情報保護學會論文誌
第15券 第3號, 2005. 6

네트워크 스토리지에서 비대칭키 방식의 시 분할 권한 관리 (ATPM)*

김 은 미,[1†] 윤 효 진,[2‡] 천 정 희[2]
[1]어울림정보기술(주), [2]서울대학교 수리과학부 및 암호연구센터

Asymmetric Temporal Privilege Management on Untrusted Storage Server*

Eun Mi Kim,[1†] HyoJin Yoon,[2‡] Jung Hee Cheon[2]
[1]OULLIM Information Technology, INC.
[2]ISaC & Dept. of Math., Seoul National University

요 약

현재까지 네트워크 스토리지의 안전성은 스토리지 서버의 접근 제어에 의해 보장되어 왔으나 스토리지를 신뢰할 수 없다면 이 방법만으로는 자료의 안전성을 보장할 수 없다. 이러한 경우 안전성을 위해 자료의 암호화가 필수적이나 복호화 방법과 권한 관리 등 효율성 및 실용성에 여러 가지 문제가 발생한다. 우리는 이 논문에서 시간에 따라 효율적으로 스토리지의 자료를 암호화 및 복호화할 수 있도록 사용자의 권한을 관리하는 세 가지 방식으로 제시한다. 첫 번째 방식은 같은 시점에서 reader와 writer가 동일한 권한을 가질 경우 사용하는 스킴으로 트리 구조 일방향 함수에 기반을 둔 효율적인 시 분할 권한 관리 (TPM)이다. 두 번째는 공개 정보로부터 누구나 자료를 암호화하여 스토리지 상에 저장할 수 있고 특정 사용자만이 암호화된 자료를 복호화해서 read할 수 있게 하는 방식으로 비대칭키 방식의 시 분할 권한 관리 (ATPM)이다. 마지막으로 ATPM에서 시간에 따라 효율적으로 writer의 권한 관리를 가능하게 하는 방법을 제시한다. 제안된 권한 관리 방식은 모두 사용자의 가입이 효율적으로 이루어질 수 있으며 특히 TPM과 ATPM의 경우 back-issue 구독 문제를 해결할 수 있는 등의 장점이 있다.

ABSTRACT

We consider a network storage model whose administrator can not be fully trusted. In this model, we assume that all data stored are encrypted for data confidentiality and one owner distributes the decryption key for each time period to users. In this paper, we propose three privilege management schemes. In the first scheme, called Temporal Privilege Management (TPM), we use a symmetric encryption based on one-way function chains for key encapsulation. In the second scheme, called Asymmetric Temporal Privilege Management (ATPM), anyone can encrypt the data using the public key of owner, but only privileged users can decrypt the encrypted data. Finally, we present a scheme to restrict writers' privilege using ID-based signatures in ATPM. In our schemes, the privilege managements are based on the time and the addition of users is efficient. Specially, applying TPM and ATPM, we can solve the back-issue problem.

Keywords : *Privilege Management, Network Storage, Hierarchical ID-Based Encryption Scheme, ID-Based Signature Scheme*

접수일 : 2004년 12월 13일 ; 채택일 : 2005년 4월 13일
* 본 연구는 KT의 지원으로 수행하였습니다.
† 주저자 : emkim@oullim.co.kr
‡ 교신저자 : jin25@math.snu.ac.kr

Ⅰ. 서 론

네트워크 스토리지란 스토리지 사업자가 사용자에게 네트워크 상에서 일정 저장장소를 제공해 줌으로써 사용자가 언제 어디서나 자료를 저장할 수 있게 하는 서비스이다. 사용자가 네트워크 스토리지의 자료에 접근할 때 마치 자신의 컴퓨터에 있는 자료를 다루는 것과 같은 방식을 사용하면서도 휴대용 저장장치 등의 도움 없이 인터넷을 통하여 언제 어디서나 자료를 조회하고 수정할 수 있는 편리함이 있다. 최근 스토리지 시스템의 가격이 저렴해지고 대형화되며 네트워크의 고속화와 대중화에 힘입어 네트워크 스토리지에 대한 관심은 더욱 높아지고 있으며 이와 함께 그에 대한 안전성 문제도 심각히 고려되고 있다.

기본적으로 네트워크 스토리지에서는 스토리지 서버가 사용자의 접근 제어 (access control)나 사용자 인증 (user authentication) 등을 통하여 스토리지에 대한 정보 보호를 제공하고 있다. 그러므로 사용자가 자신의 정보들을 암호화하지 않고 저장한다면, 단지 서버에 의해서 자신의 자료와 스토리지 공간의 안전성이 결정된다. 즉, 만일 서버를 더 이상 신뢰할 수 없게 되면 사용자의 정보와 스토리지 공간도 더 이상 안전하지 않다. 실제로 현재 상용화되고 있는 네트워크 스토리지는 각각의 기업들에 의해 관리되고 있으며 그 서버를 완전히 신뢰할 수 있는 어떠한 근거도 없다. 따라서 보안이 요구되는 민감한 자료나 개인적인 자료들을 위한 네트워크 스토리지의 사용은 제한이 될 수밖에 없다. 이것이 네트워크 스토리지 사용이 매우 편리함에도 불구하고 사용자가 크게 증가하지 않는 이유이다.

1.1 중요성

네트워크 스토리지의 가장 큰 장점 중의 하나는 자료의 공유에 있다. 자료가 스토리지 소유자 개인의 하드웨어에 저장되어 있는 것이 아니기 때문에 누구나 네트워크로 연결되어 있으면 적절한 절차를 거쳐 스토리지 소유자의 자료에 접근할 수 있는 것이다. 이러한 손쉬운 자료 공유 때문에 현재 네트워크 스토리지는 그 사용 범위가 점점 더 넓어지고 있는 것이다. 위에서 언급한 바와 같이 서버를 신뢰하지 않는 경우 우리는 자료의 안전성을 보장하기 위해 자료를 암호화해서 저장해야 한다. 그러나 자료의 암호화 및 복호화는 스토리지 소유자가 직접 관리해야하므로 효율성과 실용성에 여러 가지 문제점을 야기한다. 사실 네트워크 스토리지를 스토리지 소유자의 자료 저장이라는 용도로만 사용한다면 암호화한 키를 스토리지 소유자 혼자만 잘 저장하고 적당히 업데이트 시켜주면 충분하나, 공유를 할 경우 그 자료를 읽고자 하는 사람들 (readers)과 스토리지에 자료를 쓰고자 하는 사람들 (writers)이 존재하게 되어 이들의 정당성을 위한 권한 관리의 문제가 발생한다. 만약 정당한 reader가 서버의 접근 제어를 통과하여 자료에 접근한다면 그 reader는 암호화된 자료만을 만날 수 있으며 이를 복호화하기 위해서는 스토리지 소유자로부터 직접 복호화할 수 있는 권한을 부여 받아야 한다.

