



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2018-0003106
(43) 공개일자 2018년01월09일

(51) 국제특허분류(Int. Cl.)
H04L 9/08 (2006.01) H04L 29/06 (2006.01)
(52) CPC특허분류
H04L 9/0891 (2013.01)
H04L 63/10 (2013.01)
(21) 출원번호 10-2016-0082511
(22) 출원일자 2016년06월30일
심사청구일자 2016년06월30일

(71) 출원인
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)
(72) 발명자
이영경
서울특별시 동대문구 제기로11길 11 (제기동) 102호
이광수
서울특별시 성북구 길음로 119, 221동 902호 (길음동, 길음뉴타운 대우푸르지오아파트)
(74) 대리인
김등용, 김홍석

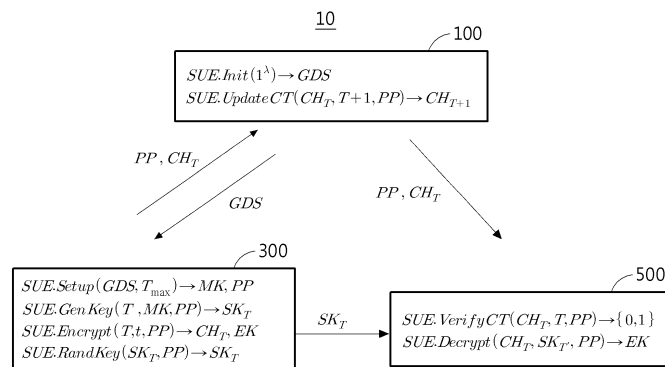
전체 청구항 수 : 총 8 항

(54) 발명의 명칭 암호문 셀프-업데이트가 가능한 암호화 방법

(57) 요약

암호문 셀프-업데이트가 가능한 암호화 방법이 개시된다. 상기 암호화 방법은, 서버가 보안 상수(1^λ)를 입력받 (뒷면에 계속)

대표도 - 도2



아 곱선행 함수(p, G, G_T, e)와 생성원(g)을 생성하고, 상기 곱선행 함수(p, G, G_T, e)와 상기 생성원(g)을 송신 단말로 송신하는 초기화 단계, 상기 송신 단말이 상기 곱선행 함수(p, G, G_T, e), 상기 생성원(g) 및 시간 정보($T_{\max}$)로부터 마스터키(MK)와 공개 파라미터(PP)를 생성하는 설정 단계, 상기 송신 단말이 시간(T), 상기 마스터키(MK) 및 상기 공개 파라미터(PP)로부터 상기 시간(T)에 대한 개인키(SK_T)를 생성하는 키생성 단계, 상기 송신 단말이 상기 시간(T), 임의의 난수($t \in Z_p$), 상기 공개 파라미터(PP)로부터 암호문 헤더(CH_T)와 세션키(EK)를 생성하는 암호화 단계, 상기 서버가 상기 송신 단말로부터 수신된 상기 암호문 헤더(CH_T), 시간($T+1$) 및 상기 공개 파라미터(PP)를 이용하여 업데이트된 암호문 헤더(CH_{T+1})를 생성하는 업데이트 단계, 수신 단말이 상기 서버로부터 수신된 상기 암호문 헤더(CH_T), 상기 시간(T) 및 상기 공개 파라미터(PP)를 이용하여 상기 암호문 헤더(CH_T)의 유효성을 검증하는 검증 단계, 및 상기 수신 단말이 상기 암호문 헤더(CH_T), 상기 송신 단말로부터 수신된 상기 개인키(SK_T) 및 상기 공개 파라미터(PP)를 이용하여 상기 세션키(EK)를 생성하는 복호화 단계를 포함한다.

(52) CPC특허분류

H04L 9/0869 (2013.01)

이 발명을 지원한 국가연구개발사업

과제고유번호	IITP-2016-R0992-16-1006
부처명	미래창조과학부
연구관리전문기관	정보통신기술진흥센터
연구사업명	대학ICT연구센터육성지원사업
연구과제명	IoT보안기술연구
기 여 율	1/1
주관기관	순천향대학교 산학협력단
연구기간	2016.01.01 ~ 2016.12.31

명세서

청구범위

청구항 1

서버가 보안 상수(1^λ)를 입력받아 곱선행 함수(p, G, G_T, e)와 생성원(g)을 생성하고, 상기 곱선행 함수(p, G, G_T, e)와 상기 생성원(g)을 송신 단말로 송신하는 초기화 단계;

상기 송신 단말이 상기 곱선행 함수(p, G, G_T, e), 상기 생성원(g) 및 시간 정보($T_{\max}$)로부터 마스터키(MK)와 공개 파라미터(PP)를 생성하는 설정 단계;

상기 송신 단말이 시간(T), 상기 마스터키(MK) 및 상기 공개 파라미터(PP)로부터 상기 시간(T)에 대한 개인키(SK_T)를 생성하는 키생성 단계;

상기 송신 단말이 상기 시간(T), 임의의 난수($t \in \mathbb{Z}_p$), 상기 공개 파라미터(PP)로부터 암호문 헤더(CH_T)와 세션키(EK)를 생성하는 암호화 단계;

상기 서버가 상기 송신 단말로부터 수신된 상기 암호문 헤더(CH_T), 시간($T+1$) 및 상기 공개 파라미터(PP)를 이용하여 업데이트된 암호문 헤더(CH_{T+1})를 생성하는 업데이트 단계;

수신 단말이 상기 서버로부터 수신된 상기 암호문 헤더(CH_T), 상기 시간(T) 및 상기 공개 파라미터(PP)를 이용하여 상기 암호문 헤더(CH_T)의 유효성을 검증하는 검증 단계; 및

상기 수신 단말이 상기 암호문 헤더(CH_T), 상기 송신 단말로부터 수신된 상기 개인키(SK_T) 및 상기 공개 파라미터(PP)를 이용하여 상기 세션키(EK)를 생성하는 복호화 단계를 포함하는,
암호문 셀프-업데이트가 가능한 암호화 방법.

