情報保護學會論文誌
第 23 卷 第 1 號, 2013. 2

SNS에서 접근제어 및 해시체인을 이용한 메시지 보호 기법*

정 한 재,† 원 동 호‡
성균관대학교 정보통신대학

A method for the protection of the message in social network service(SNS) using access control and hash-chain*

Hanjae Jeong,† Dongho Won‡
College of Information & Communication Engineering of Sungkyunkwan University

요 약

소셜네트워크서비스(social network service, SNS)가 급격히 성장함에 따라 다양한 정보들이 SNS를 통해 공유되고 있다. 그러나 SNS에서 공유되는 정보로 인하여 개인의 사생활이 침해되거나 과거에 작성된 글로 인하여 피해를 보는 경우가 발생하고 있다. 따라서, 이를 해결하기 위하여 많은 연구들이 수행되었으나, 키를 관리하는 명확한 방법이나 접근제어에 대해 다룬 연구가 미흡하였다. 본 논문에서는 접근제어 및 해시체인을 이용하여 SNS에서 작성한 글을 보호하는 방법을 제안한다.

ABSTRACT

As social network service(SNS) has grown rapidly, the variety information being shared through SNS. However, the privacy of individuals can be violated due to the shared information through a SNS or the post written in the past at the SNS. Although, many researches concerned about it, they did not suggest key management and access control especially. In this paper, we propose the method for the protection of the message in SNS using access control and hash-chain.

Keywords: Social network services, SNS, hash-chain, message protection

I. 서 론

최근 스마트폰의 사용이 급증함에 따라 Twitter 또는 Facebook과 같은 소셜네트워크서비스(social network service, SNS)의 사용이 급증하고 있다. SNS는 뉴스 및 개인의 의견, 정보 등을 실시간으로 빠르게 공유할 수 있는 장점을 가지고 있다. 그러나 SNS를 통한 정보 공유가 개인의 사생활이나 과거에 작성되었던 글이 무분별하게 불특정 다수에게 공개되는 단점이 있다. 즉, 사용자의 의도와 다르게 SNS를 통해 정보가 공유되고 노출될 수 있다.

기존 연구에서 SNS의 글을 암호화하는 방법이 제안되었으나, 키 관리 및 효율성 등이 고려되지 않았다[1][2].

본 논문에서는 SNS를 통해 개인의 사생활이나 과거의 행적이 무분별하게 공개되는 것을 해결하고자 한다. 이후 본 논문의 구성은 다음과 같다. 2장에서는 SNS 중 널리 사용되는 Twitter와 Facebook에 대

접수일(2012년 11월 22일), 수정일(2013년 2월 8일), 게재확정일(2013년 2월 8일)
* 본 연구는 방송통신위원회의 방송통신융합미디어원천기술 개발사업의 연구결과로 수행되었음" (KCA-2012-12-912-06-003)
† 주저자, hjjeong@security.re.kr
‡ 교신저자, dhwon@security.re.kr

〔표 1〕 Facebook과 Twitter 비교

	Facebook	Twitter
작성한 글의 공유에 대한 기본 설정	친구에게만 공개	전체 공개
업데이트 알림	친구에게만 알림	′Follower′에게만 알림 (단, 설정을 통해 알림 차단 가능)
정보 차단 설정	친구 끊기를 통해 가능	불가능

해 알아본다. 3장에서는 SNS에서의 프라이버시를 보호하는 방법을 제안하며, 4장에서는 제안한 방법의 성능, 보안 분석을 한다. 마지막으로 5장에서 결론을 맺는다.

