情報保護學會論文誌
第 22 卷 第 3 號, 2012. 6

소셜 네트워크에 적합한 효율적인 프라이버시 보호 데이터 공유 기법*

전 두 현,† 천 지 영, 정 익 래‡
고려대학교 정보보호대학원

An efficient privacy-preserving data sharing scheme in social network*

Doo Hyun Jeon,† Ji Young Chun, Ik Rae Jeong‡
Graduate School of information Security, Korea University

요 약

소셜 네트워크 서비스는 실시간 정보 공유의 새로운 매개체로 각광을 받고 있다. 하지만 소셜 네트워크를 통해 공유되는 정보는 사용자의 신분이나 생활 패턴 등을 노출시킬 수 있는 민감한 정보가 포함되어 있기 때문에 사용자의 프라이버시 침해가 빈번히 발생한다. 이러한 문제를 해결하기 위해 소셜 네트워크 환경에서 프라이버시를 보호하는 데이터 공유 기법에 대한 연구가 활발히 진행되고 있다. 본 논문에서는 소셜 네트워크에 적합한 프라이버시 보존 데이터 공유 기법을 제안한다. 제안하는 기법은 암호화된 데이터에서 검색자가 선택한 키워드가 모두 포함되어 있는 데이터를 검색할 수 있으며, 스토리지 서버에 대한 접근 권한을 부여받은 사용자는 누구나 데이터를 저장하고 검색할 수 있다. 또한 동적 환경인 소셜 네트워크의 특징에 적합하도록 효율적인 가입/탈퇴 기능을 제공한다.

ABSTRACT

A social network service(SNS) is gaining popularity as a new real-time information sharing mechanism. However, the user's privacy infringement is occurred frequently because the information that is shared through a social network include the private information such as user's identity or lifestyle patterns. To resolve this problem, the research about privacy preserving data sharing in social network are being proceed actively. In this paper, we proposed the efficient scheme for privacy preserving data sharing in social network. The proposed scheme provides an efficient conjunctive keyword search functionality. And, users who granted access right to storage server can store and search data in storage server. Also,, our scheme provide join/revocation functionality suited to the characteristics of a dynamic social network.

Keywords: Social Networks, Keyword Search, Data Sharing Scheme, Privacy

I. 서 론

최근 스마트폰, 테블릿 PC(tablet PC) 등 모바일 기기의 발달로 인터넷을 자유롭게 사용할 수 있게 되면서 이전보다 더 다양한 정보를 수집하고 활용할 수 있게 되었다. 특히, 트위터(twitter), 페이스북(face-book) 등과 같은 소셜 네트워크 서비스는 포털 사이트를 통해서만 이루어진 정보 검색의 제한된 환경과 달리 실시간의 정보를 쉽게 공유하고 검색할 수 있는 새로운 매개체로 각광을 받고 있다.

실시간 문자 서비스, 멀티미디어 공유, 블로그 서비스 등 다양한 형태의 소셜 네트워크 서비스가 존재하

접수일(2011년 9월 16일), 수정일(2011년 11월 22일), 게재확정일(2011년 11월 22일)
* 이 연구에 참여한 연구자(의 일부)는 '2단계BK21사업'의 지원비를 받았음
† 주저자, jundhlove@korea.ac.kr
‡ 교신저자, irjeong@korea.ac.kr

지만 이들은 모두 등록된 사용자들이 관심 분야에 따라 친분 관계를 형성하고 이를 기반으로 서비스가 제공된다는 공통점을 지닌다. 친분 관계가 형성되어 있는 사용자들은 서로의 데이터 저장 공간에 접근하여 자신의 자료를 저장하거나 관심 있는 정보는 검색하여 정보를 공유할 수 있다. 하지만 소셜 네트워크를 통해 공유되는 정보는 개인 신상 정보, 개인 위치 정보 등과 같은 민감한 정보가 많기 때문에 정보 노출에 따른 개인 프라이버시 침해가 빈번히 발생하고 있다. 특히, 데이터 저장 공간은 소셜 네트워크 서비스 제공자에 의해 제공되는 것이 아닌 신뢰할 수 없는 제 3의 저장 공간 서비스 제공자에 의해 관리되기 때문에 개인 정보의 노출은 사용자들에게 큰 악영향을 끼칠 수 있다. 따라서 저장하는 모든 데이터는 암호화하고 데이터의 효율적인 활용을 위해 암호화된 데이터에서의 검색 기능이 필요하다.

최근 소셜 네트워크에서 형성된 친분 관계를 통한 프라이버시 보존 데이터 공유에 대한 연구가 활발히 진행되고 있다[1-6]. 민감한 정보 노출에 따른 사용자 프라이버시 침해를 방지하기 위해 정당한 사용자만이 데이터에 접근할 수 있도록 접근 권한을 부여해야 한다. 이를 위해 속성 기반 접근 제어 방식[1,3], 역할 기반 접근 제어 방식[2,4]과 거리 기반 접근 제어 방식[5,6]에 대한 연구가 활발히 진행되고 있다. 본 논문에서는 거리 기반 접근 제어 방식을 활용하여 소셜 네트워크에 적합한 프라이버시 보존 데이터 공유 기법을 제안한다. 본 논문을 통한 연구의 공헌은 다음과 같다.

- 본 논문에서 소셜 네트워크를 통해 형성된 친분 관계를 활용하여 서버 관리자가 설정해 놓은 보안 레벨을 만족하는 사용자들이 데이터를 공유하는 기법을 제안한다.
- 서버 관리자는 거리 기반 접근 제어 방식을 활용하여 자신으로부터 일정 거리 이내에 위치한 사용자에게만 서버에 대한 접근 권한을 부여하고, 접근 권한을 부여받은 사용자들은 자유롭게 데이터를 저장, 검색 할 수 있다.
- 데이터는 암호화하여 저장함으로써 사용자의 프라이버시 침해를 방지하고, 암호화된 데이터에서 키워드 검색 기법을 통해 사용자가 선택한 여러 키워드가 모두 포함되어 있는 데이터를 검색(conjunctive keyword search)할 수 있게 한다.
- 사용자들의 친분 관계가 수시로 변하는 소셜 네트워크의 특징에 접합하도록 사용자에게 효율적인 추가/탈퇴 기능을 제공한다. 여기서 사용자 U와 사용자 V 사이에 형성되어 있던 친분 관계를 해제하였을 때, 사용자 U를 기준으로 사용자 V는 탈퇴한 것이고 사용자 V를 탈퇴자라고 한다.

본 논문의 구성은 다음과 같다. 2장, 3장에서 본 논문과 관련된 연구와 배경 지식에 대해 살펴본다. 4장에서 시스템 모델 및 안전성 모델을 정립한 후 5장에서 정립된 안전성 모델에 대해 안전한 기법을 제안하고 분석한다. 마지막으로 6장의 결론을 통해 논문을 마무리한다.

II. 관련 연구

소셜 네트워크에서 민감한 정보의 무분별한 노출을 방지하기 위해 특정 사용자들에게만 데이터 접근 권한을 부여하여 프라이버시를 보존하면서 안전하게 데이터를 공유하기 위한 기법들이 제안되었다[1-6].

문헌 [1]의 경우 속성 기반 암호화 기법을 활용하여 정당한 속성을 갖은 사용자만이 데이터에 접근 할 수 있도록 접근 권한을 부여하였다. 하지만 연산량이 많은 속성 기반 암호 시스템을 사용하였기 때문에 효율성이 떨어지며, 사용자가 탈퇴했을 경우 사용자들에게 분배된 속성에 따라 새로운 속성키를 재분배하여야 하기 때문에 사용자의 추가/탈퇴가 빈번히 일어나는 소셜 네트워크 환경에 적합하지 않다. 이러한 문제를 해결하기 위해 문헌 [3]에서 속성 기반 암호화 기법을 사용하여 효율적인 탈퇴 기능을 제공하는 기법을 제안하였다. 문헌 [2]의 경우 데이터 소유자가 분류한 소그룹에 따라 그룹키를 할당하고 정당한 그룹키를 소유한 사용자만이 소그룹에 분배된 데이터에 접근할 수 있도록 하였다. 또한 브로드캐스트 암호화 기법을 활용하여 사용자가 탈퇴했을 경우 그룹키를 재분배하지 않고 탈퇴자에 대한 접근 제어를 할 수 있는 기능을 제공한다. 하지만 탈퇴자가 탈퇴하기 이전에 저장된 데이터는 계속 이용할 수 있어 부분적인 탈퇴 기능만을 제공한다는 문제가 있다. 문헌 [5]의 경우 사용자들이 근접해 있는 정도에 따라 사용자들에게 신뢰도를 부여하여 민감한 데이터에 대한 효과적인 접근 통제를 할 수 있는 기법을 제안하였다. 문헌 [6]의 경우 사용자들의 친분 관계를 이용하여 일정 범위 이내에 위치

한 사용자의 데이터에 접근할 수 있도록 하기 위한 키 분배 기법을 제안하였다. 하지만 문헌 [5]와 문헌 [6]은 기법을 운영하기 위해 자신이 가지고 있는 정보 외에 다른 사용자들이 가지고 있는 정보도 사용해야하기 때문에 추가적인 정보 노출에 의한 사용자 프라이버시 침해가 발생할 수 있다.

