情報保護學會論文誌
第14券 第3號, 2004. 6

익명성을 보장하는 비대칭 공개키 공모자 추적 기법

최 은 영,[a)†] 이 동 훈,[a)‡] 홍 도 원[b)]
고려대학교,[a)] 한국 전자 통신 연구원[b)]

An Anonymous asymmetric public key traitor tracing scheme

Eun Young Choi,[a)†] Dong Hoon Lee,[a)‡] Dowon Hong[b)]
Korea University,[a)] Electronics and Telecommunications Research Institute[b)]

요 약

브로드캐스트 암호화 스킴에서, 추적 가능성은 권한을 부여 받은 사용자들 중에 불법 디코더를 생성하는데 공모한 사용자들을 추적하기 위한 프로토콜에 유익하게 이용되는 성질이다. 유감스럽게도, 이 성질은 대개의 경우 프라이버시를 희생하여 성취된다. 현재까지 대부분의 공모자 추적 기법은 사용자의 익명성을 고려하지 않은 상태에서 연구가 진행되어 왔다. 이것은 현실 세계 시장에서와 유사한 프라이버시를 전자 상거래에서 제공하기 위한 중요한 요구사항이다. 하지만 멀티미디어 컨텐츠를 구매하기 위해 사용자의 신원이 노출된다면 사용자에게는 불만족스러운 일이다. 본 논문에서는 멀티미디어 컨텐츠를 구매하는 과정에서 사용자의 취미, 생활 정보, 신원 정보 등에 대한 정보를 누출시키지 않으면서 사용자가 익명으로 멀티미디어 컨텐츠를 구매할 수 있고, 동시에 데이터 제공자가 공모자를 추적하고자 할 경우에는 사용자의 신원을 알아낼 수 있는 익명성을 보장하는 암호화 기법을 제안한다.

ABSTRACT

In broadcast encryption schemes, traceability is a useful property to trace authorized subscribers, called traitors, who collude for manufacturing a pirate decoder. Unfortunately, this is usually achieved with a sacrifice of a privacy. Most traitor tracing schemes in the literature have been developed without considering a subscriber's anonymity, which is one of important requirements for electronic marketplaces to offer similar privacy as current marketplace. It would be unsatisfactory for the subscriber to reveal his/her identity to purchase multimedia contents. In this paper we propose an anonymous broadcast encryption scheme, where a user can subscribe anonymously and one purchases multimedia contents without giving a lot of information about his lifestyle, habits, and etc, but anonymity control is provided, i.e., a data supplier can trace traitors.

Keywords : *Public key traitor tracing, Anonymous, Asymmetric*

Ⅰ. 서 론

최근, 브로드캐스트 암호화 스킴은 네트워크 상에서 디지털 컨텐츠 (멀티미디어, 소프트웨어)를 배포하는데 사용된다. 브로드캐스트 암호화 스킴에서 권한을 부여 받은 사용자만이 디지털 컨텐츠에 접근 할 수 있으며 권한이 없는 사용자는 브로드캐스트 암호화 메시지로부터 디지털 컨텐츠를 얻을 수 없어야 하는 것이 중요한 요구사항이다. 일반적으로 이러한 기

접수일 : 2003년 12월 8일 ; 채택일 : 2004년 5월 17일
* 본 연구는 한국 전자 통신 연구원 2003년 "브로드 캐스트/멀티캐스트 보안기술 연구에 관한 연구" 수행 논문입니다.
† 주저자, bluecey@cist.korea.ac.kr
‡ 교신저자, donghlee@korea.ac.kr

밀성을 보장하기 위해 브로드캐스트 메시지는 권한을 부여받은 사용자들의 비밀키로 암호화 되어진 세션키를 사용하여 암호화 된다. 따라서 각각의 권한을 부여 받은 사용자는 각각 자신의 비밀키를 이용해서 암호화된 세션키를 복호화 하고, 그 세션키를 이용해서 브로드캐스트 메시지를 복호화 한다.

브로드캐스트 스킴에서 또 다른 중요한 요구사항은 공모자 추적 가능성이다. 공모자 추적이라는 개념은 Chor[1]등에 의해서 처음 제안되었다. 이 개념은 불법 디코더를 생성하는데 자신의 개인키를 제공한 권한을 부여받은 사용자를 찾아 낼 수 있어야 한다는 것을 의미한다. 여기에서 불법 디코더를 생성하는데 참여한 권한을 부여 받은 사용자를 공모자라고 한다 (일반적인 공모자 추적 기법에서 권한을 부여 받은 사용자가 자신이 얻은 세션키를 다른 사람에게 주는 것은 제외한다). 공모자 추적 기법은 비밀키의 공유 형태에 따라서 크게 대칭과 비대칭으로 구분한다. 대칭 공모자 추적 기법은 각각의 사용자가 브로드캐스트 암호화 메시지를 복호화하기 위해 사용하는 사용자의 비밀키를 데이터 제공자와 공유하는 형태를 말한다. 이러한 대칭 공모자 추적 기법은 데이터 제공자가 불법 디코더를 생성하는데 참여하지 않은 사용자를 불법 디코더를 생성하는데 공모하였다고 할 수 있다. 왜냐하면 데이터 제공자와 사용자가 세션키를 복호화 하는데 사용하는 비밀키를 공유하기 때문에 불법 디코더에 사용자의 비밀키가 사용되었다고 한다면 사용자는 부인할 수 없다. 이 문제점을 보안하기 위해서 Pfitzmann이 비대칭 공모자 추적 기법을 제안하였다[2]. 비대칭 공모자 추적 기법에서는 사용자만이 비밀키를 알 수 있고 데이터 제공자는 알 수 없기 때문에 대칭 공모자 추적에서의 문제점을 해결할 수 있다.

그리고 Boneh와 Franklin은 공개키 기반의 공모자 추적 기법[3]을 제안하였다. 이 공개키 공모자 추적 기법에서 데이터 제공자는 하나의 암호화 공개키를 사용하여 세션키를 암호화 하고 각각의 사용자는 서로 다른 개인키를 사용하여 세션키를 복호화 한다. 응용측면에서, 공개키 공모자 기법은 여러 명의 데이터 제공자에 의해 디지털 컨텐츠가 전송되는 것을 가능하게 함으로 효율적이다.

또한, 멀티미디어 컨텐츠를 구매하는 과정에서 사용자의 취미, 생활 정보, 신원 정보 등에 대한 정보를 포함하기 때문에 사용자의 익명성에 대한 요구사항이 더 중요하게 되었다. 익명성에 대한 다양한 연구는 익명성을 제공하는 지불 및 통신 시스템[4-6]분야에서 연구되어졌다. 사용자의 익명성은 멀티미디어 컨텐츠를 구매하는 과정에서 만족되어 질 수 없는 성질이다. 그리고 이 요구사항은 공모자를 추적해서 신원을 알아내야 하는 추적성과 개념적으로 상반되는 개념이다. 즉, 사용자가 익명으로 컨텐츠를 구매하더라도 불법 재분배의 경우에 불법 디코더를 생성하는데 참여한 공모자를 추적할 수 있어야 한다. 따라서 익명성을 제공하면서 동시에 공모자를 추적할 수 있는 스킴을 설계하는 것은 쉽지 않다.

최근, Magkos 등은 Time-lock puzzles[7]과 은닉서명[8]을 사용하여 익명성을 보장하는 공모자 추적 기법을 제안하였다[9]. 그러나 익명성을 제공하기 위해 사용한 Time-lock puzzles은 실질적으로 익명성을 제공하지 못한다. 즉, 정직하지 않은 데이터 제공자는 공모에 참여하지 않은 사용자의 신원을 알아 낼 수 있다. 본 논문에서는 증명 프로토콜[10], 게시판, 신뢰 기관을 사용하여 익명성을 제공할 수 있는 기법을 제안한다. 또한 기존의 효율적인 공개키 공모자 추적 기법을 이용하여 익명성을 보장하는 공개키 공모자 기법을 처음으로 제안한다.

