(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2020년06월15일
(11) 등록번호 10-2122773
(24) 등록일자 2020년06월09일

(51) 국제특허분류(Int. Cl.)
H04L 9/08 (2006.01)
(52) CPC특허분류
H04L 9/0894 (2013.01)
H04L 9/0836 (2013.01)
(21) 출원번호 10-2018-0121216
(22) 출원일자 2018년10월11일
심사청구일자 2018년10월11일
(65) 공개번호 10-2020-0041134
(43) 공개일자 2020년04월21일
(56) 선행기술조사문헌
EP02871801 A1*
JP3260524 B2*
KR101676720 B1*
de Meer, Hermann, et al. "Redactable Signature Scheme for Trees With Signer-Controlled Non-Leaf_Redactions", International Conference on E-Business and Telecommunications (2012)
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
세종대학교산학협력단
서울특별시 광진구 능동로 209 (군자동, 세종대학교)
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)
(72) 발명자
신지선
서울특별시 송파구 올림픽로 435, 311동 2001호 (신천동, 파크리오)
이동훈
서울특별시 종로구 사직로8길 34, 1404호(내수동)
김종현
서울특별시 강남구 학동로68길 29, 101동 1301호 (삼성동, 삼성동힐스테이트1단지아파트)
(74) 대리인
두호특허법인

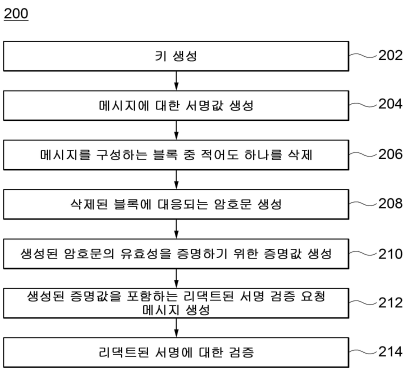
전체 청구항 수 : 총 19 항 심사관 : 양종필

(54) 발명의 명칭 **복원가능 기능을 가지는 리댁터블 서명 시스템 및 방법**

(57) 요 약

복원가능 기능을 가지는 리댁터블 서명 시스템 및 방법이 개시된다. 일 실시예에 따른 방법은, 인증 요청자 단말의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성하는 단계; 상기 메시지(m)를 구성하는 복수개의 블록 중 적어도 하나를 상기 메시지(m)로부터 삭제하는 단계; 상기 메시지(m)로부터 삭제된 블록(m_i)에 대한 암호문(c_i)을 생성하는 단계; 상기 생성된 암호문(c_i)의 유효성을 검증하기 위한 증명값(π)을 생성하는 단계; 및 생성된 상기 증명값(π)을 포함하는 리댁트된 서명 검증 요청 메시지를 생성하는 단계를 포함한다.

대 표 도



(52) CPC특허분류
H04L 2209/08 (2013.01)
H04L 2209/38 (2013.01)

이 발명을 지원한 국가연구개발사업
과제고유번호 1711076037
부처명 과학기술정보통신부
연구관리전문기관 정보통신기술진흥센터
연구사업명 정보보호핵심원천기술개발(R&D)
연구과제명 (함수암호 3세부) 함수서명 설계기법 및 응용기술 연구
기 여 율 1/1
주관기관 고려대학교산학협력단
연구기간 2018.06.01 ~ 2019.03.31

명 세 서

청구범위

청구항 1

하나 이상의 프로세서들, 및

상기 하나 이상의 프로세서들에 의해 실행되는 하나 이상의 프로그램들을 저장하는 메모리를 구비한 인증 요청자 단말 장치에서 수행되는 방법으로서,

상기 인증 요청자 단말의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성하는 단계;

상기 메시지(m)를 구성하는 복수 개의 블록 중 적어도 하나를 상기 메시지(m)로부터 삭제하는 단계;

상기 메시지(m)로부터 삭제된 블록(m_i)에 대한 암호문(c_i)을 생성하는 단계;

상기 생성된 암호문(c_i)의 유효성을 검증하기 위한 증명값(π)을 생성하는 단계; 및

생성된 상기 증명값(π)을 포함하는 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 생성하는 단계를 포함하고,

상기 증명값(π)을 생성하는 단계는, 상기 삭제된 블록(m_i) 및 기 설정된 함수 F에 대한 평가키(EK_F)에 기초하여 상기 증명값(π)을 생성하는, 방법.

청구항 2

청구항 1에 있어서,

상기 서명값(σ_{SIG})을 생성하는 단계는,

상기 메시지(m)를 상기 복수 개의 블록으로 분할하는 단계;

분할된 상기 복수 개의 블록 각각에 대응되는 랜덤값(r)을 생성하고, 상기 복수 개의 블록 및 상기 랜덤값에 대한 해시값을 이용하여 머클 트리(merkle tree)를 구성하는 단계; 및

상기 머클 트리의 루트 해시를 상기 비밀키로 암호화하여 상기 서명값(σ_{SIG})를 계산하는 단계를 더 포함하는, 방법.

청구항 3

청구항 2에 있어서,

상기 암호문(c_i)을 생성하는 단계는, 다음의 수학식

$$c_i \leftarrow PKE.Enc(pk_{PKE}, m_i, s_i)$$

(이때, PKE.Enc는 공개키 기반 암호화 알고리즘, pk_{PKE}는 인증 기관에서 발급한 공개키, s_i는 공개키 기반 암호화 알고리즘에서 사용되는 랜덤값)

에 의하여 계산되는, 방법.

청구항 4

청구항 3에 있어서,

상기 증명값(π)은, 다음의 수학식

$\pi \leftarrow VC.Compute(EK_F, x, w)$

(이때 $x = \perp(null)$, $w = (m_i, r_i, s_i)$, $y = (c_i, h_i)$, r_i는 m_i에 대응되는 랜덤값, $h_i = H(m_i, r_i)$, H는 해시함수, 상기 함수 F는 $F(x, w) = y$의 관계를 만족하는 함수로서 $F(x, w) = (H(m_i, r_i), PKE.Enc(pk_{PKE}, m_i, s_i))$로 정의됨)

에 의하여 계산되는, 방법.

청구항 5

청구항 4에 있어서,

상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})는,

상기 머클 트리의 루트 해시(h), 상기 서명값(σ_{SIG}), 상기 암호문(c_i) 및 상기 증명값(π)을 포함하는, 방법.

청구항 6

청구항 5에 있어서,

상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 수신한 검증자는,

상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})에 포함된 상기 증명값(π)에 대한 제1 검증을 수행하고,

상기 제1 검증에 성공한 경우, 상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})에 포함된 상기 서명값(σ_{SIG})에 대한 제2 검증을 수행하는, 방법.

청구항 7

청구항 6에 있어서,

상기 제1 검증은, 다음의 수학식

$d \leftarrow VC.Verify(VK_F, x, \pi)$

(이때, d는 검증 결과, VK_F는 상기 함수 F에 대응되는 검증키)

에 의하여 수행되는, 방법.

청구항 8

청구항 6에 있어서,

상기 제2 검증은, 다음의 수학식

$d \leftarrow d * SIG.Verify(pk_{SIG}, h, \sigma_{SIG})$

(이때, pk_{sig}는 상기 인증 요청자 단말의 비밀키(sk_{SIG})에 대응되는 공개키)

에 의하여 수행되는, 방법.

청구항 9

청구항 4에 있어서,

상기 메시지(m)로부터 삭제된 블록이 둘 이상인 경우,

상기 인증 요청자 단말은, 삭제된 블록 각각에 대하여 상기 암호문을 생성하는 단계 및 상기 증명값을 생성하는 단계를 반복 수행하도록 구성되는, 방법.

청구항 10

하나 이상의 프로세서들, 및

상기 하나 이상의 프로세서들에 의해 실행되는 하나 이상의 프로그램들을 저장하는 메모리를 구비한 인증 요청자 단말 장치에서 수행되는 방법으로서,

상기 인증 요청자 단말의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성하는 단계;

상기 메시지(m)의 적어도 일부를 삭제 또는 변경하여 수정된 메시지(m_{MOD})를 생성하는 단계;

상기 메시지(m)에 대응되는 암호문(c)을 생성하는 단계;

상기 생성된 암호문(c)의 유효성을 검증하기 위한 증명값(π)을 생성하는 단계; 및

생성된 상기 증명값(π)을 포함하는 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 생성하는 단계를 포함하고,

상기 증명값(π)을 생성하는 단계는, 상기 수정된 메시지(m_{MOD}) 및 기 설정된 함수 F에 대한 평가키(EK_F)에 기초하여 상기 증명값(π)을 생성하는, 방법.

청구항 11

청구항 10에 있어서,

상기 서명값(σ_{SIG})을 생성하는 단계는,

상기 메시지(m)에 대응되는 랜덤값(r)을 생성하는 단계;

상기 메시지 및 상기 랜덤값을 결합한 값에 대한 해시값(h)을 생성하는 단계; 및

상기 해시값(h)을 상기 비밀키로 암호화하여 상기 서명값(σ_{SIG})를 계산하는 단계를 더 포함하는, 방법.

청구항 12

청구항 11에 있어서,

상기 암호문(c)을 생성하는 단계는, 다음의 수학식

$c \leftarrow PKE.Enc(pk_{PKE}, m, s)$

(이때, PKE.Enc는 공개키 기반 암호화 알고리즘, pk_{PKE}는 인증 기관에서 발급한 공개키, s는 상기 공개키 기반 암호화 알고리즘에서 사용되는 랜덤값)

에 의하여 계산되는, 방법.

청구항 13

청구항 12에 있어서,

상기 증명값(π)은, 다음의 수학식

$\pi \leftarrow VC.Compute(EK_F, x, w)$

(이때 x = (h, MOD, m_{MOD}, c), w = (m, r, s), y = 1, MOD는 메시지(m)에 대한 수정 명령, m_{MOD}는 수정된 메시지, 상기 함수 F는 F(x, w) = y의 관계를 만족하는 함수로서, F(x, w) = (H(m||r) ?= h) AND (PKE.Enc(pk_{PKE}, m, s) ?= c) AND (Redact(m, MOD) ?= m_{MOD})로 정의됨)

에 의하여 계산되는, 방법.

청구항 14

청구항 13에 있어서,

상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})는,

상기 해시값(h), 상기 서명값(σ_{SIG}), 상기 암호문(c) 및 상기 증명값(π)을 포함하는, 방법.