이러한 권한 관리를 효율적으로 하기 위해서는 스토리지 소유자로부터 권한을 한 번 부여 받아 철회가 되지 않는 한 계속 이를 이용하여 자료를 복호화하는 것이 이상적이다. 이러한 개념의 자명한 해법은 Pay TV 등에 응용되는 broadcast encryption 기법 [9,10]에서 찾을 수 있다. 이러한 broadcast encryption 기법들은 사용자 단위의 권한 관리가 이루어지고 있지만 실제 스토리지에서는 사용자 단위보다는 파일 그룹 혹은 시간 단위에 따른 그룹 등 적절한 사용자 그룹에 대한 권한 관리가 더 효율적이다. 이에 Plutus [8]에서는 신뢰할 수 없는 스토리지에서 파일 그룹 단위의 키 관리를 통한 권한 관리 방법을 제시하였다.

한편, 스토리지에 writer가 원하는 시점 혹은 그룹만이 복호화할 수 있도록 자료를 쓸 수 있는 (암호화할 수 있는) 권한에 대한 관리도 필요하다. Writer의 권한 관리의 경우 자유 게시판이나 익명 게시판과 같이 누구나 자유롭게 자료를 쓸 수 있도록 하는 방법과 신문 기사 등 특정 writer만이 스토리지 소유자로부터 권한을 부여받아 쓸 수 있게 하는 방법을 생각할 수 있다.

Plutus의 경우, 자료를 쓰기 위해서는 서명키를 가져야 하는데 권한이 갱신될 때마다 소유자로부터 키를 새로 부여받아야하고 키 생성 또한 매우 비효율적이다. 즉, 스토리지 소유자는 키를 업데이트 할 때마다 reader (혹은 writer)에게 키를 안전하게 전송해야 하는 부담이 있다.

위와 같이 reader의 권한 관리와 함께 자유롭게 write 할 수 있는 방법으로 [4]등에서 제안된 비대칭키 기반의 broadcast encryption 방식을 생각해 볼

수 있다. 비대칭키 기반의 broadcast encryption 방식은 누구나 자신이 원하는 사용자들만이 복호화 할 수 있도록 자료를 암호화하는 방식을 목표로 한다. 그러나 이를 스토리지에 직접 적용하는 데는 몇 가지 문제점이 있다. 기본적으로 broadcast 방식은 사용자 단위의 권한 관리를 하기 때문에 대부분의 경우 사용자 추가가 비효율적이다. 그리고 스토리지의 경우 일반적인 broadcast encryption의 요구 사항과는 달리 back-issue 구독 또한 중요한 요구 조건이다. 즉, broadcast encryption과 같이 center가 자료의 저장 없이 실시간으로 자료를 한 번 뿌려주기만 하는 방법과 달리, 스토리지에서는 기존의 자료들이 계속 업데이트 되면서 저장되어 있으므로, 새로 가입한 가입자가 자신이 가입한 시점 이전의 자료들에 대하여 접근 권한이 있는지 없는지도 문제가 될 수 있다. 이에 스토리지에서의 권한 관리에 시간의 개념을 부여하는 것은 매우 필수적이고 적합한 것이다.

우리는 이 논문에서 서버를 신뢰할 수 없는 경우, 시간에 따라 스토리지에서의 권한 관리를 하는 효율적인 방법을 제시한다. 이 스킴에서는 누구나 원하는 시점에 자료를 올릴 수 있고, 스토리지의 소유주는 일정 기간동안만 자료를 복호화할 수 있는 키를 정당한 reader에게 주는 방식으로 reader의 권한을 관리한다. 부가적으로 특정 writer만이 자료를 올릴 수 있는 권한을 가지는 경우에 대한 효율적인 방법을 제시한다.

1.2 신뢰할 수 없는 스토리지

앞에서 언급한 바와 같이 우리는 스토리지 서버를 신뢰할 수 없다고 가정한다. 즉, 서버는 사용자가 저장해놓은 자료에 접근하여 단순히 자료를 읽을 수 있을 뿐만 아니라 변형시키거나 삭제할 수 있다고 가정한다. 서버가 자료를 변형시키거나 삭제하는 것은 해쉬함수를 이용하여 자료의 헤더에 해쉬값을 붙이거나 전체 자료의 해쉬값을 주기적으로 업데이트 시킴으로써 감지할 수 있다. 보다 기본적인 공격인 서버가 사용자의 자료에 접근하여 읽는 것을 방지하기 위해서는 자료를 암호화해서 저장하는 것이 필수적이다. 즉, 단순히 서버에 의한 접근 제어 (혹은 사용자 인증)로만 그치는 것이 아니라 스토리지의 소유자에 의한 또 다른 의미의 접근 제어- 소유자로부터 복호화키 (혹은 서명키)를 부여받은 정당한 사용자만이 자료에 접근하여 읽을 수 (혹은 쓸 수) 있다-을 수행한다는 것이다. 이 접근 제어의 경우 단순한 사용자뿐만 아니라 서버도 그 대상이 된다. 그러나 자료를 암호화하여 저장함으로써 안전성은 보장되지만 이로 인해 스토리지 관리의 효율성 측면에서 문제점이 발생한다. 암호화하지 않고 서버에만 의존했던 기존 방식에서는 자료 공유를 위한 사용자 관리가 전적으로 서버에 의해서 이루어졌지만 암호화해서 자료를 저장함으로써 정당한 사용자들이 자료를 읽기 위해서는 서버의 접근 제어를 통과해야 할 뿐 아니라 자료를 복호화하기 위해 스토리지 소유자와 직접 키 교환 절차를 거쳐야 한다. 또한 스토리지 소유자는 지속적으로 권한 관리를 해야 한다. 우리는 이 논문에서 암호화로부터 야기된 문제점을 개선할 수 있는 권한 관리 방식을 제안한다.

앞에서 언급한 바와 같이 파일을 안전하게 공유한다는 것은 신뢰할 수 없는 서버에 특정한 권한이 있는 사람만 자료를 암호화해서 저장할 수 있고 또 다른 의미의 권한이 있는 사람만 그 자료를 복호화해서 볼 수 있어야 한다는 것을 의미한다. 이 논문에서는 신뢰할 수 없는 스토리지 서버가 저장된 자료를 읽는 것을 방지하고 허가되지 않은 사용자가 서버와 공모하여 자료나 스토리지에 접근하는 것을 방지하기 위하여 자료를 암호화해서 저장하는데 초점을 맞추겠다.

이에 우리가 제시할 비대칭키 방식의 시 분할 권한 관리 방법 (ATPM) 에서는 다음을 가정한다:

① 서버는 자료를 읽거나 변형시킬 수도 있기 때문에 스토리지 공간의 스토리지 소유자는 서버를 신뢰하지 않는다.

② Reader와 writer의 권한을 독립적으로 관리한다.

1.3 Lockbox

자료를 암호화해서 저장할 때에 암호화하는 방식은 일반적으로 대칭키 방식을 이용한다. 하나의 파일을 하나의 암호화키로 암호화할 수 있다고 가정한다면 암호화해야 하는 파일의 개수와 전송 또는 저장해야 하는 복호화키의 개수가 같다. 이렇게 늘어나는 파일의 수에 비례하여 저장해야하는 키의 수가 늘어나는 것을 방지하기 위해서 우리는 그 키들을 lockbox라고 하는 상자에 넣어 또 다른 키로 암호화 한다.