본 논문은 다음과 같이 구성되어진다. 2장에서는 기존에 제안된 비대칭 공개키 공모자 추적 기법에 대해서 알아보고, 3장에서는 이전에 제안된 익명성을 보장하는 기법[9]의 문제점에 대해 살펴본다. 그리고 [9]의 기법에 익명성을 제공할 수 있는 익명성을 보장하는 기법을 제안한다. 4장에서는 비대칭 공개키 공모자 추적 기법에 익명성을 보장하는 기법을 적용시킨 새로운 기법을 제안하고 마지막으로 5장에서는 결론을 맺는다.

II. 비대칭 공개키 공모자 추적 기법

본 장에서는 익명성을 보장하는 비대칭 공개키 공모자 추적 기법을 제안하기 이전에 기존의 비대칭 공개키 공모자 추적 기법에 대해서 알아보고 본 논문에서 제안하고 있는 익명성을 보장하는 기법에 사용하는 요소들에 대해서 알아보도록 한다.

1. 표시법

암호화 키 ek를 사용하여 메시지 m을 암호화하는 것을 $E_{ek}(m)$라고 표시한다. 복호화 키 dk를 사

용하여 암호문 C를 복호화 하는 것을 $D_{dk}(C)$라고 표시한다. 서명키 sk를 사용하여 메시지 m을 서명하는 것을 $Sig_{sk}(m)$라고 표시한다. 또한, $\lceil a \rceil$는 a와 같거나 a보다 큰 가장 작은 정수를 표시한다.

2. 가정

다음의 가정을 기반하여 프로토콜이 이루어진다. G_q는 Z_p^*에서의 위수가 q인 곱셈의 부분군을 의미한다. 여기에서 p, q는 소수이며 $q|(p-1)$의 관계를 가진다.

3. Oblivious Polynomial Evaluation 프로토콜 설명

OPE(Oblivious Polynomial Evaluation) 프로토콜[11]이란 비밀 값 α를 가지고 있는 수신자 B가 함수 $p(x)$를 비밀 값으로 가지고 있는 송신자 A로부터 $p(\alpha)$을 계산하는 것이 가능한 프로토콜이다. 이 OPE 프로토콜은 다음의 성질들을 만족한다.

- 송신자 A는 수신자 B의 비밀 값 α에 대한 어떤 정보도 추출해 낼 수 없다.
- 수신자 B는 함수 값 $p(\alpha)$에 대한 것 외에 유한체 상에서의 함수 $p(x)$ 에 대한 어떤 정보도 추출해 낼 수 없다.

본 논문에서는 $OPE(\alpha)$는 비밀값만이 포함되어진 데이터를 표기하고 $OPE(p(\alpha))$는 함수에 비밀값이 포함되어진 데이터를 표기한다. OPE는 부가적으로 malleable이란 특성을 가진다. 즉, A는 $OPE(\alpha)$가 주어졌을 때, 유한체상에서 '+'연산의 성질을 이용하여 $OPE(\alpha+\alpha')$을 쉽게 구할 수 있다.

4. (2,1) 비 상호작용 OT(Non-Interactive Oblivious Transfer) 프로토콜 설명

(2,1)-비 상호작용 OT프로토콜[12]이란 수신자 B는 자신의 비밀키를 이용하여 송신자 A가 보내는 두 개의 비밀 값 (S_0, S_1)중에서 하나의 비밀 값을 얻는 프로토콜이다. 이 과정에서 송신자 A는 수신자 B가 두개의 비밀 값 중 어떤 값을 얻었는지 알 수 없다. 또한 비 상호 작용이므로 송신자 A는 메시지를 전송하고 수신자 B는 송신자 A에게 메시지를 전송하지 않는다. (2,1)-비 상호작용 OT를 OT_2^1로 표기한다. 이 절에서는 OT_2^1에 대해서 간단히 알아보도록 하겠다. g는 G_q의 생성자이고, 시스템의 모든 사용자들은 $C \in Z_q^*$값에 대해서 알고 있지만 C의 discrete log에 대해서는 아무도 모른다고 가정한다.

수신자 B는 랜덤하게 $i \in 0,1$과 $x_i \in Z_{q-1}$를 선택한다. 수신자 B는 $\beta_i = g^{x_i}$, $\beta_{1-i} = C \cdot (g^{x_i})^{-1}$를 계산하고 자신의 공개키로 공개한다. 수신자 B의 개인키는 i, x_i이다. 송신자 A는 랜덤하게 $y_i \in Z_{q-1}$ $i \in 0,1$를 선택하고, $\alpha_0 = g^{y_0}$, $\alpha_1 = g^{y_1}$, $\gamma_0 = \beta_0^{y_0}$, $\gamma_1 = \beta_1^{y_1}$를 계산한다. 송신자 A는 $r_i = S_i \oplus \gamma_i$, $i \in 0,1$을 계산하여 수신자 B에게 α_0, α_1, r_0, r_1를 보낸다. 수신자 B는 자신의 개인키를 이용하여 $\alpha_i^{x_i} = \gamma_i$ 계산하고 이 값을 이용하여 $\gamma_i \oplus r_i = S_i$를 얻는다.

5. 비대칭 공개키 공모자 추적 기법

Kiayias와 Yung는 불법 디코더를 생성하는데 참여한 사용자가 실제로 공모에 관련되었다는 것을 증명할 수 있는 효율적인 비대칭 공개키 공모자 추적 기법을 제안했다[13]. 비대칭 공개키 공모자 추적 기법의 구성원은 시스템 관리자, 데이터 제공자, 사용자들로 구성되어져 있으며, 시스템의 데이터 제공자들은 공개키 공모자 추적 기법의 특징으로 인해 하나의 데이터 암호화키를 사용하여 사용자에게 컨텐츠를 배포할 수 있다. 이 공모자 추적 기법에서 데이터 제공자 중 한명이 시스템 관리자의 역할을 할 수도 있다. 이 기법은 다음 여섯 단계로 이루어진다.

5.1 초기화

시스템 관리자는 Z_q를 선택하여 공개하고, Z_q에서 랜덤하게 함수 $Q_1(x) = a_0 + a_1 x + \cdots + a_{2v} x^{2v}$과 $b \in Z_q$를 선택한다. 그리고 $y = g^{a_0}, h_0 = g, h_1 = g^{-a_1}, \cdots, h_{2v} = g^{-a_{2v}}, h' = g^{-b}$을 계산한다. 시스템 관리자는 시스템의 공개키로써 $\langle y, h_0, \cdots, h_{2v}, h' \rangle$을 공개한다. 시스템 관리자는 $Q(x,y) = Q_1(x) + by$을

계산하고 $Q_1(x)$와 b 값을 비밀로 유지한다.

5.2 참여 단계

시스템 관리자는 Z_q에서 랜덤하게 z_u, α_u^R을 선택한다. 사용자 u는 Z_q에서 α_u^C를 랜덤하게 선택하고 commitment $\langle C_u = g^{\alpha_u^C}, Sig_{sk_u}(C_u)\rangle$을 생성한다. 이 과정에서 시스템 관리자가 선택한 α_u^R 값과 사용자 u가 선택한 α_u^C를 사용하여 $\alpha_u (= \alpha_u^R + \alpha_u^C)$를 생성하게 되는데 이 값은 유한체 상에서의 OPE의 malleable특성을 이용해서 생성되어지며 사용자 u는 생성한 commitment를 시스템 관리자에게 보낸다. 이 com-mitment는 사용자가 공모자라고 추정되었을 때 사용자가 부인할 수 없는 증거이다. 참여 단계에서 사용자가 얻게 되는 사용자의 개인키는 벡터 형태의 $\vec{k} = \langle Q(z_u, \alpha_u), z_u, z_u^2, \cdots z_u^{2v}, \alpha_u\rangle$이다. 참여 단계는 다음과 같다.