청구항 15

청구항 14에 있어서,

상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 수신한 검증자는,

상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})에 포함된 상기 증명값(π)에 대한 제1 검증을 수행하고,

상기 제1 검증에 성공한 경우, 상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})에 포함된 상기 서명값(σ_{SIG})에 대한 제2 검증을 수행하는, 방법.

청구항 16

청구항 15에 있어서,

상기 제1 검증은, 다음의 수학식

$d \leftarrow VC.Verify(VK_F, x, \pi)$

(이때, d는 검증 결과, VK_F는 상기 함수 F에 대응되는 검증키)

에 의하여 수행되는, 방법.

청구항 17

청구항 15에 있어서,

상기 제2 검증은, 다음의 수학식

$d \leftarrow d * SIG.Verify(pk_{SIG}, h, \sigma_{SIG})$

(이때, pk_{SIG}는 상기 인증 요청자 단말의 비밀키(sk_{SIG})에 대응되는 공개키)

에 의하여 수행되는, 방법.

청구항 18

하나 이상의 프로세서들;

메모리; 및

하나 이상의 프로그램들을 포함하고,

상기 하나 이상의 프로그램들은 상기 메모리에 저장되고, 상기 하나 이상의 프로세서들에 의해 실행되도록 구성되며,

상기 하나 이상의 프로그램들은,

인증 요청자 단말의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성하는 단계;

상기 메시지(m)를 구성하는 복수 개의 블록 중 적어도 하나를 상기 메시지(m)로부터 삭제하는 단계;

상기 메시지(m)로부터 삭제된 블록(m_i)에 대한 암호문(c_i)을 생성하는 단계;

상기 생성된 암호문(c_i)의 유효성을 검증하기 위한 증명값(π)을 생성하는 단계; 및

생성된 상기 증명값(π)을 포함하는 검증 요청 메시지를 생성하는 단계를 포함하는 단계들을 수행하기 위한 명령을 포함하고,

상기 증명값(π)을 생성하는 단계는, 상기 삭제된 블록(m_i) 및 기 설정된 함수 F에 대한 평가키(EK_F)에 기초하여 상기 증명값(π)을 생성하는, 컴퓨팅 장치.

청구항 19

하나 이상의 프로세서들;

메모리; 및

하나 이상의 프로그램들을 포함하고,

상기 하나 이상의 프로그램들은 상기 메모리에 저장되고, 상기 하나 이상의 프로세서들에 의해 실행되도록 구성되며,

상기 하나 이상의 프로그램들은,

인증 요청자 단말의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성하는 단계;

상기 인증 요청자 단말에서, 상기 메시지(m)의 적어도 일부를 삭제 또는 변경하여 수정된 메시지(m_{MOD})를 생성하는 단계;

상기 인증 요청자 단말에서, 상기 메시지(m)에 대응되는 암호문(c)을 생성하는 단계;

상기 인증 요청자 단말에서, 상기 생성된 암호문(c)의 유효성을 검증하기 위한 증명값(π)을 생성하는 단계; 및

생성된 상기 증명값(π)을 포함하는 검증 요청 메시지를 생성하는 단계를 포함하는 단계들을 수행하기 위한 명령을 포함하고,

상기 증명값(π)을 생성하는 단계는, 상기 수정된 메시지(m_{MOD}) 및 기 설정된 함수 F에 대한 평가키(EK_F)에 기초하여 상기 증명값(π)을 생성하는, 컴퓨팅 장치.

발명의 설명

기 술 분 야

[0001] 본 발명의 실시예들은 리댁터블 서명(redactable signature) 기술과 관련된다.

배 경 기 술

[0003] 리댁터블 서명 기법(redactable signature scheme)이란 서명된 메시지를 복호화하지 않은 상태에서 원본 메시지의 일부를 삭제 또는 수정하기 위한 기법을 의미한다. 리댁터블 서명 기법을 이용할 경우 예컨대 서명값에서 프라이버시 보장이 필요한 부분 등을 용이하게 삭제하거나 수정할 수 있다.

[0004] 이와 같이 리댁터블 서명 기법은 서명된 메시지에서 프라이버시와 관계된 부분 등을 용이하게 삭제할 수 있다는 점에서 장점이 있으나, 한번 삭제된 부분에 대해서는 추후 복원이 불가능하다는 문제점이 존재한다. 이에 따라 리댁터블 서명 기법에서 필요한 경우 삭제된 부분을 복원하기 위한 방안이 필요하게 되었다.

선행기술문헌

특허문헌

[0006] (특허문헌 0001) 미국등록특허공보 US 9,767,469 B2 (2017.09.19)

발명의 내용

해결하려는 과제

[0007] 개시되는 실시예들은 리댁터블 서명 기법에서 삭제된 부분을 복원하기 위한 기술적인 수단을 제공하기 위한 것이다.

과제의 해결 수단

[0009] 예시적인 실시예에 따르면, 하나 이상의 프로세서들, 및 상기 하나 이상의 프로세서들에 의해 실행되는 하나 이상의 프로그램들을 저장하는 메모리를 구비한 인증 요청자 단말 장치에서 수행되는 방법으로서, 상기 인증 요청자 단말의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성하는 단계; 상기 메시지(m)를 구성하는 복수 개의 블록 중 적어도 하나를 상기 메시지(m)로부터 삭제하는 단계; 상기 메시지(m)로부터 삭제된 블록(m_i)에 대한 암호문(c_i)을 생성하는 단계; 상기 생성된 암호문(c_i)의 유효성을 검증하기 위한 증명값(π)을 생성하는 단계; 및 생성된 상기 증명값(π)을 포함하는 리댁트된 서명 검증 요청 메시지를 생성하는 단계를 포함하는, 방법이 제공된다.

[0010] 상기 서명값(σ_{SIG})을 생성하는 단계는, 상기 메시지(m)를 상기 복수 개의 블록으로 분할하는 단계; 분할된 상기 복수 개의 블록 각각에 대응되는 랜덤값(r)을 생성하고, 상기 복수 개의 블록 및 상기 랜덤값에 대한 해시값을 이용하여 머클 트리(merkle tree)를 구성하는 단계; 및 상기 머클 트리의 루트 해시를 상기 비밀키로 암호화하여 상기 서명값(σ_{SIG})를 계산하는 단계를 더 포함할 수 있다.

[0011] 상기 암호문(c_i)을 생성하는 단계는, 다음의 수학식

[0012] $c_i \leftarrow PKE.Enc(pk_{PKE}, m_i, s_i)$

[0013] (이때, PKE.Enc는 공개키 기반 암호화 알고리즘, pk_{PKE}는 인증 기관에서 발급한 공개키, s_i는 공개키 기반 암호화 알고리즘에서 사용되는 랜덤값)

[0014] 에 의하여 계산될 수 있다.

[0015] 상기 증명값(π)은, 다음의 수학식

[0016] $\pi \leftarrow VC.Compute(EK_F, x, w)$

[0017] (이때 $x = \perp$, $w = (m_i, r_i, s_i)$, $y = (c_i, h_i)$, r_i는 m_i에 대응되는 랜덤값, $h_i = H(m_i, r_i)$, H는 해시함수, F(x,

w) = (H(m_i, r_i), PKE.Enc(pk_{PKE}, m_i, s_i)), EK_F는 함수 F에 대한 평가키)

[0018] 에 의하여 계산될 수 있다.

[0019] 상기 검증 요청 메시지(σ_{MOD})는,

[0020] 상기 머클 트리의 루트 해시(h), 상기 서명값(σ_{SIG-}), 상기 상기 암호문(c_i) 및 상기 증명값(π)을 포함할 수 있다.

[0021] 상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 수신한 검증자는, 상기 검증 요청 메시지에 포함된 상기 증명값(π)에 대한 제1 검증을 수행하고, 상기 제1 검증에 성공한 경우, 상기 검증 요청 메시지에 포함된 상기 서명값(σ_{SIG})에 대한 제2 검증을 수행할 수 있다.

[0022] 상기 제1 검증은, 다음의 수학식

[0023] $d \leftarrow VC.Verify(VK_F, x, \pi)$

[0024] (이때, d는 검증 결과, VK_F는 함수 F에 대응되는 검증키)

[0025] 에 의하여 수행될 수 있다.

[0026] 상기 제2 검증은, 다음의 수학식

[0027] $d \leftarrow d * SIG.Verify(pk_{SIG}, h, \sigma_{SIG})$

[0028] (이때, pk_{sig}는 상기 인증 요청자 단말의 비밀키(sk_{SIG})에 대응되는 공개키)

[0029] 에 의하여 수행될 수 있다.

[0030] 상기 메시지(m)로부터 삭제된 블록이 둘 이상인 경우, 상기 인증 요청자 단말은, 삭제된 블록 각각에 대하여 상기 암호문을 생성하는 단계 및 상기 증명값을 생성하는 단계를 반복 수행하도록 구성될 수 있다.

[0031] 다른 예시적인 실시예에 따르면, 하나 이상의 프로세서들, 및 상기 하나 이상의 프로세서들에 의해 실행되는 하나 이상의 프로그램들을 저장하는 메모리를 구비한 인증 요청자 단말 장치에서 수행되는 방법으로서, 상기 인증 요청자 단말의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성하는 단계; 상기 메시지(m)의 적어도 일부를 삭제 또는 변경하는 단계; 상기 메시지(m)에 대응되는 암호문(c)을 생성하는 단계; 및 상기 생성된 암호문(c)의 유효성을 검증하기 위한 증명값(π)을 생성하는 단계; 및 생성된 상기 증명값(π)을 포함하는 리댁트된 서명 검증 요청 메시지를 생성하는 단계를 포함하는, 방법이 제공된다.

[0032] 상기 서명값(σ_{SIG})을 생성하는 단계는, 상기 메시지(m)에 대응되는 랜덤값(r)을 생성하는 단계; 상기 메시지 및 상기 랜덤값을 결합한 값에 대한 해시값(h)을 생성하는 단계; 및 상기 해시값(h)을 상기 비밀키로 암호화하여 상기 서명값(σ_{SIG})을 계산하는 단계를 더 포함할 수 있다.

[0033] 상기 암호문(c)을 생성하는 단계는, 다음의 수학식

[0034] $c \leftarrow PKE.Enc(pk_{PKE}, m, s)$

[0035] (이때, PKE.Enc는 공개키 기반 암호화 알고리즘, pk_{PKE}는 인증 기관에서 발급한 공개키, s는 상기 공개키 기반 암호화 알고리즘에서 사용되는 랜덤값)

[0036] 에 의하여 계산될 수 있다.