문헌 [7]의 경우 친분 관계를 숨기면서 두 사용자 간의 최단 관계 경로 검색 기능을 제공하는 프라이버시 보존 경로 발견 기법을 제안하였다. 이 기법은 네트워크의 모든 구성원이 동시에 온라인 상태를 유지하지 않더라도 필요한 정보를 전달할 수 있고 이를 통해 서로의 관계 그래프를 노출시키지 않으면서 관계 경로를 찾을 수 있다. 본 논문에서는 문헌 [7]의 프라이버시 보존 경로 발견 기법을 사용자가 보안 레벨을 만족하는지 여부를 확인할 때에 활용한다.

III. 배경 지식

3.1 준 동형 공개키 암호 시스템 (homomorphic public-key system)[8]

3.1.1 구성

이 시스템은 다음의 세 가지 알고리즘으로 구성된다.

- $HPE.keygen(\tau)\rightarrow(PK,SK)$: 주어진 보안 변수 $\tau\in Z^+$에 따라, 튜플 (q_1, q_2, G, G_1, e)를 얻기 위해 $\zeta(\tau)$를 실행한다. 위수가 $n=q_1q_2$인 겹선형군 G의 생성원 g, u를 임의로 선택하고, $h=u^{q_2}$로 설정한다. 이 때, h는 G의 부분군 중 위수가 q_1인 부분군의 생성원이 된다.
  최종적으로 공개키 $PK=(n,G,G_1,e,g,h)$와 비밀키 $SK=(q_1)$를 설정한다.

- $HPE.enc_{PK}(m)\rightarrow C$: 메시지 m의 메시지 공간은 집합 $\{0,1,...,T\}$으로 한다. 단, $T<q_2$이다. 랜덤 값 $r\in Z_n$을 임의로 선택하고, $C=g^mh^r$을 계산하여 메시지 m에 대한 암호문으로 C를 출력한다.
- $HPE.dec_{SK}(C)\rightarrow M$: 암호문 C를 복호화 하기 위해 비밀키 $SK=q_1$을 사용하여 다음을 계산한다.

$$C^{q_1}=(g^mh^r)^{q_1}=(g^{q_1})^m \tag{1}$$

$\hat{g}=g^{q_1}$이라고 할 때 $m=\log_{\hat{g}}C^{q_1}$, $0\le m\le T$이므로 메시지 m을 얻기 위해서 Pollard's lambda 방법을 이용하면 $O(\sqrt{T})$의 시간이 걸린다.

3.1.2 성질

3.1.2.1 검증 가능성(verifiable)

위의 암호화 과정에서 사용한 메시지 m의 크기가 주어진 조건 $0\le m\le T$을 만족하지 않을 경우, 큰 수에 대한 이산 대수 문제의 어려움 때문에 비밀키 $SK=q_1$을 이용한 완벽한 복호화가 불가능하다. 하지만 메시지 m에 대한 암호문 $C=g^mh^r$과 새로운 메시지 m'이 주어졌을 때, $(g^mh^r)^{q_1}=(g^{m'})^{q_1}$을 확인함으로써 $m=m'$인지를 검증할 수 있다. 따라서 메시지의 크기가 큰 경우 비밀키 $SK=q_1$는 검증키로 사용 가능하다.

3.1.2.2 준 동형 성질(homomorphic properties)

메시지, 암호문 쌍 (m_1,C_1)과 (m_2,C_2)이 주어졌을 때, 암호문을 복호화하지 않고 $m_1\pm m_2$와 m_1m_2에 대한 암호문을 다음과 같은 방법으로 생성할 수 있다.

- $$\begin{aligned}&Add(HPE.enc_{pk}(m_1),HPE.enc_{pk}(m_2))\\&=C_1C_2h^r=(g^{m_1}h^{r_1})(g^{m_2}h^{r_2})h^r\\&=g^{m_1+m_2}h^{r_1+r_2+r}=g^{m_1+m_2}h^{r'}\end{aligned} \tag{2}$$

- $$\begin{aligned}&Sub(HPE.enc_{pk}(m_1),HPE.enc_{pk}(m_2))\\&=C_1(C_2)^{-1}h^r=(g^{m_1}h^{r_1})(g^{m_2}h^{r_2})^{-1}h^r\\&=g^{m_1-m_2}h^{r_1-r_2+r}=g^{m_1-m_2}h^{r'}\end{aligned} \tag{3}$$

- $$\begin{aligned}&Mul(HPE.enc_{pk}(m_1),HPE.enc_{pk}(m_2))\\&=e(C_1,C_2)h_1^r=e(g^{m_1}h^{r_1},g^{m_2}h^{r_2})h_1^r\\&=g_1^{m_1m_2}h_1^{m_1r_2+r_2m_1+\alpha q_2r_1r_2+r}=g_1^{m_1m_2}h_1^{r'}\in G_1\end{aligned} \tag{4}$$

3.2 암호화된 다항식에서의 연산

집합 $A=\{k_1,...,k_n\}$의 원소를 근으로 하는 n차 다항식 이 다음과 같이 주어졌을 때, 준 동형 암호화 시스템 (HPE)을 이용하여 다음과 같은 표기법을 정의한다.

- $$\begin{aligned}f_A(x)&=(x-k_1)\cdots(x-k_n)\\&=a_nx^n+\cdots+a_1x+a_0\end{aligned} \tag{5}$$

- $POLY.HPE.enc_{pk}(f_A(x))$
$= (HPE.enc_{pk}(a_n), HPE.enc_{pk}(a_{n-1}), ...$
$..., HPE.enc_{pk}(a_0))$ (6)

- $POLY.HPE.enc_{pk}(k)$
$= (HPE.enc_{pk}(k^n), HPE.enc_{pk}(k^{n-1}), ...$
$..., HPE.enc_{pk}(k^0))$ (7)

- $POLY.HPE.enc_{pk}(k_1 + k_2 + k_3)$
$= (HPE_{pk}(k_1^n + k_2^n + k_3^n),$
$HPE_{pk}(k_1^{n-1} + k_2^{n-1} + k_3^{n-1}), ...$
$..., HPE_{pk}(k_1^0 + k_2^0 + k_3^0))$ (8)

위의 식을 이용하여 암호화된 다항식 $f_A(x)$에 값 k를 입력한 $POLY.HPE.enc_{pk}(f_A(k))$를 준 동형 암호화 시스템의 성질을 이용하여 다음과 같이 계산할 수 있다.

- $POLY.HPE.enc_{pk}(f_A(k))$
$= Add(Mul(HPE.enc_{pk}(a_n), HPE.enc_{pk}(k^n)),$
$..., Mul(HPE.enc_{pk}(a_0), HPE.enc_{pk}(k^0)))$ (9)

- $POLY.HPE.enc_{pk}(f_A(k_1) + f_A(k_2) + f_A(k_3))$
$= Add(Mul(HPE.enc_{pk}(a_n),$
$HPE.enc_{pk}(k_1^n + k_2^n + k_3^n)), ...$
$, Mul(HPE.enc_{pk}(a_0), HPE.enc_{pk}(k_1^0 + k_3^0 + k_3^0))$ (10)

3.3 프라이버시 보존 관계 경로 발견 기법[7]

소셜 네트워크 사용자들의 연결성을 나타내는 관계 그래프는 네트워크를 분석하는데 중요한 도구로 사용될 수 있지만 사용자들 사이의 관계성, 관계 경로 상에 존재하는 사용자들의 정보 등과 같은 민감한 정보가 포함되어 있기 때문에 악용될 경우 사용자들의 프라이버시 침해가 발생할 수 있다. 따라서 사용자 프라이버시 보호를 위해 최소한의 정보만을 노출하는 관계 경로 발견 기법에 대한 연구가 활발히 진행되고 있다.

G. Mezzour 등의 연구[7]에서 프라이버시 보존 관계 경로 발견 기법이 제안되었다. 이 프라이버시 보존 관계 경로 발견 기법을 통해 다른 사용자의 친분 관계나 전체 네트워크 정보를 드러내지 않으면서 두 사용자 사이에 존재하는 최단 경로와 경로의 길이만을 알 수 있다. 제안된 프라이버시 보존 관계 경로 발견 기법은 토큰 전송 알고리즘($Token\_Flooding$)과 경로 발견 알고리즘($Path\_Discovery$)으로 구성되어 있다.

