情報保護學會論文誌
第14券 第2號, 2004. 4

P2P 환경의 자기 평판 관리 시스템

조 남 수[a)†‡], 김 우 환[a)], 윤 효 진[a)], 이 인 석[a)], 천 정 희[a)], 김 태 성[b)], 진 승 헌[b)], 추 경 균[c)]
서울대학교[a)], 한국전자통신연구원[b)], 행정자치부[c)]

Self-Reputation System in P2P Networks

Nam Su Jho[a)†‡], Woo Hwan Kim[a)], Hyo Jin Yoon[a)], In-Sok Lee[a)], Jung Hee Cheon[a)],
Tae-sung Kim[b)], Seung-hun Jin[b)], Kyung-kyun Choo[c)]
Seoul National University[a)], ETRI[b)],
Ministry of Government Administration and Home Affairs[c)]

요 약

평판 기반 시스템(reputation-based system)은 현재 활발히 사용되고 있는 P2P(Peer-to-Peer) Network System에서 가장 주목받고 있는 인증 방식이다. 본 논문에서는 새로운 평판 기반 시스템인 Self-Reputation system을 제안한다. Self-Reputation system은 사용되어지는 P2P network에 무관하게 적용 가능하기 때문에, 여러 가지 P2P 네트워크에서의 평판을 하나로 통합 관리할 수 있는 방안을 제공한다. 또한, 이 시스템은 기존의 Xrep 방식에서 broadcast를 통해서 평판을 수렴하는 방법 대신 각 사용자들이 자신의 평판 테이블을 관리하도록 하여 평판을 알아보는 과정에서의 통신량을 현격하게 줄였고, 그 결과로 효율성을 크게 향상시켰다.

ABSTRACT

Though P2P network systems are widely used, not so much research has been done on security issues. One of the serious problem in P2P network is the authentication problem. To resolve this problem, we introduce a new concept, a "self-reputation system" in which a user manages her own reputation. We define self-reputation systems by presenting several requirements. We also give one instance of self-reputation system. The proposed instance satisfies the requirement including the prevention of erasing and the platform independence.

Keywords: *Self-Reputation, Reputation, P2P, XRep*

Ⅰ. 서 론

Peer-to-Peer(P2P) Network System이란 모든 사용자가 각각 서로 동등한 입장에서 파일을 공유하고 교환하는 체계이다. 초기의 모델인 Napster와 소리바다는 centralized system으로, P2P를 일반 대중에게까지 가능하게 해주었다는데 큰 의미가 있다. 그 결과 Napster의 경우 6천만 명이라는 엄청난 회원 수를 자랑하는 정도에까지 이르렀으나 저작권 소송 문제로 현재는 더 이상 서비스를 제공하지 않고 있으며, 최근에는 이러한 실패를 거울삼아 Gnutella와 같은 decentralized 시스템을 이용한 KaZaA, eDonkey 등의 프로그램들이 널리 사용되고 있다.

접수일: 2003년 12월 9일; 채택일: 2004년 3월 15일
* 본 연구는 한국전자통신원 정보보호연구본부 지원으로 수행되었음.
† 주저자, ‡ 교신저자 : drake@math.snu.ac.kr

그러나, P2P 시스템이 점점 우리의 생활에 밀접한 영향을 끼치고 있음에도 불구하고 현재까지의 P2P 시스템에서의 보안은 아주 취약한 실정이다. 이러한 가운데 P2P 시스템에 평판 기반의 보안 체계를 구축하고자 하는 연구들이 있었다. 2002 년도에 제안된 Gnutella에 평판을 수렴하는 과정을 첨가한 Xrep 방식이 그러한 한 예라고 할 수 있다. 앞으로 살펴보겠지만 Xrep 방식은 broadcast를 이용하여 평판의 수렴을 하기 때문에 네트워크의 통신량을 증가시킨다는 단점이 있다. 또한 앞으로의 P2P의 주류가 될 것으로 보여지는 DHT 방식의 P2P 네트워크에서의 적용도 어려워 보인다.