[0037] 상기 증명값(π)은, 다음의 수학식

[0038] $\pi \leftarrow VC.Compute(EK_F, x, w)$

[0039] (이때 x = (h, MOD, m_{MOD}, c), w = (m, r, s), y = 1, MOD는 메시지(m)에 대한 수정 명령, m_{MOD}는 수정된 메시지, F는 F(x, w) = y의 관계를 만족하는 함수로서, F(x, w) = (H(m||r) ?= h) AND (PKE.Enc(pk_{PKE}, m, s)

?= c) AND (Redact(m, MOD) ?= m_{MOD})로 정의됨)

[0040] 에 의하여 계산될 수 있다.

[0041] 상기 검증 요청 메시지(σ_{MOD})는, 상기 머클 트리의 루트 해시(h), 상기 서명값(σ_{SIG}), 상기 상기 암호문(c) 및 상기 증명값(π)을 포함할 수 있다.

[0042] 상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 수신한 검증자는, 상기 검증 요청 메시지에 포함된 상기 증명값(π)에 대한 제1 검증을 수행하고, 상기 제1 검증에 성공한 경우, 상기 검증 요청 메시지에 포함된 상기 서명값(σ_{SIG})에 대한 제2 검증을 수행할 수 있다.

[0043] 상기 제1 검증은, 다음의 수학식

[0044] $d \leftarrow VC.Verify(VK_F, x, \pi)$

[0045] (이때, d는 검증 결과, VK_F는 함수 F에 대응되는 검증키)

[0046] 에 의하여 수행될 수 있다.

[0047] 상기 제2 검증은, 다음의 수학식

[0048] $d \leftarrow d * SIG.Verify(pk_{SIG}, h, \sigma_{SIG})$

[0049] (이때, pk_{SIG}는 상기 인증 요청자 단말의 비밀키(sk_{SIG})에 대응되는 공개키)

[0050] 에 의하여 수행될 수 있다.

[0051] 다른 예시적인 실시예에 따르면, 하나 이상의 프로세서들; 메모리; 및 하나 이상의 프로그램들을 포함하고, 상기 하나 이상의 프로그램들은 상기 메모리에 저장되고, 상기 하나 이상의 프로세서들에 의해 실행되도록 구성되며, 상기 하나 이상의 프로그램들은, 인증 요청자 단말의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성하는 단계; 상기 메시지(m)를 구성하는 복수 개의 블록 중 적어도 하나를 상기 메시지(m)로부터 삭제하는 단계; 상기 메시지(m)로부터 삭제된 블록(m_i)에 대한 암호문(c_i)을 생성하는 단계; 상기 생성된 암호문(c_i)의 유효성을 검증하기 위한 증명값(π)을 생성하는 단계; 및 생성된 상기 증명값(π)을 포함하는 검증 요청 메시지를 생성하는 단계를 포함하는 단계들을 수행하기 위한 명령을 포함하는, 컴퓨팅 장치가 제공된다.

[0052] 다른 예시적인 실시예에 따르면, 하나 이상의 프로세서들; 메모리; 및 하나 이상의 프로그램들을 포함하고, 상기 하나 이상의 프로그램들은 상기 메모리에 저장되고, 상기 하나 이상의 프로세서들에 의해 실행되도록 구성되며, 상기 하나 이상의 프로그램들은, 인증 요청자 단말의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성하는 단계; 상기 인증 요청자 단말에서, 상기 메시지(m)의 적어도 일부를 삭제 또는 변경하는 단계; 상기 인증 요청자 단말에서, 상기 메시지(m)에 대응되는 암호문(c)을 생성하는 단계; 상기 인증 요청자 단말에서, 상기 생성된 암호문(c)의 유효성을 검증하기 위한 증명값(π)을 생성하는 단계; 및 생성된 상기 증명값(π)을 포함하는 검증 요청 메시지를 생성하는 단계를 포함하는 단계들을 수행하기 위한 명령을 포함하는, 컴퓨팅 장치가 제공된다.

발명의 효과

[0054] 개시되는 실시예들에 따르면, 리댁터블 서명 기법에서 삭제된 부분을 추후 필요시 용이하게 복원할 수 있어 리댁터블 서명 기법의 편의성을 높일 수 있다.

도면의 간단한 설명

[0056] 도 1은 일 실시예에 따른 리댁터블 서명 시스템을 설명하기 위한 블록도

도 2는 본 발명의 제1 실시예에 따른 리댁터블 서명 방법을 설명하기 위한 블록도

도 3은 본 발명의 일 실시예에 따른 머클 트리 기반의 서명 생성 기법을 설명하기 위한 예시도

도 4는 본 발명의 일 실시예에 따른 머클 트리 기반의 서명 생성 기법에서 일부 블록이 삭제된 상태를 설명하기

위한 예시도

도 5는 본 발명의 제2 실시예에 따른 리댁터블 서명 방법을 설명하기 위한 블록도

도 6은 본 발명의 제2 실시예에 따른 리댁터블 서명 방법에서 암호문의 유효성을 증명하기 위한 함수(F)를 설명하기 위한 예시도

도 7은 예시적인 실시예들에서 사용되기에 적합한 컴퓨팅 장치를 포함하는 컴퓨팅 환경을 예시하여 설명하기 위한 블록도

발명을 실시하기 위한 구체적인 내용

[0057] 이하, 도면을 참조하여 본 발명의 구체적인 실시형태를 설명하기로 한다. 이하의 상세한 설명은 본 명세서에서 기술된 방법, 장치 및/또는 시스템에 대한 포괄적인 이해를 돕기 위해 제공된다. 그러나 이는 예시에 불과하며 본 발명은 이에 제한되지 않는다.

[0058] 본 발명의 실시예들을 설명함에 있어서, 본 발명과 관련된 공지기술에 대한 구체적인 설명이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명을 생략하기로 한다. 그리고, 후술되는 용어들은 본 발명에서의 기능을 고려하여 정의된 용어들로서 이는 사용자, 운용자의 의도 또는 관례 등에 따라 달라질 수 있다. 그러므로 그 정의는 본 명세서 전반에 걸친 내용을 토대로 내려져야 할 것이다. 상세한 설명에서 사용되는 용어는 단지 본 발명의 실시예들을 기술하기 위한 것이며, 결코 제한적이어서는 안 된다. 명확하게 달리 사용되지 않는 한, 단수 형태의 표현은 복수 형태의 의미를 포함한다. 본 설명에서, "포함" 또는 "구비"와 같은 표현은 어떤 특성들, 숫자들, 단계들, 동작들, 요소들, 이들의 일부 또는 조합을 가리키기 위한 것이며, 기술된 것 이외에 하나 또는 그 이상의 다른 특성, 숫자, 단계, 동작, 요소, 이들의 일부 또는 조합의 존재 또는 가능성을 배제하도록 해석되어서는 안 된다.

[0060] 본 발명의 실시예들을 구체적으로 설명하기에 앞서, 리댁터블 서명 기법(redactable signature scheme)을 설명하면 다음과 같다. 리댁터블 서명 기법은 KeyGen, Sign, Redact 및 Verify를 포함하는 네 개의 알고리즘으로 구성된다.

[0061] KeyGen 알고리즘은 시큐리티 파라미터 λ를 받아서 비밀키(sk)와 공개키(pk)를 생성하기 위한 알고리즘이다. 이 중 비밀키(sk)는 비밀로 보관하고, 공개키(pk)는 공개한다. 이를 수식으로 나타내면 다음과 같다.

[0063] $KeyGen(1^{\lambda}) \rightarrow (sk, pk)$

[0065] Sign 알고리즘은 비밀키(sk)를 이용하여 메시지(m)에 대한 서명값(σ_m)을 생성한다. 이를 수식으로 나타내면 다음과 같다.

[0067] $Sign(sk, m) \rightarrow \sigma_m$

[0069] Redact 알고리즘은 공개키(pk) 및 메시지(m)에 대한 수정 명령(MOD; Redaction Instruction)을 이용하여 수정된 메시지(MOD(m)) 및 이에 대응되는 서명값($\sigma_{MOD(m)}$)을 생성한다. 이를 수식으로 나타내면 다음과 같다.

[0071] $Redact(pk, m, \sigma_m, MOD) \rightarrow (MOD(m), \sigma_{MOD(m)})$

[0073] Verify 알고리즘은 Sign 알고리즘으로 생성된 서명값 또는 Redact 알고리즘으로 변경된 서명값에 대한 검증을 수행한다. 예를 들어 Verify 알고리즘은 서명이 유효한 경우 1, 그렇지 않은 경우 0을 검증 결과로서 반환할 수 있다. 이를 수식으로 나타내면 다음과 같다.

[0075] $Verify(pk, m, \sigma_m) \rightarrow 0 \text{ or } 1$

[0076] $Verify(pk, MOD(m), \sigma_{MOD(m)}) \rightarrow 0 \text{ or } 1$

[0078] 도 1은 일 실시예에 따른 리댁터블 서명 시스템(100)을 설명하기 위한 블록도이다. 도시된 바와 같이, 일 실시예에 따른 아이디 기반 서명 시스템(100)은 인증 요청자(102), 검증자(104) 및 인증 기관(Certified Authority)(106)을 포함한다.

[0079] 인증 요청자(102)는 자신의 비밀키를 이용하여 메시지를 서명하고 상기 서명값을 이용하여 인증을 요청하는 단말이다. 메시지를 서명한 이후, 인증 요청자(102)는 상기 메시지의 적어도 일부를 삭제/변경(redact)하여 수정

된 메시지(MOD(m)) 및 이에 대응되는 서명값($\sigma_{MOD(m)}$)을 생성한다. 한편, 이 과정에서 인증 요청자(102)는 상기 삭제 또는 변경 전의 원본 메시지를 복원하기 위한 암호문(c) 및 상기 암호문을 검증하기 위한 증명값(π)을 생성하여 상기 서명값과 함께 검증자(104)에게 송신한다.

[0080] 검증자(104)는 인증 요청자(102)로부터 상기 서명값, 상기 암호문 및 상기 증명값을 포함하는 검증 요청 메시지를 수신하고, 상기 증명값 및 상기 서명값에 대한 검증을 수행하기 위한 단말이다.

[0081] 인증 기관(Certified Authority, 106)은 상기 암호문으로부터 상기 원본 메시지를 복원하기 위한 단말이다. 일 실시예에서, 인증 요청자(102)는 인증 기관(106)이 발급한 공개키를 이용한 공개키 기반 암호화 기법을 통하여 상기 암호문을 생성할 수 있다. 그러면 인증 기관(106)은 상기 인증 기관(106)의 공개키에 대응되는 비밀키를 이용하여 상기 암호문으로부터 상기 원본 메시지를 복원할 수 있다.