우선, 시스템 관리자는 $m = \lceil \log_2 |Z_q| \rceil$, $j \in m$에 대한 $v_j = 2^{j-1}\alpha_u^R$를 생성하고 랜덤한 $v_j = 2^{j-1}\ \alpha_u^R$를 선택하여 m개의 $(r_j, r_j + v_j)$ 쌍을 생성한다. 시스템 관리자는 z_u을 선택하여 $OPE(\alpha_u^R) = \{(r_j, r_j + v_j), r_1\}, z_u$를 생성하고 사용자에게 보낸다. 사용자는 자신이 선택한 α_u^C를 사용하여 시스템 관리자로부터 받은 m개의 쌍만큼의 $w_j = 2^{j-1}\alpha_u^C$를 생성하고 랜덤한 $R_j \in Z_q$를 선택하여 m개의 $OPE(\alpha_u) = (r_j + R_j, r_j + v_j + R_j + w_j)$쌍을 생성한다. 또한 시스템 관리자가 보낸 r_1과 $r_1 + v_1$을 이용하여 α_u^R를 얻고 데이터 복구를 위해 사용되는 개인키의 일부인 $\alpha_u = \alpha_u^R + \alpha_u^C$를 계산한다.

시스템 관리자는 자신이 선택한 b를 $b = \sum_{j \in m} b_j 2^{j-1}$로 표현한다. 시스템 관리자는 $b_j = 0$이면 $r_j + R_j$ 값을 얻고, $b_j = 1$이면 $r_j + v_j + R_j + w_j$를 얻을 수 있도록 사용자와 $OT_2^1(r_j + R_j, r_j + R_j + v_j + w_j)$ 프로토콜을 시행한다. 즉, $b_j = 0$이면 시스템 관리자는 OT_2^1에 사용할 공개키를 x_0를 이용하여 생성하며, $b_j = 1$이면 OT_2^1에 사용할 공개키를 x_1를 이용하여 생성한다. OT_2^1프로토콜을 m번 실행하여 m개의 값을 얻는다. 시스템 관리자는 자신이 선택한 함수에 z_u을 이용하여 $Q_1(z_u)$을 계산하고 그 값에 사용자로부터 받은 m개의 값을 합하여 $Q(z_u, \alpha_u) + \sum_{j \in m} R_j + \sum_{j \in m} r_j$값을 얻는다. 시스템 관리자는 $Q(z_u, \alpha_u) + \sum_{j \in m} R_j + \sum_{j \in m} r_j$에서 자신이 $OPE(\alpha_u^R)$에 첨가한 r_j의 합 $\sum_{j \in m} r_j$를 뺀다. 즉, $OPE(Q(z_u, \alpha_u)) = Q_1(z_u) + b(\alpha_u) + \sum_{j \in m} R_j$을 얻게 된다. 시스템 관리자는 $OPE(Q(z_u, \alpha_u))$을 사용자에게 보낸다.

사용자는 $OPE(Q(z_u, \alpha_u))$에서 자신이 첨가한 R_j의 합 $\sum_{j \in m} R_j$을 뺀다. 사용자는 자신의 개인키를 구성하는 $Q(z_u, \alpha_u)$를 얻게 된다. 그러므로 사용자의 개인키는 사용자만이 알 수 있다.

5.3 암호화 단계

데이터 제공자는 Z_q에서 랜덤하게 선택한 r과 암호화 키 (공개키) $ek = \langle y, h_0, \cdots, h_{2v}, h'\rangle$를 사용하여 메시지 M을 암호화 한다. 암호화 과정은 다음과 같다.

$$E_{ek}(M) = \langle y^r \cdot M, h_0^r, \cdots, h_{2v}^r, (h')^r\rangle$$

5.4. 복호화 단계

사용자들은 암호문 $\tilde{G} = \langle G, G_0, \cdots, G_{2v}, G'\rangle$에 각각 자신의 개인키 $\vec{k} = \langle \delta_0, \cdots, \delta_{2v}, \delta'\rangle$를 사용하여 암호문을 복호화 한다. 복호화 과정은 다음과 같다.

$$D_{\vec{k}}(\tilde{G}) = G/((G')^{\delta'} \prod_{j=0}^{2v} (G_j)^{\delta_j})$$

5.5 공모자 추적 단계

공모자 추적 단계에서 시스템 관리자는 공모자 추적 알고리즘을 사용하여 공모자를 알아낸다. 공모자

보를 찾아낸다. 공모자 추적 알고리즘의 입력값은 공모자들이 형성한 불법 디코더이다 (공모자의 불법 디코더는 블랙박스 추적기법에 의해서 얻을 수 있다). 공모자 추적 알고리즘은 입력값에 대해서 공모자를 찾을 수 있는 벡터를 출력한다. 시스템 관리자는 출력값을 이용하여 공모자를 추적한다.

공모자들이 생성하는 불법 디코더는 $\vec{K}=\sum_{l=1}^{t}\mu_l \vec{k}_{u_l}$의 형태이다. 불법 디코더는 공모자들의 개인키의 선형 결합으로 구성되어지게 된다. 다음 정리는 불법 디코더의 형태가 공모자들의 개인키의 선형 결합이여야 하는 이유에 대하여 설명한다. (즉, 이산대수 문제는 풀기 어려운 문제이므로 불법 디코더는 공모자들의 개인키의 선형 결합의 형태를 갖게 된다.)

○ 정리 : 시스템의 공개키 $ek=\langle y, h_0, \cdots, h_{2v}, h' \rangle$와 사용자 t명의 개인키($\langle Q(z_i, \alpha_i), z_i, \alpha_i \rangle$)가 주어지면 공격자가 기존에 주어진 사용자들의 개인키의 선형 결합의 형태가 아닌 불법 디코더를 생성한다고 가정한다. 그렇게 되면, 그룹 X에서의 이산대수 (discrete-logarithm) 문제가 풀린다.

공모자 추적 알고리즘은 불법 디코더의 입력에 대해 공모자를 찾을 수 있는 벡터 $\vec{\nu}=\langle \nu_1, \cdots, \nu_n \rangle$를 출력한다. 벡터 $\vec{\nu}$는 불법 디코더를 생성하는데 참여한 사용자들을 나타내는 벡터이다. 이 벡터는 공모자를 나타내는 t 개에 대해서는 $\nu_{u_l}=\mu_l$를 출력하고, 그 이외 값 (n-t)개에 대해서는 $\nu_i=0$를 출력한다. 공모자 추적 알고리즘은 Algebraic Codes의 decoding에 기반하며 공모자 추적 알고리즘은 Linear codes decoding에 기반하여 공모자를 추적하는 방법[3,14]과 유사하다. 즉, 공모자 추적 알고리즘은 Generalized Reed-Solomon Code이면서 주어진 조건을 만족하는 코드를 선택하여 Generalized Reed-Solomon Code의 성질을 이용하여 복호화 하는 과정에서 공모에 참여한 사용자를 찾을 수 있는 벡터 값을 출력한다. 공모자 추적 과정은 다음과 같다.

공모자 추적 알고리즘은 다음의 가정을 기반하여 아래 정의를 만족하는 코드 C를 사용한다.

○ 가정 : • 사용자의 수 n는 시스템에 사용되는 변수 v보다 무수히 크며, $n>2v$라고 가정한다.