우리는 이 논문을 통해 평판 관리 시스템의 새로운 개념인 Self-Reputation을 소개하고, 이를 적용한 시스템을 제안하였다. 이 것은 투표를 통해서 평판을 수렴하는 기존의 평판 관리 시스템과 비교할 때 효율성에 있어서 매우 뛰어나다고 할 수 있으며, 특히 특정 P2P 네트워크의 특성에 영향을 받지 않기 때문에 많은 적용 가능성을 지니고 있다. 앞으로의 P2P 환경에서는 여러 새로운 P2P 네트워크 방식이 생겨나고, 또 사용자들이 한 가지 P2P만을 사용하는 것이 아니라 여러 P2P 네트워크를 복합적으로 사용할 것이라고 예측되므로 이러한 특성은 큰 의미를 지닌다.

본 논문은 다음과 같이 구성되어 있다. 먼저 2장에서는 현재 널리 사용되고 있는 P2P 모델인 Gnutella의 프로토콜을 설명하고, Gnutella를 보완하여 만들어진 Xrep 시스템을 소개한다. 3장에서는 자신이 직접 자기의 reputation을 관리하는 'self-reputation system'의 개념을 설명하고, 일례로 RCert 시스템을 살펴보겠다. 그 후에 새로운 시스템을 제안한다. 4장과 5장에서는 각각 제안된 시스템을 분석하고 기존의 Xrep, RCert와의 비교를 다룬다. 마지막으로 6장에서 결론을 정리한다.

II. 기존의 평판 기반 시스템(Xrep)

2.1 P2P 프로토콜

P2P 프로토콜이란 서버(server)와 사용자(client 혹은 user)에 기반한 프로토콜과 상반되는 개념으로 개인 대 개인(peer-to-peer)으로 정보를 검색하고 공유하는 것이다. P2P 소프트웨어는 메신저 프로그램으로부터 시작하여 Napster와 같은 파일 공유 프로그램 등을 통해 급성장하였다. P2P 프로토콜은 centralized system, unstructured decentralized system, structured decentralized system으로 크게 세 가지로 분류할 수 있다.

현재 가장 많이 사용되어지고 있는 Gnutella[1]는 unstructured decentralized system의 가장 대표적인 기술 중의 하나이다. Gnutella는 중앙서버에서 개입하는 부분이 전혀 없이 각각의 peer가 server이자 client의 역할을 수행하는 대칭성을 가진다. 이런 의미에서 각 노드는 servent(server+client)의 역할을 수행한다고 한다. 여기에서 Gnutella의 동작 원리와 사용되는 주요 protocol들을 간단히 살펴보도록 한다.

처음 Gnutella system에 접속을 시도할 경우 Gnutella를 이용한 소프트웨어 패키지 중 하나를 download 받고, Gnutella network 상의 다른 servent를 찾아서 연결을 시도한다. 여러 개의 노드에 연결을 요청(request)한 뒤 실제 network 상에 존재하고 있고 요청을 받아들이는 노드와 연결이 됨으로써 사용자의 컴퓨터 또한 Gnutella network 상의 servent가 된다.

파일을 download 받고자 할 때 다음의 5가지 protocol을 따른다. 이러한 프로토콜에 사용되는 routing의 기본은 broadcasting이다. 즉 자신과 연결된 모든 노드들에 자신이 받은 protocol 들을 전파하는 것이다.

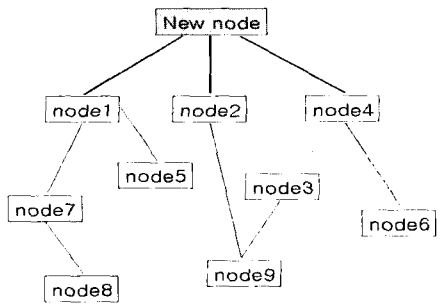


그림 1. Gnutella 새 노드 추가

표 1. Gnutella Protocol

Ping	network 상의 활동 중인 node을 점검
Pong	Ping에 대한 활동 중인 node들의 응답
Query	network 안에서 원하는 file을 찾는 요청
QueryHit	Query에 대한 응답
Download(Push)	file이 요청자에게 전해지는 과정

Unstructured system에서는 자료의 검색이 이웃하고 있는 노드 간의 routing에 의해서 이루어지므로 비효율적일 수 있으며 이를 보완한 것이 DHT(distributed hash table)을 이용한 시스템으로 structured decentralized system이라 할 수 있다. DHT를 이용한 system에서는 각 자료에 key(보통 파일명의 hash 함수값)를 대응시킨 후 key 값에 대응되는 node에 실제 자료를 제공하는 node의 IP 주소를 저장함으로써 원하는 자료를 효율적으로 검색할 수 있도록 한다. CAN, Pastry, Tapestry 등이 DHT를 이용한 시스템이다.