[0083] 도 2는 본 발명의 제1 실시예에 따른 리댁터블 서명 방법(200)을 설명하기 위한 블록도이다. 도시된 흐름도에서는 상기 방법을 복수 개의 단계로 나누어 기재하였으나, 적어도 일부의 단계들은 순서를 바꾸어 수행되거나, 다른 단계와 결합되어 함께 수행되거나, 생략되거나, 세부 단계들로 나뉘어 수행되거나, 또는 도시되지 않은 하나 이상의 단계가 부가되어 수행될 수 있다.

[0084] 본 발명의 제1 실시예에 따른 리댁터블 서명 방법(200)은 머클 트리 기반의 리댁터블 서명 기법(Merkle-Tree Redactable Signature Scheme)을 사용한다. 머클 트리 기반의 리댁터블 서명 기법은 (SIG.KeyGen, SIG.Sign, SIG.Redact, SIG.Verify)를 포함하는 네 가지 알고리즘으로 구성된다.

[0085] SIG.KeyGen 알고리즘은 시큐리티 파라미터 λ를 받아서 비밀키(sk_{SIG})와 공개키(pk_{SIG})를 생성하기 위한 알고리즘이다. 이를 수식으로 나타내면 다음과 같다.

[0087] $SIG.KeyGen(1^{\lambda}) \rightarrow (sk_{SIG}, pk_{SIG})$

[0089] SIG.Sign 알고리즘은 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_m)을 생성하기 위한 알고리즘이다. 이를 수식으로 나타내면 다음과 같다.

[0091] $SIG.Sign(sk, m) \rightarrow \sigma_m$

[0093] 본 실시예에서, SIG.Sign 알고리즘은 머클 트리 기반으로 상기 서명값을 생성하도록 구성된다. 구체적인 머클 트리 기반의 서명값 생성 방법에 대해서는 후술하기로 한다.

[0094] Redact 알고리즘은 공개키(pk_{SIG}) 및 메시지(m)에 대한 수정 명령(MOD; Redaction Instruction)을 이용하여 수정된 메시지(MOD(m)) 및 이에 대응되는 서명값($\sigma_{MOD(m)}$)을 생성한다. 이를 수식으로 나타내면 다음과 같다.

[0096] $SIG.Redact(pk_{SIG}, m, \sigma_m, MOD) \rightarrow (MOD(m), \sigma_{MOD(m)})$

[0098] SIG.Verify 알고리즘은 SIG.Sign 알고리즘으로 생성된 서명값 또는 SIG.Redact 알고리즘으로 변경된 서명값에 대한 검증을 수행한다. 예를 들어 Verify 알고리즘은 서명이 유효한 경우 1, 그렇지 않은 경우 0을 검증 결과로서 반환할 수 있다. 이를 수식으로 나타내면 다음과 같다.

[0100] $SIG.Verify(pk_{SIG}, m, \sigma_m) \rightarrow 0 \text{ or } 1$

[0101] $SIG.Verify(pk_{SIG}, MOD(m), \sigma_{MOD(m)}) \rightarrow 0 \text{ or } 1$

[0103] 또한, 본 발명의 제1 실시예에 따른 리댁터블 서명 방법(200)은 SNARK(Succint Non-interactive Argument of Knowledge) 알고리즘 및 공개키 기반의 암호화 기법을 사용한다. 먼저 SNARK알고리즘을 설명하면 다음과 같다. SNARK 알고리즘은 영지식 증명(Zero-Knowledge Proof)의 하나로, 영지식 증명은 암호학에서 누군가가 상대방에게 어떤 사항이 참이라는 것을 증명할 때, 그 문장의 참 거짓 여부를 제외한 어떤 것도 노출하지 않는 알고리즘을 의미한다.

[0104] SNARK 알고리즘은 VC.KeyGen, VC.Compute 및 VC.Verify를 포함하는 세 개의 알고리즘으로 구성된다.

[0105] VC.KeyGen 알고리즘은 함수 F 및 시큐리티 파라미터 λ를 받아서 평가키(evaluation key, EK_F)와 검증키(verification key, VK_F)를 생성하기 위한 알고리즘이다. 이때 EK_F 및 VK_F는 함수 F에는 의존적이나

(dependent), 함수 F의 입력에는 독립적이다(independent). 이를 수식으로 나타내면 다음과 같다.

[0107] $VC.KeyGen(F, 1^{\lambda}) \rightarrow (EK_F, VK_F)$

[0109] VC.Compute 알고리즘은 어떤 진술(statement)이 참(true)이라는 사실을 증명하기 위한 증명자(prover)가 사용하는 알고리즘이다. 증명자는 함수 F 및 입력값 x, w에 대하여 F(x, w) = y를 계산하고 VC.Compute 함수 및 EK_F를 이용하여 상기 연산 결과가 참이라는 것을 증명하기 위한 증명값(Proof of Correctness)인 π를 생성한다. 이를 수식으로 나타내면 다음과 같다.

[0111] $VC.Compute(EK_F, x, w) \rightarrow (y, \pi)$

[0113] VC.Verify 알고리즘은 검증자(verifier)가 사용하는 알고리즘이다. 검증자는 함수 F에 대한 입력값 x, 출력값 y 및 VK_F 및 π를 알고 있는 경우, VC.Verify 알고리즘을 이용하여 F에 대한 정보가 없어도 F(x, w) = y를 만족하는 w가 존재한다는 진술이 참이라는 것을 알 수 있다. 이를 수식으로 나타내면 다음과 같다.

[0115] $VC.Verify(VK_F, x, y, \pi) \rightarrow 0 \text{ or } 1$

[0117] 마지막으로 공개키 기반의 암호화 기법(Public Key Encryption)은 (PKE.KeyGen, PKE.Enc, PKE.Dec)의 세 가지 알고리즘으로 구성된다.

[0118] PKE.KeyGen 알고리즘은 시큐리티 파라미터 λ를 받아서 비밀키(sk_{PKE})와 공개키(pk_{PKE})를 생성하기 위한 알고리즘이다. 이를 수식으로 나타내면 다음과 같다.

[0120] $PKE.KeyGen(1^{\lambda}) \rightarrow (sk_{PKE}, pk_{PKE})$

[0122] PKE.Enc 알고리즘은 공개키(pk_{PKE})를 이용하여 메시지(m)로부터 암호문(c)을 생성하기 위한 알고리즘이다. 이를 수식으로 나타내면 다음과 같다. 이때 s는 랜덤값이다.

[0124] $PKE.Enc(pk_{PKE}, m, s) \rightarrow c$

[0126] PKE.Dec 알고리즘은 비밀키(sk_{PKE})를 이용하여 암호문(c)으로부터 메시지(m)를 복원하기 위한 알고리즘이다. 이를 수식으로 나타내면 다음과 같다.

[0128] $PKE.Dec(sk_{PKE}, c, s) \rightarrow m$

[0130] 이하, 도 2에 도시된 본 발명의 제1 실시예에 따른 리댁터블 서명 방법(200)을 설명하면 다음과 같다.

[0131] 단계 202에서, 인증 요청자(102) 및 인증 기관(106)은 리댁터블 서명을 위한 키를 생성한다.

[0132] 먼저, 인증 요청자(102)는 시큐리티 파라미터 λ를 받아서 SIG.KeyGen, VC.KeyGen 알고리즘을 이용하여 다음과 같이 키를 생성한다.

[0134] $SIG.KeyGen(1^{\lambda}) \rightarrow (sk_{SIG}, pk_{SIG})$

[0135] $VC.KeyGen(F, 1^{\lambda}) \rightarrow (EK_F, VK_F)$

[0137] 이때, 상기 함수 F는 인증 요청자(102)가 생성한 암호문을 검증하기 위한 증명값을 생성하는 데 사용되는 함수로서, 그 구체적인 형태에 대해서는 이하에서 상세히 설명한다.

[0138] 또한, 인증 기관(106)은 시큐리티 파라미터 λ를 받아서 PKE.KeyGen 알고리즘을 이용하여 다음과 같이 키를 생성한다.

[0140] $PKE.KeyGen(1^{\lambda}) \rightarrow (sk_{PKE}, pk_{PKE})$

[0142] 본 단계에서 생성되는 비밀키(SK) 및 공개키(PK) 쌍은 다음과 같다.

[0144] $SK = (sk_{SIG}, sk_{PKE})$

[0145] $PK = (pk_{SIG}, pk_{PKE}, EK_F, PK_F)$

[0147] 이중, sk_{SIG}는 메시지(m)를 서명하는 데 사용되고, pk_{SIG}는 서명된 메시지를 검증하는데 사용된다. 또한 pk_{PKE}는 삭제 또는 수정된 메시지를 복원하기 위한 암호문을 생성하는 데 사용되고, sk_{PKE}는 상기 암호문으로부터 원본 메시지를 복원하는 데 사용된다.

[0148] 단계 204에서, 인증 요청자(102)는 자신의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성한다. 본 실시예에서, 인증 요청자(102)는 머클 트리 기반 서명 기법(merkle tree based signature scheme)을 이용하여 서명값을 생성하도록 구성된다.

[0149] 도 3은 본 발명의 일 실시예에 따른 머클 트리 기반의 서명 생성 기법을 설명하기 위한 예시도이다. 머클 트리 기반의 서명 생성 기법에서, 원본 메시지(m)는 n(n은 자연수) 개의 블록으로 분할된다. 도시된 실시예에서는 메시지(m)를 4개의 블록(m_{00}, m_{01}, m_{10}, m_{11})으로 분할한 예를 나타내었다.

[0150] 이후, 인증 요청자(102)는 분할된 각각의 블록에 대응되는 랜덤값 $r=\{r_{00}, r_{01}, r_{10}, r_{11}\}$을 생성하고, 분할된 각각의 블록에 대하여 다음과 같이 해시값을 계산하여 머클 트리의 리프 노드(leaf node)를 생성한다.

[0152] $h_{00} = H(m_{00}, r_{00})$

[0153] $h_{01} = H(m_{01}, r_{01})$

[0154] $h_{10} = H(m_{10}, r_{10})$

[0155] $h_{11} = H(m_{11}, r_{11})$

[0157] 이후, 인증 요청자(102)는 순차적으로 상기 머클 트리의 상위 노드 해시를 연산하여 최종적으로 머클 트리의 루트 해시(root hash)를 계산한다. 도 3에 도시된 실시예에서 루트 해시(h)는 다음과 같이 계산된다.