청구항 2

제1항에 있어서, 상기 키생성 단계는,

상기 개인키(SK_T)와 상기 공개 파라미터(PP)를 이용하여 상기 개인키(SK_T)를 재 난수화하는 난수화 단계를 더 포함하는,

암호문 셀프-업데이트가 가능한 암호화 방법.

청구항 3

송신 단말에서 수행되는, 암호문 셀프-업데이트가 가능한 암호화 방법에 있어서,

상기 송신 단말이 서버로부터 수신된 곱선행 함수(p, G, G_T, e), 생성원(g) 및 시간 정보($T_{\max}$)로부터 마스터키(MK)와 공개 파라미터(PP)를 생성하는 설정 단계;

상기 송신 단말이 시간(T), 상기 마스터키(MK) 및 상기 공개 파라미터(PP)로부터 상기 시간(T)에 대한 개인키(SK_T)를 생성하는 키생성 단계; 및

상기 송신 단말이 상기 시간(T), 임의의 난수($t \in Z_p$), 상기 공개 파라미터(PP)로부터 암호문 헤더(CH_T)와 세션키(EK)를 생성하는 암호화 단계를 포함하는,

암호문 셀프-업데이트가 가능한 암호화 방법.

청구항 4

제3항에 있어서,

상기 암호문 헤더(CH_T)는 상기 서버를 경유하여 수신 단말로 송신되고,

상기 개인키(SK_T)는 상기 송신 단말에 의해 상기 수신 단말로 송신되는,

암호문 셀프-업데이트가 가능한 암호화 방법.

청구항 5

제4항에 있어서, 상기 키생성 단계는,

상기 개인키(SK_T)와 상기 공개 파라미터(PP)를 이용하여 상기 개인키(SK_T)를 재 난수화하는 난수화 단계를 더 포함하는,

암호문 셀프-업데이트가 가능한 암호화 방법.

청구항 6

제5항에 있어서,

상기 설정 단계는,

임의의 제1 난수들($u_s, h_s \in G$)을 선택하는 단계;

상기 곱셈형 함수(p, G, G_T, e), 상기 생성원(g) 및 포화 이진트리(full binary tree)의 높이(depth, l)를 이용하여 제1 마스터키(MK_{CDE})와 제1 공개 파라미터(PP_{CDE})를 생성하는 제1 설정 단계; 및

선택된 인덱스(z)를 이용하여 해쉬함수($H^z \in \mathcal{H}$)를 선택하는 단계를 포함하고,

상기 마스터키(MK)는 상기 제1 마스터키(MK_{CDE})이고,

상기 공개 파라미터(PP)는 상기 제1 공개 파라미터(PP_{CDE}), 상기 인덱스(z) 및 상기 제1 난수들($u_s, h_s \in G$)을 포함하는,

암호문 셀프-업데이트가 가능한 암호화 방법.

청구항 7

제6항에 있어서,

상기 시간 정보($T_{\max}$)는 수학식에 의해 정의되고,

상기 수학식은 $T_{\max} = 2^{l+1} - 1$ 인,

암호문 셀프-업데이트가 가능한 암호화 방법.

청구항 8

제7항에 있어서,

상기 키생성 단계는,

상기 포화 이진트리에서 상기 시간(T) 정보가 포함된 노드의 라벨스트링(L)을 획득하는 단계;

상기 라벨스트링(L), 상기 마스터키(MK), 상기 공개 파라미터(PP)를 이용하여 상기 개인키(SK_T)를 생성하는 단계를 포함하는,

암호문 셀프-업데이트가 가능한 암호화 방법.

발명의 설명

기술 분야

[0001] 본 발명의 개념에 따른 실시 예는 암호문 셀프-업데이트가 가능한 암호화 방법에 관한 것으로, 특히 클라우드 스토리지와 같은 환경에서 암호문에 대한 권한이 없는 관리자에 의해 시간에 따른 암호문 업데이트가 가능하도록 하여 사용자에게 프라이버시 보장과 키 위임을 통한 접근제어를 제공할 수 있는 암호문 셀프-업데이트가 가능한 암호화 방법에 관한 것이다.

배경 기술

[0003] 본 발명의 배경이 되는 기술에 대한 간략한 설명은 다음과 같다.

[0004] 선택 암호문 공격에 대한 안전성(Chosen-Ciphertext Attack Security)

[0005] '선택 암호문 공격에 대한 안전성'이란 챌린지 암호문을 제외하고 선택한 암호문들의 평문을 얻어볼 수 있는 환경의 공격자로부터 챌린지 암호문의 안전성이 보장됨을 의미한다. 셀프-업데이트가 가능한 공개키 암호에서는 선택 암호문 공격자가 챌린지 암호문에 대한 평문을 얻어볼 수 없으며, 추가로 챌린지 암호문으로부터 업데이트된 암호문들의 평문을 역시 얻어볼 수 없도록 제한된다. 여러 공격 모델 중 가장 강력한 공격 모델인 선택 암호문 공격에 안전하기 위해서는 'Non-Malleability'가 보장되어야 한다. 'Non-Malleability'란 공격자에게 암호문이 주어졌을 때 암호문에 대한 평문과 관련 있는 새로운 평문에 대한 암호문을 생성할 수 없어야 한다는 개념이다. 복호화 과정에서 검증 과정을 추가하여 'Non-Malleability'를 보장할 수 있다.