- $Token\_Flooding(u, u_i, d) \rightarrow TR_{u,u_i}$: 사용자 u가 직접적인 친분 관계를 형성하고 있는 사용자 u_i를 통해 관계 경로를 발견하기 위해 자신만의 암호학적 토큰을 최대 토큰 전달 범위 d까지 전달하는 알고리즘이다. 최종적으로 사용자 u와 u_i 사이에 전달된 토큰에 대한 정보를 담고 있는 토큰 트리 TR_{u,u_i}가 출력된다.

- $Path\_Discovery(u, v) \rightarrow \perp \text{ or } (u, u_i, d_i, v)$: 사용자 u와 사용자 v 사이에 존재하는 관계 경로를 찾기 위해 운영되는 알고리즘이다. 이때 사용자 u는 경로의 시작 노드, 사용자 v는 경로의 끝 노드가 된다. 최종적으로 사용자 u는 사용자 v가 자신의 친구 u_i를 통하여 길이 d만큼 떨어져있다는 것을 확인할 수 있다.

IV. 시스템 모델

4.1 시스템 구성 요소

제안하는 프로토콜을 사용하는 시스템은 소셜 네트워크 서비스 제공자, 스토리지 서버, 서버 관리자, 구성원으로 구성되어 있다.

- **소셜 네트워크 서비스 제공자**: 소셜 네트워크 도메인이 안전하게 운영될 수 있도록 암호학적 초기화를 하고 사용자들에게 편리한 서비스를 제공하기 위한 응용 프로그램을 공급하며 서비스에 등록한 사용자들이 편리하게 친분 관계를 형성할 수 있도록 다양한 정보를 제공한다. 일반적으로 서비스에 등록한 사용자는 서비스 제공자를 신뢰하는 것으로 가정한다.

- **스토리지 서버**: 소셜 네트워크에서 생성되는 방대한 양의 데이터를 저장할 수 있도록 저장 공간을 제공하는 제 3의 서비스 공급자이다. 스토리지 서버는 프로토콜에 정상적으로 참여하지만 사용자들이 저장하고 검색하기 위해 전송하는 값을 이용하여 사용자의 프라이버시를 침해할 수 있기 때문에 우리는 서버를 반-신뢰(semi-trust)한다고 가정한다.

- **관리자**: 스토리지 서버에 자신만의 저장 공간을 갖는 소셜 네트워크 서비스 사용자이다. 관리자는 보안 레벨을 설정하여 자신과 친분 관계를 형성하고 있는 친구들뿐만 아니라 보안 레벨을 만족하는 모든 사용자들이 데이터를 공유할 수 있도록 서버를 관리한다.

- **구성원**: 소셜 네트워크 서비스 사용자들과 친분 관계를 형성하고 있는 소셜 네트워크 사용자이다. 자신과 친분 관계를 형성하고 있는 사용자가 운영하는 서버에 자신의 데이터를 업로드하거나 필요한 데이터를 다운로드 할 수 있는 권한을 행사할 수 있다.

4.2 보안 요구 사항

4.2.1 데이터 프라이버시

소셜 네트워크를 통해 공유되는 데이터에는 사용자의 신분이나 생활 패턴, 인맥 관계 등을 노출시킬 수 있는 민감한 정보가 포함되어 있기 때문에 이러한 민감한 데이터의 노출로 인해 사용자 프라이버시 침해가 발생할 수 있다. 따라서 암호화를 통한 데이터의 기밀성을 보장하여 인가되지 않은 사용자에게는 다른 사용자의 프라이버시를 침해할 수 있는 어떠한 정보도 노출되지 않아야 한다. 또한 완벽히 신뢰할 수 없는 제 3의 스토리지 서버를 이용하기 때문에 데이터를 저장하거나 검색하는 과정에서 어쩔 수 없이 노출되는 정보 이외에는 어떠한 정보도 노출되어서는 안 된다.

4.2.2 접근 제어

소셜 네트워크에서 사용자들의 친분 관계는 빈번하게 변화한다. 따라서 새롭게 인맥 관계가 형성된 사용자에게는 데이터를 공유할 수 있도록 데이터 접근 권한을 부여해야 하고, 탈퇴된 사용자에게는 접근 권한을 박탈하여 탈퇴된 후에는 어떠한 데이터에도 접근할 수 없도록 해야 한다.

4.2.3 데이터 검색/저장 단계에서 발생하는 추가적인 정보 노출에 대한 안전성

사용자들의 데이터가 저장되는 저장 공간은 신뢰할 수 없는 제 3의 스토리지 서버에 의해 제공된다. 스토리지 서버는 반-신뢰한다고 가정하여 모든 프로토콜에는 정당하게 참여하지만 사용자가 데이터를 저장하고 검색하는 과정에서 불가피하게 노출될 수 있는 정보들을 통해 사용자들의 추가적인 정보를 알 수 없어야 한다. 예를 들어, 사용자의 검색 패턴, 데이터와 키워드의 연관 관계 등은 불가피하게 노출되는 정보가 아닌 서버가 추가적으로 알게 되는 정보이다. 따라서 키워드 검색에 대한 안전성을 제공하여 서버로부터 추가적인 정보 노출을 피하고 사용자의 프라이버시를 보호해야 한다.

V. 제안하는 프로토콜

5.1 시스템 초기화

제안하는 프로토콜에서 소셜 네트워크 서비스 제공자는 사용자들에게 안전한 시스템을 제공하기 위해 암호학적 초기화를 진행한다. 안전한 대칭키 암호화 기법 SE, 충돌 저항성(collision resistance) 키 해시 함수 H_k, 안전한 준 동형 공개키 암호화 기법 HPE을 설정한다.

5.2 메인 프로토콜

본 논문에서 제안하는 프로토콜은 6가지 프로토콜로 구성되어 있다.

〔표 1〕 제안하는 기법에서 사용하는 표기법과 의미

표기법	의미
ID_u	사용자 u의 아이디
pk_{ss}/sk_{ss}	스토리지 서버의 공개키/비밀키
H_k/k_u	키 해시함수와 사용자 u의 해시키
$List_u/SL_u$	사용자 u가 설정한 보안 레벨/사용자 u의 친구 목록
$List_u.ID$	사용자 u의 친구 목록 중 아이디 영역
$List_u.KEY$	사용자 u의 친구 목록 중 키 영역
$HPE.enc$	준 동형 암호화 기법의 암호화 알고리즘
$SE.enc/SE.dec$	대칭키 암호화 기법의 암호화/복호화 알고리즘
$TR_{u,u_i}/$ $TR_u = \cup TR_{u,u_i}$	사용자 u가 생성한 사용자 u_i와 관련된 토큰 트리/사용자 u가 생성한 전체 토큰 트리
TS_u	사용자 u가 다른 사용자들로부터 받은 토큰들의 집합

- $Setup(M;S_M)$: 관리자 M과 스토리지 서버 S_M이 참여하는 프로토콜이다. 소셜 네트워크 사용자 중 한명인 관리자 M은 스토리지 서버 S_M을 관리하기 위해 S_M의 공개키와 개인키, 보안 레벨을 설정한다.

- $Direct\_Registration(U_i;M;S_M)$: 임의의 사용자 U_i, 관리자 M, 스토리지 서버 S_M이 참여하는 프로토콜이다. 사용자 U_i는 관리자 M과 친분 관계를 형성하고, 관리자 M은 스토리지 서버 S_M을 통해 사용자 U_i가 다른 사용자들과 데이터를 공유할 수 있도록 설정한다.

- $Indirect\_Registration(V;S_M)$: 임의의 사용자 V와 스토리지 서버 S_M이 참여하는 프로토콜이다. 스토리지 서버 S_M에는 $Direct\_Registration$ 프로토콜을 수행하여 관리자 M과 친분 관계를 형성한 사용자들의 정보만 저장되어 있다. 하지만 스토리지 서버 S_M에 설정되어 있는 보안 레벨을 만족하는 다른 사용자들도 스토리지 서버 S_M을 통해 데이터를 공유할 수 있어야 한다. 따라서 임의의 사용자 V는 소셜 네트워크에서 자신이 맺은 친분 관계를 이용하여 스토리지 서버 S_M에 설정되어 있는 보안 레벨을 만족함을 보이고 이를 통해 데이터를 공유할 수 있는 권한을 얻는다. 권한을 얻은 사용자는 저장/검색 단계에서 사용할 키를 서버에 일시적으로 등록한다.

- $Revocation(M;S_M)$: 관리자 M과 스토리지 서버 S_M이 참여하는 프로토콜이다. 관리자 M은 자신과 친분 관계를 형성한 임의의 사용자 U가 탈퇴했을 경우 스토리지 서버 S_M에 저장되어 있는 사용자 U의 정보를 삭제함으로써 사용자 U가 더 이상 스토리지 서버 S_M으로부터 데이터를 공유할 수 없도록 한다.