II. 관련 연구

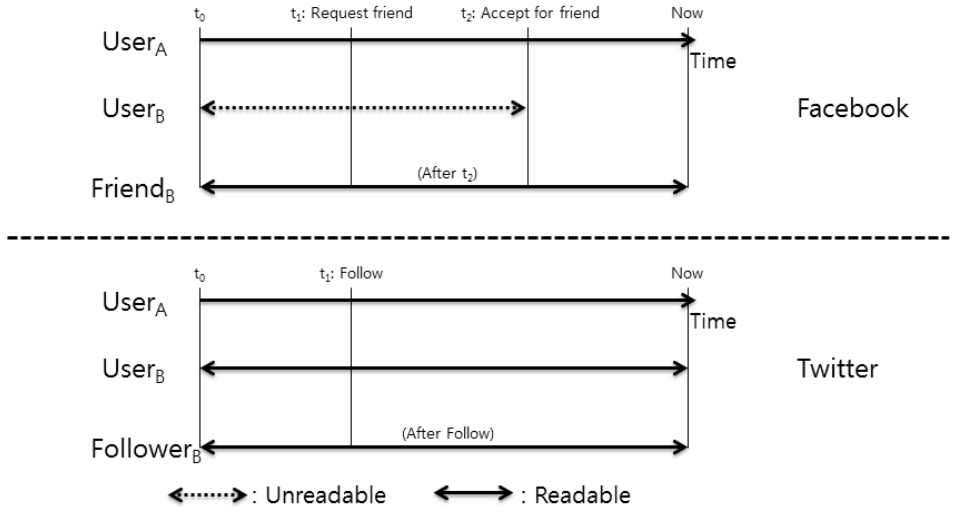
2.1 Facebook

Facebook은 2004년 2월 서비스를 시작하였으며, 2012년 12월 기준으로 매일 약 6억 명의 사용자가 Facebook을 이용하고 있다〔3〕〔4〕. Facebook은 API를 공개하여 웹의 사진, 동영상과 같은 정보뿐만 아니라 사용자들이 만드는 다양한 콘텐츠를 Facebook을 통하여 쉽게 공유할 수 있다.

Facebook에서 사용자 간 정보를 공유하기 위해서는 먼저 '친구' 라는 관계를 형성해야 한다. Facebook의 특징 중 하나는 기본적으로 '친구' 관계를 맺은 사용자 간에만 정보가 공유되는 것이다. '친구' 관계를 맺기 위해서는 상대방에게 '친구'요청을 해야 하고 상대방 또한 수락을 해야 '친구' 관계가 성립된다. 사용자A와 사용자B가 친구관계라고 가정하면, 사용자A와 사용자B는 서로 작성한 글을 모두 볼 수 있다. 또한 사용자A가 글 또는 콘텐츠를 Facebook에 게시할 때 마다 사용자B는 사용자A가 글 또는 콘텐츠를 게시하였다는 것을 실시간으로 확인할 수 있다.

2.2 Twitter

Twitter는 2006년 3월에 서비스를 시작하였으며, 2012년 7월 기준으로 약 5억 명의 사용자를 보유하고 있다〔5〕. Twitter 역시 Facebook과 같이 API를 공개하여 웹에 있는 다양한 정보를 Twitter를 통해 공유할 수 있다. Twitter도 Facebook과 같이 사용자 간에 관계를 맺어 정보를 공유한다. Facebook에서는 사용자 간에 '친구'라는 관계를 맺지만 Twitter는 ′Follow′라는 관계를 맺는다. 그러나 Facebook과 다르게 관계를 형성해야만 정보를 공유할 수 있는 것은 아니다. Twitter에서는 'Follow′ 관계가 아닌 사용자도 특정 사용자를 검색하여 검색된 사용자가 작



〔그림 1〕 Facebook과 Twitter에서 사용자가 읽을 수 있는 글의 범위

〔표 2〕 SNS에서 만족해야 하는 요구사항

보안요구사항	설명
데이터 보호	전송되는 데이터 및 서버에 저장되는 데이터는 권한이 없는 사용자로부터 보호되어야 함
- 기밀성	데이터는 암호화되어 전송 및 저장되어야 함
- 접근제어	서버에 저장되는 데이터는 사용자의 관계 및 권한에 따라 접근이 허가 또는 제한되어야 함
- 무결성	데이터의 위/변조를 막기 위해 무결성을 만족해야 함
사용자 인증	사용자를 인증하여 적절한 서비스를 제공해야 함
프라이버시 보호	사용자의 프라이버시를 보호해야 함
- Backward 프라이버시	관계 형성 이후의 글은 관계를 형성한 사용자만 열람할 수 있도록 제한해야 함
- Forward 프라이버시	관계를 형성한 사용자에게도 관계 형성 이전에 작성된 글에 대한 접근을 제한해야 함