• H는 시스템 관리자가 사용자에게 전송하는 z_i로 구성된 ($z_i, \cdots,$)를 행으로 구성하는 ($n\times 2v$)행렬이다.

• $\lambda_1, \cdots, \lambda_n$는 차수가 n보다 작은 모든 $g\in Z_q[x]$에 대하여 $\lambda_1 g(z_1)+\cdots+\lambda_n g(z_n)=g(0)$인 Lagrange coefficients라고 하자.

○ 정의 : C는 H를 parity-check행렬로 갖는 Z_q^n에서의 코드라고 정의한다. 즉, C는 $\deg ree(M)<n-2$ 형태의 원소들의 집합이고, massage-rate는 $(n-2v)/n$ 이고 distance가 $2v+1$인 linear code이다.

위에서 정의한 C는 Generalized Reed-Solomon Code이다. 그래서 Generalized Reed-Solomon Codes는 $e\le(n-k)/2$일 때, Reed-Solomon decoding의 해는 유일 해라는 것이 보장되며, Berlekamp-Welch algorithm[15]를 사용하여 polynomial time안에 유일 해를 구 할 수 있다는 성질을 이용하여 주어진 코드를 복호화 한다. 이 복호화 과정에 사용되는 Berlekamp-Welch algorithm은 주어진 벡터를 이용하여 벡터와 관련된 함수를 생성하는 알고리즘 이다.

C형태의 코드를 기반으로 하는 공모자 추적 알고리즘을 사용하기 위해서 주어진 불법 디코더 $\vec{K}=\sum_{l=1}^{t}\mu_l \vec{k}_{u_l}$는 $\vec{K}=\langle K_0, K_1, \cdots, K_{2v}, K' \rangle$라 표시한다(t : 공모자의 수). 또한, 시스템 관리자는 불법 디코더를 이용하여 $\vec{\eta}=\langle K_1, \cdots, K_{2v} \rangle$를 구성한다. $\vec{\eta}$의 구성 요소들에 의해, $\vec{\nu}\cdot H=\vec{\eta}$의 성질을 만족한다는 것을 알 수 있다. 시스템 관리자는 $\vec{\delta}\cdot H=\vec{\eta}$을 만족하는 $\vec{\delta}$를 계산한다. 벡터 $\vec{\omega}=\vec{\delta}-\vec{\nu}$가 linear code C의 원소라는 것은 쉽게 증명할 수 있다. $\vec{\omega}\cdot H=\vec{\delta}\cdot H-\vec{\nu}\cdot H=\vec{\eta}-\vec{\eta}=O$. 즉, $\vec{\omega}$가 C의 형태로 이루어진다는 것을 알 수 있다. 또한 벡터 $\vec{\delta}$는 $\vec{\delta}=\vec{\omega}+\vec{\nu}$라고 표시할 수 있다.

다시 말하면, $t\le v$라는 조건하에서, 벡터 $\vec{\delta}$는 C에 속하는 $\vec{\omega}$와 최대 서로 v개의 값이 다른 n-벡터이며, $\vec{\nu}$의 Hamming weight (codeword에서

0이 아닌 값의 coordinate의 개수)는 v와 같거나 작을 수 있다는 것을 알 수 있다. 또한 $\vec{\delta}$ 의 에러 범위(e)가 $e \le n-(n-2v)/2 = v$이기 때문에 Generalized Reed-Solomon Code의 성질을 만족하게 된다. 그러므로 시스템 관리자는 Berlekamp-Welch algorithm을 사용하여 유일한 해 $\vec{\omega}$를 구할 수 있다. 즉, 시스템 관리자는 주어진 불법 디코더를 이용하여 $\vec{\delta}$ 과 $\vec{\omega}$를 계산하고 벡터 $\vec{\nu}$를 구할 수 있다. 공모자 추적 알고리즘을 간략하게 나타내면 다음과 같다.

- 입력 : 불법 디코더 $\vec{K}$와 $z_1, \ldots, z_n$값.
  ($\vec{K} = \sum_{l=1}^{t} \mu_l \vec{k}_{u_l}$, $u_1, \cdots, u_t \subseteq 1, 2, \cdots, n$, t : 공모자 수 $(t < 2v+2)$, n : 시스템의 사용자 수)
- 출력 : 벡터 $\vec{\nu} = \langle \nu_1, \cdots, \nu_n \rangle$
  ($\nu_{u_l} = \mu_l$ for $l = 1, \cdots, t$ 그리고 $\nu_i = 0$ for all $i \in 1, \cdots, n - u_1, \cdots, u_t$)

5.6 심사 단계

시스템 관리자는 벡터 $\vec{\nu}$, 불법 디코더 $\vec{K}$, 사용자의 $a^R_{u_1}, \cdots, a^R_{u_t}$값과 commitment C_{u_1}; $Sig_{sk_{u_1}}(C_{u_1}), \cdots, C_{u_t}$; $Sig_{sk_{u_t}}(C_{u_t})$를 심사관에게 보낸다. 그 후 심사관은 다음의 식을 이용해서 시스템 관리자가 보내 준 값이 타당한지를 확인한다.

$$\prod_{l=1}^{t} (C_{u_l} g^{a^R_{u_l}})^{\nu_{u_l}} = ? g^{\vec{K}}$$

만약 이 과정을 통과한다면, 심사관은 불법 디코더 $\vec{K}$를 생성하는데 사용자 $u_1, \cdots, u_t$가 참여했다는 것을 확신한다.

6. 증명 프로토콜

[10]의 논문에서는 공개된 값 $y = g^{-x}$를 만족하는 $x = -\log_g y$를 안다는 것을 증명하는 프로토콜을 제안했다. 이 프로토콜에서 증명자가 자신이 비밀값 x에 대하여 알고 있다는 것을 검증자에게 증명하는 과정은 다음과 같다. 증명자는 랜덤하게 $r \in Z_q$를 선택하고 $t = g^r$을 생성한다. 생성한 t 값을 검증자에게 보낸다. 검증자는 랜덤하게 $c \in Z_q$를 선택하여 그것을 증명자에게 보낸다. 증명자는 $s = r + cx \mod q$를 계산하여 검증자에게 보낸다. 만약 $g^s y^c = t$을 만족한다면, 검증자는 수락한다. 즉 , 증명자가 x값을 안다고 믿을 수 있다.

Ⅲ. 기존 기법의 문제점과 익명성 보장하는 새로운 스킴 제안

본 장에서는 [9]에 제안된 기법이 익명성을 보장하지 못하다는 것을 보이고 신뢰기관 (TA)을 이용하여 익명성을 보장하는 스킴을 제안한다.

1. [9]의 문제점

[9]의 스킴은 Kurosawa와 Desmedt가 제안한 [16]의 대칭 공모자 추적 기법을 비대칭 공모자 추적 기법으로 변형한 것이다. [9]의 저자들는 비대칭으로 변형시키기 위해서 OT^1_2[12]를 사용하였다. OT^1_2는 송신자가 두개의 비밀값을 전송하였을 때 수신자가 두개의 비밀값 중 어떤 비밀값을 얻었는지를 송신자가 모르게 하는 기법이기 때문에 대칭 기법을 비대칭으로 변형하는데 많이 이용되는 방법이다. 따라서 논문 [9]에서 데이터 제공자가 OT^1_2 프로토콜로 전달되는 두개의 비밀키 중 사용자가 어떤키를 복호화 키로 얻었는지를 알아 낼 수 없다는 것으로 비대칭의 성질을 만족 시킨다. 또한 저자들은 공모자 추적과 익명성을 제공하기 위해서 time-lock puzzles[7]을 사용한다. 사용자는 time-lock puzzles을 이용하여 자신의 신원을 T시간 동안 알 수 없게 할 수 있으며, T시간 이후에는 자신의 신원을 복호화 할 수 있게 함으로써 사용자가 공모에 참여하였을 경우 사용자의 신원을 알아 낼 수 있도록 한다.