실제로 스토리지 소유자가 서버에 단지 하나의 파일만을 저장하는 경우는 드물 것이다. 스토리지 소유자가 여러 개의 파일을 서버에 저장할 때 파일마다 다른 키를 사용한다면 각 사용자에게 파일의 수에 비례하는 키를 전송해야하고 그는 파일 개수만큼의 키를 관리해야 할 것이다. 그러나 여러 개의 파일을 암호화할 때 하나의 키로 암호화 한다면 한 개의 키를 업데이트할 때 모든 자료의 재 암호화가 수행되어야 한다. 이러한 문제를 해결하기 위해 많은 파일 시스템에서 파일 lockbox 키를 이용한다[5,11]. 즉 각기 다른 파일들을 각기 다른 키로 암호화한 후에 그 키들을 사용자들에게 모두 나누어주는 것이 아니라 그 키들을 모두 묶어 하나의 키로 암호화 한 후 그 마지막에 사용되는 암호화키만을 사용자들에게 나누어주는 것이다. 이때 마지막에 사용되는 암호화키를 파일 lockbox 키라 하고 파일 lockbox 키를 가진 사용자는 이것을 이용해서 lockbox를 풀어서 그 안에 들어있는 각 파일에 대응하는 키들을 모두 볼 수 있으므로 개별 파일들을 열어볼 수 있다. 이것은 결국 여러 개의 파일에 각기 다른 암호화키를 사용하면서도 사용자가 관리해야 하는 키는 단 하나가 되기 때문에 매우 편리하고 효율적인 방법이라 할 수 있다.

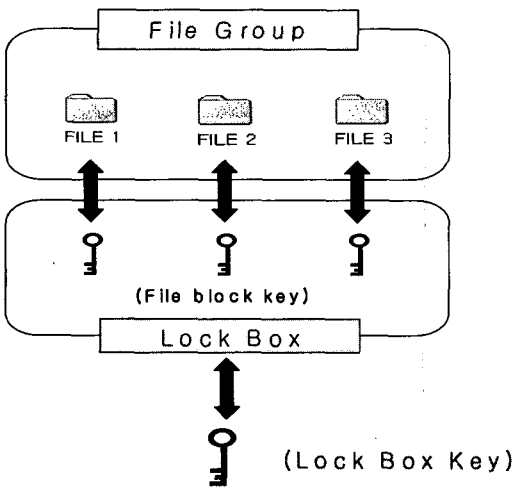

그림 1. Lockbox Key

1.4 의의

스토리지 서버를 신뢰할 수 없다고 가정하고 이러한 환경에서도 자료를 안전하게 저장하고 공유할 수 있는 효율적인 권한 관리 방법을 제시한다. 서버는 언제라도 자료를 읽거나 변형시킬 수 있다고 가정하기 때문에 기존의 서버에 의한 자료의 접근 제어만으로는 자료의 안전성을 보장할 수 없고 자료의 암호화가 필수적이다. 우리는 자료의 암호화로부터 야기되는 비효율성을 개선하기 위하여, 시간 단위로 사용자들의 권한 관리를 가능하게 하는 방법으로 *TPM*, *ATPM*, 그리고 *writer의 권한 관리 방법*을 제시한다. 이와 같은 시 분할 권한 관리 방법들은 사용자들이 원하는 기간 동안에 오직 하나의 키로 모든 암호화된 자료를 복호화해서 읽거나 쓸 수 있게 한다.

본 논문은 다음과 같이 구성되어 있다. 먼저 1장에서는 기본적인 파일시스템과 lockbox에 대해 알아보았다. 2장에서는 Plutus를 소개한다. 3장에서는 가능한 서비스 모델과 비대칭키 및 대칭키 기반의 권한 관리 방법을 제시한다. 4장에서는 알고리즘을 이해하기 위한 기본 도구들에 대해 설명한 뒤 구체적인 스킴들을 소개하면서 reader와 writer가 실제로 어떤 알고리즘을 수행하여 자료를 읽고 쓰는지에 대하여 알아보고 제시된 알고리즘들의 안전성을 분석하고 5장에서는 ATPM의 성질을 기존의 방법들과 비교 분석한다. 6장에서는 제시된 모델별로 적용 가능한 응용분야를 생각해보고 7장에서 결론을 정리한다.

II. Plutus

신뢰할 수 없는 스토리지 서버에서의 권한 관리 방식에 대한 기존의 연구로는 Plutus [8]가 대표적이다. 이 절에서는 Plutus의 권한 관리 방식에 대해 살펴본다.

기본적으로 대칭키 방식을 채택한 Plutus에서는 reader의 권한을 lockbox와 key rotation 방식을 이용하여 관리한다. 그러나 비대칭키 방식을 채택하여 reader의 권한 관리를 하기 때문에 writer의 권한 관리를 위해서는 또 다른 키를 관리 할 필요가 있다. 실제로 Plutus에서 writer의 권한 관리는 비대칭키 방식을 기반으로 하여 서명을 통해 이루어지며 서명키와 검증키가 서로 다르다. 구체적으로 Plutus의 권한 관리 방식은 다음과 같다.

2.1 Reader의 권한 관리

스토리지의 소유자는 RSA 암호 체계의 공개키와

비밀키를 사용한다. 즉, (e,N)이 스토리지 소유자의 공개키이고 d가 스토리지 소유자의 비밀키이며 $ed = 1 \pmod{\varphi(N)}$ 이다. K_0를 초기키라고 할 때 키의 revocation이 일어날 때마다 다음과 같이 키 순환 (key rotation)을 한다.

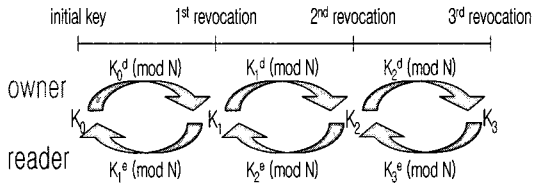

그림 2. 키 순환

위와 같은 키 순환 방식에 따르면 하나의 lock-box key를 알면 이전의 lockbox key들을 모두 알 수 있다. 즉, 어떤 순간의 lockbox key를 얻은 reader는 revocation 이전의 자료들은 모두 읽을 수 있다. 그러나 RSA 가정에 의해 이후의 자료들에 대한 lockbox key들은 구해낼 수 없다. 대칭키 방식을 이용하는 방식이므로 이 경우 writer에 대한 권한 관리는 반드시 별도로 이루어져야한다.

2.2 Writer의 권한 관리

Plutus에서 writer의 권한 관리는 서명에 의해 이루어지는데 서명 (signing)과 검증 (verifying)의 과정으로 이루어진다. 앞에서 언급한 바와 같이 이 과정은 서명키와 검증키가 다른 비대칭키 방식을 이용하고 있으며 revocation이 생길 때 마다 스토리지 소유자가 키를 업데이트한다. Reader는 업데이트 된 검증키를 lockbox key와 파일 헤더에 붙어 있는 자료를 이용하여 얻을 수 있으며, reader의 권한 관리에서 살펴본 바와 같이 이전 lockbox key를 얻을 수 있기 때문에 이전 검증키도 복구할 수 있다. 한편 서명키는 revocation이 발생하고 자료가 업데이트 될 때마다 스토리지 소유자가 생성하여 writer에게 직접 전해줘야 하며 이로 인해 저장공간, 전달 및 계산상의 효율성에 손실이 발생한다.