〔표 3〕 표기법

표기법	설명
사용자A	사용자B와 관계를 형성하는 사용자
사용자B	사용자A에게 관계를 요청하는 사용자
ri	랜덤한 수
H	암호학적으로 안전한 해시 함수
hi	사용자i의 마스터 해시값
cur_hi	사용자i의 현재 해시값
counti	사용자i가 작성한 글 번호
Ek(m)	키 k를 이용하여 m을 암호화
Dk(m)	키 k를 이용하여 m을 복호화

성한 글을 모두 볼 수 있다. Twitter에서 'Follow' 관계는 임의의 사용자A가 Twitter에 글을 작성하면, 사용자A를 'Follow'하고 있는 모든 사용자가 실시간으로 확인할 수 있는 관계이다. 또한 Facebook에서 관계를 형성하기 위해서는 승인이 필요하였지만, Twitter는 관계를 형성하기 위해서 승인이 필요 없다.

2.3 Facebook과 Twitter 비교

Facebook과 Twitter의 차이점을 정리하면 〔표 1〕과 같다. Facebook이나 Twitter에서 사용자B가 사용자A와 관계를 형성하였을 때, 사용자A가 작성한 글 중 사용자B가 읽을 수 있는 범위를 표현하면 〔그림 1〕과 같다.

Twitter의 경우 관계 유무와 관계없이 모든 글을 기본적으로 열람할 수 있는 형태이며, Facebook은 관계 후 글을 열람할 수 있도록 제한되어 있다. 그러나 Facebook과 Twitter에는 한 번 관계를 형성하면, 관계 형성 이전의 글을 모두 열람할 수 있기 때문에 프라이버시 침해가 발생할 수 있다. 일반적으로 정보보안 시스템에 요구되는 보안 요구사항과 SNS가 만족해야 하는 보안요구사항을 결합하여 정의하면 〔표 2〕와 같다〔6〕.

III. 제안하는 방법

본 논문에서는 Facebook, Twitter와 같은 SNS에서 접근제어 및 해시체인을 사용하여 관계를 맺은 이후의 글만 읽을 수 있도록 제한하여 프라이버시를 보호하는 방법을 제안한다. 단, SNS의 전체 동작과정을 설명하지는 않으며, 기존의 SNS에서 프라이버시 보호를 강화하기 위해 개선한 부분만 설명한다.

3.1 가입

사용자A가 SNS에 가입 신청을 하게 되면, 기존에 서버에 저장되는 사용자A의 정보와 함께 〔표 4〕의 값들을 추가로 저장한다.

3.2 글 작성

사용자A가 새로운 글 m을 작성하면, m을 그대로 서버에 저장하지 않고 cur_hA를 키로 사용하여 m을 암호화하여 저장한다. 그리고 추가로 현재 m이 사용자A가 몇 번째 작성한 글인지 표시하는 countA를 함께 서버에 저장한다. 마지막으로 암호화된 m이 서버에 저장된 후, 서버는 사용자A의 cur_hA와 countA를 〔그림 2〕와 같은 알고리즘을 이용하여 업데이트 한다.

〔표 4〕 제안하는 방법에서 가입할 때 추가되는 정보

추가로 저장되는 정보	계산식	설명
hA	hA=H(ri)	마스터 해시값
cur_hA	cur_hA=H(hA)	작성한 글을 암호화 할 때 사용되는 키이며, 글을 작성할 때마다 해시 함수를 이용하여 업데이트 됨
countA	countA=1	사용자가 작성한 글의 일련 번호

〔표 5〕 제안하는 방법에서 관계 형성 시 추가되는 정보

추가로 저장되는 정보	계산식	설명
master_hBA	master_hBA=cur_hA	사용자B가 사용자A와 관계를 형성하였을 때, 사용자A의 cur_hA를 master_hBA로 저장
cur_hBA	cur_hBA=cur_hA	사용자B가 사용자A의 글을 읽을 때 복호화에 사용되는 키이며, 사용자A의 글을 읽을 때마다 해시 함수를 이용하여 업데이트 됨
master_countBA	master_countBA=countA	사용자B가 사용자A와 관계를 형성하였을 때, 사용자A의 countA를 master_countBA로 저장
countBA	countBA=countA	사용자A가 작성한 글 중 사용자B가 읽은 마지막 글의 일련번호

```
m′=Ecur_h(m);
store (m′, count);
Alg.Update(cur_h, count);

Alg.Update(cur_h, count) {
        cur_h = H(cur_h);
        count = count +1;
}
```

〔그림 2〕 cur_hA와 countA업데이트 알고리즘

3.3 관계 형성

사용자B가 사용자A와 관계를 형성하였을 때, 서버는 사용자A에 대하여 〔표 5〕와 같은 정보를 추가로 저장한다.