하지만 데이터 제공자가 일정 시간 이후에는 공모에 참여하지 않은 정직한 사용자의 time-lock-puzzle를 복호화 하여 신원을 알아 낼 수 있기 때문에 익명성을 제공하지 못한다. 즉, time-lock-puzzles은 익명성을 보장하지 못한다.

2. 익명성을 보장하는 스킴 제안

익명성을 보장하는 스킴은 신뢰 기관 (TA)과 증명 프로토콜[10], 게시판을 이용한다.

2.1 신뢰기관

사용자가 신뢰 기관과 등록 절차를 실행한다. 자신이 사용할 비밀 값과 신원 정보를 신뢰 기관에 보낸다. 이 과정을 통해서 불법 디코더가 발견되었을 때 공모에 참여한 사용자의 신원을 알아낼 수 있다.

2.3 증명 프로토콜

사용자가 데이터 제공자에게 제시한 값이 게시판에서 무작위로 뽑은 것이 아니라 사용자가 실제로 생성한 것이라는 것을 확인하는 과정에 사용된다. 즉, 사용자 인증 과정에 필요한 방법이다.

2.3 게시판

사용자가 신뢰 기관에 제시한 값을 게시판에 공개함으로써 데이터 제공자는 그 값을 통해 사용자가 신뢰기관에 자신의 신원 정보를 등록했다는 것을 알 수 있고, 사용자가 데이터 제공자에게 제시한 값이 사용자가 데이터 복호화 키를 얻기 위해 사용한 값이라는 것을 확인 가능하게 한다.

3. 개괄적인 스킴 설명

제안하는 익명성을 보장하는 스킴은 기존 [9]의 기법에 신뢰기관 초기화 단계를 추가하고, 키 생성 과정에 익명성을 제공하도록 설계되었다.

3.1 초기화 단계

- 단계 1. 신뢰 기관은 암호화용 키 쌍 (ek_A, dk_A)를 생성한다. ek_A는 TA의 암호화키이고, dk_A는 복호화 키이다. 또한 신뢰 기관은 G_q의 생성자 g를 생성한다. 신뢰 기관 공개키는 (g, ek_A)이다.
- 단계 2. 사용자는 서명에 사용할 서명키 쌍 (sk_u, vk_u)를 생성한다. sk_u는 사용자의 서명키이고 vk_u는 서명 확인용 키이다. 또한 사용자는 랜덤하게 $x \in Z_q$를 선택하고 $PI = g^x$를 계산한다. 그리고 PI에 서명하고, 이것을 신뢰 기관의 공개키로 암호화한다. PI는 익명성을 보장하기 위해 사용되는 값이며 pseudo-identity라고 정의한다. 사용자는 임의의 비밀 값 S를 생성하고 그것을 자신의 서명키로 서명하고 신뢰 기관의 공개키로 암호화 한다. 사용자는 $E_{ek_A}(Sig_{sk_u}(PI), Sig_{sk_u}(S))$와 자신의 신원 ID을 신뢰 기관에 보낸다.
- 단계 3. 신뢰기관은 서명을 통해서 사용자의 PI, 비밀값 S을 인증하고, (PI, S)값을 게시판에 공고한다. 그 후 자신의 데이터 베이스에 (ID, $Sig_{sk_u}(PI)$, $Sig_{sk_u}(S)$)을 저장한다. 이 과정을 통해서 신뢰 기관은 데이터 제공자가 자신과 거래하는 사용자가 신뢰 기관에 등록한 사용자라는 것을 확인할 수 있는 PI를 게시판에 공고하며, 이후 공모자 추적을 할 경우 신뢰기관은 이 값을 이용해서 공모자 신원을 복원 할 수 있다.

3.2. 키 생성 단계 (증명 프로토콜 사용)

- 단계 1. 사용자는 $r \in Z_q$을 선택하고 $t = g^r$를 계산하여 (PI, t)를 데이터 제공자에게 보낸다.
- 단계 2. 데이터 제공자는 사용자가 보낸 값을 게시판에서 확인하고 PI을 찾아낸다. 게시판에 PI이 존재하는 경우 데이터 제공자는 사용자가 신뢰 기관에 인증을 받은 것이라는 것을 확신할 수 있다. 그리고 랜덤하게 $c \in Z_q$를 선택하여 사용자에게 보낸다. 이 과정은 사용자가 자신이 생성한 pseudo identity을 제시한 것이라는 것을 확인하는 증명 프로토콜을 실행하기 위해 필요하다.
- 단계 3. 사용자는 $s = r - cx \mod q$를 계산하여 데이터 제공자에게 보낸다.
- 단계 4. 데이터 제공자는 사용자가 보낸 값이 $g^s(\Pi)^c = t$를 만족하는지를 확인한다. 이 과정을 통해서 사용자 인증을 한다. 이 식을 만족하는 경우 사용자가 신뢰기관에 보낸 값 PI에 대한 x를 알고 있다는 것을 확신할 수 있다. 즉, 신뢰기관에 인증 받은 사용자라는 것을 알 수 있다. 그 후 데이터 제공자는 암호화된 데이터를 복호화 할 수 있는 복호화 키를 전송한다. 식을 만족하지 못한다면 프로토콜 수행을 중단한다.

표 1. 익명성을 보장하는 방법의 개괄적인 프로토콜 설명

	사용자 (sk_u, vk_u)	전송 데이터	신뢰 기관 $(dk_A;(g, ek_A))$
초기화	$PI = g^x$생성 비밀 값 S 생성	$E_{ek_A}(Sig_{sk_u}(PI), sig_{sk_u}(S))$, ID →	$E_{ek_A}(Sig_{sk_u}(PI), sig_{sk_u}(S))$ 게시판에 게시 (ID, $Sig_{sk_u}(PI)$, $Sig_{sk_u}(S)$) 저장
	사용자		데이터제공자
키생성	$t = g^r$ 생성	(PI, t) →	PI 확인, (PI, S)획득
		c ←	$c \in Z_q$
	$s = r - cx \bmod q$	s →	
			$g^s(II)^c = ? t$
		$OT_2^1(S)$ ←	컨텐츠 복호화 키 $OT_2^1(S)$생성
	복호화 키 획득		

3.3. 익명성

본 논문에서 다루는 익명성이란 사용자가 데이터 제공자와의 키 생성 과정에서 데이터 제공자가 사용자의 신원을 알 수 없어야 하며, 그 이후에 동일 사용자와 동일 데이터 제공자가 다시 거래를 하더라도 데이터 제공자는 사용자의 신원을 알 수 없을 뿐만 아니라 사용자에 대한 취미, 관심사 등에 대한 어떤 정보도 알 수 없어야 한다. 단, 신뢰기관은 사용자의 신원정보를 알 수 있으며, 데이터 제공자와 공모하지 않는다.

그 이후 사용자가 데이터 제공자로부터 컨텐츠를 얻고자 한다면 사용자가 개인키 생성하기 위해서 기존의 PI 값을 사용한다면 데이터 제공자가 사용자의 컨텐츠 구입에 대한 정보를 얻을 수 있다. 이런 정보의 유출을 막기 위해서 하나의 세션이 끝 난후 사용자는 다음 과정을 실행해야 한다.