- $Token\_Change(U_i;M;S_M)$: 관리자 M과 보안 레벨을 만족하는 모든 사용자 U_i, 스토리지 서버 S_M이 참여하는 프로토콜이다. 관리자 M은 관계 그래프에서 자신과 친분 관계를 형성한 친구들의 정보 이외에 다른 사용자들의 정보는 알 수 없다. 따라서 친구 이외에 사용자들의 추가/탈퇴에 대해 유동적으로 대처하기 위해 관리자 M은 보안 레벨 안에 위치한 모든 사용자에게 새로운 토큰을 전달하고 스토리지 서버 S_M에 토큰 정보를 갱신한다.

- $Store(U_i;S_M)$: 보안 레벨을 만족하는 사용자 U_i와 스토리지 서버 S_M이 참여하는 프로토콜이다. 사용자 U_i는 저장할 데이터와 데이터에 해

〔표 2〕 Setup Protocol

Manager M		Storage Server S_M
$pk_{ss}=(n,G,G_1,e,g,h)$ $sk_{ss}=(q_1)$ Generate ID_M, k_M Set the security level SL_M	$(pk_{ss},sk_{ss}),SL_M$ ⟶	$(pk_{ss},sk_{ss}),SL_M$

〔표 3〕 Direct_Registration Protocol

User U_i		Manager M		Storage Server S_M
	ID_{U_i} ⟶	$u_i \leftarrow_R Z_n^*$		
$(u_i,k_M,pk_{ss},e(g,g)^{q_1})$	$(u_i,k_M,pk_{ss},e(g,g)^{q_1})$ ⟵	$Q=(H_{k_M}(ID_{U_i}),g^{u_i})$	Q ⟶	$List_M \leftarrow List_M \cup Q$
	$Token\_Flooding(M,U_i,SL_M,k_M)$ ⟷		TR_{M,U_i} ⟶	$TR_M \leftarrow TR_M \cup TR_{M,U_i}$

당하는 키워드를 선택하여 서버에 저장한다. 스토리지 서버 S_M에는 사용자 U_i의 정보가 이미 저장되어 있기 때문에 관리자 M이 직접 참여하지 않아도 저장 단계를 수행할 수 있다.

- $Search(U_i ; S_M)$: 보안 레벨을 만족하는 사용자 U_i와 스토리지 서버 S_M이 참여하는 프로토콜이다. 사용자 U_i는 자신이 선택한 여러 키워드가 모두 포함되어 있는 데이터를 스토리지 서버 S_M으로부터 검색한다. 스토리지 서버 S_M에는 사용자 U_i의 정보가 이미 저장되어 있기 때문에 관리자 M이 직접 참여하지 않아도 검색 단계를 수행할 수 있다.

5.2.1 *Setup* 프로토콜

관리자 M는 자신의 스토리지 서버 S_M의 공개키 $pk_{ss}=(n,G,G_1,e,g,h)$와 비밀키 $sk_{ss}=(q_1)$를 생성하여 스토리지 서버에 전달하고 키 해시함수에서 사용할 키 k_M와 자신의 아이디 ID_M를 임의로 선택한다. 또한, 자신과 관련된 다른 사용자들이 스토리지 서버를 이용할 수 있도록 보안 레벨 SL_M를 설정한다.

5.2.2 *Direct_Registration* 프로토콜

사용자 U_i는 관리자 M과 친분 관계를 형성하기 위해 ID_{U_i}를 전송한다. 관리자 M는 사용자 U_i가 사용할 비밀키 $u_i \leftarrow_R Z_n^*$를 임의로 선택하여 U_i에게 $(u_i, k_M, pk_{ss}, e(g,g)^{q_1})$을 전송하고, 사용자 U_i가 스토리지 서버를 통해 데이터를 공유할 수 있도록 $(H_{k_M}(ID_{U_i}), g^{u_i})$을 스토리지 서버 S_M에 전송하여 사용자 목록 $List_M$에 포함시킨다. 관리자 M은 사용자 U_i를 통해 일정 거리 안에 위치한 다른 사용자들도 *Indirect_Registration* 프로토콜에서 보안 레벨을 만족함을 보이고 스토리지 서버 S_M을 통해 데이터를 공유할 수 있도록 해야 한다. 따라서 $Token\_Flooding(M, U_i, SL_M, k_M)$ 알고리즘을 수행하여 사용자 U_i를 통해 일정 거리 안에 위치하여 보안 레벨을 만족하는 사용자들에게 자신의 토큰과 해시 키 k_M을 함께 전달한다.

5.2.3 *Indirect_Registration* 프로토콜

관리자 M의 친구가 아닌 임의의 사용자 V가 스토리지 서버 S_M에 접근 권한을 요청하기 위해 $v \leftarrow_R Z_n^*$를 임의로 선택하여 해시된 아이디와 함께 질의 $(H_{k_M}(ID_V), g^v)$을 스토리지 서버에 전송한다. 여기서 v는 사용자 V가 데이터를 저장하거나 검색하는 과정에서 일시적으로 사용할 비밀키로 데이터를 저장/검색하기 위해 스토리지 서버에 접근하고자 할 때마다 새로

〔표 4〕 Indirect_Registration Protocol

User V		Storage Server S_M
		If $H_{k_M}(ID_V) \notin List_M.ID$
$(H_{k_M}(ID_V), g^v)$	$(H_{k_M}(ID_V), g^v)$ →	$Path\_Discovery(M, V) \rightarrow (M, H_{k_M}(ID_{M_i}), d_i, H_{k_M}(ID_V))$
		If $(d_i \le SL_M)$
		insert $(H_{k_M}(ID_V), g^v)$ in $List_M$
$e(g,g)^{q_1}$	← $e(g,g)^{q_1}$	
	Store, *Search* ↔	
		$List_M \setminus (H_{k_M}(ID_V), g^v)$

〔표 5〕 Revocation Protocol

Manager M		Storage Server S_M
	$H_{k_M}(ID_{U_i})$, TR_{M,U_i} →	$List_M \leftarrow List_M \setminus \{H_{k_M}(ID_{U_i}), g^{u_i}\}$
		$TR_M \leftarrow TR_M \setminus TR_{M,U_i}$

운 비밀키를 선택한다.

스토리지 서버 S_M은 $List_M.ID$에 $H_{k_M}(ID_V)$이 존재하는지 확인한다. 만약 존재하지 않는다면, 3.3 절에서 언급한 $Path\_Discovery(M, V)$ 알고리즘을 수행하여 관리자 M과 사용자 V 사이에 관계 경로 $(M, H_{k_M}(ID_{M_i}), d_i, H_{k_M}(ID_V))$를 찾는다. 스토리지 서버에는 관리자 M의 토큰 트리 TR_M이 저장되어 있기 때문에 관리자 M이 직접 $Path\_Discovery(M, V)$ 알고리즘에 참여하지 않더라도 스토리지 서버는 사용자의 관계 경로 정보를 스스로 얻을 수 있다. $(M, H_{k_M}(ID_{M_i}), d_i, H_{k_M}(ID_V))$로부터 얻은 관계 경로의 길이가 $d_i \le SL_M$을 만족한다면 $(H_{k_M}(ID_V), g^v)$를 $List_M$에 임시 저장하고 $e(g,g)^{q_1}$을 사용자 V에게 전송하여 사용자 V가 데이터를 저장하고 검색할 수 있도록 한다. 사용자 V의 서비스 이용이 끝났을 때 스토리지 서버는 $(H_{k_M}(ID_V), g^v)$을 $List_M$에서 삭제한다. 만약 사용자 V의 서비스 이용이 끝났을 때 $(H_{k_M}(ID_V), g^v)$을 $List_M$에서 삭제하지 않고 재사용한다면 사용자 V는 관리자 M의 직접적인 친구가 아님에도 불구하고 직접적인 친구처럼 행동할 수 있다. 따라서 스토리지 서버는 $(H_{k_M}(ID_V), g^v)$을 삭제해야 한다.

5.2.4 *Revoacation* 프로토콜

관리자 M은 자신과 직접 친분 관계를 형성한 친구 U_i가 탈퇴했을 경우, 서버에 저장되어 있는 U_i의 정보를 삭제함으로써 U_i가 더 이상 스토리지 서버를 이용할 수 없도록 할 수 있다. 관리자 M는 탈퇴자 U_i를 자신의 사용자 목록 $List_M$에서 삭제하기 위해 $H_{k_M}(ID_{U_i})$를 서버에 전송하고, 서버는 $List_M$에서 $(H_{k_M}(ID_{U_i}), g^{u_i})$를 삭제한다. 또한, 탈퇴자 U_i와 관련된 토큰 트리 TR_{M, U_i}를 서버에 전송하여 새로운 토큰 트리 $TR_M \leftarrow TR_M \setminus TR_{M, U_i}$을 저장한다.