Ⅲ. 모델

앞에서 언급한 바와 같이 서버를 신뢰할 수 없는 경우 자료의 안전한 저장과 공유를 위해 자료의 암호화가 필수적이고 암호화된 자료를 효율적으로 공유하는 것은 일반적으로 쉬운 문제가 아니다. 이 장에서는 권한 관리의 어려움을 개선하기 위해 스킴에 앞서 서비스 모델 및 제안할 스킴의 모델을 설명한다. 여기에서 제안할 서비스 모델은 이해를 돕기 위해 예시로 만든 것이므로 서비스 상황에 따라 유동적으로 변형이 가능하다.

3.1 서비스 모델

우리가 생각하는 시나리오는 신뢰할 수 없는 스토리지 상에 자료를 저장해서 사용자들이 자료를 읽거나 쓸 수 있게 하는 것이다. 이에 사용가능한 서비스 모델의 예로 다음과 같은 모델을 생각할 수 있다.

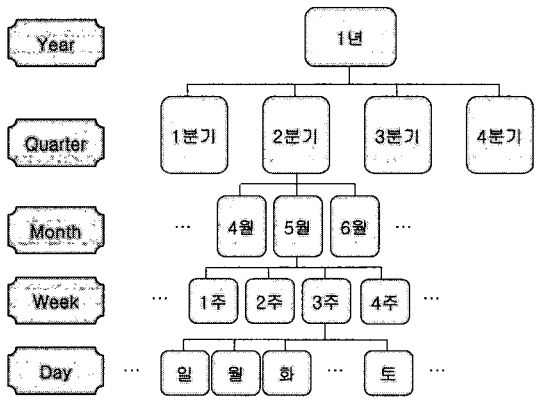

그림 3. 서비스 모델

위 모델의 장점은 서비스 기간을 1년으로 잡아도 모든 사용자들이 의무적으로 1년간 서비스를 이용해야 하는 것이 아니라 원하는 분기나 원하는 달, 또는 원하는 날을 골라서 그 기간에만 서비스를 이용할 수 있다는 것이다. 예를 들어, 매 시간 키가 업데이트 되는 자료가 있다고 하자. 그 자료에 대하여 시간마다 키를 전송받으면 1년간 365×24개의 키를 관리해야 한다. 이렇게 많은 키의 개수를 줄이고 원하는 기간동안 서비스를 받을 수 있도록 하는 것이 우리 서비스 모델의 목적이다. 연 회원이라면 키 1개로 1년간 자료를 볼 수 있고 분기 회원이라면 키 1개로 한 분기동안 자료를 볼 수 있다. 원하는 기간에 맞추어 적당히 시간단위를 조절할 수도 있다. 이것은 다양한 기호의 사용자들이 각기 다른 필요와 상황에 맞추어 서비스를 이용하는 것을 가능하게 하

기 때문에 실제 응용에서 매우 유용한 모델임에 틀림없다. 이 서비스 모델은 비대칭키 방식과 대칭키 방식에 모두 적용 가능하다. 3.2에서는 대칭키를 이용한 권한 관리 방법인 TPM 모델을 소개한다. 3.3에서는 비대칭키를 이용한 권한 관리 방법인 ATPM 모델을 소개한다.

3.2 TPM 모델

대칭키 방식의 시 분할 권한 관리 (TPM)는 자료의 암호화키와 복호화키가 같은 대칭키 방식을 사용한다. 해쉬함수의 체인을 이용하는 이 스킴은 보다 상위레벨의 키를 가진 사용자는 하위레벨의 키를 모두 계산해 낼 수 있다. 그러나 하위레벨의 키를 가진 사용자는 해쉬함수의 일방향성에 의해 상위레벨의 키를 계산할 수 없다. 여기서는 일정 기간동안 사용자들에게 writer의 권한과 reader의 권한이 동일하게 주어진다. 이것을 매우 현실적이고도 흔한 모델로서 인터넷 정보 공유 사이트를 생각하면 될 것이다. 또한 기업의 홈페이지에서 볼 수 있는 독자 게시판 등도 좋은 예가 될 수 있다. 누구나 글을 쓸 수 있고 누구나 글을 읽을 수 있다.

3.3 ATPM 모델

비대칭키 방식의 시 분할 권한 관리 (ATPM)는 스토리지 소유자의 공개키를 이용하여 누구나 자료를 암호화해서 올릴 수는 있지만 스토리지 소유자로부터 비밀키를 받은 특정 사용자만이 암호화된 자료를 읽을 수 있도록 하는 일종의 reader의 권한 관리 방식이다. 예를 들면 특허심사나 논문심사 등을 생각할 수 있다. 누구나 특허심사를 요청하는 자료를 서버에 올릴 수는 있지만 특정한 키를 가진 사용자만이 이 자료에 접근할 수 있다. 이것은 reader의 권한을 제한한 전형적인 경우로서 우리가 4장에서 제시할 ATPM은 이와 같은 경우의 일반화된 스킴이다.

3.4 Writer의 권한 관리 모델

TPM의 경우는 reader의 권한과 writer의 권한이 동일하게 주어지는데 반해 ATPM에서는 서로 독립적이다. 실제로 ATPM에서는 누구나 자료를 암호화하여 저장할 수 있다. 그러나 신문사 홈페이지에 기사를 올릴 수 있는 권한을 지닌 사용자는 오직 허가받은 기자들뿐이어야 함을 생각하면 ATPM을 사용하는 경우에도 writer의 권한 관리가 독립적으로 필요한 경우가 있다. 또한 기자들을 계약직으로 관리할 경우 계약 기간에 따라 권한의 박탈이 편리해야함 또한 자명하다. 이러한 경우 일정 기간동안 writer의 권한을 관리해 줄 수 있는 방법을 ID 기반 서명 스킴을 이용하여 구현할 수 있다.

Ⅳ. 권한 관리 스킴

우리는 이 장에서 TPM과 ATPM, 그리고 writer의 권한 관리에 대한 구체적인 스킴을 제시한다. 먼저 TPM은 해쉬 체인을 이용하여 만들었으며, ATPM은 겹선형 사상을 이용하고 [3,6]의 스킴을 적용 및 개선하여 만들었다. 또한 writer의 권한 관리는 Cha-Cheon의 ID 기반 서명 스킴을 응용 및 개선하여 제시한다.

먼저 스킴에 직접적으로 사용되는 admissible bilinear pairing 및 gap Diffie-Hellman group과 안전성 기반 문제들에 대한 내용을 정의한다.

4.1 정의

G를 위수가 소수 l인 덧셈에 관한 순환군이라 하고 P를 G의 생성원이라 하자. 또한 V를 위수가 l인 곱셈에 대한 순환군이라 하자.