- 단계 1. 사용자는 랜덤하게 $x' \in Z_q$를 선택하여 새로운 $PI' = g^{x'}$값과 비밀값 S'를 생성하여 $E_{ek_A}(Sig_{sk_u}(II'), Sig_{sk_u}(S'))$와 ID를 신뢰 기관에 보낸다. 신뢰 기관은 받은 값에 대한 인증하고 값을 저장하고 게시판에 (PI', S')을 게시한다.
- 단계 2. 만약 새로운 세션이 시작한다면, 사용자는 $r' \in Z_q$선택하고 $t = g^{r'}$계산하여 (PI', t')를 데이터 제공자에게 보낸다. 이 과정에서 동일 사용자가 데이터 제공자에게 데이터를 전송 받고자 하더라도 데이터 제공자는 이 사용자의 신원을 알 수 없을 뿐만 아니라, 이 사용자에 대한 어떤 정보도 얻을 수 없다. 즉, 이 과정에서도 사용자에 대한 어떤 정보도 노출 되지 않는다.

3.4. 익명성 회복 단계

불법 디코더가 발견되었을 때, 데이터 제공자는 공모자에 참여한 사용자의 PI값을 신뢰 기관에 보낸다. 신뢰기관은 PI값에 대응되는 사용자의 신원을 복원하여 데이터 제공자에게 신원을 제공한다. 이런 방법으로 키 생성 과정에서 사용자가 실제 신원이 아닌 PI값을 제시하여 그 이후 PI값에 대응되는 사용자의 신원 정보를 얻어 낼 수 있다. 위의 표 1에서 익명성을 보장하는 방법을 개괄적으로 설명한다.

Ⅳ. 익명성 보장하는 비대칭 공개키 공모자 추적 기법

본 장에서는 본 논문의 2장 5절에서 설명한 비대칭 공개키 공모자 추적 기법에 제안한 익명성을 보장하는 스킴을 이용하여 익명성을 보장하는 공모자 추적 기법을 제시한다(본 장에서는 3장에서 설명한 익명성을 보장하는 스킴을 간략히 설명하고자 한다. 구체적인 것은 3장을 참고하라). 이 공모자 추적 기법은 기존의 비대칭 공개키 공모자 추적 기법의 여섯 단계를 수행한다. 본 장에서는 동일하게 수행하는 부분인 암호화 • 복호화 단계를 제외하고 변형되는 나

머지 단계에 대해서 설명한다. 아래 표 2에서 초기화 단계를 보여준다.

1. 초기화 단계

- 단계 1. 신뢰 기관은 암호화용 키쌍과 생성자를 생성한다. 즉, 신뢰 기관의 공개키는 (g, ek_A) 이다.
- 단계 2. 사용자는 서명키 쌍과 익명성 보장하기 위한 PI (pseudo-identity)을 생성한다. 또한 사용자는 Z_q에서 α_u^C를 랜덤하게 선택하고 commitment $\langle C_u = g^{\alpha_u^C}, Sig_{sk_u}(C_u)\rangle$)을 생성하고 신뢰기관의 공개키로 암호화 한다. 그 후 사용자는 자신의 신원 ID, $E_{ek_A}(Sig_{sk_u}(PI)$, $\langle C_u, Sig_{sk_u}(C_u)\rangle)$을 신뢰 기관에 보낸다. 신뢰기관이 저장한 commitment는 사용자가 공모자로 추정되었을 때, 사용자가 부인 할 수 없는 증거이다.
- 단계 3. 신뢰 기관은 서명을 통해서 사용자의 값을 인증하고, PI값을 게시판에 공고한다. 그 후 자신의 데이터 베이스에 (ID, $Sig_{sk_u}(PI)$, $\langle C_u, Sig_{sk_u}(C_u)\rangle$)을 저장한다.

2. 참여 단계

기존에 제안된 비대칭 공개키 공모자 추적 스킴의 시스템 관리자의 초기화 단계를 시스템 관리자의 초기화단계와 참여 단계로 변형하였다. 시스템 관리자의 초기화 단계는 시스템의 공개키를 생성하고 사용자들의 개인키를 생성하는데 필요한 값을 생성한다. 참여 단계에서는 사용자가 시스템에서 사용할 수 있는 개인키를 획득한다.

2.1 시스템 관리자 초기화 단계

시스템 관리자는 $Q_1(x)$와 b을 생성하여 비밀로 유지하고 $Q(x, y) = Q_1(x) + by$를 계산하고, 공개키 $\langle y, h_0, \cdots, h_{2v}, h'\rangle$을 생성하여 공개한다.

2.2 참여 단계

- 단계 1. 사용자는 $r \in Z_q$을 선택하고 $t = g^r$를 계산하여 (PI, t)를 데이터 제공자에게 보낸다. 이 과정은 데이터 제공자에게 자신이 신뢰 기관에 등록한 사용자라는 것을 알리기 위한 것이다.
- 단계 2. 시스템 관리자는 사용자가 보낸 값을 게시판에서 확인하고 랜덤하게 $c \in Z_q$를 선택하고 Z_q에서 랜덤하게 z_u, α_u^R을 선택하여 사용자에게 c, z_u, $OPE(\alpha_u^R)$보낸다.
- 단계 3. 사용자는 $s = r - cx \mod q$를 계산하고 $OPE(\alpha_u = \alpha_u^R + \alpha_u^C)$를 계산한다. 사용자는 s를 생성하여 시스템 관리자에게 보낸다. 이 과정을 통해서 데이터 제공자는 자신과 거래하고자 하는 사용자가 실제 PI 값을 생성하였다는 것을 확인 할 수 있는 s값을 얻게 된다.
- 단계 4. 시스템 관리자는 사용자가 보낸 값이 $g^s(PI)^c = t$를 만족하는지를 확인한다. 이 과정을 통해서 사용자를 인증을 한다. 이 식을 만족하는 경우 시스템 관리자는 사용자와 OT_2^1 프로토콜을 m번 실행하여 얻은 값에서 자신이 $OPE(\alpha_u^R)$을 생성하는데 사용한 랜덤 값 들을 제거하여 $OPE(Q(z_u, \alpha_u))$을 생성하여 $OPE(Q(z_u, \alpha_u))$을 사용자에게 전송한다. 사용자는 $OPE(Q(z_u, \alpha_u))$에서 랜덤 값 들을 제거하여 개인키를 획득한다. 식을 만족하지 못한다면 프로토콜 수행을 중단한다. 뒤의 표 3은 사용자 참여 단계를 보여준다.

또한 개인키를 획득한 사용자는 자신이 받은 키가 암호화된 컨텐츠를 복호화 할 수 있는 정당한 키 인지를 시스템의 공개키를 이용하여 혼자서 확인가능하

표 2. 초기화 단계

사용자 (sk_u, vk_u)	전송 데이터	신뢰 기관 $(dk_A; (g, ek_A))$
α_u^C 선택, $PI = g^x$생성	ID, $E_{ek_A}(S_{sk_u}(PI)$	PI 게시판에 게시. (ID, $Sig_{sk_u}(PI)$,
$\langle C_u = g^{\alpha_u^C}, Sig_{sk_u}(C_u)\rangle$생성	$\langle C_u = g^{\alpha_u^C}, Sig_{sk_u}(C_u)\rangle$)	$\langle C_u = g^{\alpha_u^C}, Sig_{sk_u}(C_u)\rangle$)저장

표 3. 사용자 참여 단계

	사용자	전송 데이터	시스템 관리자
참여단계	$t=g^{r}$ 생성	(PI, t) →	PI 확인, 획득
		c, z_u, $OPE(\alpha_u^R)$ ←	$c\in Z_q, z_u$ 선택, $OPE(\alpha_u^R)$ 생성
	$s=r-cx \bmod q$ $OPE(\alpha_u=\alpha_u^R+\alpha_u^C)$생성	s, $OT_2^1(OPE(\alpha_u))$ →	OT_2^1 공개키 생성
			확인 $g^{s}(PI)^{c}=^{?}t$
		$OPE(Q(z_u, \alpha_u))$ ←	$OPE(Q(z_u, \alpha_u))$생성
	$\vec{k}$ 획득		