[0007] 셀프-업데이트가 가능한 공개키 암호화 (Self-Updatable Encryption)

[0008] '셀프-업데이트가 가능한 공개키 암호'는 Kwangsu Lee 등의 "Self-updatable encryption: Time constrained access control with hidden attributes and better efficiency(In Kazue Sako and Palash Sarkar, editors, ASIACRYPT 2013, volume 8269 of Lecture Notes in Computer Science, pages 235254. Springer, 2013.)"에서 처음으로 제안되었다. 본 기술에서는 키와 암호문에 시간정보가 포함되어 있으며 시간정보가 유효한 키만 암호문을 복호화할 수 있다. 마스터키를 소유한 사용자는 기한이 정해진 키를 생성해서 다른 사용자에게 위임을 할

수 있다. 시간에 따라 암호문을 업데이트 하는 방식으로, 기한이 정해진 키는 정해진 기간 이후로 업데이트된 암호문을 복호화할 수 없다. 마스터키가 필요 없이 암호문 업데이트가 가능하기 때문에 스토리지 관리자와 같은 제3자에게 주기적인 암호문 업데이트를 위임하더라도 사용자의 프라이버시를 보장할 수 있다. Kwangsu Lee 등이 제안한 기법은 선택 암호문 공격보다 약한 안전성을 갖는 선택 평문 공격에 안전한 기법으로 특정한 환경에서는 안전성을 보장하지 못한다.

[0010] 포화 이진트리와 라벨스트링(Full Binary Tree and Label string)

[0011] 포화 이진트리는 말단노드를 제외한 모든 노드의 자식 수가 2개인 트리를 말한다. 말단노드의 개수를 N 이라 할 때, 모든 노드의 개수는 $2N - 1$ 이 된다. 포화 이진트리(BT)에서 모든 $i (1 \leq i \leq 2N-1)$ 에 대해, $v_i \in BT$ 로 모든 노드에 인덱스(index)를 부여할 수 있다. 다음으로, 각각의 노드 v_i 에 대해서 고유한 라벨스트링 $L \in \{0,1\}^*$ 을 부여한다. 라벨스트링을 부여할 때는 다음과 같은 법칙을 따른다. 왼쪽 자식노드의 라벨스트링은 부모노드의 라벨스트링 왼쪽(또는 오른쪽)에 1을 덧붙여 생성하고, 오른쪽 자식노드의 라벨스트링은 부모노드의 라벨스트링 왼쪽(또는 오른쪽)에 0을 덧붙여 생성한다. 루트노드의 라벨스트링은 엠티(empty) 스트링으로 정의한다.

[0013] 타임 구조(Time Structure)

[0014] 'Time Structure'는 시간에 따른 암호문 업데이트를 수행하기 위해 시간 정보와 라벨스트링을 이진트리에 담은 형태, 즉 이진트리 형태의 데이터 구조이다. 도 1은 시간 정보(T)와 라벨스트링(L)을 포함하는 포화 이진트리, 즉 시간 구조를 설명하기 위한 개념도를 도시한다. 노드 출력 알고리즘($TimeNodes$)은 시간 정보를 포함하는 이진트리에서 하나의 노드를 입력으로 받아 조건을 만족하는 노드들을 출력하는 알고리즘이다. v 를 이진트리의 노드라 할 때 $Parent(v)$ 는 v 노드의 부모노드를 의미한다. $Path(v)$ 는 루트 노드로부터 v 노드까지 오는 경로에 포함되는 모든 노드들의 집합을 의미한다. $RightSibling(v) = RightChild(Parent(v))$ 는 v 노드의 부모 노드에서 오른쪽 자식노드를 의미한다. 이때 $TimeNodes(v) = \{v\} \cup RightSibling(Path(v)) \setminus Path(Parent(v))$ 로 정의한다.

[0015] L 을 각 노드의 고유한 라벨스트링이라 할 때, $Parent(L)$, $Path(L)$, $RightSibling(L)$, 그리고 $TimeNodes(L)$ 를 위의 정의와 비슷하게 정의할 수 있다. 이때, 라벨스트링을 입력으로 할 경우 출력 또한 이에 대응하는 라벨스트링이 될 수 있다.

[0016] 이진트리에 시간 정보를 담는 방식은 다음과 같다. 이진트리(총 노드 수: $2N-1$)의 모든 노드에 고유한 시간 정보 $T \in \{1, \dots, T_{\max}\}$ 를 전위순회(pre-order traversal) 순으로 부여한다. 이진트리의 높이(depth)를 d 라 할 때 루트 노드에는 1, 가장 왼쪽 말단 노드에는 $d+1$, 그리고 가장 오른쪽 말단 노드에는 $2^{d+1}-1 (= T_{\max})$ 값이 배정된다. 시간 정보 T 를 배정받은 노드의 라벨스트링을 찾기 위한 다음과 같은 함수 $\psi(T) \rightarrow L$ 를 정의한다. 함수 ψ 는 시간 정보 $T \in \{1, \dots, T_{\max}\}$ 를 입력받아 해당 시간 정보가 배정된 노드의 고유한 라벨스트링을 출력한다.

특허문헌

- [0018] (특허문헌 0001) Kwangsu Lee, Seung Geol Choi, Dong Hoon Lee, Jong Hwan Park, and Moti Yung, "Self-updatable encryption: Time constrained access control with hidden attributes and better efficiency", In Kazue Sako and Palash Sarkar, editors, ASIACRYPT 2013, volume 8269 of Lecture Notes in Computer Science, pages 235254. Springer, 2013.

발명의 내용

해결하려는 과제

- [0019] 본 발명이 이루고자 하는 기술적인 과제는 클라우드 스토리지와 같은 환경에서 암호문에 대한 권한이 없는 관리자에 의한 시간에 따른 암호문 업데이트가 가능하도록 하여 사용자에게 프라이버시 보장과 키 위임을 통한 접근 제어를 제공할 수 있는 암호문 셀프-업데이트가 가능한 암호화 방법을 제공하는 것이다.