3.4 글 읽기

사용자B가 관계를 형성한 사용자A의 글을 서버에 요청하였을 때, 서버는 〔그림 3〕과 같은 과정을 통해 사용자B에게 결과를 전송한다.

Case1 countBA > countA

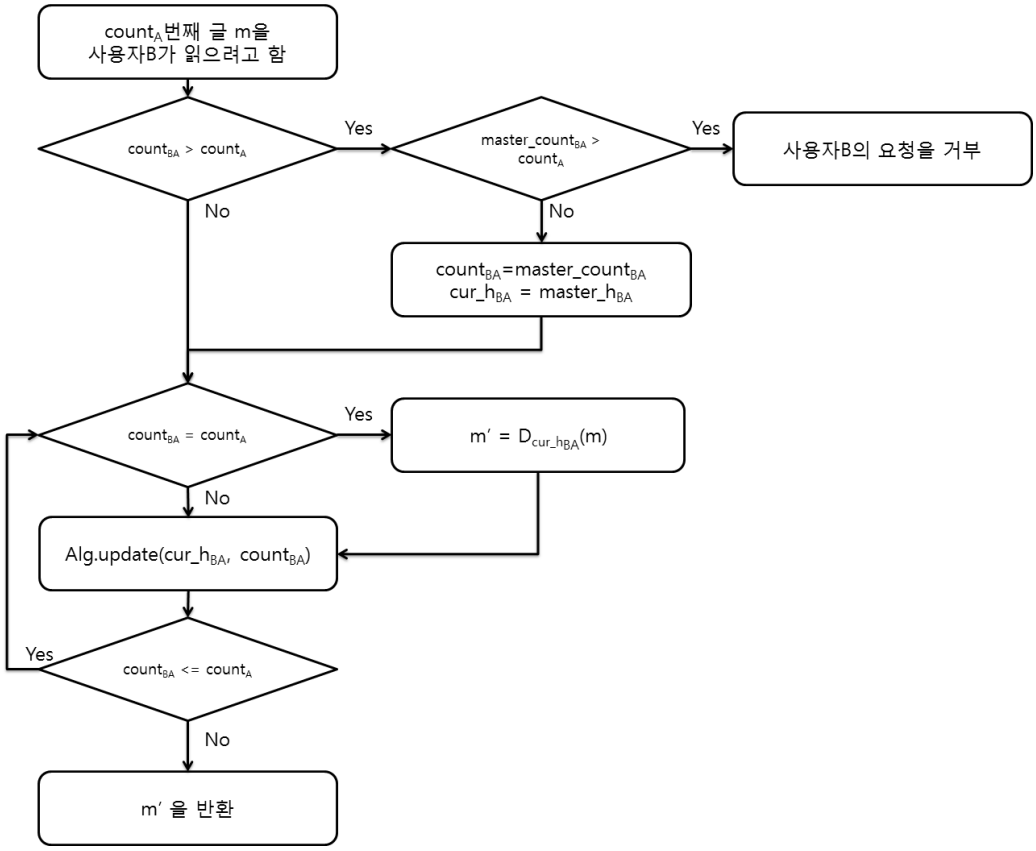
countBA가 m의 countA보다 크다는 것은 이전에 작성된 글을 읽으려는 것을 의미한다. 이러한 경우 우선 사용자B와 A의 관계 형성 시점을 확인하기 위해 master_countAB와 countA를 한 번 더 비교한다. master_countAB가 countA보다 크다는 것은 사용자B가 사용자A와 관계를 형성하기 이전의 글을 읽으려고 한 것이므로 사용자B의 요청을 거부한다. master_countAB가 countA보다 크지 않은 것은 현재의 글은 아니지만 관계 형성 이후의 글을 읽으려고 하는 것이므로 countBA=master_countBA, cur_hBA=master_hBA로 설정하고 Case2로 넘어간다.