5.2.5 *Token_Change* 프로토콜

일정한 주기마다 관리자 M은 스토리지 서버에 저장된 토큰 트리 TR_M을 새롭게 갱신한다. 이는 네트워크의 구성이 빈번히 변하는 동적인 소셜 네트워크에서 관리자 M의 친구는 아니지만 보안 레벨을 만족하는

〔표 6〕 Token_Change Protocol

All user U_i	Manager M	Storage Server S_M
	Del →	Delete TR_M
$Token\_Flooding(M, U_i, SL_M)$ ↔	TR_{M, U_i}' →	$TR_M' \leftarrow TR_M' \cup TR_{M, U_i}'$

〔표 7〕 Store Protocol

User U_i	Storage Server S_M
$D_{i,j}$, $W_{i,j} = (w_1, \ldots, w_{l_{ij}})$	
$a_{i,j} \leftarrow_R Z_n^*$, $f_{i,j}(x) = a_{i,j}(x - H_{k_M}(w_1)) \cdots (x - H_{k_M}(w_{l_{ij}}))(x - H_{k_M}(rw_1)) \cdots (x - H_{k_M}(rw_{p-l_{ij}}))$	
$POLY.HPE.enc_{pk_{ss}}(f_{i,j}(x))$	
$r_{i,j} \leftarrow_R Z_n^*$, $HPE.enc_{pk_{ss}}(r_{i,j} + u_i)$, $k_{i,j} = H_{k_M}(e(g,g)^{q_1 r_{ij}})$, $C_{i,j} = SE.enc_{k_{ij}}(D_{i,j})$	
$Q' = (H_{k_M}(ID_{U_i}), POLY.HPE.enc_{pk_{ss}}(f_{i,j}(x)), HPE.enc_{pk_{ss}}(r_{i,j} + u_i), C_{i,j})$	
Q' →	
	Check $H_{k_M}(ID_{U_i}) \in List_M.ID$
	$(Sub(HPE.enc_{pk_{ss}}(r_{i,j} + u_i), HPE.enc_{pk_{ss}}(u_i)))^{q_1} = (g^{q_1})^{r_{ij}}$
	Store $(POLY.HPE.enc_{pk_{ss}}(f_{i,j}(x)), (g^{q_1})^{r_{ij}}, C_{i,j})$

〔표 8〕 Search Protocol

User U_j | Storage Server S_M

$w_1,\dots,w_t$

$Q'' = (HPE.enc_{pk_{ss}}(H_{k_M}(w_1)^p + \cdots + H_{k_M}(w_t)^p), \dots, HPE.enc_{pk_{ss}}(H_{k_M}(w_1)^0 + \cdots + H_{k_M}(w_t)^0))$

$(H_{k_M}(ID_{U_j}), Q'')$ →

Check $H_{k_M}(ID_{U_j})$

For $1 \le i \le |documents|$

If $(POLY.HPE.enc_{pk_{ss}}(f_i(H_{k_M}(w_1)) + \cdots + f_i(H_{k_M}(w_t)))^{q_1} = 1$

append $(C_i, e(g^{q_1 r_i}, g^{u_i}))$ to the output list Γ

← Γ

For $1 \le i \le |\Gamma|$

$k_i = H_{k_M}(e(g^{q_1 r_i}, g^{u_j})^{u_j^{-1}}) = H_{k_M}(e(g,g)^{q_1 r_i})$, $D_i = SE.dec_{k_i}(C_i)$

위치에 새로 가입한 사용자들이 서비스를 이용하게 하거나, 보안 레벨을 만족하는 위치에서 탈퇴된 사용자들에게 데이터가 노출되는 것을 막기 위해 반드시 필요한 과정이다.

먼저 관리자 M의 삭제 요청에 의해 서버는 기존의 토큰 트리 TR_M을 삭제한다. 관리자 M은 자신의 모든 친구 U_i와 $Token_Flooding(M, U_i, SL_M)$을 수행하여 사용자 U_i에 대한 새로운 토큰 트리 TR_{M,U_i}'을 생성하고 이를 스토리지 서버에 전송하여 새로운 토큰 트리 $TR_M' = TR_M' \cup TR_{M,U_I}'$를 저장한다.

5.2.6 *Store* 프로토콜

사용자 U_i는 데이터 $D_{i,j}$에 대한 키워드 집합 $W_{i,j} = \{w_1,\dots,w_{l_{ij}}\}$를 생성한다. 만약 $|W_{i,j}| < p$라면, 랜덤값 $rw_{l'}(1 \le l' \le p - |W_{i,j}|)$을 생성하여 새로운 키워드 집합 $W_{i,j}' = W_{i,j} \cup \{rw_1,\dots,rw_{p-l_{ij}}\}$을 생성한다. 검색 단계에서 데이터 $D_{i,j}$를 키워드 집합에 포함된 키워드들로 검색할 수 있도록 하기 위해 $W_{i,j}'$의 원소를 근으로 하는 다음과 같은 p차 다항식에 대한 암호문 $POLY.HPE.enc_{pk_{ss}}(f_{i,j}(x))$을 생성한다. 여기서 $a_{i,j}$는 데이터를 저장하는 사용자가 임의로 선택한 난수이다.

$$f_{i,j}(x) = a_{i,j}(x - H_{k_M}(w_1)) \cdots (x - H_{k_M}(w_{l_{ij}}))(x - H_{k_M}(rw_1)) \cdots (x - H_{k_M}(rw_{p-l_{ij}})) \quad (11)$$

사용자 U_i는 난수 $r_{i,j} \leftarrow_R Z_n^*$를 선택하여 대칭키 $k_{i,j} = H_{k_M}(e(g,g)^{q_1 r_{ij}})$을 생성하고 $SE.enc_{k_{ij}}(D_{i,j})$를 얻는다. 암호화된 데이터와 함께 난수 $r_{i,j}$에 대한 정보를 저장하기 위해 자신의 비밀키 u_i를 이용하여 암호문 $HPE.enc_{pk_{ss}}(r_{i,j} + u_i)$을 생성하고 스토리지 서버에 다음과 같은 질의 Q'을 전송한다.

$$Q' = (H_{k_M}(ID_{U_i}), POLY.HPE.enc_{pk_{ss}}(f_{i,j}(x)), , HPE.enc_{pk_{ss}}(r_{i,j} + u_i), C_{i,j}) \quad (12)$$

스토리지 서버는 먼저 $List_M.ID$에 $H_{k_M}(ID_{U_i})$가 존재하는지 확인하고, $H_{k_M}(ID_{U_i})$에 해당하는 키 g^{u_i}를 얻는다. g^{u_i}와 임의로 선택한 $z \leftarrow_R Z_n^*$를 이용하여 $HPE.enc_{pk_{ss}}(u_i) = g^{u_i} h^z$로 하고 자신의 비밀키 $sk_{ss} = (q_1)$을 사용하여 식 (13)을 통해 $(g^{q_1})^{r_{ij}}$를 얻어 $(POLY.HPE.enc_{pk_{ss}}(f_{i,j}(x)), (g^{q_1})^{r_{ij}}, C_{i,j})$을 저장한다.

$$(Sub(HPE.enc_{pk_{ss}}(r_{i,j} + u_i), HPE.enc_{pk_{ss}}(u_i)))^{q_1} = (HPE.enc_{pk_{ss}}(r_{i,j}))^{q_1} = (g^{q_1})^{r_{ij}} \quad (13)$$

5.2.7 *Search* 프로토콜

사용자 U_j는 자신이 선택한 키워드 $w_1,\dots,w_t$가 모두 포함된 데이터를 검색한다. 키워드의 정보가 포함되어 있는 질의를 다음과 같이 생성한다.

$$Q'' = (HPE.enc_{pk_{ss}}(H_{k_M}(w_1)^p + \cdots + H_{k_M}(w_t)^p), \ldots, HPE.enc_{pk_{ss}}(H_{k_M}(w_1)^0 + \cdots + H_{k_M}(w_t)^0)) \quad (14)$$

사용자 U_i가 $(H_{k_M}(ID_{U_i}), Q'')$을 서버에 전송하면 서버는 먼저 $List_M.ID$에 $H_{k_M}(ID_{U_i})$가 존재하는지 확인한다. 존재하지 않는다면 프로토콜을 종료한다. 만약 존재한다면, 다음과 같은 검증 과정을 거쳐 데이터를 검색한다.

$$\begin{aligned}&(POLY.HPE.enc_{pk_{ss}}(f_i(H_{k_M}(w_1)) + \cdots + f_i(H_{k_M}(w_t)))^{q_1} \\ &= (POLY.HPE.enc_{pk_{ss}}(0))^{q_1} = (HPE.enc_{pk_{ss}}(0))^{q_1} \\ &= (g^0 h^r)^{q_1} = (g^{q_1})^0 = 1 \end{aligned} \quad (15)$$

만약 사용자 U_i가 선택한 키워드 모두가 데이터에 포함되어 있다면 $f_i(H_{k_M}(w_j)) = 0 (1 \le j \le t)$으로 식 (15)을 만족한다. 위의 검증 과정을 통과한 데이터에 대해서 $e(g^{q_1 r_i}, g^{u_j})$을 계산하여 암호화된 데이터 C_i와 함께 데이터 목록 Γ에 저장하고 이를 사용자 U_i에게 전송한다. 사용자 U_i는 데이터에 대해 복호화키 $k_i = H_{k_M}(e(g^{q_1 r_i}, g^{u_j})^{u_j^{-1}}) = H_{k_M}(e(g,g)^{q_1 r_i})$를 생성하고, 각 암호문을 $D_i = SE.dec_{k_i}(C_i)$로 복호화하고 데이터를 얻는다.