2.2 Xrep 기법

2.2.1 도입배경

P2P 시스템에서는 사용자의 익명성 때문에 그들이 제공하는 자료들에 대해 신뢰성이 부족하다. 실제로 소리바다나 Napster, KaZaA, 그리고 Fast Track 등의 P2P 프로그램에서 어떤 노래를 찾고자 할 때 우리는 아무 정보 없이 파일 제공자가 제공하는 정보만으로 그 파일에 대한 모든 것을 판단한다. 그 파일이 실제로는 virus나 worm 등을 포장해 놓은 것일 수도 있고 매우 질이 낮은 것일 수도 있지만 우리는 그것을 판단할 수 있는 어떠한 정보도 가지고 있지 않은 것이다.

이러한 문제점들에 대한 일차적인 해결 방안으로 사용자들의 경험에 의거한 의견이나 소문 등을 생각할 수 있으며 실제로 여러 인터넷 사이트에서 사용되고 있다. 일상 생활에서의 예를 들면 우리는 식사를 하고자 할 때 일단 인터넷 상의 여러 외식업체들에 대한 평가와 경험담들을 소개해놓은 사이트를 참고하기도 하고, Amazon.com과 같은 사이트에서 원하는 책에 대한 서평을 보고 책을 구입한다든지, 인터넷 쇼핑몰에서는 물건에 대한 품평들을 참조하여 물건을 구입하는 등의 행동을 한다. 또한 e-bay나 옥션 등에서 거래를 할 때 판매자와 구매자에 대한 평가가 거래에 대한 큰 잣대가 된다. 그러나 그러한 의견이나 소문들의 신뢰도 혹은 위조 가능성이 문제가 될 수 있다. 이러한 문제점에 대한 해결 방안으로 몇 가지의 방법이 제시되고 있는데 그 중 평판(reputation)에 기반한 시스템이 가장 주목할 만하다. 현재, 이 부분의 연구는 활발히 진행중이지만, 결과는 미비한 실정이다.

2.2.2 Xrep-시스템[2]

우리는 이 장에서 Damiani 등에 의해 2002년 발표된 Xrep-프로토콜에 대해 살펴본다. Xrep-프로토콜에서는 각 개인이 다른 사용자들에 대한 경험과 정보를 저장하고 이에 대하여 사용자의 요구가 있을 시 이를 공유하도록 한다. 이 때 각 사용자(servent)는 자신의 공개키를 hash한 값을 $servent_{id}$로 사용하고 자료의 내용을 hash한 값을 $resouce_{id}$로 사용한다.

그리고 각 사용자는 두 가지의 평판을 관리하는데 하나는 사용자 평판으로 $servent_{id}$에 대응되는 값이고, 다른 하나는 자료에 대한 평판으로 $resouce_{id}$에 대응되는 값이다. 이 시스템은 다음과 같은 다섯 가지 단계로 각 사용자와 자료에 대한 평판을 알아내고 자료를 공유한다.

표 2. XRep 프로토콜

첫 번째 단계	자원찾기(resource searching)
두 번째 단계	자원선택 및 투표묻기 (resource selection and vote polling)
세 번째 단계	투표평가(vote evaluation)
네 번째 단계	최적제공자검증(best servent check)
다섯 번째 단계	자원받기(resource downloading)

이 프로토콜을 도식화하면 다음과 같다.

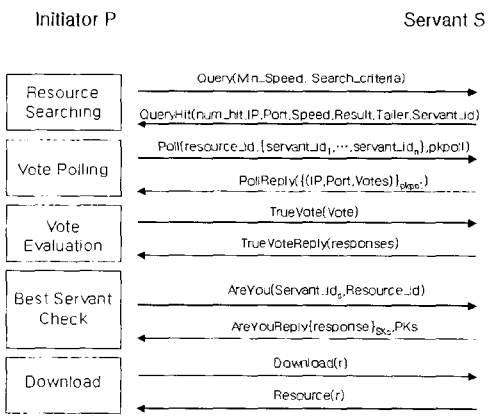


그림 2. XRep 프로토콜 참고도

각 단계를 세부적으로 살펴보도록 하자.