■ Admissible Bilinear Pairing
사상 $e : G\times G\rightarrow V$가 다음의 성질을 만족할 때 우리는 이를 admissible bilinear pairing이라고 부른다.

1. Bilinear (겹선형성): $aP, bQ\in G$에 대하여, $e(aP, bQ)=e(P, Q)^{ab}$이다.
2. Non-degenerate: 모든 G의 원소 P(또는 Q)에 대하여 $e(P, Q)=1_V$이면, Q(또는 P)는 G의 항등원이다.
3. Efficient: $P, Q\in G$에 대하여, $e(P, Q)$를 효율적으로 계산하는 알고리즘이 존재한다.

타원곡선상의 Weil pairing과 Tate pairing을 이용하면 다음과 같이 admissible bilinear pairing을 얻을 수 있다. $q=p^n$이고 p가 소수일

때, E를 F_q위의 타원곡선이라고 하자. 소수 l과 E의 l torsion subgroup $E[l]$과 적당한 α에 대하여, 우리는 Weil pairinge$=E[l]\times E[l]\rightarrow F_{q^\alpha}^*$를 정의할 수 있다. 이제 $G=E(F_q)[l]$라 하고 ϕ를 G의 동형사상이라 할 때 사상 $e: G\times G\rightarrow F_{q^\alpha}^*$를 $e(P,Q)=e_W(P,\phi(Q))$이라고 정의하자. 그러면 e는 admissible bilinear pairing이 된다. Tate pairing을 이용하면 위와 비슷하게 admissible bilinear pairing을 정의할 수 있고 이는 Weil pairing보다 더 효율적이다[1].

■ 수학적 난제 및 Gap Diffie-Hellman Group

정의 1 (CBDHP) 주어진 G, V, e와 random한 $P, aP, bP, cP\in G$에 대하여 $e(P,P)^{abc}$를 구하는 문제를 Computational Bilinear Diffie-Hellman Problem이라 한다.

Computational Bilinear Diffie-Hellman Assumption이란 Computational Bilinear Diffie-Hellman Problem을 다항식 시간 안에 해결하기 어렵다고 가정하는 것이다.

정의 2 (DDHP) 주어진 $P, aP, bP, cP\in G$에 대하여, Z/lZ에서 $c=ab$인지를 결정하는 것을 Decisional Diffie-Hellman Problem이라 한다.

정의 3 (CDHP) 주어진 $P, aP, bP\in G$에 대하여, abP를 계산하는 것을 Computation Diffie-Hellman Problem이라 한다.

정의 4 (Gap Diffie-Hellman Group) G에서 DDHP가 효율적으로 계산 가능하고 G에서 무시할 수 없는 확률로 다항식 시간 안에 CDHP를 풀 수 있는 알고리즘이 존재하지 않으면 G를 Gap Diffie-Hellman Group (Gap DH Group)이라 한다.

정의 5 (DH tuple) 주어진 $P, aP, bP, cP\in G$에 대하여, (P, aP, bP, cP)가 정의 3을 만족시키면, (P, aP, bP, cP)을 유효한 Diffie-Hellman (DH) tuple이라 한다.

4.2 시 분할 권한 관리 (TPM)

우리는 이 절에서 해쉬체인을 이용하여 시 분할 권한 관리 방법 (TPM)을 제시한다.

H를 해쉬함수라 하고, S를 스토리지 소유자가 선택하여 비밀로 저장하고 있는 root 비밀키라고 할 때, 각 기간에 해당하는 비밀키를 다음과 같이 설계하여 각 기간의 권한을 가지고 있는 reader와 writer에게 분배한다.

- Year: $H(S,year)$
- Quarter: $H(H(S,year),quarter)$
- Month: $H(H(H(S,year),quarter),month)$
- Week: $H(H(H(H(S,year),quarter),month),week)$
- Day: $H(H(H(H(H(S,year),quarter),month),week),day)$

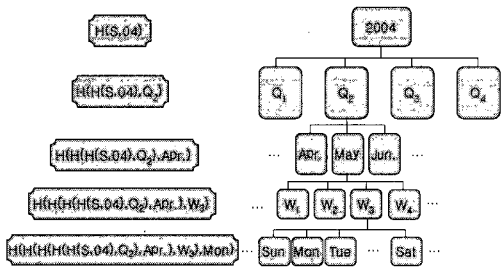

그림 4. TPM Key Distribution

예를 들어 2004년 2사분기 4월 셋째 주 월요일의 자료는 $H(H(H(H(S,2004),Q_2),April),W_3),Monday)$의 키로 암호화되고 2004년 연 회원, 2004년 2사분기 회원, 2004년 2사분기 4월 회원, 2004년 2사분기 4월 셋째 주 회원, 2004년 2사분기 4월 셋째 주 월요일 회원 모두 자신의 키로부터 복호화키를 복원할 수 있어 자료를 복호화할 수 있다. 또한 위의 키를 사용하여 자료를 암호화하여 스토리지에 저장할 수도 있다. 즉, reader와 writer의 권한이 같다.

4.3 비대칭키 방식의 시 분할 권한 관리 (ATPM)

자료의 업데이트와 사용자의 revocation이 자주 일어나는 경우 스토리지 소유자는 키를 자주 업데이트 시켜야 한다. 예를 들어, 스토리지에 뉴스를 매 시간 업데이트하는 경우 하루 혹은 한 달 동안 뉴스를 읽고자 하는 사용자는 매 시간 새로운 키를 스토리지 소유자로부터 받아야하고 이로 인해 bandwidth를 낭비하게 된다. 또한 사용자가 이미 지불

한 기사를 다시 읽기 위해서는 수많은 키들을 저장하고 있어야 하므로 사용자의 저장 공간에 부하가 걸리게 된다.

우리는 [3,6,7]과 같은 기존의 스킴들을 응용하여 스토리지 소유자는 한 개의 키를 전송하고 사용자는 한 개의 키만을 저장해서 원하는 기간 동안의 모든 자료들을 간단한 계산을 통해 복호화할 수 있는 다음과 같은 ID 기반 알고리즘을 제시한다. 또한 서비스 모델을 간략화하여 year, month, day 세 종류의 reader 권한에 대한 스킴을 서술한다. 물론 이 스킴은 앞에서 제시한 5가지 종류의 권한 관리는 물론 더욱 세분화되거나 통합된 시간 단위에도 적용 가능하다.

환경 설정: G는 위수가 소수 l이고 4.1에서 정의된 것과 같은 겹선형 사상 e를 가지는 군이라고 하고 두 해쉬함수 $H_1:\{0,1\}^*\rightarrow G$와 $H_2:G\rightarrow\{0,1\}^n$를 생각한다. 단, n은 메시지 공간의 크기이다. Z/lZ의 임의의 원소 s를 선택하고 G의 생성원 P를 선택하여 $P_{pub}=sP$를 계산한다. 이제 $param=(ID, G, e, P, P_{pub}, H_1, H_2)$를 공개하고 root 비밀키 $SK_\varepsilon=sH_1(ID)$를 계산하여 저장한다. 단, ID는 스토리지 소유자의 identity이다.