5.3 안전성 분석

5.3.1 데이터 프라이버시

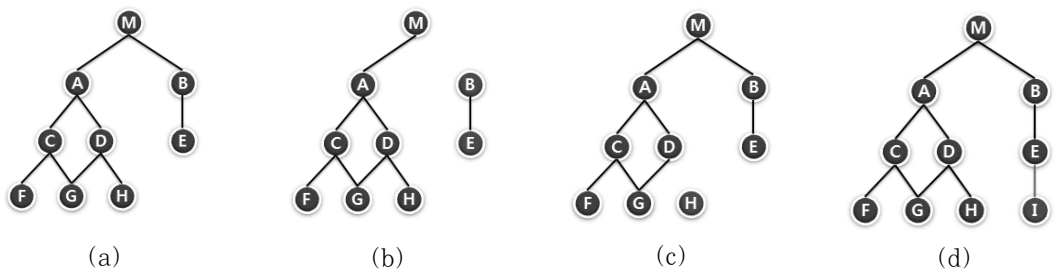
스토리지 서버가 데이터 $D_{i,j}$를 알아내기 위해서는 $C_{i,j} = SE.enc_{k_{i,j}}(D_{i,j})$를 복호화해야 한다. 하지만 스토리지 서버는 관리자 M의 친구 목록 $List_M$으로부터 g^{u_i}를 알 수 있지만 실제 대칭키를 생성하기 위해 필요한 값 u_i는 알 수 없다. 이는 g^{u_i}로부터 u_i를 얻는 이산 대수 문제의 어려움에 기반한다. 또한, 스토리지 서버는 관리자 M의 해시키 k_M를 모르기 때문에 스토리지 서버가 가지고 있는 정보로부터 $C_{i,j} = SE.enc_{k_{i,j}}(D_{i,j})$를 복호화하는 것은 불가능하다.

비인가 사용자는 관리자 M이 설정해 놓은 보안 레벨 SL_M을 만족하지 못하는 사용자라고 생각할 수 있다. 비인가 사용자는 스토리지 서버로부터 데이터를 저장하거나 검색할 수 없다. 따라서 정당한 사용자 U_j가 검색한 정보 $(C_i, e(g^{q_1 r_i}, g^{u_j}))$를 도청함으로써 데이터를 알아내려고 시도할 것이다. 하지만 도청을 통해 다른 사용자의 검색 정보를 얻는다 하더라도 비인가 사용자는 $Direct\_Registration$ 프로토콜에서 전송되었던 k_M을 전달받지 못 했고 사용자 U_j의 비밀키 u_j을 모르기 때문에 C_i의 복호화키 $k_i = H_{k_M}(e(g^{q_1 r_i}, g^{u_j})^{u_j^{-1}})$를 생성할 수 없다. 따라서 도청에 의해 노출된 정보를 통해서는 실제 데이터를 알아낼 수 없다.

하지만 정당한 사용자가 스토리지 서버, 비인가 사용자와 공모 공격(collusion attack)을 한다면 데이터 $D_{i,j}$가 노출될 수 있다. 이러한 공모 공격이 발생했을 경우, 악의적인 행위를 한 내부 공모자(traitor)를 추적하는 기법[9,10]이 연구되고 있다. 연구되어 있는 기법을 적용하여 내부 공모자를 추적함으로써 공모 공격에 의해 노출되는 데이터를 보호할 수 있다.

5.3.2 접근 제어

제안하는 기법에서 거리 기반 접근 제어 방식을 사용하여 관리자 M으로 일정한 거리 내에 위치하는 사용자만이 데이터에 접근할 수 있는 접근 권한이 주어진다. 사용자들의 친분 관계가 수시로 변하는 소셜 네



[그림 1] 각 경우에 따른 관리자 M의 관계 그래프 변화. (a) 관리자 M의 관계 그래프 (b) 관리자 M으로부터 친구 B의 탈퇴 (c) 보안 레벨 내에 위치한 사용자 H의 탈퇴 (d) 보안 레벨 내에 새로운 사용자 I 가입

트워크에서 다음과 같은 친분 관계 변화가 발생했을 경우에 대해 안전성을 분석한다.

5.3.2.1 거리 1에 위치한 사용자(친구)의 탈퇴

관리자 M과 거리 1에 위치한 사용자(친구)가 탈퇴 었을 경우 해당 탈퇴자뿐만 아니라 탈퇴자와 관련되어 있는 보안 레벨 SL_M을 만족하는 모든 사용자도 스토리지 서버에 데이터를 저장하거나 저장되어 있는 어떠한 데이터에도 접근할 수 없어야 한다. 〔그림 1〕의 (b)의 경우를 살펴보자.

관리자 M으로부터 친구 B가 탈퇴되었을 경우 *Revocation* 프로토콜에서 관리자 M은 스토리지 서버에 저장되어 있는 탈퇴자 B의 정보 $(H_{k_M}(ID_B), g^b)$를 $List_M$에서 삭제한다. 비록 탈퇴자 B는 관리자 M의 해시키 k_M을 알고 있어 *Store* 프로토콜과 *Search* 프로토콜에서 스토리지 서버에 질의할 수 있지만, 질의에 포함되어 있는 $H_{k_M}(ID_B)$로 인해 스토리지 서버는 $List_M$에 $(H_{k_M}(ID_B), g^b)$이 존재하지 않음을 확인하고 프로토콜을 종료한다.

따라서 탈퇴자 B는 더 이상 스토리지 서버에 데이터를 저장하거나 검색 할 수 없다.

탈퇴자 B의 친구 E도 스토리지 서버에 접근할 수 없도록 해야 한다. 이를 위해, *Revocation* 프로토콜에서 관리자 M은 탈퇴자 B와 관련된 토큰 트리 $TR_{M,B}$를 스토리지 서버에 저장되어 있는 TR_M에서 삭제한다. 사용자 E는 스토리지 서버에 데이터를 저장하거나 검색하기 위해 *Indirect_Registration* 프로토콜에서 관리자 M이 설정해 놓은 보안 레벨 내에 위치함을 보여야한다. 하지만, 이 과정에서 사용되는 토큰 트리 $TR_{M,B}$이 삭제되었기 때문에 E가 가지고 있는 토큰은 더 이상 TR_M에 존재하지 않는다. 따라서 *Path_Discovery*(M, E)을 통해 정당한 경로를 검색할 수 없기 때문에 사용자 E는 *Store* 프로토콜과 *Search* 프로토콜을 수행할 수 없다.

5.3.2.2 거리 $1 < j \leq SL_M$ 에 위치한 사용자의 탈퇴

관리자 M이 설정해 놓은 보안 레벨 내에 위치한 임의의 사용자가 탈퇴되었을 경우, 탈퇴한 사용자는 스토리지 서버에 더 이상 접근할 수 없어야 한다. 〔그림 1〕의 (c)의 경우를 살펴보자.