첫 번째 자원찾기에서는 Gnutella와 같이 사용자 p(initiator)가 자신의 주변에 있는 사용자에게 찾고자하는 자원의 질의(Qurey)를 퍼뜨리고 질의를 받은 servant 중 자료를 가지고 있는 servant는 자료의 수, 최저 속도, IP, port 등의 정보를 대답한다.

두 번째 투표묻기에서는 받은 응답(QureyHit)에 대하여 질의자 p(initiator)가 자원(resource_id)과 자원을 제공할 사람(servant_ id)에 대한 평판을 주변 사람들에게 묻고 그 응답을 받는다.

세 번째 투표평가는 받은 투표응답(PollReply)에 대하여 실제로 해당하는 node에 의해서 투표된 것인지를 묻고 응답하는 과정이다.

Initiator p는 투표 결과를 바탕으로 자신의 잣대(criterion)에 맞는 자료의 제공자와 자료를 고른다.

네 번째 최적제공자검증에서는 선택된 자료의 제공자와 자료에 대하여, 제공자에게 자료(resource_id)를 제공 가능한 지 묻고 답을 받는 과정이다. 이 때 만약 최적제공자를 유일하게 선택한다면 대부분의 좋은 평판의 제공자에게 집중되어 실행 효율에 있어 병목현상(bottleneck)을 일으키게 된다. 그래서 선정된 최적제공자에게 자료(resource_id)의 제공여부를 질의하고 만약 현재는 불가능하다면 다른 제공자를 찾는다.

다섯 번째 자원받기는 최적 제공자가 자료 제공이 가능하다고 응답하면, QueryHit의 〈IP, port〉를 이용하여 직접 최적 제공자로부터 자원을 다운로드하는 과정이다. 이 때 제공자를 여러 명 선택하여 각 자료의 조각(fragment)들을 병렬(parallel)방식으로 자료를 받는 것도 가능하다.

Ⅲ. Self-Reputation system

3.1 Self-Reputation의 정의

앞 장에서 살펴본 Xrep방식은 사용자들의 평판을 관리할 수 있는 어느 정도의 수단을 제공해주고는 있지만, broadcasting 방식을 사용하기 때문에, Gnutella 등의 특정한 P2P 방식 위에서만 적용이 가능하고, 또한 한 파일을 받기 위해서, 많은 사용자들과의 통신을 이용한 평판의 수렴과정이 필요해 많은 네트워크 통신량을 유발시킨다. 이러한 단점들을 보안하기 위한 다른 개념으로 Self-Reputation 방식을 생각해 볼 수 있다. Self-Reputation 이라는 것은 P2P 내에 있는 사용자들이 투표(voting)에 의해서 만들어진 자신의 평판을 테이블로 만들어서 자신이 직접 관리하면서 사용자의 요청이 있는 경우 그 테이블을 공개하는 방식이다. 이러한 방식에서는 평판을 알고 싶은 사용자가 있을 경우 그 사용자와의 1 대 1 통신만을 통해서 사용자의 평판을 알 수 있는 큰 장점이 있다. 어떠한 P2P 환경에서든 원하는 자료를 검색할 때 결과적으로 그 자료를 지니고 있는 사용자의 IP 주소를 얻게되므로, 이러한 시스템은 모든 P2P 환경에 적용될 수 있다는 장점을 지니게 된다. 하지만, 자신의 평판을 스스로 관리한다는 측면 때문에 부작용이 생길 수 있고, 이를 해결하기 위해 다음과 같은 요구사항이 주어진다.

Self-Reputation의 요구사항

1. Reputation voting의 위조가 힘들어야 한다.
2. Reputation voting의 임의적인 삭제, 추가가 불가능하여야 한다.
3. Table로 이루어진 reputation voting의 합산을 쉽게 구할 수 있어야 하고, 구해진 합산이 그 사용자의 reputation을 정확하게 반영하여야 한다.
4. 악의적인 사용자들(임의의 제3자)로부터 평판이 쉽게 조작되지 않아야 한다.

이러한 요구사항 이외에도 일반적인 reputation 방식에서의 요구사항 또한 갖추어야 한다.

3.2 RCert(Revocation Certification)[6] 시스템

이러한 Self-Reputation 방식의 한 예로 2003년에 제안된 RCert 방식을 먼저 살펴보도록 하겠다. RCert 방식은 다음과 같은 10가지 단계를 거쳐 평판의 작성과 참조가 이루어진다.