Case2 countBA = countA

countBA가 m의 countA와 같은 것은 올바른 요청이므로 cur_hBA를 키로 사용하여 m을 복호화하여 m′을 얻는다. 그 후 Alg.update(cur_hBA, countBA)를 수행하여, cur_hBA와 countBA를 업데이트 한 후 m′을 반환한다.

Case3 countBA <= countA

countBA가 m의 countA보다 작거나 같은 것은 올바른 요청이다. 그러나 키가 업데이트가 되지 않은 상태이므로 countBA와 countA가 같을 때 까지 Alg.update(cur_hBA, countBA)를 수행한다. 그 후, cur_hBA를 키로 사용하여 m을 복호화하여 m′을 얻는다. 마지막으로 Alg.update(cur_hBA, countBA)를 한 번 더 수행하여, cur_hBA와 countBA를 업데이트 한 후 m′을 반환한다.



〔그림 3〕 제안하는 방법에서 글을 읽을 때의 동작 과정

3.5 완전 공개

사용자A는 사용자B에게 관계 형성 이전의 글을 읽을 수 있도록 허가할 수 있다. 사용자A가 사용자B에게 자신이 작성한 모든 글의 열람을 허용할 경우, 서버는 다음과 같이 값을 변경한다.

master_hBA = cur_hBA = H(hA)
master_countBA = countBA = 1

이와 같이 설정하면, 사용자B는 사용자A가 작성한 모든 글을 열람할 수 있는 권한을 얻게 된다.

IV. 분석

본 장에서는 본 논문에서 제안한 방식과 기존 SNS의 프라이버시 보호 수준을 비교한다. 그리고 제안한 방식을 기존 SNS에 적용하였을 경우 요구되는 추가 공간 및 연산 시간을 분석한다. 마지막으로 제안한 방식의 보안 분석을 수행한다.

4.1 추가 공간 분석

사용자 정보에 추가되는 데이터는 hA, cur_hA이다. hA, cur_hA는 해시 값으로 현재, 암호학적으로 안전하다고 알려진 해시함수의 길이는 256비트이다. 그리고 countA는 글을 작성할 때 필요한 일련번호이다. 1초에 글을 하나 작성하고, 인간의 수명이 80년이라고 가정하였을 때, count는 최대 2,522,880,000까지 증가할 수 있다. 이를 비트로 표현하면 32비트이므로 countA는 32비트면 충분하다. 따라서 사용자 정보에 추가로 필요한 공간은 544비트, 즉 68바이트이다.

그리고 사용자 간에 관계를 형성할 때, 추가되는 데이터는 master_hBA, cur_hBA, master_countBA, countBA이다. master_hBA, cur_hBA는 해

시값으로 현재 암호학적으로 안전하다고 알려진 해시 함수의 길이는 256비트이다. 그리고 master_count-BA와 countBA는 글을 작성할 때 필요한 일련번호이며, 32비트면 충분하다. 따라서 사용자 간에 관계를 형성할 때, 추가로 필요한 공간은 576비트, 즉 72바이트이다.

본 논문에서 제안한 방식을 적용함으로써, 추가로 필요한 공간을 정리하면 다음과 같다.

(256bit × 2 + 32bit) + (256bit × 2 + 32bit × 2) × n
(단, n은 관계를 형성한 사용자 수)