키 생성 및 분배: 연 회원 (y), 월 회원 (m), 일일 회원 (d)은 각자 회원의 상태에 따라 그에 적합한 키를 부여받는다. 단, y, m, d는 각각 해당하는 날의 연도, 월, 일을 대입한다.

① 연 회원 (y): Z/lZ의 임의의 원소 r_y를 선택하여 $R_y=r_yP$와 $S_y=SK_\varepsilon+r_yH_1(ID\|y)$를 계산하여 (R_y,S_y)를 복호화키로 준다.

② 월 회원 (m): Z/lZ의 임의의 원소 r_y (연 회원에서 선택한 값)와 r_m을 선택하여 $R_y=r_yP$, $R_{y|m}=r_mP$, $S_{y|m}=S_y+r_mH_1(ID\|y\|m)$을 계산하여 $(R_y,R_{y|m},S_{y|m})$을 복호화키로 준다.

③ 일일 회원 (d): Z/lZ의 임의의 원소 r_y (연 회원에서 선택한 값), r_m (월 회원에서 선택한 값), r_d를 선택하여 $R_y=r_yP$, $R_{y|m}=r_mP$, $R_{y|m|d}=r_dP$, $S_{y|m|d}=S_{y|m}+r_dH_1(ID\|y\|m\|d)$를 계산하여 $(R_y,R_{y|m},R_{y|m|d},S_{y|m|d})$을 복호화키로 준다.

암호화: 스토리지 소유자 혹은 writer들은 주어진 $param$, 해당하는 연 (y), 월 (m), 일 (d), 그리고 메시지 $M\in\{0,1\}^n$에 대해서 Z/lZ의 임의의 원소 γ를 선택하여 암호문 $C=(\gamma P,\gamma H_1(ID\|y),\gamma H_1(ID\|y\|m),\gamma H_1(ID\|y\|m\|d),M\oplus H_2(\alpha))$을 생성한다. 단, $\alpha=e(P_{pub},H_1(ID))^\gamma$이다.

이 알고리즘은 ID 기반 알고리즘이므로 누구나 시스템의 $param$만 알면 원하는 메시지를 암호화해서 스토리지에 저장할 수 있다.

복호화: 주어진 암호문 $C=(U_0,U_y,U_{y|m},U_{y|m|d},V)$에 대해 회원의 상태에 따라 가지고 있는 비밀키가 다르므로 주어진 암호문을 복호화하는 방법이 다르다.

① 연 회원 (y): 주어진 복호화키 (R_y,S_y)를 사용하여 $\alpha=\dfrac{e(U_0,S_y)}{e(R_y,U_y)}$를 계산하여 메시지 $M=V\oplus H_2(\alpha)$을 구한다.

② 월 회원 (m): 주어진 복호화키 $(R_y,R_{y|m},S_{y|m})$을 사용하여 $\alpha=\dfrac{e(U_0,S_{y|m})}{e(R_y,U_y)e(R_{y|m},U_{y|m})}$를 계산하여 메시지 $M=V\oplus H_2(\alpha)$을 구한다.

③ 일일 회원 (d): 주어진 복호화키 $(R_y,R_{y|m},R_{y|m|d},S_{y|m|d})$을 사용하여 $\alpha=\dfrac{e(U_0,S_{y|m|d})}{e(R_y,U_y)e(R_{y|m},U_{y|m})e(R_{y|m|d},U_{y|m|d})}$를 계산하여 메시지 $M=V\oplus H_2(\alpha)$을 구한다.

위에서 제시한 스킴에서는 연 회원 (y), 월 회원 (m), 일일 회원 (d)으로 그 시간단위를 제한하였지만, 필요한 서비스 모델에 따라서 시간단위를 훨씬 더 세분화시키거나 통합시킬 수 있다 (3개월, 6개월 등). 즉, ATPM은 서비스 모델에 따라 쉽게 변형이 가능하다. 또한 certificate을 첨가하면 ID 기반뿐 아니라 공개키를 기반으로 한 스킴에의 응용도 가능하다. 이 스킴의 안전성은 CBDHP에 기반을 둔다. 실제로 만일 다른 권한의 메시지를 복호화할 수 있는 공격자가 존재한다면 우리는 이를 이용하여 CBDHP를 풀 수 있다[6,3]. 그러므로 CBDHP가 어렵다고 가정하는 CBDHA 하에서 제안된 스킴은 안전하다.

4.2와 4.3에서 제시한 TPM과 ATPM의 효율성을 비교하면 다음과 같다. 표 1에서는 시간단위를 더 세분화하여 사용자의 상태를 연 회원 (y), 분기 회원 (q), 월 회원 (m), 주 회원 (w), 일 회원 (d)로 구분하였고 각각의 연산이 수행된 횟수를 표시하였다.

표 1. 효율성 비교표

스킴		TPM			ATPM		
연산		해쉬	pairing	상수배	해쉬	pairing	상수배
시간(ms)		1.67	·	·	0.01	55.44	2.69
암호화		5	0	0	7	1	6
복호화	y	4	0	0	1	2	0
	q	3	0	0	1	3	0
	m	2	0	0	1	4	0
	w	1	0	0	1	5	0
	d	0	0	0	1	6	0

위 표의 시간은 각 연산을 Mobile Intel Pentium 4-M, CPU 1.8 GHz, LAM 512 Mbytes에서 수행하였을 때 한번 연산하는데 걸리는 시간이다. 해쉬함수는 전용해쉬 MD-5 알고리즘으로 시간을 측정하였고, pairing은 *ID*-based signature scheme을 사용했을 때의 시간이며, 마지막으로 상수배는 160 bit $GF(p)$ 타원곡선에서의 시간을 측정하였다.

4.4 Writer의 권한 관리

우리가 제시한 ATPM 스킴은 누구나 스토리지 소유자의 공개키를 이용하여 자료를 암호화해서 write할 수 있는 반면에 특정 사용자만이 자료를 복호화해서 read할 수 있게 하는 reader의 권한 관리에 중점을 두었다. 그러나 서비스 모델에 따라서는 자료를 write할 수 있는 권한 관리도 필요한 경우가 있다. 이 경우, 기존의 reader group의 권한 관리와는 독립적으로 writer group의 권한 관리가 이루어져야 한다. 또한 모든 reader는 server의 도움 없이 정당한 writer가 쓴 자료인지 확인할 수 있어야 한다. 우리는 이 절에서 Cha-Cheon의 *ID*-based signature scheme [2]에 시간의 개념을 도입하여 스킴의 변경 없이 스토리지에 writer의 권한 관리를 효율적으로 할 수 있는 방법을 제시한다. 이 방법은 RSA 서명 스킴을 적용한 Plutus와 달리 writer의 권한이 시간과 함께 서명키에 포함되어 있으므로 사용자의 가입과 탈퇴에 따른 별도의 시스템 상의 부하가 걸리지 않는다. 구체적인 스킴은 기본적으로 Cha-Cheon의 *ID*-based signature scheme과 같고 다음과 같이 진행된다.

