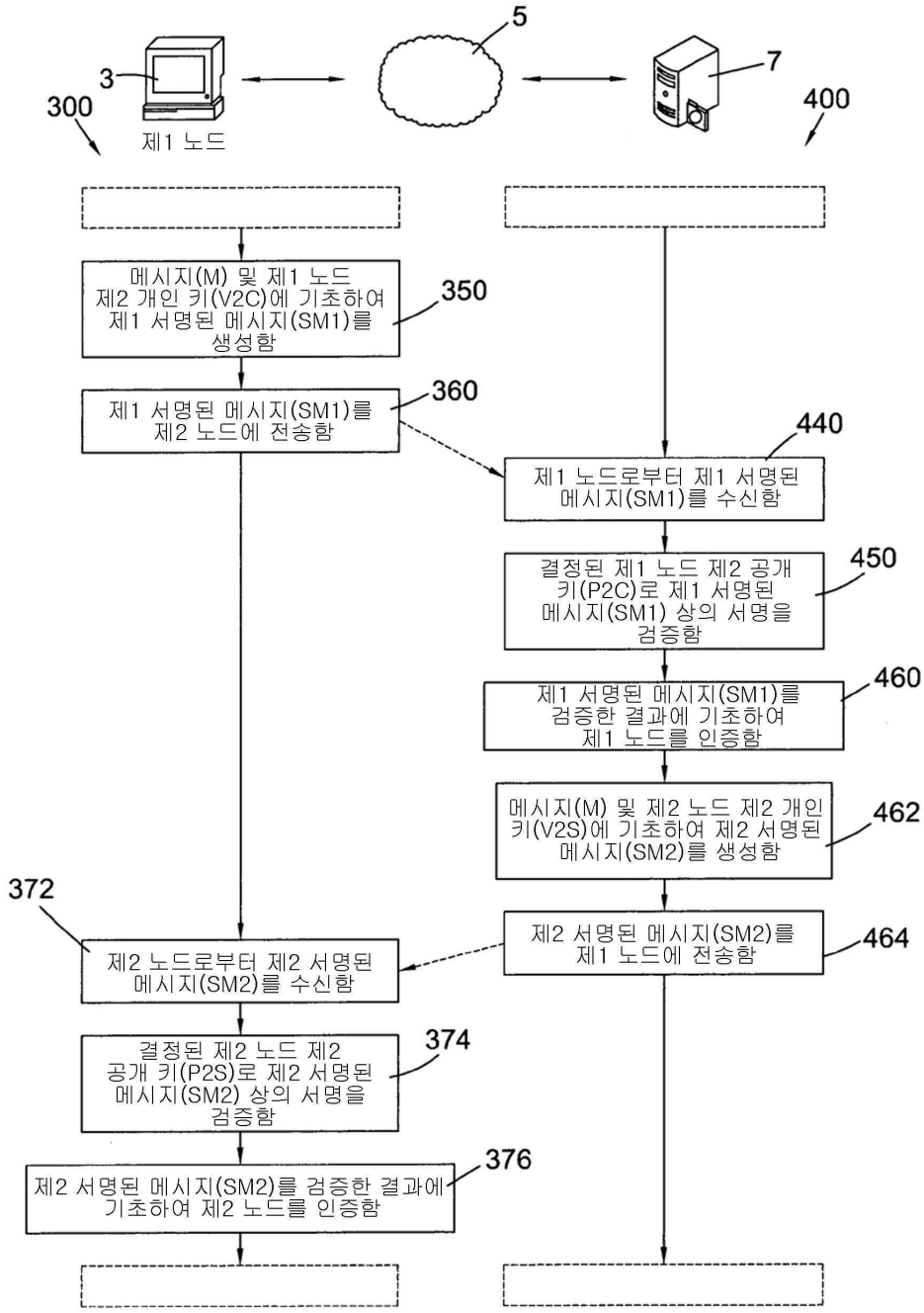
도면9


3
제1 노드
5
네트워크
7
제2 노드
100
200
110
베이스 값(G)을 사용하여 ECC 시스템 상에 정착함
210
베이스 값(G)을 사용하여 ECC 시스템 상에 정착함
120
제1 노드 개인 마스터 키(V1C) 및 G에 기초하여 제1 노드 마스터 개인 키(V1C); 및 제1 노드 마스터 공개 키(P1C)를 포함하는 제1 비대칭 암호 쌍을 생성함
130
클라이언트 마스터 공개 키(P1C)를 제2 노드에 전송함
220
제1 노드 마스터 공개 키(P1C)를 수신함
230
제1 노드 마스터 공개 키(P1C)를 저장함
240
제2 노드 마스터 개인 키(V1S) 및 베이스 값(G)에 기초하여 제2 노드 마스터 개인 키(V1S); 및 제2 노드 마스터 공개 키(P1S)를 포함하는 제2 비대칭 암호 쌍을 생성함
250
제2 노드 마스터 공개 키(P1S)를 제1 노드로 전송함
140
제2 노드 마스터 공개 키(P1S)를 수신함
150
제2 노드 마스터 공개 키(P1S)를 저장함