1. Request(평판 요구)
2. RCert(평판 전송)
3. Verify(서명 확인)
4. Last-Time Stamp(최근 평판 확인)
5. Acknowledge(확인 전송)
6. Time Stamp(현재 시간 전송)
7. Transaction(자료 전송)

8. Update of RCert(평판 갱신)
9. Revoke(최근 사용자 변경)
10. Release Notification(최근 사용자 변경 확인)

간단히 요약하면 다음과 같다. 자료를 얻고 싶은 사용자(Rater)가 다른 사용자의 평판을 알고 싶으면, Step 1의 단계에서 그 사용자에게 Request를 보낸다. Request를 받은 사용자(Ratee)는 자신의 평판을 모아놓은 RCert를 보내주는데, 이것의 확인은 두 가지 과정을 거치게 된다. Step 3의 Verify는 서명자체를 확인하는 과정이다. 이것과 더불어 Step 4의 과정을 통해서 평판의 추가와 삭제여부를 확인한다. 이 과정은 이전에 Ratee의 평판을 작성한 사용자(Previous Rater)들과의 통신을 통해서 이루어진다. 즉, 서명에 Time Stamp를 포함시키고, 이 Time Stamp를 이전의 Rater에게서 인증을 받는 형식이다. 가장 최근의 Rater가 연결이 되지 않을 경우는 그 이전의 Rater들 중에서 연결이 되는 Rater를 찾을 때까지 거슬러 올라간다. 모든 확인 작업이 완료되었다면, 자료를 받을 Ratee를 선정하고 그 Ratee에게 Acknowledge를 보낸다. Acknowledge를 받은 Ratee는 자신의 Time Stamp를 보내주고, 이것이 현재 시간과 일치하면, 자료의 전송이 시작된다. 자료의 전송 이후에 Time Stamp를 포함하는 서명을 만들어 Ratee에게 보낸다(Step 8). 마지막 단계로 최근의 Rater에게 Revoke 신호를 보내고, Ratee는 Release Notification을 역시 Rater에게 보내준다. 양쪽의 신호를 모두 받으면, Rater는 자신이 revoke 되었다는 사실을 알게되고, 자료를 전송 받은 사용자가 최근의 Rater가 된다.

RCert는 PKI를 사용하고, 서명의 안전성을 이용하여 평판의 위조를 방지한다. 그리고, 서명의 임의적인 추가와 삭제를 방지하기 위해서 Time Stamp를 이용한다. 이것은 기본적인 방지책은 되겠지만, 우리가 생각하고 있는 Self-Reputation과는 조금 거리가 있다고 할 수 있다. 즉, P2P 환경의 특성상 사용자들이 항상 접속상태를 유지하고 있다는 것은 가정하기 어려우므로, Time Stamp의 확인과정에서 상당수의 사용자들에게 접속을 시도하여야 한다. 또한, 악의적인 사용자들에 의해서 한 사용자의 평판이 쉽게 손상될 우려가 있다. 악의적인 사용자들이 한 사용자의 평판에 참여한 이후에 자신들에게 Time Stamp의 확인을 하러 오는 모든 사용자들에게 잘못된 Time Stamp(여러번 받은 Time Stamp 중에서 이전의 것)를 제공하게 된다면, 항상 Verify가 잘못된 것으로 되어 Ratee의 평판이 위조된 것으로 판단될 우려가 있다. 악의적인 Ratee에 대한 대비는 되어있지만, 제3자를 확인과정에 포함시켜서 악의적인 사용자들이 평판을 회손할 여지를 준 것이라 할 수 있다.

3.3. 겹선형함수

앞으로 제안할 방식에서는 elliptic curve 위에서 정의된 겹선형함수가 중요하게 사용된다. 겹선형함수의 예로 Weil pairing과 Tate pairing이 있으며 이 중 Weil pairing에 대해서 간략하게 정리해 보면 다음과 같다.

위수가 p^l(p는 소수)인 유한체 위에서 정의된 elliptic curve E에 대해서 $E[q]=\{P\in E \mid qP=0\}$라 정의하자. 단, $q(\neq p)$는 소수이다. $q \mid p^{la}-1$을 만족하는 최소의 a를 선택하면 효율적으로 계산할 수 있는 bilinear form, $e: E[q]\times E[q] \rightarrow F_{p^{la}}$가 존재하며 이를 Weil pairing이라 부른다. 그리고 이 Weil pairing은 다음과 같은 성질을 가진다.