지금부터, 우리는 $G=<P>$가 큰 소수 l을 위수로 갖는 gap Diffie-Hellman group이라 하고 특별한 언급이 없는 한 모든 스킴은 group G에서 수행된다고 가정한다.

환경 설정: $G=<P>$를 위수가 소수 l인 gap DH group이라 하자. Z/lZ의 임의의 원소 α를 선택하고 G의 생성원 P를 선택하여 $P_{pub}=\alpha P$를 계산한다. 두 해쉬함수 $H_1:\{0,1\}^* \times G \rightarrow (Z/lZ)^*$와 $H_2:\{0,1\}^* \rightarrow G^*$를 선택하자. System parameter를 $param=(G,P,P_{pub},H_1,H_2)$으로 공개하고 master key α를 비밀로 저장한다.

키 생성: 주어진 identity *ID*에 대하여, $Q_{ID}=H_2(ID \| t_s \| t_t)$와 $D_{ID}=\alpha H_2(ID \| t_s \| t_t)$을 계산하고 identity *ID*의 비밀키로써 D_{ID}를 출력한다. 단, 여기서 t_s는 starting time이고 t_t는 terminal time이다.

서명: 주어진 비밀키 D_{ID}와 메시지 M에 대하여, 임의의 원소 $r \in Z/lZ$을 뽑고 $U=rQ_{ID}$, $h=H_1(M,U)$, 그리고 $V=(r+h)D_{ID}$일 때, 서명 $\sigma=(M,U,h,V,t_s,t_t)$를 출력한다.

검증: 메시지 M의 identity *ID*에 대한 서명 $\sigma=(M,U,h,V,t_s,t_t)$이 주어지면, $h=H_1(M,U)$을 계산한다. 서명이 받아들여진다는 것은 $(P,P_{pub},U+hQ_{ID},V)$가 DH-tuple인 것과 필요충분조건이다.

우리가 만약 G에서 admissible bilinear pairing e를 가지면, 우리는 다음과 같이 G에서 DDHP를 효율적으로 풀 수 있다:

(P,aP,bP,cP)이 valid DH tuple이다

$\Leftrightarrow e(aP,bP)=e(P,cP)$.

이 스킴의 안전성은 gap DH group에 의존한다. 실제로 서명을 위조할 수 있는 forger가 존재하

면 우리는 이를 이용하여 CDHP를 풀 수 있다 [2]. 우리는 gap DH group에서 스킴을 설계하였으므로 이로부터 제안된 스킴은 안전하다.

V. ATPM의 분석

이 장에서는 ATPM과 [8]에서 제안된 Plutus와 현재 가장 효율적인 broadcast encryption 스킴인 SD 스킴 [9], 이를 비대칭키 기반으로 적용시킨 [4]의 스킴을 비교, 분석한다.

표 2. 분석 및 비교

		Plutus [8]	BE [9]	PK_BE [4]		ATPM
암호화(writer)		별도 관리	center	Anyone		Anyone
관리 단위 (granularity)		파일 그룹	사용자	사용자		시간
Key Size (사용자=N)		O(1)	log N	pk	O(1)	d(*)
				sk	log2 N	d+1
Encryption Size		O(1)	2r(**)	2r • (log N)		d+2
키 유출	센터키	재발급	재발급	재발급		재발급
	사용자키	재발급	사용자 revocation	사용자 revocation		재발급
Back-Issue		O	X	X		O
가입		자유로움	기존키변경	기존키변경		자유로움

신뢰할 수 없는 네트워크 스토리지에서의 대표적인 권한 관리 방법으로 Ⅱ장에서 살펴본 Plutus가 있다. 앞에서 살펴본 바와 같이 Plutus에서 reader의 권한 관리는 대칭키 방식을 따르고 있기 때문에 writer의 권한 관리가 반드시 별도로 이루어져야 한다. 즉, writer의 권한을 행사할 수 있는 키를 별도로 발급 받아서 사용해야 하는 것이다. 이러한 권한 관리의 문제는 특정 사용자 집단만이 복호화할 수 있도록 암호화 한다는 측면에서 "Pay TV" 등에서 사용된 broadcast encryption (BE) 기법과 밀접한 관련이 있다.

그러므로 writer의 권한 관리를 부가적으로 하지 않기 위해 먼저 비대칭키 기반의 broadcast encryption (PK_BE) 기법을 스토리지에 적용시키는 방법을 생각해 볼 수 있다. 실제로 비대칭키 기반의 broadcast encryption 기법에서는 누구나 자신이 원하는 집단의 사람들만 복호화할 수 있도록 암호화하는 것을 목표로 하고 있으므로 별도의 키 분배 없이 writer의 권한 관리까지 가능하게 하는 우리의 목적에 부합한다. 그러나 [9]에서 제안된 broadcast encryption 기법 및 [4]에서 제안된 비대칭키 기반의 broadcast encryption 기법은 기본적으로 사용자 단위의 권한 관리를 하고 있으며 새로운 사용자가 가입할 때 기존 사용자들의 키를 바꾸어야 하는 등 가입이 자유롭지 않다. 또한 새로운 사용자는 그가 가입하기 이전 문서를 복호화할 수 없다. 즉, back-issue 문제를 해결할 수 없는 것이다.

ATPM은 시간 단위로 사용자들의 권한을 관리함으로써 보다 효율적으로 목적에 부합하는 권한 관리를 가능하게 한다. 또한 어떠한 시간 단위로든 사용자의 가입이 자유로우며 새로운 사용자의 권한이 미치는 범위 한에서 이전 자료들에 대하여 자유롭게 복호화 가능하다. 예를 들어, 3월에 연 회원 권한으로 가입한 회원은 1월, 2월의 자료들도 복호화할 수 있는 것이다.

기본적으로 ATPM에서는 누구나 원하는 시간에 자유롭게 write할 수 있다. 또한, 만약 writer의 권한을 제어하기를 원한다면 4.4절에서 제시된 바와 같이 ID 기반 서명을 바탕으로 Plutus보다 간단한 방법으로 writer의 권한 관리가 가능하다. 실제로 Plutus에서는 writer의 권한 관리를 위해서 reader의 키와는 별도로 RSA를 기반으로 하는 비대칭키를 이용하는 서명 방식을 이용하고 있으나 사용자가 revoke 될 때마다 RSA parameter들을 다시 생성해야 하고 그 키를 모든 정당한 사용자들에게 전달해야 하는 등 매우 비효율적이다. 그러나 제안된 스킴에 의하면, 서명키에 사용자의 ID와 유효 기간이 포함되어 있으므로 다른 사용자들에게 다시 키 전달을 하거나 새로운 키를 생성해야 하는 등의 부가적인 작업 없이 정당하지 않은 writer를 자동적으로 revoke 시킬 수 있다.

Ⅵ. 응 용

우리가 제시한 main scheme에서는 스토리지 소유자의 공개키를 이용해 누구나 자료를 암호화해서 write할 수 있지만 특정 사용자만이 암호화된 자료를 복호화해서 read할 수 있었다. 이렇게 reader의 권한 관리에 중점을 둔 scheme이 실제 사용될 수 있는 시나리오는 우리 주변에서 쉽게 찾을 수 있다.