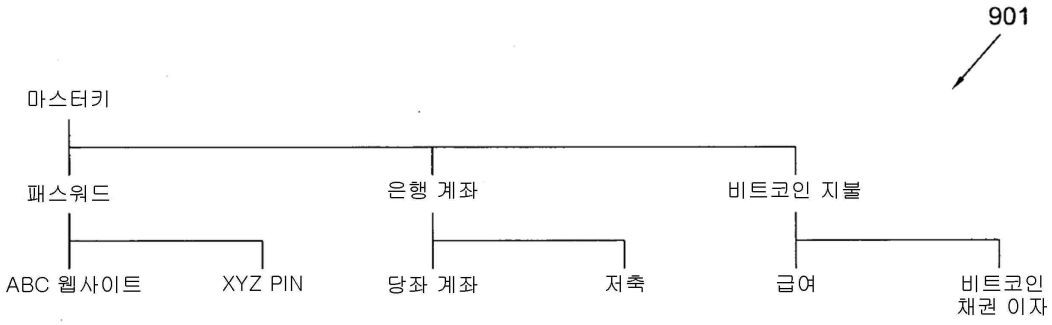
도면10


5
네트워크
3
7
300
제1 노드
제2 노드
400
310
메시지(M)를 생성함
315
메시지(M)를 제2 노드에 전송함
410
제1 노드로부터 메시지(M)를 수신함
320
메시지(M)에 기초하여
생성기 값(GV)을 결정함
420
메시지(M)에 기초하여
생성기 값(GV)을 결정함
330
제1 노드 마스터 개인 키(V1P) 및
생성기 값(GV)에 기초하여
제1 노드 제2 개인 키(V2P)를 결정함
430'
제1 노드 마스터 공개 키(P1C),
생성기 값(GV) 및 베이스 값(G)에
기초하여 제1 노드 제2 공개 키(P2C)를
결정함
350
메시지(M) 및 제1 노드 제2
개인 키(V2C)에 기초하여
제1 서명된 메시지(SM1)를
생성함
440
제1 노드로부터 제1 서명된
메시지(SM1)를 수신함
360
제1 서명된 메시지(SM1)를
제2 노드에 전송함
450
결정된 제1 노드 제2 공개 키(P2C)로
제1 서명된 메시지(SM1) 상의
서명을 검증함
460
제1 서명된 메시지(SM1)를 검증한
결과에 기초하여 제1 노드를 인증함
370'
제2 노드 마스터 공개 키(P1S),
생성기 값(GV) 및 베이스 포인트(G)에
기초하여 제2 노드 제2 공개 키(P2S)를
결정함
470
제2 노드 마스터 개인 키(V1S) 및
생성기 값(GV)에 기초하여
제2 노드 제2 개인 키(V2S)를 결정함
380
제1 노드 제2 개인 키(V2C) 및
제2 노드 제2 공개 키(P2S)에 기초하여
공통 비밀(CS)을 결정함
480
제2 노드 제2 개인 키(V2S) 및
제1 노드 제2 공개 키(P2C)에 기초하여
공통 비밀(CS)을 결정함

도면11


5
3
7
300
400
제1 노드
메시지(M) 및 제1 노드 제2 개인 키(V2C)에 기초하여 제1 서명된 메시지(SM1)를 생성함
350
제1 서명된 메시지(SM1)를 제2 노드에 전송함
360
440
제1 노드로부터 제1 서명된 메시지(SM1)를 수신함
결정된 제1 노드 제2 공개 키(P2C)로 제1 서명된 메시지(SM1) 상의 서명을 검증함
450
제1 서명된 메시지(SM1)를 검증한 결과에 기초하여 제1 노드를 인증함
460
메시지(M) 및 제2 노드 제2 개인 키(V2S)에 기초하여 제2 서명된 메시지(SM2)를 생성함
462
372
제2 노드로부터 제2 서명된 메시지(SM2)를 수신함
제2 서명된 메시지(SM2)를 제1 노드에 전송함
464
결정된 제2 노드 제2 공개 키(P2S)로 제2 서명된 메시지(SM2) 상의 서명을 검증함
374
제2 서명된 메시지(SM2)를 검증한 결과에 기초하여 제2 노드를 인증함
376

도면12


901
마스터키
패스워드
은행 계좌
비트코인 지불
ABC 웹사이트
XYZ PIN
당좌 계좌
저축
급여
비트코인 채권 이자

도면13