과제의 해결 수단

- [0021] 본 발명의 실시 예에 따른, 암호문 셀프-업데이트가 가능한 암호화 방법은, 서버가 보안 상수(1^λ)를 입력받아 곱셈형 함수(p, G, G_T, e)와 생성원(g)을 생성하고, 상기 곱셈형 함수(p, G, G_T, e)와 상기 생성원(g)을 송신 단말로 송신하는 초기화 단계, 상기 송신 단말이 상기 곱셈형 함수(p, G, G_T, e), 상기 생성원(g) 및 시간 정보($T_{\max}$)로부터 마스터키(MK)와 공개 파라미터(PP)를 생성하는 설정 단계, 상기 송신 단말이 시간(T), 상기 마스터키(MK) 및 상기 공개 파라미터(PP)로부터 상기 시간(T)에 대한 개인키(SK_T)를 생성하는 키생성 단계, 상기 송신 단말이 상기 시간(T), 임의의 난수($t \in \mathbb{Z}_p$), 상기 공개 파라미터(PP)로부터 암호문 헤더(CH_T)와 세션키(EK)를 생성하는 암호화 단계, 상기 서버가 상기 송신 단말로부터 수신된 상기 암호문 헤더(CH_T), 시간($T+1$) 및 상기 공개 파라미터(PP)를 이용하여 업데이트된 암호문 헤더(CH_{T+1})를 생성하는 업데이트 단계, 수신 단말이 상기 서버로부터 수신된 상기 암호문 헤더(CH_T), 상기 시간(T) 및 상기 공개 파라미터(PP)를 이용하여 상기 암호문 헤더(CH_T)의 유효성을 검증하는 검증 단계, 및 상기 수신 단말이 상기 암호문 헤더(CH_T), 상기 송신 단말로부터 수신된 상기 개인키(SK_T) 및 상기 공개 파라미터(PP)를 이용하여 상기 세션키(EK)를 생성하는 복호화 단계를 포함한다.

발명의 효과

- [0023] 본 발명의 실시 예에 따른 암호문 셀프-업데이트가 가능한 암호화 방법에 의할 경우, 암호문에 대한 권한이 없는 관리자에 의한 시간에 따른 암호문 업데이트가 가능해 사용자에게 프라이버시 보장과 키 위임을 통한 접근 제어를 제공할 수 있는 효과가 있다.

도면의 간단한 설명

- [0025] 본 발명의 상세한 설명에서 인용되는 도면을 보다 충분히 이해하기 위하여 각 도면의 상세한 설명이 제공된다. 도 1은 시간 정보와 라벨스트링을 포함하는 포화 이진트리, 즉 시간 구조를 설명하기 위한 개념도이다.

도 2는 본 발명의 일 실시 예에 따른 시스템을 도시한다.

발명을 실시하기 위한 구체적인 내용

- [0026] 본 명세서에 개시되어 있는 본 발명의 개념에 따른 실시 예들에 대해서 특정한 구조적 또는 기능적 설명은 단지 본 발명의 개념에 따른 실시 예들을 설명하기 위한 목적으로 예시된 것으로서, 본 발명의 개념에 따른 실시 예들은 다양한 형태로 실시될 수 있으며 본 명세서에 설명된 실시 예들에 한정되지 않는다.
- [0027] 본 발명의 개념에 따른 실시 예들은 다양한 변경들을 가할 수 있고 여러 가지 형태들을 가질 수 있으므로 실시 예들을 도면에 예시하고 본 명세서에서 상세하게 설명하고자 한다. 그러나, 이는 본 발명의 개념에 따른 실시 예들을 특정한 개시 형태들에 대해 한정하려는 것이 아니며, 본 발명의 사상 및 기술 범위에 포함되는 모든 변경, 균등물, 또는 대체물을 포함한다.
- [0028] 제1 또는 제2 등의 용어는 다양한 구성 요소들을 설명하는데 사용될 수 있지만, 상기 구성 요소들은 상기 용어들에 의해 한정되어서는 안 된다. 상기 용어들은 하나의 구성 요소를 다른 구성 요소로부터 구별하는 목적으로만, 예컨대 본 발명의 개념에 따른 권리 범위로부터 벗어나지 않은 채, 제1 구성 요소는 제2 구성 요소로 명명될 수 있고 유사하게 제2 구성 요소는 제1 구성 요소로도 명명될 수 있다.
- [0029] 어떤 구성 요소가 다른 구성 요소에 "연결되어" 있다거나 "접속되어" 있다고 언급된 때에는, 그 다른 구성 요소에 직접적으로 연결되어 있거나 또는 접속되어 있을 수도 있지만, 중간에 다른 구성 요소가 존재할 수도 있다고 이해되어야 할 것이다. 반면에, 어떤 구성 요소가 다른 구성 요소에 "직접 연결되어" 있다거나 "직접 접속되어" 있다고 언급된 때에는 중간에 다른 구성 요소가 존재하지 않는 것으로 이해되어야 할 것이다. 구성 요소들 간의 관계를 설명하는 다른 표현들, 즉 "~사이에"와 "바로 ~사이에" 또는 "~에 이웃하는"과 "~에 직접 이웃하는" 등도 마찬가지로 해석되어야 한다.
- [0030] 본 명세서에서 사용한 용어는 단지 특정한 실시 예를 설명하기 위해 사용된 것으로서, 본 발명을 한정하려는 의도가 아니다. 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 명세서에서, "포함하다" 또는 "가지다" 등의 용어는 본 명세서에 기재된 특징, 숫자, 단계, 동작, 구성 요소, 부분품 또는 이들을 조합한 것이 존재함을 지정하려는 것이지, 하나 또는 그 이상의 다른 특징들이나 숫자, 단계, 동작, 구성 요소, 부분품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.
- [0031] 다르게 정의되지 않는 한, 기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 모든 용어들은 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가진다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥상 가지는 의미와 일치하는 의미를 갖는 것으로 해석되어야 하며, 본 명세서에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다.
- [0033] 본 발명, 즉 암호문 셀프-업데이트가 가능한 암호화 기법 또는 암호문 셀프-업데이트가 가능한 암호화 시스템에서는 CCA(Chosen Ciphertext Attacks)에 안전한 암호문 셀프-업데이트가 가능한 암호를 설계하기 위해 먼저 암호문 위임 암호화 기법(CDE: Ciphertext Delegate Encryption)을 설계하고, CDE와 Time Structure를 이용하여 기법을 설계한다.
- [0035] 우선 암호문 위임 암호화 기법(CDE)은 초기화 알고리즘(CDE.Init), 설정 알고리즘(CDE.Setup), 키생성 알고리즘(CDE.GenKey), 난수화 알고리즘(CDE.RandKey), 암호화 알고리즘(CDE.Encrypt), 위임 알고리즘(CDE.DelegateCT), 검증 알고리즘(CDE.VerifyCT) 및 복호화 알고리즘(CDE.Decrypt)을 포함하는 총 8개의 알고리즘으로 구성된다. 각각의 알고리즘은 다음과 같이 구성된다.
- [0037] 1. 초기화 알고리즘($CDE.Init(1^\lambda) \rightarrow GDS$)
- [0038] 본 알고리즘은 보안 상수 1^λ 를 입력으로 받고, 곱셈형 함수 (p, G, G_{ne}) 와 생성원(generator) g 를 생성