관리자 M이 설정해 놓은 보안 레벨을 만족하는 사용자 D의 친구 H가 탈퇴되었다고 가정해 보자. 이 경우, 사용자 H는 관리자 M의 스토리지 서버를 통해 데이터를 공유할 수 없어야 한다. 이를 위해, 관리자 M은 사용자 H가 받았을 토큰을 토큰 트리 TR_M으로부터 삭제하여 *Indirect_Registration* 프로토콜에서 수행하는 *Path_Discovery* 알고리즘에서 정당한 경로를 찾을 수 없도록 해야 한다. 하지만 관리자 M은 자신과 직접 친분 관계를 형성하고 있는 사용자들 이외에는 소셜 네트워크에서 다른 사용자들이 형성하고 있고 친분 관계나 각 사용자들이 받았을 토큰의 정보를 알 수 없다. 따라서 사용자 D의 친구 H가 탈퇴했을 경우에도 관리자 M은 사용자 H가 탈퇴되었는지 알 수 없고 H가 가지고 있는 자신의 토큰도 정확히 알 수 없다. 따라서 제안하는 기법에서는 *Token_Change* 프로토콜을 활용하여 주기적으로 *Token_Flooding* 알고리즘을 수행하여 관계 그래프 내에 사용자들에게 새로운 토큰을 전달하고 스토리지 서버에 저장되어 있는 TR_M을 새롭게 갱신함으로써 탈퇴한 사용자 H가 새로운 토큰을 받을 수 없도록 한다. 이를 통해, 보안 레벨 내의 탈퇴한 임의의 사용자에 대한 접근 제어를 할 수 있다.

5.3.2.3 거리 $1 < j \leq SL_M$ 에 새로운 사용자의 가입

관리자 M이 설정해 놓은 보안 레벨 내에 새로운 사용자가 가입되었을 경우, 새로운 사용자에게 스토리지 서버를 이용할 수 있는 접근 권한을 부여해야 한다. 〔그림 1〕의 (d)의 경우를 살펴보자.

관리자 M이 설정해 놓은 보안 레벨을 만족하는 사용자 E의 친구 I가 새롭게 가입되었다고 가정해 보자. 이 경우 관리자 M은 사용자 I에게 자신의 스토리지 서버를 이용할 수 있는 접근 권한을 부여해야 한다. 하지만 관리자 M은 사용자 I의 가입 여부를 알 수 없기 때문에 직접적으로 접근 권한을 부여할 수 없다. 따라서 관리자 M은 5.3.2절에서 언급한 것과 같이 *Token_Change* 프로토콜을 통하여 사용자 I가 새로운 토큰을 받을 수 있도록 하고 자신의 토큰 트리 TR_M을 새롭게 갱신함으로써 사용자 I에게 스토리지 서버를 활용할 수 있도록 접근 권한을 부여할 수 있다.

Token_Change 프로토콜을 통해 사용자들에게 새로운 토큰을 부여하고 토큰 트리를 주기적으로 갱신함으로써 관계 그래프에서 발생하는 변화에 효과적으로 대처할 수 있지만 토큰을 주기적으로 변경해주어야 하기 때문에 시스템의 전반적인 효율성이 떨어지는 상충되는 문제(tradeoff)가 있다. 또한 토큰 트리가 갱신되

는 시점에 따라 탈퇴된 사용자가 얼마동안은 데이터를 저장/검색할 수 있고, 새로 가입한 사용자가 데이터를 공유할 수 없는 시간이 존재하게 된다. 이는 향후 연구를 통해 효율적인 개선 방안을 제시하도록 한다.

5.3.3 데이터 검색/저장 단계에서 발생하는 추가적인 정보 노출에 대한 안전성

Store 프로토콜과 *Search* 프로토콜에서 스토리지 서버는 사용자들로부터 전달받은 질의를 통해 추가적인 정보를 얻으려고 시도할 것이다. 예를 들어, 저장하는 데이터와 키워드의 관계, 데이터를 저장하거나 검색할 때 사용자가 선택한 키워드의 정보 등은 질의를 통해 얻을 수 있는 추가적인 정보이다. 데이터 저장/검색 단계에서 발생하는 추가적인 정보 노출에 대한 제안하는 기법의 안전성을 다음과 같이 분석한다.

5.3.3.1 저장 과정에서 발생하는 추가적인 정보 노출에 대한 안전성

사용자 U_i는 다음의 질의를 통해 데이터를 서버에 저장한다.

$$(H_{k_M}(ID_{U_i}),\ POLY.HPE.enc_{pk_{ss}}(f_{i,j}(x)) \\ ,\ HPE.enc_{pk_{ss}}(r_{i,j}+u_i),\ SE.enc_{k_{ij}}(D_{i.j})) \quad (16)$$

5.3.1 절에서 언급한 것과 같이 스토리지 서버는 전달받은 질의로부터 데이터 복호화키 $k_{i,j}$를 생성할 수 없기 때문에 저장하는 데이터 $D_{i,j}$를 정확히 알 수 없다. 하지만, 사용자 U_i가 설정해 놓은 키워드에 대한 정보를 알 수 있다면 데이터의 내용을 어느 정도 유추할 수 있다.

키워드를 알아내는 가장 명확한 방법은 $POLY.HPE.enc_{pk_{ss}}(f_{i,j}(x))$을 복호화하여 $f_{i,j}(x)$의 계수를 알아내고 인수분해를 통해 근을 찾는 것 이다. 하지만 계수의 크기는 충분히 크고 크기가 큰 평문에 대해서는 복호화할 수 없는 준 동형 암호화 기법을 사용하였기 때문에 $f_{i,j}(x)$의 근을 찾는 것은 불가능하다. 다른 방법으로, 스토리지 서버는 데이터에 해당하는 키워드는 한정적이라는 사실을 이용해 임의의 키워드를 대입함으로써 사전 공격(dictionary attack)을 시도할 수 있다. 스토리지 서버는 임의의 키워드 w를 선택하고 질의에 포함되어 있는 $POLY.HPE.enc_{pk_{ss}}(f_{i,j}(x))$에 대입하여 키워드의 정당성을 확인해야 한다. 하지만 스토리지 서버가 정당한 키워드 $w \in W_{i,j}$를 선택하였다 할지라도 $f_{i,j}(x)$는 $H_{k_M}(w)$를 근으로 갖는 다항식이므로 관리자 M의 해시키 k_M을 모르면 $H_{k_M}(w)$을 생성할 수 없다. 따라서 스토리지 서버가 선택한 키워드 w가 저장하는 데이터에 해당하는 정당한 키워드인지를 확인할 수 없다.

스토리지 서버가 동일한 키워드가 사용된 데이터를 분류할 수 있다면 사용자의 데이터 저장/검색 패턴이 노출될 수 있다. 따라서 스토리지 서버는 서로 다른 데이터에 적용된 키워드의 유사성을 판단할 수 없어야 한다. 만약 사용자 U_i가 서로 다른 데이터 $D_{i,j}$와 $D_{i,j}'$에 동일한 키워드 집합 W를 적용한다 하더라도 생성되는 다항식 $f_{i,j}(x)$와 $f_{i,j}'(x)$에는 서로 다른 랜덤 값 $a_{i,j}$와 $a_{i,j}'$이 사용되기 때문에 서버는 다항식 $f_{i,j}(x)$와 $f_{i,j}'(x)$에 대한 암호문을 보고 사용된 키워드의 유사성을 판별할 수 없다.

5.3.3.2 검색 과정에서 발생하는 추가적인 정보 노출에 대한 안전성

사용자 U_i가 검색하기 위해 서버에 전달하는 질의는 $(H_{k_M}(ID_{U_i}), Q'')$이다. 사용자 U_i는 동일한 아이디 $H_{k_M}(ID_{U_i})$을 사용하여 검색하기 때문에 서버는 사용자 U_i로부터 전송받은 m개의 질의를 이용하여 사용자 U_i의 검색 패턴 분석을 시도할 수 있다. 질의는 다음

〔표 9〕 기존 기법과 제안하는 기법의 비교

	〔2〕	〔3〕	제안하는 기법
접근 제어	역할 기반	속성 기반	거리 기반
암호화 기법	브로드캐스트 암호	속성 기반 암호	준 동형 공개키 암호
탈퇴 기능	△	△	○
데이터 공유 범위(거리)	1	1	k
데이터 공급자 수	1	1	n
데이터 검색자 수	n	n	n

과 같은 구성된다.

$$\{(H_{k_M}(ID_{U_i}), Q_j'') : 1 \le j \le m\} \quad (17)$$

하지만 Q_j''은 준 동형 암호화 기법으로 암호화 되어 있고 이 암호화 기법은 크기가 작은 평문에 대한 암호문만 복호화 할 수 있기 때문에 키워드 w_i에 대한 크기가 큰 해시값 $H_{K_M}(w_i)$을 평문으로 사용하면 완벽한 복호화를 할 수 없다. 만약, 복호화를 통해 $H_{K_M}(w_i)$을 얻었다 하더라도 H_{k_M}은 안전한 키 해시 함수이기 때문에 키 k_M을 모르는 스토리지 서버는 키워드 w_i에 대한 어떠한 정보도 얻을 수 없다. 따라서 이전에 전송받은 m개의 질의을 통해서 어떠한 정보도 얻을 수 없기 때문에 키워드의 연관 관계를 통한 사용자의 검색 패턴 분석은 불가능하다.