다. 사용자는 개인키 $\vec{k_u}$와 시스템 공개키 $\langle y, h_0, \cdots, h_{2v}, h' \rangle$을 이용하여 다음 식을 만족하면 생성한 개인키는 정당한 키이다.

$$(h_0)^{Q(z_u,\alpha_u)}(h_1)^{z_u}\cdots(h_{2v})^{z_u^{2v}}(h')^{\alpha_u}=g^{\alpha_0}=^{?}y$$

3. 공모자 추적과 심사 단계

불법디코더($\vec{K}=\sum_{l=1}^{t}\mu_l\vec{k}_{u_l}$, $u_1, \cdots, u_t \subseteq 1, \cdots, n$)가 발견되었을 때, 시스템 관리자는 추적 알고리즘에 불법 디코더와 $z_1, \ldots, z_n$ 값을 입력값으로 하여 출력값 벡터 $\vec{\nu}=\langle \nu_1, \cdots, \nu_n \rangle$ 를 얻는다. (불법 디코더를 생성한 공모자를 추적하는 과정은 2장 5.5를 통해 알 수 있다.) 그 후 시스템 관리자는 벡터 값에 해당하는 pseudo-identity $PI_1, \cdots, PI_t$, $\vec{\nu}$, $\vec{K}$, $\alpha_{u_1}^R, \cdots, \alpha_{u_t}^R$를 신뢰 기관에 보낸다. 신뢰 기관은 PI_i , $i=1, \ldots, t$ 에 해당하는 commitment를 찾아낸다. 그리고 $\prod_{l=1}^{t}(C_{u_l}g^{\alpha_{u_l}^R})^{\nu_{u_l}}=^{?}g^{\vec{K}}$ 을 만족하는지 확인한다. 이 식을 만족한다면 시스템 관리자가 공모자라고 주장하는 사용자들이 공모에 참여한 것이라는 것을 확신 할 수 있다. 이 과정으로 통해서 신뢰기관은 시스템 관리자에게 PI_i, $i=\{1,\ldots,t\}$에 해당하는 공모자의 신원정보를 준다.

4. 익명성

제안하는 익명성을 보장하는 공개키 공모자 추적 기법은 시스템 관리자로부터 사용자가 데이터를 복구하는 키를 얻는 과정에서 신분이 노출되지 않아서 사용자가 구매하고자 하는 데이터에 대한 정보를 얻을 수 없게 된다. 그래서 사용자의 개인 정보가 누출되지 않으며, 하나의 세션이 끝나고 새로운 세션에서 동일 시스템 관리자로부터 사용자가 데이터를 복구하는 키를 얻더라도 그 사용자가 예전에 거래했던 사용자라는 것을 알 수 없다. 그리하여 사용자의 관심 분야, 취미 생활 등의 개인적인 정보가 유출되지 않게 된다.(단, 신뢰기관은 사용자의 신원정보를 알 수 있으며, 시스템 관리자와 공모하지 않는다.)

그러므로 참여 세션 이후에 사용자가 다시 이 시스템에서 컨텐츠를 얻고자 한다면 사용자가 개인키 생성을 위해서 기존의 PI값을 사용한다면 시스템 관리자 혹은 데이터 제공자가 사용자의 컨텐츠 구입에 대한 정보를 얻을 수 있다. 이런 정보의 노출을 막기 위해서 하나의 세션이 끝 난후 사용자는 새롭게 PI 값과 α_u^C 값을 생성하여 4.1의 초기화 단계를 실행한다. 만약 새로운 세션을 시작한다면 사용자는 새로운 t' 값을 생성하여 시스템 관리자 (데이터 제공자)에게 (PI, t')을 보낸다(구체적인 과정은 본 논문 3장 3.3을 참고하라). 이 과정을 통해서 시스템 관리자 (데이터 제공자)는 동일 사용자와 거래하더라도 새로운 pseudo-identity 와 t'를 제시하기 때문에 사용자의 신원을 알 수 없을 뿐만 아니라 사용자의 어떤 정보도 얻을 수 없다.

5. 안전성

제안하는 익명성을 보장하는 비대칭 공개키 공모자 추적 기법은 기존의 기법이 가지고 있던 모든 안전성을 유지한다. 하지만 제안하는 익명성을 보장하

는 기법에서는 공격자의 두 가지 공격을 예상할 수 있다.

하나, 공격자가 게시판의 게시되어 있는 값 (PI, t)값을 모아 두었다가 공격자에게 보낼 수 있다. 이런 경우 시스템 관리자는 사용자와 실행하여야 하는 증명 프로토콜 과정을 통해서 자신과 거래하는 사용자가 정당한 사용자가 아니라는 것을 알 수 있다. 즉, 공격자가 시스템 관리자가 보내는 c값에 대한 s값을 수식 $g^s(ID)^c = t$을 만족하도록 생성할 수 없으므로 pseudo-identity값을 생성하는데 쓰인 x 값을 아는 사람만이 개인키를 얻을 수 있다. 그러므로 이 기법은 실제 신원이 아닌 pseudo-identity를 이용하여 익명성을 제공할 수 있다. 또한, 정당한 사용자만이 개인키를 얻을 수 있다.

둘, 공격자가 사용자가 전송한 s값을 모아 두었다가 새로 생성한 $OPE(\alpha_u = \alpha_u^R + \alpha_u^C)$을 시스템 관리자에게 전송한다. 하지만 사용자가 생성하는 s값에 관련된 비밀값은 세션 마다 변경되기 때문에 $g^s(PI)^c = t$을 만족 시킬 수 없다. 그러므로 공격자는 시스템 관리자로부터 정당한 키를 전송 받을 수 없다.

이와 같이 익명성을 보장하는 기법은 키를 전송받는 과정에서 사용자가 정당한지를 확인 할 수 있는 방법을 사용하기 때문에 안전하다.

V. 익명성과 효율성

본 논문에서 새롭게 제안하는 익명성을 보장하는 방법은 기존의 방법보다 효율적이며 실질적인 익명성을 보장한다. 이 점에 대해 구체적으로 알아보기 위해서 제안하는 익명성을 보장하는 방법을 사용하는 기법과 Time-Lock puzzles, 전자 화폐를 사용하는 기존의 기법들과 익명성과 효율성에 대하여 비교해 보도록 하겠다. 익명성 측면에서는 세 가지 방법에 대해서 비교할 것이며, 효율성 측면에서는 익명성을 보장하는 기존 전자 화폐와 제안하는 방법에 대해서 비교할 것이다.

- 익명성 :

- Time-Lock puzzles을 사용하는 경우, 데이터 제공자는 키 생성 과정에서 사용자의 신원을 알 수 없다. 그러나 일정 시간이 지난 후에는 사용자의 신원 정보를 알 수 있다. 또한 하나의 세션 이후에 데이터 제공자와 사용자가 키 생성 과정을 실행할 경우 일정 시간 후에 데이터 제공자는 사용자의 신원 정보를 알 수 있다. 즉, 익명성을 보장하지 못한다.
- 전자 화폐를 사용하는 경우, 사용자는 신뢰기관과의 인증 단계를 통해 전자 화폐를 발행 받는다. 발행 받은 전자 화폐는 사용자의 신원 정보를 나타내지 않기 때문에 익명성을 보장할 수 있다.
- 익명성을 보장하는 방법을 사용하는 경우, 데이터 제공자는 키 생성 과정에서도 사용자의 신원 정보를 알 수 없으며, 일정 시간이 지난 후에도 사용자의 신원에 대한 어떤 정보도 얻을 수 없다. 또한 하나의 세션 이후에 데이터 제공자와 사용자가 키 생성 과정을 실행하더라도 사용자의 신원 정보뿐만 아니라 사용자의 어떤 정보도 데이터 제공자에게 유출되지 않는다. 즉, 익명성을 보장한다.