한다. 생성된 값들은 $GDS = ((p, G, G_T, e), g)$ 라 정의되고 출력된다.

2. 설정 알고리즘($CDE.Setup(GDS, l) \rightarrow MK, PP$)

본 알고리즘은 Z_p 에서 임의의 값 β 를 선택하고, G 에서 임의의 값 $w, v, u, \{h_{i,0}, h_{i,1}\}_{i=1}^l$ 을 선택한다. $i \in [l]$ 그리고 $b \in \{0,1\}$ 에 대해서 $F_{i,b}(L) = u^L h_{i,b}$ 로 정의한다. 이후, $\Lambda = e(g, g)^\beta$ 를 생성한 다음 마스터키 $MK = \beta$ 와 공개 파라미터 $PP = ((p, G, G_T, e), g, w, v, u, \{h_{i,0}, h_{i,1}\}_{i=1}^l, \Lambda = e(g, g)^\beta)$ 를 출력한다.

3. 키생성 알고리즘($CDE.GenKey(L, MK, PP) \rightarrow SK_L$)

본 알고리즘은 Z_p 에서 임의의 값 $r, r_1, \dots, r_n$ 을 선택한 다음 $K_0 = g^\beta w^r$, $K_1 = g^{-r}$, 그리고 $\{K_{i,1} = v^r F_{i,L[i]}(L|_i)^{r_i}, K_{i,2} = g^{-r_i}\}_{i=1}^n$ 값을 생성한다. 다음으로, 라벨 L 에 대한 개인키 $SK_L = (K_0, K_1, \{K_{i,1}, K_{i,2}\}_{i=1}^n)$ 를 출력한다.

4. 난수화 알고리즘($CDE.RandKey(SK_L, \delta, PP) \rightarrow SK_L$)

본 알고리즘은 Z_p 에서 임의의 값 $r', r'_1, \dots, r'_n$ 을 선택한 다음 개인키 SK_L 를 다음과 같이 재 난수화한다. $K'_0 = K_0 \cdot g^\beta w^{r'}$, $K'_1 = K_1 \cdot g^{-r'}$, $\{K'_{i,1} = K_{i,1} \cdot v^{r'} F_{i,L[i]}(L|_i)^{r'_i}, K'_{i,2} = K_{i,2} \cdot g^{-r'_i}\}_{i=1}^n$. 다음으로, 재 난수화된 개인키 $SK_L = (K'_0, K'_1, \{K'_{i,1}, K'_{i,2}\}_{i=1}^n)$ 를 출력한다.

5. 암호화 알고리즘($CDE.Encrypt(L, t, \vec{s}, PP) \rightarrow CH_L, EK$)

본 알고리즘은 d 길이를 갖는 라벨 L 과 벡터 $\vec{s} = (s_1, \dots, s_d) \in Z_p^d$ 를 입력받는다. 다음으로 Z_p 에서 임의의 값 t 를 선택하고 세션키 $EK = \Lambda^t$ 를 생성한다. 다음으로, $C_0 = g^t$, $C_1 = w^t \prod_{i=1}^d v^{s_i}$, $\{C_{i,1} = g^{s_i}, C_{i,2} = F_{i,L[i]}(L|_i)^{s_i}\}_{i=1}^d$ 을 생성한다. 암호문 $CH_L = (C_0, C_1, \{C_{i,1}, C_{i,2}\}_{i=1}^d)$ 와 세션키 EK 를 출력한다.

6. 위임 알고리즘($CDE.DelegateCT(CH_L, c, PP) \rightarrow CH_L$)

[0053] 본 알고리즘은 $CH_L = (C_0, C_1, \{C_{i,1}, C_{i,2}\}_{i=1}^d)$ 그리고 $L \in \{0,1\}^d$ 일 때, 길이가 $d+1$ 인 새로운 라벨 $L' = L||c (c \in \{0,1\})$ 를 생성한다. 다음으로, $\mathbb{Z}_p$ 에서 임의의 값 s_{d+1} 을 선택하여 다음과 같이 계산한다. $C'_0 = C_0$, $C'_1 = C_1 \cdot v^{s_{d+1}}$, $\{C'_{i,1} = C_{i,1}, C'_{i,2} = C_{i,2}\}_{i=1}^d$, $C'_{d+1,1} = g^{s_{d+1}}$, $C'_{d+1,2} = F_{d+1,c}(L')^{s_{d+1}}$. 계산된 값으로 구성된 위임된(delegated) 암호문 헤더 $CH_L = (C'_0, C'_1, \{C'_{i,1}, C'_{i,2}\}_{i=1}^{d+1})$ 를 출력한다.

[0055] 7. 검증 알고리즘($CDE.VerifyCT(CH_L, L, PP) \rightarrow \{0,1\}$)

[0056] 본 알고리즘은 $CH_L = (C_0, C_1, \{C_{i,1}, C_{i,2}\}_{i=1}^d)$ 그리고 $L \in \{0,1\}^d$ 일 때, 모든 $i \in \{1, \dots, d\}$ 에 대해서 다음과 같은 두 식을 계산하여 참이 되는지 검증한다.