[0159] $h_0 = H(h_{00}, h_{01})$

[0160] $h_1 = H(h_{10}, h_{11})$

[0161] $h = H(h0_1, h_1)$

[0163] 상기 루트 해시로부터 메시지(m)에 대한 서명값(σ_{SIG})은 SIG.Sign 알고리즘 및 sk_{SIG}을 이용하여 다음과 같이 계산된다.

[0165] $\sigma_{SIG} \leftarrow SIG.Sign(sk_{SIG}, h)$

[0167] 또한, 머클 트리 기반의 서명 생성 기법에서 인증 요청자(102)가 검증자(104)에게 송신하는 검증 요청 메시지(σ)는 다음과 같이 구성된다.

[0169] $\sigma = (r_{00}||r_{01}||r_{10}||r_{11}, m, \sigma_{SIG})$

[0171] 다시 도 2로 돌아가서, 단계 206에서 인증 요청자(102)는 원본 메시지(m)를 구성하는 복수 개의 블록 중 적어도 하나를 원본 메시지(m)에서 삭제한다. 머클 트리 기반 서명 기법에서, 메시지의 수정(redaction)은 원본 메시지를 구성하는 블록 중 하나 이상을 삭제하는 방식으로 수행된다. 도 4는 원본 메시지(m)로부터 m_{00} 블록이 삭제된 예를 나타낸 것이다.

[0172] 원본 메시지의 일부가 삭제된 경우, 상기 검증 요청 메시지(σ)에서 삭제된 블록에 해당하는 랜덤값은 머클 트리의 해시값으로 대체된다. 예를 들어, 도 4에 도시된 실시예에서 m_{00} 블록이 원본 메시지에서 삭제된 경우, r_{00}은 h_{00}으로 대체된다. 즉, 머클 트리 기반의 서명 기법의 경우 메시지의 일부가 삭제되더라도, 해당 블록에 대응되는 해시값을 이용하여 루트 해시를 동일하게 생성할 수 있게 된다.

[0173] 단계 208에서, 인증 요청자(102)는 원본 메시지(m)로부터 삭제된 블록(m_i)에 대한 암호문(c_i)을 생성한다. 이때 상기 암호문(c_i)은 이후 인증 기관(106) 등에서 삭제된 블록을 복원하는 데 사용된다.

[0174] 예를 들어 원본 메시지(m)에서 i번째 블록(m_i)이 삭제된 경우, m_i 블록에 대한 호문(c_i)은 공개키 기반의 암호화 기법 및 인증 기관(106)의 공개키(pk_{PKE})를 이용하여 다음과 같이 생성될 수 있다.

[0176] $c_i \leftarrow PKE.Enc(pk_{PKE}, m_i, s_i)$

[0178] 이때, s_{00}은 공개키 기반의 암호화 기법에서 사용되는 랜덤값이다.

[0179] 단계 210에서, 인증 요청자(102)는 208 단계에서 생성된 암호문(c_i)의 유효성을 검증하기 위한 증명값(π)을 생성한다. 본 실시예에서 상기 증명값(π)은 VC.Compute 알고리즘을 이용하여 생성된다. 이를 수식으로 나타내면 다음과 같다.

[0181] $\pi \leftarrow VC.Compute(EK_F, x, w)$

[0183] 이때, $x = \perp(null)$, $w = (m_i, r_i, s_i)$, $y = (c_i, h_i)$, m_i는 삭제된 블록, r_i는 m_i에 대응되는 랜덤값, $h_i = H(m_i, r_i)$, H는 해시함수, c_i는 m_i로부터 생성된 암호문, s_i는 상기 암호문 생성에 사용되는 랜덤값, EK_F는 함수 F에 대한 평가키이다.

[0184] 또한 F는 $F(x, w) = y$의 관계를 만족하는 함수로서, $F(x, w) = (H(m_i, r_i), PKE.Enc(pk_{PKE}, m_i, s_i))$으로 정의된다. 즉 상기 증명값($\pi$)은 $h_i = H(m_i, r_i)$ 및 $c_i = PKE.Enc(pk_{PKE}, m_i, s_i)$를 만족하는 $w = (m_i, r_i, s_i)$가 존재함을 증명하기 위한 값이다.

[0185] 이후 단계 212에서, 인증 요청자(102)는 생성된 상기 증명값(π)을 포함하는 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 생성한다. 리댁트된 서명 검증 요청 메시지(σ_{MOD})는 다음과 같이 구성된다.

[0187] $\sigma_{MOD} = (h, \sigma_{SIG}, c_i, \pi)$

[0189] 만약 상기 과정 이후 또 다른 블록(m_j)에 대한 삭제(redact)가 추가적으로 수행된 경우, 인증 요청자(102)는 상기 추가 삭제 블록(m_j)에 대하여 전술한 208 단계 내지 212 단계를 반복하게 된다. 즉, 이 경우 상기 리댁트된 서명 검증 요청 메시지(σ_{MOD}')는 다음과 같이 구성된다.

[0191] $\sigma_{MOD}' = (h, \sigma_{SIG}, c_i, \pi, c_j, \pi')$

[0193] 이때, c_j는 mj에 대응되는 암호문, π'는 c_j의 유효성을 검증하기 위한 증명값이다. 즉, 인증 요청자(102)는, 삭제된 블록의 개수 만큼 상기 208 단계 내지 212 단계를 반복 수행하게 된다.

[0194] 단계 214에서, 상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 수신한 검증자(104)는 이에 대한 검증을 수행한다.

[0195] 먼저, 검증자(104)는 VC.Verify 알고리즘을 이용하여 리댁션 과정에서 생성된 증명값(π)에 대한 검증(제1 검증)을 수행한다. 이를 수식으로 나타내면 다음과 같다.

[0197] $d \leftarrow VC.Verify(VK_F, x, \pi)$

[0199] 이때 VC.Verify 알고리즘은 상기 검증에 성공한 경우 1을, 실패한 경우 0을 반환한다. 만약 상기 리댁트된 서명이 복수 개의 암호문 및 증명값을 포함하는 경우, 즉 복수 회의 리댁션이 이루어진 경우, 상기 과정은 상기 리댁션 횟수만큼 수행된다.

[0200] 이후, 검증자(104) SIG.Verify 알고리즘을 이용하여 σ_{SIG}에 대한 검증(제2 검증)을 수행한다. 이를 수식으로 나타내면 다음과 같다.

[0202] $d \leftarrow d * SIG.Verify(pk_{SIG}, h, \sigma_{SIG})$

[0204] SIG.Verify 알고리즘 또한 검증에 성공한 경우 1을, 그렇지 않은 경우 0을 반환한다. 즉, 상기 d 값은 상기 제1 검증 및 제2 검증에 모두 성공한 경우에만 1이 되며(검증 성공), 제1 검증 및 제2 검증 중 하나라도 실패한 경우에는 0을 반환하게 된다(검증 실패).

[0205] 한편, 단계 206에서 삭제된 블록에 대한 복원이 필요한 경우, 인증 기관(106)은 PKE.Dec 알고리즘 및 자신의 비밀키(sk_{PKE})를 이용하여 암호문(c_i)으로부터 삭제된 블록(m_i)을 복원할 수 있다. 이를 수식으로 나타내면 다음과 같다.

[0207] $PKE.Dec(sk_{PKE}, c_i, s) \rightarrow m_i$

[0209] 도 5는 본 발명의 제2 실시예에 따른 리댁터블 서명 방법(500)을 설명하기 위한 블록도이다. 도시된 흐름도에서는 상기 방법을 복수 개의 단계로 나누어 기재하였으나, 적어도 일부의 단계들은 순서를 바꾸어 수행되거나, 다른 단계와 결합되어 함께 수행되거나, 생략되거나, 세부 단계들로 나뉘어 수행되거나, 또는 도시되지 않은 하나 이상의 단계가 부가되어 수행될 수 있다.

[0210] 본 발명의 제2 실시예에 따른 리댁터블 서명 방법(500)은 머클 트리 기반의 리댁터블 서명 기법(Merkle-Tree Redactable Signature Scheme)이 아닌 일반적인 리댁터블 서명 기법(Generic redactable signature)을 사용한다는 점에서 제1 실시예와 상이하다. 제1 실시예에서와 같이 머클 트리 기반의 리댁터블 서명 기법을 사용할 경우 여러 번의 리댁션을 수행할 수 있으나, 이 경우 삭제되는 블록 별로 SNARK에 의한 검증이 수행되어야 하는 번거로움이 존재한다. 반면, 제2 실시예의 경우 SNARK에 의한 검증이 한 번만 수행된다는 점이 장점이나, 알고리즘의 특성 상 단 한 번의 리댁션만이 가능하게 된다. 본 실시예에서의 리댁터블 서명 기법 또한 (SIG.KeyGen, SIG.Sign, SIG.Redact, SIG.Verify)를 포함하는 네 가지 알고리즘으로 구성된다.

[0211] SIG.KeyGen 알고리즘은 시큐리티 파라미터 λ를 받아서 비밀키(sk_{SIG})와 공개키(pk_{SIG})를 생성하기 위한 알고리즘이다. 이를 수식으로 나타내면 다음과 같다.

[0213] $SIG.KeyGen(1^{\lambda}) \rightarrow (sk_{SIG}, pk_{SIG})$

[0215] SIG.Sign 알고리즘은 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_m)을 생성하기 위한 알고리즘이다. 이를 수식으로 나타내면 다음과 같다.

[0217] $SIG.Sign(sk, m) \rightarrow \sigma_m$

[0219] Redact 알고리즘은 공개키(pk_{SIG}) 및 메시지(m)에 대한 수정 명령(MOD; Redaction Instruction)을 이용하여 수정된 메시지(MOD(m)) 및 이에 대응되는 서명값($\sigma_{MOD(m)}$)을 생성한다. 이를 수식으로 나타내면 다음과 같다.

[0221] $SIG.Redact(pk_{SIG}, m, \sigma_m, MOD) \rightarrow (MOD(m), \sigma_{MOD(m)})$

[0223] SIG.Verify 알고리즘은 SIG.Sign 알고리즘으로 생성된 서명값 또는 SIG.Redact 알고리즘으로 변경된 서명값에 대한 검증을 수행한다. 예를 들어 Verify 알고리즘은 서명이 유효한 경우 1, 그렇지 않은 경우 0을 검증 결과로서 반환할 수 있다. 이를 수식으로 나타내면 다음과 같다.