4.2 추가 처리 시간 분석

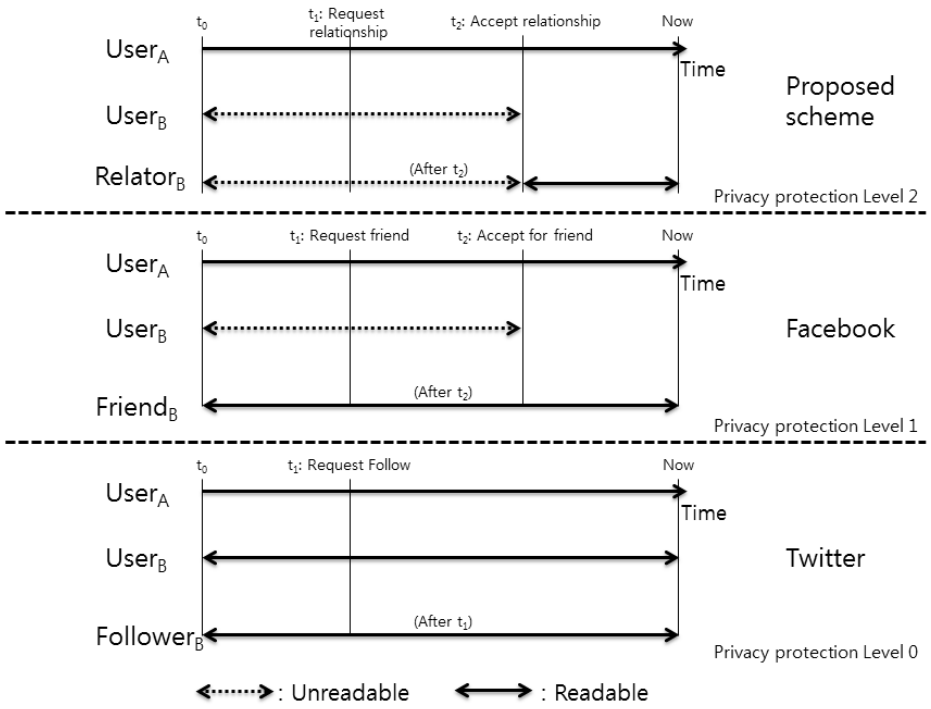
본 논문에서 제안한 방식을 적용하였을 때, 추가로 필요한 처리 시간을 분석하기 위해 현재까지 안전하다고 알려진 알고리즘을 선택하여 분석을 하였다. 해시 함수는 SHA-256을 사용하고, 암호 알고리즘은 128비트의 키를 사용하여 AES-CBC를 이용하는 것으로 가정하고 분석을 수행하였다. AES-CBC에 입력으로 들어가는 256비트의 키는 0~127비트는 IV로 사용되며, 128~255비트는 키로 사용된다.

〔표 6〕 제안하는 방법 적용 시 필요한 해시함수 및 암호 알고리즘

	SHA-256	AES-CBC(128)
가입	2	0
글 작성	1	1
글 읽기	1	1
합계	4	2

본 논문에서 제안한 방식을 적용하였을 때, 가입, 글 작성, 글 읽기에 추가로 필요한 해시, 암/복호화 수를 정리하면 〔표 6〕과 같다. 단, 글 읽기는 case2로 가정한다.

crypto++ 라이브러리를 Intel Core 2 1.83 GHz 프로세서, 32비트 Windows 비스타에서 구동했을 때 SHA-256은 0.275microsecond, AES-CBC는 1.225microsecond, AES-CBC의 IV 및 키 설정 시간이 0.569microsecond이다〔7〕. 단, SNS에서 작성되는 글의 길이는 다양한데, Twitter의 경우 기본적으로 글의 길이를 140바이트로 제한하므로 본 논문에서도 글의 길이를 140바이트로 가정하고 암/복호화했을 때의 시간을 계산하였다〔2〕. 제안한



〔그림 4〕 제안하는 방법과 기존SNS의 비교

〔표 7〕 제안하는 방법 적용 시 추가되는 시간,
(단위:microsecond)

	해시	암/복호화	합계
가입	0.550	0	0.550
글 작성	0.275	1.794	2.069
글 읽기	0.275	1.794	2.069
합계	1.100	3.588	4.688

방식을 적용했을 때 추가로 소요되는 시간을 정리하면 〔표 7〕과 같다.

위의 분석 결과와 같이 일반적인 사항 외에 최악의 경우에 대한 분석이 필요하다. 예를 들어, 한 사용자가 10,000명의 다른 사용자와 관계를 맺고 있고, 각 사용자마다 100개의 새로운 글을 작성했다고 가정했을 때, 소요되는 시간을 정리하면 다음과 같다.

$(0.275 \times 1.794) \times 100 \times 10{,}000 = 2{,}069{,}000$ microsecond $= 2.069$ second

즉, 10,000 명의 사용자가 각각 100개의 글을 작성한 것에 대해 글 읽기를 요청하여도 2.069 초 내에 처리가 가능함을 알 수 있다.

4.3 보안 분석

본 절에서는 본 논문에서 제안한 방법을 적용하였을 때 추가로 달성 가능한 보안 요구사항에 대해 분석을 한다. 기존 SNS에서 수행되고 있는 사용자 인증, 데이터 보호 중 접근제어, 무결성 및 Backward 프라이버시에 대한 분석은 생략한다.