[0057]
$$e(C_1, g) = e(C_0, w) \cdot e(\prod_{i=1}^d C_{i,1}, v)$$

[0058]
$$e(C_{i,2}, g) = e(C_{i,1}, F_{i,L[i]}(L_i))$$

[0059] 모든 $i \in \{1, \dots, d\}$ 에서 참이 되는 경우에 1을 출력하고, 그렇지 않은 경우에 0을 출력한다. 이로써, 암호문 헤더에 대한 유효성을 검증할 수 있다.

[0061] 8. 복호화 알고리즘($CDE.Decrypt(CH_L, SK_L, PP) \rightarrow EK$)

[0062] 본 알고리즘은 $SK_L = (K_0, K_1, \{K_{i,1}, K_{i,2}\}_{i=1}^n)$, $L' \in \{0,1\}^n$ 일 때, 그리고 L 이 L' 의 프리픽스(prefix)일 때, CH_L 로부터 위임 알고리즘($CDE.DelegateCT$)을 반복적으로 사용하여 CH_L 를 생성한다. 다음으로, 세션키 $EK = e(C'_0, K_0) \cdot e(C'_1, K_1) \cdot \prod_{i=1}^n (e(C'_{i,1}, K_{i,1}) \cdot e(C'_{i,2}, K_{i,2}))$ 를 생성하여 출력한다.

[0064] 상술한 바와 같이 정의된 CDE 알고리즘과 타임 구조(Time Structure)를 사용하여 CCA에 안전한 암호문 셀프-업데이트가 가능한 암호를 설계할 수 있다. 추가로 인덱스(index) 집합 $\{z\}$ 로 인덱스된(indexed) 해쉬함수 H^z 의 집합 $\mathcal{H}$ 를 정의한다. 즉, $H^z \in \mathcal{H}$ 이다.

[0066] 암호문 셀프-업데이트가 가능한 암호(SUE: Self-Updatable Encryption)는 초기화 알고리즘(SUE.Init), 설정 알고리즘(SUE.Setup), 키생성 알고리즘(SUE.GenKey), 난수화 알고리즘(SUE.RandKey), 암호화 알고리즘(SUE.Encrypt), 업데이트 알고리즘(SUE.UpdateCT), 검증 알고리즘(SUE.VerifyCT), 복호화 알고리즘(SUE.Decrypt)의 총 8개의 알고리즘으로 이루어져 있다. 알고리즘들 각각에 대한 구체적인 설명은 다음과 같다.

[0068] 1. 초기화 알고리즘($SUE.Init(1^\lambda) \rightarrow GDS$)

[0069] 본 알고리즘은 암호문 위임 암호화 기법(CDE)의 초기화 알고리즘($CDE.Init(1^\lambda) \rightarrow GDS$)을 통하여, 보안 상수(1^λ)를 입력받고 GDS 를 생성하고, 생성된 GDS 를 출력한다. 상기 GDS 는 곱셈형 함수(p, G, G_{Te})와 생성원(generator) g 를 포함할 수 있다.

[0071] 2. 설정 알고리즘($SUE.Setup(GDS, T_{\max}) \rightarrow MK, PP$)

[0072] 본 알고리즘은 곱셈형 그룹(G)에서 임의의 값 $u_S h_S$ 를 선택하고, $T_{\max} = 2^{l+1} - 1$ 일 때(여기서, l 은 포화 이진트리의 높이를 의미할 수 있고, $T_{\max}$ 는 포화 이진트리의 오른쪽 말단 노드에 부여된 시간 정보를 의미할 수 있음), 암호문 위임 암호화 기법(CDE)의 설정 알고리즘($CDE.Setup(GDS, l)$)을 통해 MK_{CDE} , PP_{CDE} 를 생성한다. 다음으로, 임의의 인덱스(index) z 를 선택하여 해쉬함수 $H^z \in \mathcal{H}$ 를 선택한다. 마지막으로, 마스터키 $MK = MK_{CDE}$ 와 공개 파라미터 $PP = (PP_{CDE}, z, u_S, h_S)$ 를 출력할 수 있다.

[0074] 3. 키생성 알고리즘($SUE.GenKey(T, MK, PP) \rightarrow SK_T$)

[0075] 본 알고리즘은 ψ 함수를 통해 입력된 시간 정보가 포함된 노드의 라벨스트링을 얻어온 뒤 암호문 위임 암호화 기법(CDE)의 키생성 알고리즘($CDE.GenKey(\psi(T), MK, PP)$)을 통해 시간 T 에 대한 개인키 SK_T 를 생성하고 출력할 수 있다.

[0077] 4. 난수화 알고리즘($SUE.RandKey(SK_T, PP) \rightarrow SK_T$)

[0078] 본 알고리즘은 암호문 위임 암호화 기법(CDE)의 난수화 알고리즘($CDE.RandKey(SK_T, PP)$)을 통하여 개인키 SK_T 를 재 난수화하여 출력할 수 있다. 실시 예에 따라, 본 알고리즘(난수화 알고리즘)은 키생성 알고리즘($SUE.GenKey(T, MK, PP) \rightarrow SK_T$)에 포함되는 개념으로 이해될 수도 있다. 이 경우, 키생성 알고리즘은 생성된 재 난수화된 개인키 SK_T 를 출력할 수 있다.

[0080] 5. 암호화 알고리즘($SUE.Encrypt(T, t, PP) \rightarrow CH_T, EK$)

[0081] 본 알고리즘은 $L = \psi(T)$ 로 시간 정보 T 에 대한 라벨스트링을 얻어온 뒤 다음의 과정을 통해 결과를 출력한다.