[0225] $SIG.Verify(pk_{SIG}, m, \sigma_m) \rightarrow 0 \text{ or } 1$

[0226] $SIG.Verify(pk_{SIG}, MOD(m), \sigma_{MOD(m)}) \rightarrow 0 \text{ or } 1$

[0228] 한편, 본 발명의 제2 실시예에서 사용되는 SNARK 기법 및 공개키 기반 암호화 기법은 제1 실시예에서와 동일한 바, 여기서는 반복되는 설명을 생략한다.

[0229] 이하, 도 5에 도시된 본 발명의 제1 실시예에 따른 리댁터블 서명 방법(500)을 설명하면 다음과 같다.

[0230] 단계 502에서, 인증 요청자(102) 및 인증 기관(106)은 리댁터블 서명을 위한 키를 생성한다.

[0231] 먼저, 인증 요청자(102)는 시큐리티 파라미터 λ를 받아서 SIG.KeyGen, VC.KeyGen 알고리즘을 이용하여 다음과 같이 키를 생성한다.

[0233] $SIG.KeyGen(1^{\lambda}) \rightarrow (sk_{SIG}, pk_{SIG})$

[0234] $VC.KeyGen(F, 1^{\lambda}) \rightarrow (EK_F, VK_F)$

[0236] 또한, 인증 기관(106)은 시큐리티 파라미터 λ를 받아서 PKE.KeyGen 알고리즘을 이용하여 다음과 같이 키를 생

성한다.

[0238] $PKE.KeyGen(1^{\lambda}) \rightarrow (sk_{PKE}, pk_{PKE})$

[0240] 본 단계에서 생성되는 비밀키(SK) 및 공개키(PK)는 다음과 같다.

[0242] $SK = (sk_{SIG}, sk_{PKE})$

[0243] $PK = (pk_{SIG}, pk_{PKE}, EK_F, PK_F)$

[0245] 이중, sk_{SIG}는 메시지(m)를 서명하는 데 사용되고, pk_{SIG}는 서명된 메시지를 검증하는데 사용된다. 또한 pk_{PKE}는 삭제 또는 수정된 메시지를 복원하기 위한 암호문을 생성하는 데 사용되고, sk_{PKE}는 상기 암호문으로부터 원본 메시지를 복원하는 데 사용된다.

[0246] 단계 504에서, 인증 요청자(102)는 자신의 비밀키(sk_{SIG})를 이용하여 메시지(m)에 대한 서명값(σ_{SIG})을 생성한다. 구체적으로, 인증 요청자(102)는 원본 메시지(m)에 대응되는 랜덤값(r)을 생성하고 해시 함수(H)를 이용하여 다음과 같이 해시값(h)을 생성한다.

[0248] $h = H(m||r)$

[0250] 원본 메시지(m)에 대한 서명값(σ_{SIG})은 상기 해시값에 SIG.Sign 알고리즘 및 sk_{SIG}을 이용하여 다음과 같이 계산된다.

[0252] $\sigma_{SIG} \leftarrow SIG.Sign(sk_{SIG}, h)$

[0254] 이후 검증자에게 전달되는 검증 요청 메시지(σ)는 다음과 같이 구성된다.

[0255] $\sigma = (h, r, \sigma_{SIG})$

[0257] 단계 506에서, 인증 요청자(102)는 원본 메시지(m)의 적어도 일부를 삭제 또는 변경한다. 이때 상기 원본 메시지(m)의 삭제 또는 변경은 기 설정된 수정 명령(MOD; Redaction Instruction)에 따라 수행된다.

[0258] 단계 508에서, 인증 요청자(102)는 원본 메시지(m)에 대응되는 암호문(c)을 생성한다. 구체적으로 인증 요청자(102)는 PKE.Enc 알고리즘 및 공개키(pk_{PKE})를 이용하여 다음과 같이 메시지(m)에 대한 암호문(c)을 생성하게 된다.

[0260] $c \leftarrow PKE.Enc(pk_{PKE}, m, s)$

[0262] 이때, s는 공개키 기반 암호화 알고리즘에서 사용되는 랜덤값이다.

[0263] 단계 510에서, 인증 요청자(102)는 생성된 암호문(c)의 유효성을 검증하기 위한 증명값(π)을 생성한다. 상기 증명값(π)은 VC.Compute 알고리즘을 이용하여 생성된다. 이를 수식으로 나타내면 다음과 같다.

[0265] $\pi \leftarrow VC.Compute(EK_F, x, w)$

[0267] 이때 $x = (h, MOD, m_{MOD}, c)$, $w = (m, r, s)$, $y = 1$, MOD는 메시지(m)에 대한 수정 명령, m_{MOD}는 506 단계에서 수정된 메시지이며, F는 $F(x, w) = y$의 관계를 만족하는 함수로서, $F(x, w) = (H(m||r) \stackrel{?}{=} h)$ AND $(PKE.Enc(pk_{PKE}, m, s) \stackrel{?}{=} c)$ AND $(Redact(m, MOD) \stackrel{?}{=} m_{MOD})$으로 정의된다.

[0268] 즉 상기 증명값(π)은 $H(m||r) = h$, $PKE.Enc(pk_{PKE}, m, s) = c$, 및 $(Redact(m, MOD) \stackrel{?}{=} m_{MOD}$ 를 모두 만족하는 (즉, 각각의 수식이 모두 참(1)인) $w = (m, r, s)$가 존재함을 증명하기 위한 값이다. 이를 그림으로 나타내면 도 6과 같다.

[0269] 이후 단계 512에서, 인증 요청자(102)는 생성된 상기 증명값(π)을 포함하는 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 생성한다. 리댁트된 서명 검증 요청 메시지(σ_{MOD})는 다음과 같이 구성된다.

[0271] $\sigma_{MOD} = (h, \sigma_{SIG}, c, \pi)$

[0273] 단계 514에서, 상기 리댁트된 서명 검증 요청 메시지(σ_{MOD})를 수신한 검증자(104)는 이에 대한 검증을 수행한다.

[0274] 먼저, 검증자(104)는 VC.Verify 알고리즘을 이용하여 리댁션 과정에서 생성된 증명값(π)에 대한 검증(제1 검증)을 수행한다. 이를 수식으로 나타내면 다음과 같다.

[0276] $d \leftarrow VC.Verify(VK_F, x, \pi)$

[0278] 이때 VC.Verify 알고리즘은 상기 검증에 성공한 경우 1을, 실패한 경우 0을 반환한다. 만약 상기 리댁트된 서명이 복수 개의 암호문 및 증명값을 포함하는 경우, 즉 복수 회의 리댁션이 이루어진 경우, 상기 과정은 상기 리댁션 횟수만큼 수행된다.

[0279] 이후, 검증자(104) SIG.Verify 알고리즘을 이용하여 σ_{SIG}에 대한 검증(제2 검증)을 수행한다. 이를 수식으로 나타내면 다음과 같다.

[0281] $d \leftarrow d * SIG.Verify(pk_{SIG}, h, \sigma_{SIG})$

[0283] SIG.Verify 알고리즘 또한 검증에 성공한 경우 1을, 그렇지 않은 경우 0을 반환한다. 즉, 상기 d 값은 상기 제1 검증 및 제2 검증에 모두 성공한 경우에만 1이 되며(검증 성공), 제1 검증 및 제2 검증 중 하나라도 실패한 경우에는 0을 반환하게 된다(검증 실패).

[0284] 한편, 단계 506에서 삭제 또는 변경된 메시지에 대한 복원이 필요한 경우, 인증 기관(106)은 PKE.Dec 알고리즘 및 자신의 비밀키(sk_{PKE})를 이용하여 암호문(c)으로부터 원본 메시지(m)을 복원할 수 있다. 이를 수식으로 나타내면 다음과 같다.

[0286] $PKE.Dec(sk_{PKE}, c, s) \rightarrow m$

[0288] 도 7은 예시적인 실시예들에서 사용되기에 적합한 컴퓨팅 장치를 포함하는 컴퓨팅 환경(10)을 예시하여 설명하기 위한 블록도이다. 도시된 실시예에서, 각 컴포넌트들은 이하에 기술된 것 이외에 상이한 기능 및 능력을 가질 수 있고, 이하에 기술되지 것 이외에도 추가적인 컴포넌트를 포함할 수 있다.

[0289] 도시된 컴퓨팅 환경(10)은 컴퓨팅 장치(12)를 포함한다. 일 실시예에서, 컴퓨팅 장치(12)는 본 발명의 실시예들에 따른 인증 요청자(102), 검증자(10 4) 및 인증 기관(106)일 수 있다. 컴퓨팅 장치(12)는 적어도 하나의 프로세서(14), 컴퓨터 판독 가능 저장 매체(16) 및 통신 버스(18)를 포함한다. 프로세서(14)는 컴퓨팅 장치(12)로 하여금 앞서 언급된 예시적인 실시예에 따라 동작하도록 할 수 있다. 예컨대, 프로세서(14)는 컴퓨터 판독 가능 저장 매체(16)에 저장된 하나 이상의 프로그램들을 실행할 수 있다. 상기 하나 이상의 프로그램들은 하나 이상의 컴퓨터 실행 가능 명령어를 포함할 수 있으며, 상기 컴퓨터 실행 가능 명령어는 프로세서(14)에 의해 실행되는 경우 컴퓨팅 장치(12)로 하여금 예시적인 실시예에 따른 동작들을 수행하도록 구성될 수 있다.

[0290] 컴퓨터 판독 가능 저장 매체(16)는 컴퓨터 실행 가능 명령어 내지 프로그램 코드, 프로그램 데이터 및/또는 다른 적합한 형태의 정보를 저장하도록 구성된다. 컴퓨터 판독 가능 저장 매체(16)에 저장된 프로그램(20)은 프로세서(14)에 의해 실행 가능한 명령어의 집합을 포함한다. 일 실시예에서, 컴퓨터 판독 가능 저장 매체(16)는 메모리(랜덤 액세스 메모리와 같은 휘발성 메모리, 비휘발성 메모리, 또는 이들의 적절한 조합), 하나 이상의 자기 디스크 저장 디바이스들, 광학 디스크 저장 디바이스들, 플래시 메모리 디바이스들, 그 밖에 컴퓨팅 장치(12)에 의해 액세스되고 원하는 정보를 저장할 수 있는 다른 형태의 저장 매체, 또는 이들의 적합한 조합일 수 있다.