데이터 보호 - 기밀성

본 논문에서 제안한 방식은 암호학적으로 안전한 해시함수를 이용하여 128bit의 키를 매번 갱신하며, 키를 AES-CBC로 암호화한다. 따라서, 데이터 보호를 만족한다.

프라이버시 - Forward 프라이버시

본 논문에서 제안한 방식은 사용자B가 사용자A와 관계를 형성하였을 때, 사용자B는 관계 형성 이후에 사용자A가 작성한 글만 열람할 수 있도록 제한하였다. 따라서, 관계 형성 이전에 작성된 글이 공유됨에 따라 침해받을 수 있는 프라이버시를 보호할 수 있다.

〔그림 4〕는 본 논문에서 제안하는 방법과 Facebook, Twitter에서 사용자B와 사용자A가 관계를 형성했을 때, 사용자A가 작성한 글을 사용자B가 읽을 수 있는 범위를 비교한 것이다. 또한 프라이버시 보호 수준에 따라 단계를 나누어서 표기하였다.

V. 결 론

SNS에서는 사용자 간에 관계만 형성이 되면, 관계를 맺은 사용자 간에는 작성한 모든 글은 볼 수 있으므로 프라이버시가 침해될 수 있다. 본 논문에서는 사용자가 작성한 모든 글을 암호화하고, 글을 작성할 때마다 키를 업데이트 하는 방법을 제안하였다. 제안한 방법을 SNS에 적용하면 관계가 형성된 이후의 글만 볼 수 있으므로 사용자의 프라이버시가 무분별하게 침해되는 것을 방지하였다. 또한 사용자가 관계 형성 이전의 글도 열람할 수 있는 방법도 제안하였다. 마지막으로, 본 논문에서 제안한 방식을 적용할 때 추가로 필요한 공간과 연산시간을 분석하였으며, 보안 분석을 하였다.

참고문헌

〔1〕 Cutillo, L.A., Molva, R. and Strufe, T., "Privacy preserving social networking through decentralization," Wireless On-Demand Network Systems and Services, 2009. WONS 2009. Sixth International Conference on, pp. 145-152, Feb. 2009

〔2〕 Cutillo, L.A., Molva, R. and Strufe, T., "Safebook: Feasibility of Transitive Cooperation for Privacy on a Decentralized Social Network," World of Wireless, Mobile and Multimedia Networks & Workshops, 2009. WoWMoM 2009. IEEE International Symposium on a, pp. 1-6, Jun. 2009

〔3〕 Key Facts - Facebook Newsroom, available at http://newsroom.fb.com/Key-Facts

〔4〕 Facebook - Wikipedia, the free encyclopedia, available at http://en.wikipedia.org/wiki/Facebook

〔5〕 Twitter - Wikipedia, the free encyclo-

pedia, available at http://en.wikipedia.org/wiki/Twitter
〔6〕 "Minimum Security Requirements for Federal Information and Information Systems", NIST FIPS PUB 200, Mar. 2006
〔7〕 Speed Comparison of Popular Crypto Algorithms, available at http://cryptopp.com/benchmarks.html

〈著者紹介〉

정 한 재 (Hanjae Jeong) 정회원
2006년 2월: 성균관대학교 컴퓨터공학과 졸업
2008년 2월: 성균관대학교 전자전기컴퓨터공학과 석사
2012년 8월: 성균관대학교 휴대폰학과 박사
2012년 9월~현재: 삼성전자 선임연구원
〈관심분야〉 정보보호, 모바일 보안, 취약성 분석

원 동 호 (Dongho Won) 종신회원
1976년~1988년: 성균관대학교전자공학과(학사, 석사, 박사)
1997년~1998년: 국무총리실 정보화추진위원회 자문위원
2002년~2003년: 한국정보보호학회 회장
2002년~2004년: 산학연 정보보안협의회 회장, 대검찰청 컴퓨터 범죄 수사 자문위원
2005년~2008년: 한국정보보호진흥원 이사
현재: 성균관대학교 정보통신공학부 교수, 정보통신대학원장
〈관심분야〉 암호이론, 정보이론, 정보보호