(1) $R_1, R_2 \in E[q]$, $a, b \in Z$에 대하여 $e(aR_1, bR_2)=e(R_1, R_2)^{ab}$ 이다.
(2) 모든 $R' \in E[q]$에 대하여 $e(R, R')=1$이면 $R=0$ 이다.

Elliptic curve의 겹선형함수를 이용한 서명체계의 안전성은 elliptic curve 위에서 computational Diffie Hellman(CDH) 문제는 어렵고, 겹선형함수를 이용하면 Decisional Diffie Hellman (DDH) 문제가 쉽게 풀린다는 점에 기반한다.

3.4 제안 프로토콜

모든 사용자들은 P2P에 참여할 때 PKI를 통한 인증을 받은 것으로 가정한다. 이것은 Sybil 공격 등으로부터 시스템을 보호할 수 있도록 하기 위해서이다.

시스템을 설계하면서, 사용자가 자신의 reputation table 하나만을 관리하는 체계와 reputation table 과 voting table 두 개를 동시에 관리하는

dual-table체계의 두 가지를 생각하였다. 몇 가지 세부적인 사항을 제외하고는 모두 동일하므로, 여기서는 두 개의 table을 이용하는 방식에 대해서 설명하겠다.

3.4.1 시스템 설계단계

충분히 큰 소수 q를 골라서, elliptic curve $E(F_q)$를 만든다. 이 중 위수가 소수 p인 원소 P를 고른다. Hash 함수 H와 Weil-Pairing $e: E[p]\times E[p]\rightarrow F_{q^2}^*$ 를 만들어 공개한다. 모든 사용자들은 시스템에 참여할 때, 자신의 pseudonym(즉, 자신의 ID)을 바탕으로 Hash함수를 이용하여 $Q_{ID} = H(ID)$를 만든다.

모든 사용자들은 두 개의 테이블을 만든다. 하나는 자신의 reputation table이고, 다른 하나는 voting table이다. 각 각 T_{rep}와 T_{vot}라고 부르기로 한다. 그리고, 임의의 원소 r_{id}, t_{id}를 선택하여 token을 만들 때 사용한다. 마지막으로 s_{id}를 선택하여 비밀키로 사용한다.

공개키 : $P_{id}=s_{id}P$, Q_{id}.

비밀키 : $D_{id}=s_{id}Q_{id}$ 또는 s_{id} .

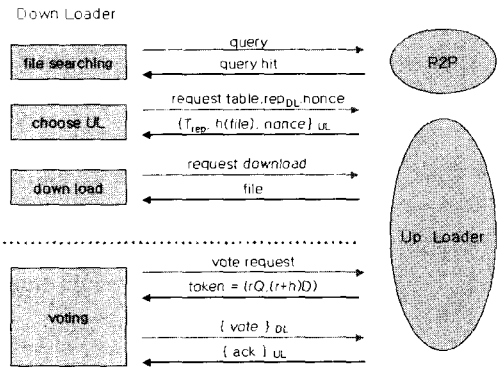


그림 3. Dual Table을 이용한 방식 Choose UL과 Voting의 과정으로 프로토콜이 구성된다.

3.4.2 평가 단계

사용자 DL(downloader)이 UL(uploader)의 자료를 전송받은 후 평판을 작성하고자 할 때, 다음의 단계를 거친다. (DL의 j 번째 voting 이고, UL의 i 번째 reputation 이라고 가정하자.)

1) 요청

DL은 UL에게 voting을 하겠다는 신호로 request token(R_j)를 전송한다. 단, $R_j=(r_{DL}^{2'} Q_{DL},\ (r_{DL}^{2'}+h_j)D_{DL})$이다.

여기에서 $h_j=H(r_{DL}^{2'}, ID_{UL}, filename)$이다. 사용자 UL은 DL에게 받은 request token이 정확한 것인지 확인한다. 확인은 다음 두 단계를 거쳐서 이루어진다.