[0291] 통신 버스(18)는 프로세서(14), 컴퓨터 판독 가능 저장 매체(16)를 포함하여 컴퓨팅 장치(12)의 다른 다양한 컴포넌트들을 상호 연결한다.

[0292] 컴퓨팅 장치(12)는 또한 하나 이상의 입출력 장치(24)를 위한 인터페이스를 제공하는 하나 이상의 입출력 인터페이스(22) 및 하나 이상의 네트워크 통신 인터페이스(26)를 포함할 수 있다. 입출력 인터페이스(22) 및 네트워크 통신 인터페이스(26)는 통신 버스(18)에 연결된다. 입출력 장치(24)는 입출력 인터페이스(22)를 통해 컴퓨팅 장치(12)의 다른 컴포넌트들에 연결될 수 있다. 예시적인 입출력 장치(24)는 포인팅 장치(마우스 또는 트랙패드 등), 키보드, 터치 입력 장치(터치패드 또는 터치스크린 등), 음성 또는 소리 입력 장치, 다양한 종류의 센서 장치 및/또는 촬영 장치와 같은 입력 장치, 및/또는 디스플레이 장치, 프린터, 스피커 및/또는 네트워크 카드와 같은 출력 장치를 포함할 수 있다. 예시적인 입출력 장치(24)는 컴퓨팅 장치(12)를 구성하는 일 컴포넌트로서

컴퓨팅 장치(12)의 내부에 포함될 수도 있고, 컴퓨팅 장치(12)와는 구별되는 별개의 장치로 컴퓨팅 장치(102)와 연결될 수도 있다.

[0294] 한편, 본 발명의 실시예는 본 명세서에서 기술한 방법들을 컴퓨터상에서 수행하기 위한 프로그램, 및 상기 프로그램을 포함하는 컴퓨터 판독 가능 기록매체를 포함할 수 있다. 상기 컴퓨터 판독 가능 기록매체는 프로그램 명령, 로컬 데이터 파일, 로컬 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체는 본 발명을 위하여 특별히 설계되고 구성된 것들이거나, 또는 컴퓨터 소프트웨어 분야에서 통상적으로 사용 가능한 것일 수 있다. 컴퓨터 판독 가능 기록매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체, CD-ROM, DVD와 같은 광 기록 매체, 및 롬, 램, 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 상기 프로그램의 예는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함할 수 있다.

[0295] 이상에서 본 발명의 대표적인 실시예들을 상세하게 설명하였으나, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자는 상술한 실시예에 대하여 본 발명의 범주에서 벗어나지 않는 한도 내에서 다양한 변형이 가능함을 이해할 것이다. 그러므로 본 발명의 권리범위는 설명된 실시예에 국한되어 정해져서는 안 되며, 후술하는 특허청구범위뿐만 아니라 이 특허청구범위와 균등한 것들에 의해 정해져야 한다.