'인터넷 사진관'의 경우에는 누구나 자신의 디지털 사진기로 찍은 사진을 현상하기 위해 인터넷 사진관

의 스토리지에 자료를 저장할 수 있다. 그러나 사용자들은 스토리지에 저장된 사진을 아무나 열어보기를 원하지 않기 때문에 이 자료를 암호화해서 올려야 한다. 이 사진을 복호화해서 read할 수 있는 사용자는 인터넷 사진관의 직원들로 제한되어야 한다.

'출판사'의 경우에는 책을 출간하고 싶은 사용자는 누구나 자신이 집필한 내용을 암호화해서 출판사의 스토리지에 저장할 수 있고 비밀키를 가진 특정 출판사 직원만이 이 자료에 접근할 수 있어야 하기 때문에 ATPM의 응용에 해당한다고 할 수 있다.

'특허심사'나 '논문심사'의 경우도 마찬가지이다. 누구나 자신의 아이디어나 논문으로 특허심사나 논문심사를 요청하는 자료를 심사기관의 스토리지에 저장할 수 있지만 이렇게 저장된 자료는 철저한 보안을 요구하는 자료들이므로 반드시 암호화되어야 한다. 이 자료들을 복호화해서 열어볼 수 있는 사용자는 특정 심사위원들로 제한되어야 한다.

4.4에서 우리는 writer group의 권한 관리방법을 추가로 제시하였다. 우리는 writer의 권한 관리가 필요한 시나리오 역시 쉽게 찾을 수 있다.

'신문사의 홈페이지'의 경우를 살펴보면, 신문사 홈페이지에는 아무나 기사를 써서 올려서는 안 될 것이다. 따라서 신문사 (스토리지 소유자)는 몇몇 기자들 (writer)에게 신문사 홈페이지 (스토리지)에 글을 올릴 수 있는 권한을 사전에 부여하고 이 권한을 부여받은 소수의 사용자만이 글을 쓸 수 있는 권한을 가져야 한다. 물론 이때 진짜 권한을 가진 사람이 글을 썼다는 서명이 필요하다. 권한을 부여받은 기자들이 쓴 신문사 홈페이지의 기사는 누구나 읽을 수 있다.

또한 이 신문사에서 홈페이지를 유료로 운영하고 싶다면 writer group의 권한 관리와 reader group의 권한 관리를 동시에 수행함으로써 사전에 권한을 부여받은 기자들만 기사를 쓸 수 있고 사용료를 지불한 독자들만 기사를 읽을 수 있게 할 수 있다. 이러한 시나리오에서는 writer group의 권한 관리와 reader group의 권한 관리는 철저히 독립적으로 이루어진다. 이것은 4.4에서 제시된 scheme의 전형적인 응용이 될 수 있다.

Ⅷ. 결 론

우리는 이 논문에서 서버를 신뢰할 수 없다는 가정 하에 네트워크 스토리지 상에서 안전하게 자료를 저장 및 공유하는 효율적인 권한 관리 방식을 제안했다. 네트워크 스토리지는 자료의 저장 뿐 아니라 손쉬운 자료 공유의 목적으로도 중요한 의미를 지니고 있다. 이에 일정 기간동안 reader와 writer가 같은 권한을 가지는 대칭키 기반 권한 관리 방식인 TPM과 정당한 reader가 보다 효율적으로 암호화된 자료에 접근할 수 있는 비대칭키 기반의 권한 관리 방식인 ATPM을 제시하고, 마지막으로 서명을 이용한 독자적인 writer의 권한 관리 방식 또한 제시했다. 우리는 권한 관리에 시간 개념을 도입하여 기존의 결과에 비해 효율적인 권한 관리 방법을 제시하였으며 back-issue 구독 문제와 사용자의 자유로운 가입 문제를 해결하였다.

참 고 문 헌

[1] D. Boneh, B. Lynn and H. Shacham, *"Short Signatures from the Weil Pairing,"* in Advances in Cryptology - Asiacrypt 2001, LNCS 2248, pp. 514-532, 2001.

[2] J. C. Cha and J. H. Cheon, *"An Identity-Based Signature from Gap Diffie-Hellman Groups,"* in PKC 03, LNCS 2567, pp. 18-30, 2003.

[3] R. Canetti, S. Halevi and J. Katz, *"A Forward-Secure Public-Key Encryption Scheme,"* in Advances in Cryptology - Eurocrypt 2003, LNCS 2656, pp. 255-271, 2003.

[4] Y. Dodis and N. Fazio, *"Public Key Broadcast Encryption for Stateless Receivers"*, in ACM-DRM, 2002.

[5] K. Fu, *"Group Sharing and Random Access in Cryptographic Storage File Systems,"* in Master's thesis, MIT, 1999.

[6] C. Gentry and A. Silverberg, *"Hierarchical ID-Based Cryptography,"* in Advances in Cryptology - Asiacrypt 2002, LNCS 2501, pp. 548-566, 2002.

[7] J. Horwitz and B. Lynn, *"Toward Hierarchical Identity-Based Encryption,"* in Advances in Cryptology -

Eurocrypt 2003, LNCS 2332, pp. 466-481, 2002.

[8] M. Kallahalla, E. Riedel, R. Swaminathan, Q. Wang and K. Fu, *"Plutus : Scalable Secure File Sharing on Untrusted Storage,"* in FAST 03, pp. 29-42, 2003.

[9] D. Naor, M. Naor, and J. Lotspiech. *"Revocation and Tracing Schemes for Stateless Receivers,"* in Advances in Cryptology - Crypto '01, pages 41 {62, Berlin, 2001. Springer-Verlag. Lecture Notes in Computer Science Volume 2139.

[10] Arvind Narayanan, C. Pandu Rangan, Kwangjo Kim, *"Practical Pay TV Schemes"*, in ACISP 2003, 192-203, 2003.

[11] E. Riedel, M. Kallahalla and R. Swaminathan, *"A Framework for Evaluating Storage System Security,"* in FAST 02, pp. 15-30, 2002.

〈著者紹介〉

김 은 미 (Eun Mi Kim)
2002년 2월: 중앙대학교 수학과 학사
2005년 2월: 서울대학교 수학과 석사
2005년 3월~현재: 어울림정보기술(주) NSP 개발실 연구원
〈관심분야〉 공개키 암호 이론, 정보보호

윤 효 진 (HyoJin Yoon)
1999년 2월: 서울대학교 수학교육과 학사
2001년 8월: 서울대학교 수학과 석사
2001년 9월~현재: 서울대학교 수학과 박사과정
〈관심분야〉 공개키 암호 이론, 정보보호

천 정 희 (Jung Hee Cheon)
1997년 2월: 한국과학기술원 수학과 박사
1997년 3월~2000년 1월: 한국전자통신연구원 선임연구원
2000년 1월~2000년 12월: Brown 대학 박사후 연구원
2000년 12월~2003년 2월: 한국정보통신대학교 공학부 조교수
2003년 3월~현재: 서울대학교 수리과학부 부교수
〈관심분야〉 응용정수론, 암호론, 응용암호론