5.4 비 교

기존에 제안된 소셜 네트워크 환경에서 프라이버시 보호 데이터 공유 기법들[2,3]과 제안하는 기법을 [표 9]와 같이 비교한다.

문헌 [2]와 문헌 [3]는 각각 역할 기반 접근 제어 방식과 속성 기반 접근 제어 방식을 사용하여 유일한 데이터 공급자가 제공하는 데이터를 다수의 사용자가 검색하여 데이터를 이용할 수 있도록 하는 프라이버시 보호 데이터 공유 기법을 제안하였다. 하지만 데이터 소유자만이 데이터를 저장하고 친분 관계가 있는 사용자들만 데이터를 검색할 수 있는 제한적인 데이터 공유 기법이다. 또한 탈퇴한 사용자가 일부 데이터를 계속적으로 이용할 수 있기 때문에 효과적인 탈퇴 기능을 제공하지 못한다.

본 논문에서 제안하는 기법은 거리 기반 접근 제어 방식을 사용하여 직접적으로 친분 관계를 형성한 사용자들뿐만 아니라 보안 레벨을 만족하는 사용자들과도 데이터를 공유할 수 있는 효율적인 기법이다. 또한 준 동형 공개키 암호화 기법을 사용하여 연산을 효율적으로 수행하였으며 사용자들의 관계 그래프가 수시로 변화하는 소셜 네트워크 특징을 고려하여 탈퇴한 사용자와 새로 가입한 사용자가 원활한 서비스를 제공받을 수 있게 하였다. 특히, 탈퇴한 사용자들이 일부 데이터를 지속적으로 이용할 수 있었던 기존 연구의 문제를 개선하여 사용자의 탈퇴 처리가 이루어진 후 어떠한 데이터도 이용할 수 없도록 효과적인 탈퇴 기능을 제공한다. 본 논문에서는 기존 연구[2,3]에서 한명의 데이터 소유자만이 데이터를 저장하고 다른 사용자들은 데이터를 검색만 할 수 있는 환경을 확장하여 서버 관리자와 친분 관계를 맺고 있는 사용자들뿐만 아니라 일정 거리 이내에 위치한 사용자들 모두에게 데이터를 저장하고 검색할 수 있도록 하였다.

[표 10] Store protocol 연산

연산	사용자	스토리지 서버
해시(hash) 연산	p+2	·
Poly.HPE	1	·
페어링 연산	1	·
대칭키 암호 연산	1	·
준 동형 성질	·	1

[표 11] Search protocol 연산

연산	사용자	스토리지 서버
해시(hash) 연산	(p+1)t+1	·
Poly.HPE	·	$\|documents\|$
페어링 연산	1	$\|I\|$
대칭키 암호 연산	$\|I\|$	·

본 논문에서 제안한 프로토콜에서 저장/검색 단계에 대한 연산량은 각각 [표 10], [표 11]과 같다.

Store protocol에서 사용자는 키워드의 해시값과 키를 생성하기 위해 해시 연산을 p+2번 수행하며, 키워드의 정보를 담은 다항식을 생성하기 위해 Poly.HPE 연산을 1번 수행한다. 또한, 대칭키를 생성하기 위해 페어링 연산 1번, 데이터를 암호화하기 위해 대칭키 암호 연산 1번을 수행한다. 스토리지 서버는 데이터를 저장하기 위해 준 동형 성질을 1번 사용한다.

Search protocol에서 사용자는 검색 키워드에 대한 해시값을 생성하기 위해 해시 연산을 (p+1)t번, 대칭키를 생성하기 위해 해시 연산을 1번을 수행하여 총 (p+1)t+1 번의 해시 연산을 수행한다. 또한, 대칭키 생성 과정에서 페어링 연산 1번이 필요하고 스토리지 서버로부터 받은 데이터를 복호화하기 위해 전송받은 데이터량 만큼의 대칭키 암호 연산이 사용된다. 스토리지 서버는 사용자가 원하는 데이터를 검색하는 단계에서 POLY.HPE 연산을 저장된 데이터의 수 $|documents|$ 만큼 수행하며, 검색된 데이터를 사용자에게 전달하기 위해 $|I|$ 번의 페어링 연산을 수행한다.

저장/검색 과정에서는 연산 효율이 좋은 해시 연산과 대칭키 암호 연산을 주로 사용하여 사용자의 연산량을 줄여 효율성을 높였다.

VI. 결 론

본 논문에서는 소셜 네트워크 환경에 적합한 프라이버시 보존 데이터 공유 기법을 제안하였다. 제안하는 기법을 통해 서버 관리자와 친분 관계를 맺고 있는 사용자들과 일정한 거리 내에 위치하여 보안 레벨을 만족하는 사용자들이 스토리지 서버를 자유롭게 사용함으로써 효율적으로 데이터를 공유할 수 있다. 또한 암호화된 데이터에서 키워드 검색 기능을 제공함으로써 효과적으로 필요한 데이터만을 검색하여 활용할 수 있다. 사용자들의 관계 그래프가 수시로 변하는 소셜 네트워크의 특징을 고려하여 효율적이고 안전한 추가/탈퇴 기능을 제공하였다. 특히, 기존 연구들에서 드러났던 탈퇴한 사용자가 탈퇴 후에도 일부 데이터에 지속적으로 접근하는 문제점을 해결하여 소셜 네트워크에 적합한 탈퇴 기능을 제공하였다. 또한 여러 사용자가 하나의 스토리지 서버에 데이터를 저장하고 검색할 수 있어 데이터 공유의 효율성을 높였다.

참고문헌

[1] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: An Online Social Network with User-Defined Privacy," In Proc. of SIGCOMM, pp. 135-146, 2009.

[2] J. Sun, X. Zhu, and Y. Fang, "A Privacy-Preserving Scheme for Online Social Networks with Efficient Revocation," In Proc. 29th conference on Information communications, pp. 2516-2524, 2010.

[3] S. Jahid, P. Mittal, and N. Borisov, "EASiER: Encryption-based access control in social networks with efficient revocation," In Proc. ASIACCS, 2011.

[4] Y. Zhu, Z. Hu, H. Wang, H. Hu, and G.-J. Ahn, "A Collaborative Framework for Privacy Protection in Online Social Networks," Cryptology ePrint Archive, Report 2010/491, 2010.

[5] K. Frikken and P. Srinivas, "Key allocation schemes for private social networks,' In WPES, pp. 11-20, 2009.

[6] W. Villegas, B. Ali, and M. Maheswaran, "An Access Control Scheme for Protecting Personal Data," In Proc. PST, pp. 24-35, 2008.

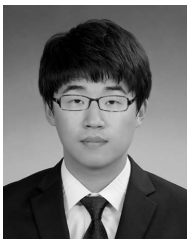
[7] G. Mezzour, A. Perrig, V. Gligor, and P. Papadimitratos, "Privacy - Preserving Relationship Path Dis covery in Social Networks," In Proc. CANS, pp. 189-208, 2009.

[8] D. Boneh, E.J. Goh, and K. Nissim, "Evaluating 2-DNF Formulas on Ciphertexts," In Proc. TCC, pp. 325-341, 2005.

[9] M. Naor and B. Pinkas, "Threshold Traitor Tracing," In Proc. CRYPTO, pp. 502-517, 1998

[10] D. Boneh and M. Franklin, "An Efficient Public Key Traitor Tracing Scheme," In Proc. CRYPTO, pp. 338-353, 1999.

〈著者紹介〉



전 두 현 (Doo-Hyun Jeon) 학생회원
2009년 8월: 서울시립대학교 수학과 졸업
2010년 3월~현재: 고려대학교 정보보호학과 석사과정
〈관심분야〉 프라이버시향상기술(PET), 데이터베이스 보안, 소셜 네트워크 보안

천 지 영 (Ji Young Chun) 학생회원
1997년 2월: 이화여자대학교 수학과 학사 졸업
2006년 2월: 고려대학교 정보경영공학과 석사 졸업
2011년 8월: 고려대학교 정보경영공학과 박사 졸업
2011년 9월~현재: 고려대학교 정보보호대학원 박사 후 연구원
〈관심분야〉 암호 이론, 프라이버시향상기술(PET), 유비쿼터스 보안, 소셜 네트워크 보안

정 익 래 (Ik Rae Jeong) 정회원
1998년 2월: 고려대학교 전산학과 학사 졸업
2000년 2월: 고려대학교 전산학과 석사 졸업
2004년 8월: 고려대학교 정보보호학과 박사 졸업
2006년 6월~2008년 2월: 한국전자통신연구원 암호기술연구팀 선임연구원
2008년 3월~현재: 고려대학교 정보경영공학부 교수
〈관심분야〉 프라이버시향상기술(PET), 데이터베이스 암호, 암호 이론, 소셜 네트워크 보안