부호의 설명

[0297] 100: 리댁터블 서명 시스템

102: 인증 요청자

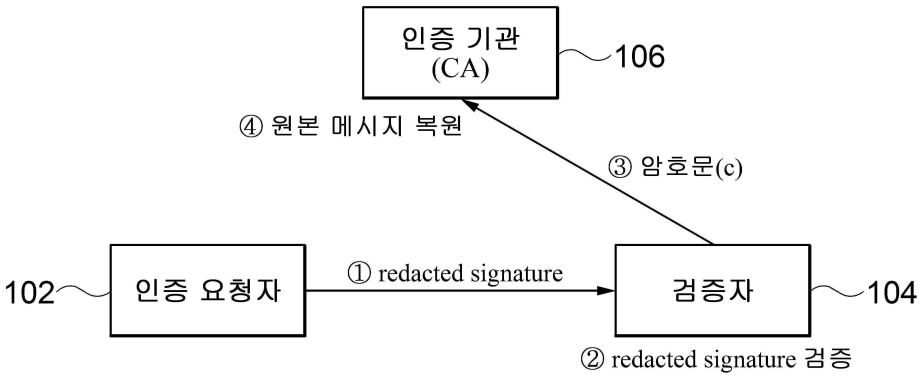
104: 검증자

106: 인증 기관

도면

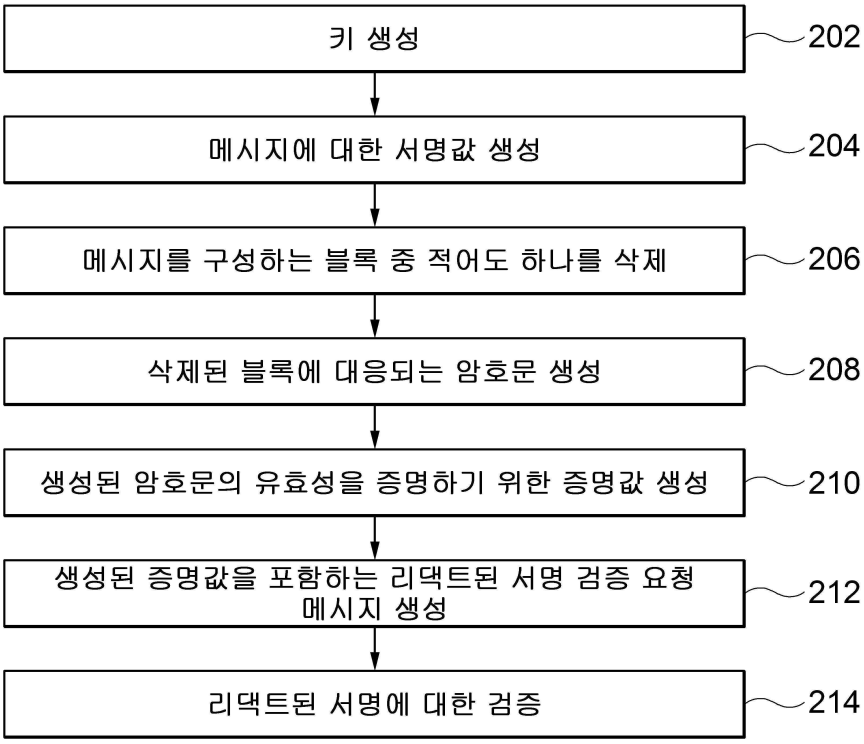
도면1

100

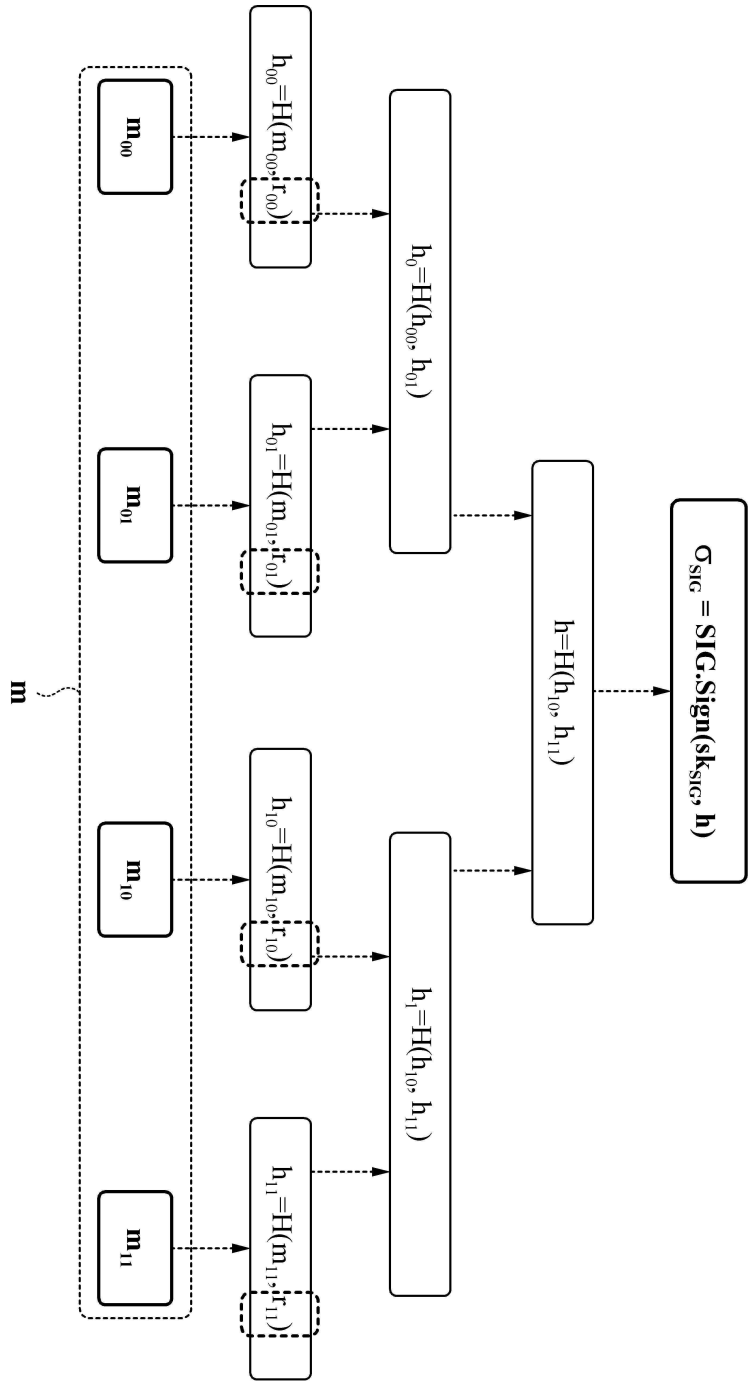


도면2

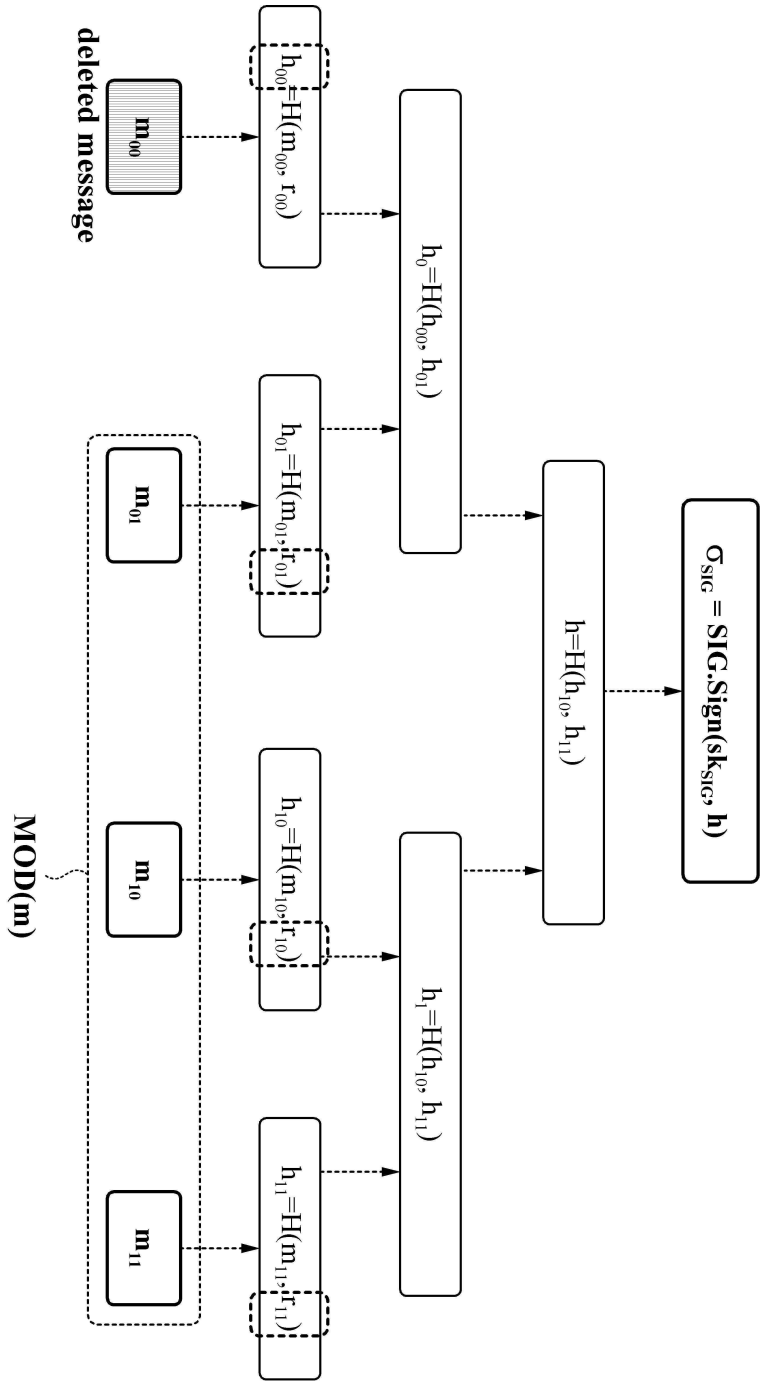
200


키 생성
202
메시지에 대한 서명값 생성
204
메시지를 구성하는 블록 중 적어도 하나를 삭제
206
삭제된 블록에 대응되는 암호문 생성
208
생성된 암호문의 유효성을 증명하기 위한 증명값 생성
210
생성된 증명값을 포함하는 리댁트된 서명 검증 요청 메시지 생성
212
리댁트된 서명에 대한 검증
214

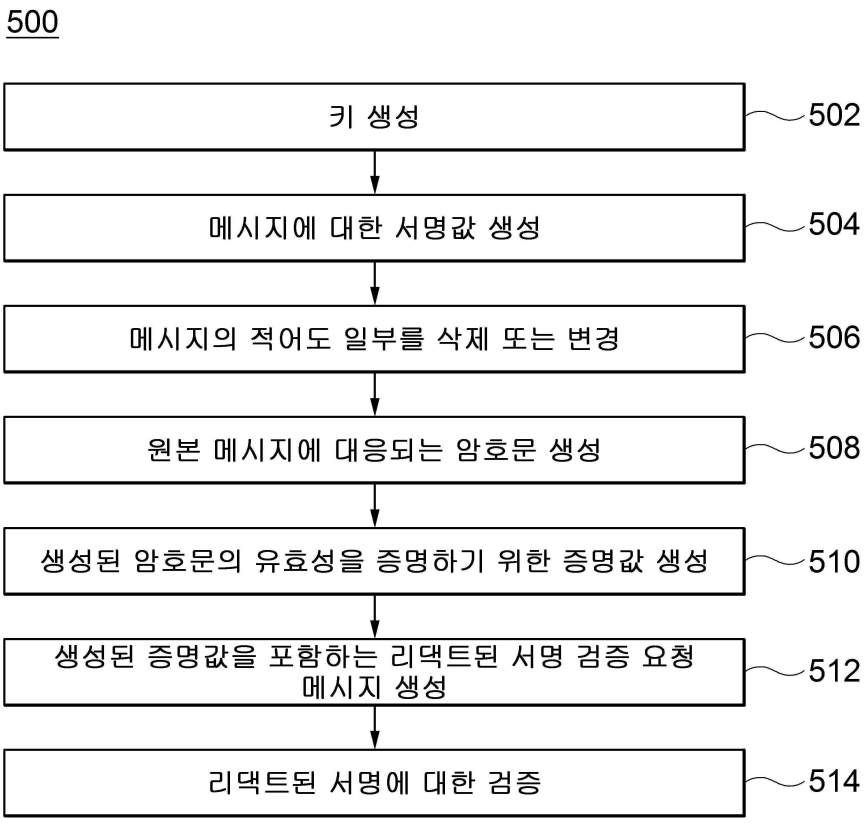
도면3


m
m00
m01
m10
m11
h00=H(m00,r00)
h01=H(m01,r01)
h10=H(m10,r10)
h11=H(m11,r11)
h0=H(h00, h01)
h1=H(h10, h11)
h=H(h0, h1)
σSIG = SIG.Sign(skSIG, h)

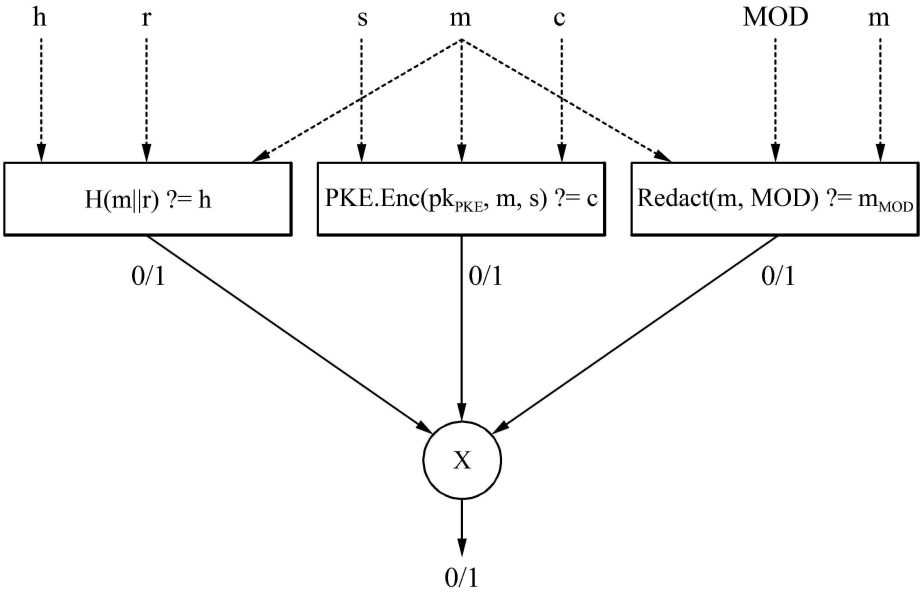
도면4


MOD(m)
m_{00}
deleted message
m_{01}
m_{10}
m_{11}
h_{00}=H(m_{00}, r_{00})
h_{01}=H(m_{01}, r_{01})
h_{10}=H(m_{10}, r_{10})
h_{11}=H(m_{11}, r_{11})
h_0=H(h_{00}, h_{01})
h_1=H(h_{10}, h_{11})
h=H(h_0, h_1)
σ_{SIG} = SIG.Sign(sk_{SIG}, h)

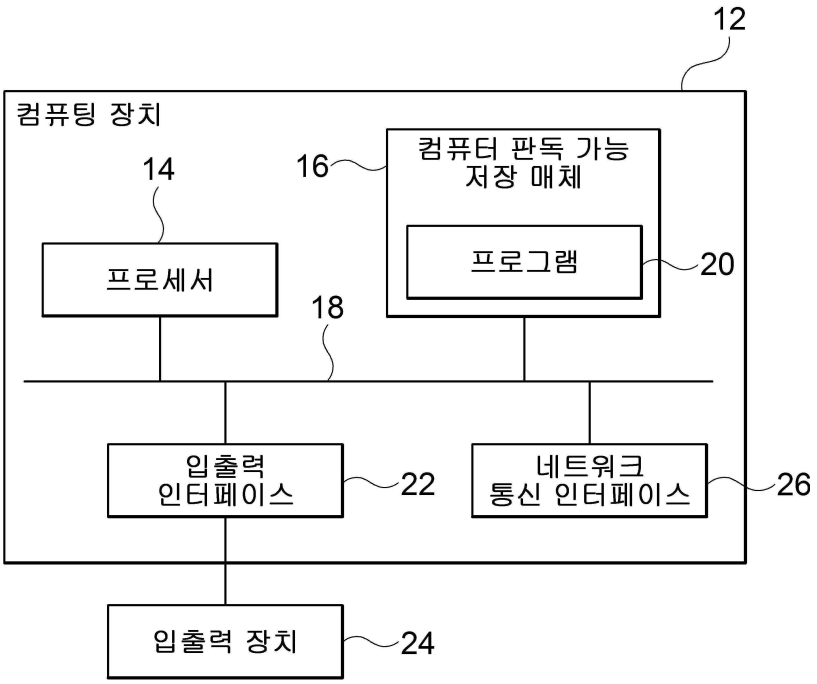
도면5


500
키 생성
502
메시지에 대한 서명값 생성
504
메시지의 적어도 일부를 삭제 또는 변경
506
원본 메시지에 대응되는 암호문 생성
508
생성된 암호문의 유효성을 증명하기 위한 증명값 생성
510
생성된 증명값을 포함하는 리댁트된 서명 검증 요청 메시지 생성
512
리댁트된 서명에 대한 검증
514

도면6


h
r
s
m
c
MOD
m
H(m||r) ?= h
PKE.Enc(pk_PKE, m, s) ?= c
Redact(m, MOD) ?= m_MOD
0/1
0/1
0/1
X
0/1

도면7


10
12
컴퓨팅 장치
14
프로세서
16
컴퓨터 판독 가능
저장 매체
프로그램
20
18
입출력
인터페이스
22
네트워크
통신 인터페이스
26
입출력 장치
24