- $R_j=(U_j, V_j)$라고 할 때, 먼저 UL의 T_{rep}의 가장 마지막에 있는 정보, 즉 가장 최근에 일어난 voting을 이용하여, $e(r_{DL}^{2'}Q_{DL}, Q_{DL})= e(r_{DL}^{2'-1} Q_{DL},\ r_{DL}^{2'-1}Q_{DL})$를 만족하는 지 확인한다. 이것은 $e(U_j,\ Q_{DL})= e(U_{j-1},\ U_{j-1})$을 통해서 삭제가 발생하지 않았음을 확인한다.
- 그 다음 $e(P,\ V_j)=e(s_{DL}\ P,\ U_j+h_jQ_{DL})$을 확인해서 정당한 token 인지를 검증한다. 즉 $e(P, b)= e(sP, a+h_jQ_{DL})$을 확인해 본다.

2) token의 발행

DL에게 받은 request token이 정확하다면, 이미 선택한 t_{UL}을 이용하여 i번째 token $T_i=(r^{2'}Q_{UL},(r^{2'}+h_i)D_{UL})$를 만들어 DL에게 전송한다. 사용자 DL은 UL에게 받은 token을 (1)에서와 같은 방법으로 확인한다.

3) 투표

UL에게 받은 token $T_i=(U_i, V_i)$를 이용하여 다음과 같은 voting 메시지 V_i를 작성한 후 UL에게 전송한다. 메시지 M_i가 $M_i=(T_i, vote_i, ID_{DL}, time)$일 경우, 그에 해당하는 voting 메시지는 $m_i=(M_i,\ sign_{DL}(M_i))$이다.

4) 수신확인

사용자 UL는 DL에게 voting, m_i을 받았다는 증거로 acknowledge 메시지, $sign_{UL}(ACK)$를 전송한다.

$ACK=(ID_{DL}, ID_{UL}, time)$이다. 만약, (3)의 단계에서 DL이 m_i를 전송하지 않는 경우는 (4)를 수행하지 않고 프로토콜은 종료된다. 그리고, 다음에 request가 있을 경우 i+1 번째의 token을 발행한다.

3.4.3 저장단계

모든 사용자들은 앞에서 설명한 두 개의 table을 관리한다. T_{rep}의 경우 $R_j, M_i, sign_{DL}(M_i)$를 저장하고, T_{vot}의 경우 $Q_{UL}, T_i, sign_{UL}(ACK)$ 를 저장한다. (3)의 단계에서 V_i가 전송되지 않은 경우에는 R_j만을 저장하고, 뒤의 두 칸은 공란으로 비워둔다. (4)의 경우에도 UL이 ACK 메시지를 보내지 않으면, Q_{UL}, T_i 를 기록하고, 마지막은 비워둔다.

3.4.4 Table의 이용

앞에서 제안된 과정을 통해서 사용자들이 T_{rep}과 T_{vot}을 관리하고 있다고 하자. DL이 원하는 자료를 P2P 네트워크를 통해서 검색을 하면 P2P 네트워크는 각각의 검색 과정을 거쳐 DL에게 자료를 제공하는 사용자들의 주소값을 알려줄 것이다. DL은 이들 사용자들의 평판 점수를 요구한다. DL은 사용자들이 보내온 메시지 중에서 점수와 네트워크의 속도 파일의 크기 등을 고려하여 직접 자료를 전송받을 UL을 선정한다. 이 과정에서 UL의 T_{rep} 전체를 요구한다. 이것은 다음과 같은 작업을 통해서 이루어진다. DL은 UL에게 (request table message , repu. of DL, nonce)를 전송한다. UL은 이 것을 이용하여 (T_{rep}, h(file), sigUL(T_{rep}, h(file), nonce))을 계산하여 전송하여 준다. $h(file)$은 파일의 전송이 종료한 이후에 파일의 무결성을 검사하기 위하여 미리 전송하여 준다. 그리고, UL의 서명 부분은 T_{rep}을 중간에 다른 사용자가 변조하거나, 또는 다른 사용자가 이를 자신의 table인 것처럼 위조하여 사용하지 못하도록 막아준다. 이것에 대해서는 뒷 부분에서 좀 더 자세히 다루도록 한다.

DL은 전송받은 T_{rep}을 보고 UL이 미리 보내온 평판과 비교해 볼 수 있다. 또한 DL의 reputation을 UL에게 보내주어서 UL 또한 DL이 자신의 기준에 적합한 사용자인지를 확인해 볼 수 있다. 이 경우 reputation은 DL의 T_{rep}과 T_{vot}의 합산으로 생각되어져야 할 것이다. 적당한 0에서 1사이의 상수 α에 대해서 reputation = α×(reputation of T_{rep})+(1-α)×(reputation of T_{vot})로 계산한다.