[0082] 1) $TimeNodes(L)$ 을 통해 $TL = (L^{(0)} = L, L^{(1)}, \dots, L^{(d)})$ 를 얻어온다. 이 때, $d^{(j)}$ 는 $L^{(j)}$ 라벨스트링의 길이로 정의한다. 또한, TL 은 전위순회(pre-order traversal) 순을 따른다.

[0083] 2) 각각의 $j \in \{0, \dots, d\}$ 에 대해서 다음의 과정을 수행한다.

- [0084] a. $j=0$ 일 때, $Z_p^{d^{(0)}}$ 에서 임의의 $\vec{s} = s_1, \dots, s_{d^{(0)}}$ 를 선택하고, 암호문 위임 암호화 기법(CDE)의 암호화 알고리즘($CDE.Encrypt(L^{(0)}, t, \vec{s}, PP)$)을 통해 $CH^{(0)} = (C_0, C_1, \{C_{i,1}, C_{i,2}\}_{i=1}^{d^{(0)}})$ 와 EK 를 생성한다.
- [0085] b. $j \geq 1$ 일 때, Z_p 에서 임의의 $s_{d^{(j)}}$ 를 선택하고, $\vec{s}$ 에서 $s_1, \dots, s_{d^{(j)}-1}$ 부분만 가져와 $\vec{s}' = (s_1, \dots, s_{d^{(j)}-1}, s_{d^{(j)}})$ 를 구성한다. 다음으로, 암호문 위임 암호화 기법(CDE)의 암호화 알고리즘($CDE.Encrypt(L^{(j)}, t, \vec{s}', PP)$)을 통해 $CH^{(j)} = (C'_0, C'_1, \{C'_{i,1}, C'_{i,2}\}_{i=1}^{d^{(j)}})$ 를 생성한다. 마지막으로, $CH^{(j)}$ 에서 $CH^{(0)}$ 에 포함되어 있는 동일한 부분인 $C'_0, \{C'_{i,1}, C'_{i,2}\}_{i=1}^{d^{(j)}-1}$ 를 제거한다.
- [0086] 3) 해쉬함수 연산을 통해 $\pi = H^z(C_0)$ 를 생성하고, $C_3 = (u_s^\pi h_s)^t$ 를 생성한다.
- [0087] 4) 암호문 헤더 $CH_T = (CH^{(0)}, CH^{(1)}, \dots, CH^{(d)}, C_3)$ 와 세션키 EK 를 출력한다.
- [0089] 6. 업데이트 알고리즘($SUE.UpdateCT(CH_T, T+1, PP) \rightarrow CH_{T+1}$)
- [0090] 본 알고리즘은 $CH_T = (CH^{(0)}, CH^{(1)}, \dots, CH^{(d)}, C_3)$ 그리고 $L^{(j)}$ 를 $CH^{(j)}$ 의 라벨스트링이라 할 때, 다음과 같은 과정을 통해 결과를 출력한다.
- [0091] 1) $L^{(0)}$ 의 길이가 l 보다 작을 때, 모든 $c \in \{0, 1\}$ 에 대해 암호문 위임 암호화 기법(CDE)의 위임 알고리즘($CDE.DelegateCT(CH^{(0)}, c, PP)$)을 통해 $CH_{L^{(0)}||0}$ 과 $CH_{L^{(0)}||1}$ 를 생성한다. 다음으로, 생성된 두 부분과 CH_T 값을 가지고 업데이트된 암호문 헤더 $CH_{T+1} = (CH'^{(0)} = CH_{L^{(0)}||0}, CH'^{(1)} = CH_{L^{(0)}||1}, CH'^{(2)} = CH^{(1)}, \dots, CH'^{(d+1)} = CH^{(d)}, C_3)$ 를 생성하고 출력한다.
- [0092] 2) $L^{(0)}$ 의 길이가 l 일 때, $CH^{(1)}$ 에서 $CH^{(0)}$ 과 공통돼서 제거했던 부분을 다시 $CH^{(1)}$ 으로 복사한다. 다음으로, $CH^{(0)}$ 를 제거하고 업데이트된 암호문 헤더 $CH_{T+1} = (CH'^{(0)} = CH^{(1)}, \dots, CH'^{(d-1)} = CH^{(d)}, C_3)$ 를 출력한다.
- [0094] 7. 검증 알고리즘($SUE.VerifyCT(CH_T, T, PP) \rightarrow \{0, 1\}$)
- [0095] 본 알고리즘은 $CH_T = (CH^{(0)}, CH^{(1)}, \dots, CH^{(d)}, C_3)$ 그리고 라벨스트링 $L = \psi(T)$ 이라 할 때, 다음과 같은 과정을 통해 결과를 출력한다.
- [0096] 1) 먼저 $TimeNodes(L)$ 을 통해 $TL = (L^{(0)}, L^{(1)}, \dots, L^{(d)})$ 를 생성한다. 다음으로, TL 에 있는 라벨의 개수와 CH_T 에 있는 CDE 암호문 헤더의 개수가 동일한지 검증한다.