-효율성 :

- 전자화폐를 사용하는 경우, 익명성을 보장하기 위해 사용하는 전자 화폐를 생성하는 과정에서 신뢰 기관과 사용자사이에 은닉 서명을 사용하기 때문에 전송 되는 메시지의 길이도 길며, 서로 주고받는 전송 횟수도 많다. 또한 사용자나 신뢰기관이 행해야 하는 연산양도 많다.
- 익명성을 보장하는 방법을 사용하는 경우, 처음에 사용자와 신뢰 기관 간에 초기화를 실행한다. 하지만 이 과정에서 전송횟수는 한번이다. 그 이후의 키 생성과정은 두 방법 모두 동일하다. 즉, 제안하는 방법은 익명성을 보장하는 전자 화폐를 사용하는 것과 달리 신뢰 기관과 사용자 사이에 단 한번의 메시지 전송을 실행하면 된다. 그리고 데이터 제공자와 사용자 간에 새로운 세션을 시작 할 경우에는 두 가지 방법 모두 새로운 값을 생성하여야 한다는 점에서 동일하다. 즉, 전자화폐를 사용하는 경우 새롭게 전자화폐를 생성하고 익명성을 보장하는 방법의 경우에는 (pseudo-identity, t)를 생성해야 한다. 그러므로 메시지 전송과 메시지 길이 측면에서 기존 익명성을 보장하는 방법 보다 제안하는 익명성을 보장하는 방법이 더 효율적이다.

위와 같이 새롭게 제안하는 익명성을 보장하는 방

법은 익명성 측면에서는 전자화폐와 동일한 성질을 갖는다. 반면 효율성 측면에서 전자화폐는 사용자와 신뢰 기관 사이에 메시지의 전송 횟수도 많으며, 계산해야 하는 연산양도 많다. 그러므로 새롭게 제안하는 방법이 익명성을 보장하고 효율적이다.

Ⅳ. 결 론

본 논문에서는 기존에 제안된 [9]의 논문에서 익명성을 보장하기 위해 사용한 Time-lock puzzles이 실질적으로 익명성을 보장하지 못한다는 것을 본 논문 3장 1절에서 보였다. 즉, 정직하지 않은 데이터 제공자는 공모에 참여하지도 않은 정직한 사용자의 신원을 알아 낼 수 있다. 따라서 본 논문에서는 증명 프로토콜[10], 게시판, 신뢰 기관을 사용하여 익명성을 보장할 수 있는 스킴을 제안했다. 그리고 제안하는 익명성을 보장하는 스킴은 기존의 익명성을 보장하려고 사용한 Time-Lock puzzles를 사용하는 것과 달리 실질적인 익명성을 보장하며, 일반적으로 익명성 보장하기 위해 사용되었던 전자화폐 보다 더 효율적이다. 또한 본 논문 2장 5절에서 설명한 기존의 효율적인 공개키 공모자 추적 기법에 제안한 익명성을 보장하는 스킴을 사용하여 익명성을 보장하는 공개키 공모자 기법을 처음으로 제안했다.

참 고 문 헌

[1] B. Chor, A. Fait and M. Naor Tracing traitors, Advances in cryptology CRT YPO'94, LNCS 839, Springer-Verlag, pp.257-270. 1994.

[2] B. Pfitzmann. Trials of traced traitors, Infor-mation Hiding'96, LNCS 1174, Springer-Verlag, pp.49-64, 1996.

[3] D. Boneh and M. Franklin. An efficient public key traitor tracing scheme, Advances in cryptology, CRYPTO'99, LNCS 1666, Spr-inger-Verlag, pp. 338-353, 1999.

[4] S. Brands, Untraceable off-line cash in Wallets with observers, Advances in crypt- ology, CRYPTO'93, LNCS 0773, Springer-Verlag, pp.302-318, 1993.

[5] D. Chaum, A. Fiat and M. Naor, Untraceable Electronic Cash, Advances in cryptology CRYPTO'88, LNCS403, Springer-Verlag, pp.319-327, 1990.

[6] A. Lysyanskaya, R. L. Rivest and A. Sahia, and S. Wolf, Pseudonym systems, http://theory.lcs.mit.edu/anna/lrsw99.ps, 1999.

[7] R. Rivest, A. Shamir, and D. Wagner, Time-Lock Puzzles and Timed-Released Crypto, LCS Techinical Mono MIT/LCS/TR-684, 1996.

[8] D. Chaum, Blind Signatures for Untraceable Payments, Advances in cryptology, CRYPTO'82, Plenum Press, pp. 199-203, 1982.

[9] E. Magkos, P. Kotzanikolaou and V. Chr-issikopoulod, An asymmetric traceability scheme for copyright protection without trust assumptions, EC_Web '2001, LNCS 2115, Springer-Verlag, pp. 186-195, 2001.

[10] C. Schnorr, Efficient identification and signatures for smart cards, Advancen in cry-ptology, CRYPTO'89, LNCS 435, Springer-Verlag, pp.239- 251, 1989.

[11] M. Naor and B. Pinkas, Oblivious Transfer and Polynomial Evaluation, the 31th ACM Conference on Computer and Communication Security, ACM, 1999.

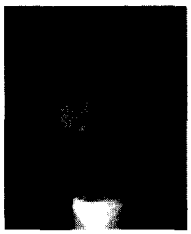
[12] M. Bellare and S. Micali, Non- Interactive Oblivious Transfer and Applications, Advances in cryptology, CRYPTO '89, LNCS 435, Spr-inger-Verlag, pp.544-557, 1990.

[13] A. Kiayias and M. Yung, Breaking and repairing asymmetric public-key traitor tracing, ACM Conference on Computer and Communication Security, ACM, 2002.

[14] M. Naor and B. Pinkas, Efficieint Trace and Revoke Schemes, In the Proceedings of Financial Crypto'2000, Anguilla, 2000.

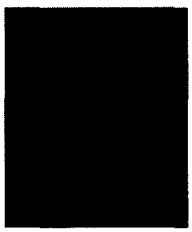
[15] E. Berlekamp and L. Welch, Error correction for algebraic block codes, 1986.
[16] K. Kurosawa and Y. Desmedt, Optimum traitor tracing and asymmetric scheme, Advances in cryptology EUROCYRYPT '98, LNCS 1403, Springer-Verlag, pp.145-157, 1998.

〈著者紹介〉



최 은 영 (Eun Young Choi) 학생회원
2001년 8월 : 고려대학교 수학과 학사
2003년 8월 : 고려대학교 정보보호대학원 공학석사
2004년 3월 : 고려대학교 정보보호대학원 박사과정
〈관심분야〉 암호 이론, 정보보호 프로토콜

이 동 훈 (Dong Hoon Lee) 종신회원
1983년 8월 : 고려대학교 경제학사
1987년 12월 : Oklahoma University 전산학 석사
1992년 5월 : Oklahoma University 전산학 박사
1993년 3월~1997년 2월 : 고려대학교 전산학과 조교수
1997년 3월~2001년 2월 : 고려대학교 전산학과 부교수
2001년 3월~현재 : 고려대학교 정보보호대학원 교수
〈관심분야〉 정보보호 프로토콜, 계산이론, 암호이론, 네트워크 보안



홍 도 원 (Dowon Hong)
1994년 2월 : 고려대학교 이과대학 수학과(학사)
1996년 2월 : 고려대학교 수학과(석사)
2000년 2월 : 고려대학교 수학과(박사)
2000년 4월~현재 : 한국전자통신연구원 팀장
〈관심분야〉 암호 이론, 정보보호 이론, 이동통신 정보보호