- [0097] 2) TL 내의 각각의 라벨스트링 $L^{(j)}$ 에 대해 암호문 위임 암호화 기법(CDE)의 검증 알고리즘 $(CDE.VerifyCT(CH^{(j)}, L^{(j)}, PP))$ 을 통해 검증을 수행한다.
- [0098] 3) 해쉬함수 연산을 통해 $\pi = H^z(C_0)$ 를 생성하고, 다음 수식 $e(C_3, g) = e(C_0, u_{sh}^\pi)$ 이 참이 되는지 검증한다.
- [0099] 4) 위 단계의 모든 과정에서 검증이 되면 1을, 하나라도 검증이 안되면 0을 출력한다.
- [0100] 상술한 검증 알고리즘을 수행함으로써 암호문 헤더의 유효성이 검증될 수 있다.
- [0102] 8. 복호화 알고리즘($SUE.Decrypt(CH_T, SK_{T'}, PP) \rightarrow EK$)
- [0103] 본 알고리즘은 만약 $T > T'$ 라면 복호화가 불가능하므로 $\perp$ 를 출력한다. $T \leq T'$ 인 경우에는 다음과 같은 과정을 통해 결과를 출력한다.
- [0104] 1) 먼저, 검증 알고리즘($SUE.VerifyCT(CH_T, T, PP)$)의 연산 결과로써 1이 출력되었는지, 즉 암호문 헤더의 유효성 검증에 성공했는지 여부를 확인한다. 1이 출력되지 않은 경우에는 $\perp$ 를 출력한다. 이 경우, 암호문 헤더가 유효하지 않기 때문에 추가적인 세션키 생성 과정을 수행할 필요가 없다.
- [0105] 2) CH_T 안에서 $L' = \psi(T')$ 의 프리픽스(prefix)인 라벨스트링 $L^{(j)}$ 을 갖는 $CH^{(j)}$ 를 찾는다. 다음으로, 암호문 위임 암호화 기법(CDE)의 복호화 알고리즘($CDE.Decrypt(CH^{(j)}, SK_{T'}, PP)$)을 통해 세션키 EK 를 생성하고 출력한다.
- [0106] 상술한 내용 이외에 본 발명에서 이용하는 알고리즘에 대한 보다 구체적인 설명은 본 발명의 발명자인 Kwangsu Lee 등의 "CCA Security for Self-Updatable Encryption: Protecting Cloud Data When Clients Read/Write Ciphertexts(IACR Cryptology ePrint Archive, 2015)"와 Kwangsu Lee 등의 "Self-updatable encryption: Time constrained access control with hidden attributes and better efficiency(In Kazue Sako and Palash Sarkar, editors, ASIACRYPT 2013, volume 8269 of Lecture Notes in Computer Science, pages 235254. Springer, 2013)"가 참조될 수 있다.
- [0108] 도 2는 본 발명의 일 실시 예에 따른 암호 시스템을 도시한다.
- [0109] 도 2를 참조하면, 암호문 셀프-업데이트가 가능한 공개키 암호 시스템 등으로 불릴 수도 있는 암호 시스템(10)은 서버(100), 송신 단말(300) 및 수신 단말(500)을 포함한다.
- [0110] 앞서 상세하게 설명된 알고리즘들(또는 단계들) 중 초기화 알고리즘(SUE.Init)과 업데이트 알고리즘(SUE.UpdateCT)은 서버(100)에 의해 수행되고, 설정 알고리즘(SUE.Setup), 키생성 알고리즘(SUE.GenKey), 난수화 알고리즘(SUE.RandKey) 및 암호화 알고리즘(SUE.Encrypt)은 송신 단말(300)에 의해 수행되고, 검증 알고리즘(SUE.VerifyCT)과 복호화 알고리즘(SUE.Decrypt)은 수신 단말(500)에 의해 수행될 수 있다.
- [0111] 알고리즘들 각각에 대한 상세한 설명은 앞서 설명한 바 있으므로, 이에 대한 상세한 설명은 생략하기로 한다.
- [0112] 송신 단말(300)과 수신 단말(500) 각각은 PC(personal computer), 태블릿 PC, 노트북(notebook), 넷-북(net-book), e-리더(e-reader), PDA(personal digital assistant), PMP(portable multimedia player), MP3 플레이어, 또는 MP4 플레이어와 같은 데이터 처리 장치로 구현되거나, 모바일폰(mobile phone), 스마트폰(smart phone)과 같은 핸드헬드 장치(handheld device)로 구현될 수 있다.
- [0114] 본 발명은 도면에 도시된 실시 예를 참고로 설명되었으나 이는 예시적인 것에 불과하며, 본 기술 분야의 통상의

지식을 가진 자라면 이로부터 다양한 변형 및 균등한 타 실시 예가 가능하다는 점을 이해할 것이다. 따라서, 본 발명의 진정한 기술적 보호 범위는 첨부된 등록청구범위의 기술적 사상에 의해 정해져야 할 것이다.

부호의 설명

[0116]

10 : 암호 시스템

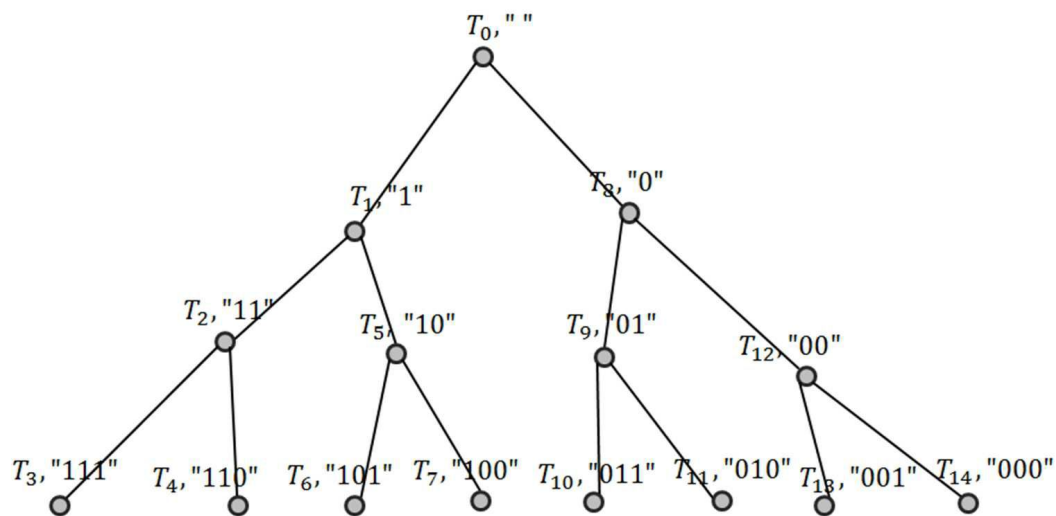
100 : 서버

300 : 송신 단말

500 : 수신 단말

도면

도면1



도면2