3.4.5 하나의 테이블만을 이용하는 방식

각 사용자는 자신의 reputation table 만을 관리하고, vote request를 보낼 때, token 형식이

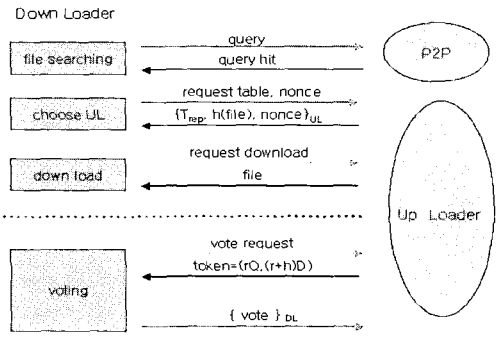


그림 4. 하나의 Table 만을 이용하는 방식

아닌 DL의 서명을 한 메시지를 보낸다. 그리고, 마지막 단계인 UL의 ACK 메시지의 전송은 생략된다.

이러한 방식의 장점은 테이블을 하나만 사용하므로 생기는 저장공간과 통신량 그리고 계산량에서의 효율성이다. 하지만, 악의적인 사용자가 반복적으로 한 UL에게 vote request만을 하여 token을 받고 vote를 하지 않으면서 UL이 나쁜 vote를 받아서 메시지를 삭제한 것처럼 만들 수 있다는 단점 또한 존재한다.

Ⅳ. 분석

4.1 Voting의 위조 방지

각각의 vote는 DL의 서명을 통해서 이루어진다. 다른 r을 사용한 한번의 서명은 signature scheme의 안전성에 기반하여 random oracle model에서 chosen message attack에 대한 위조불능 안전성을 지닌다. 단, 위의 제안된 시스템에서 사용하는 서명은 연관성이 있는 r을 사용하므로 이러한 특성에 대한 검증이 필요하다. rP를 알고 r^2P를 구하는 문제인 square exponential problem이 어렵다고 알려져 있으며 이러한 문제에 기반한 서명이 안전하다는 사실이 최근 증명되었다.

4.2 Voting의 임의적인 삭제, 추가 방지

사용자가 자신의 table을 위조하지 못하도록, 다시 말하자면 table의 한 줄을 삭제하거나 또는 만들어진 거짓 voting 값을 추가하거나 하는 것을 방지하기 위해 voting에 이용되는 token들이 바로 전

에 발행된 token과 유기적인 결합구조를 지니고 있도록 만들었다. token의 앞부분을 보면, 처음 선택한 r_{id}를 시작으로 하여, token이 발행될 때마다, r_{id}^2, r_{id}^4, r_{id}^8 ...으로 일종의 chain 구조를 이루며 변화하는 것을 볼 수 있다. 이러한 token의 성질 때문에 table의 중간에 있는 값을 임의로 삭제할 수 없게 된고, 추가하는 것 또한 임의로 할 수 없다.

사용자가 자신에게 불리한 voting값을 삭제한 경우는 다음과 같은 방법으로 사용자의 비밀정보가 드러나게 되어있다. 같은 token을 DL1 과 DL2 에게 발행한 것이 되므로, DL1과 DL2가 가지고 있는 token은 각각 $(rQ,(r+h_1)D)$, $(rQ,(r+h_2)D)$이다. Hash 함수의 성질(collision-free)에 의해서 사용자 A가 h_1과 h_2의 값을 조작할 수 없다고 가정할 수 있으므로, 일반적으로 $h_1 \neq h_2$ 가 된다. 이것으로부터 쉽게 A의 비밀정보 D를 구할 수 있다. 이 값이 알려지는 것은 A의 pseudonym을 더 이상 사용할 수 없게 되는 것과 같은 의미를 지닌다.

4.3 Voting의 합산(evaluation)

사용자가 자료를 받고자 할 때, 여러 uploader들 중 reputation이 가장 좋은 한 명을 선택하여 그의 자료를 받게 될 것이다. 따라서 여러 사용자들의 reputation을 쉽게 비교할 수 있는 방법이 제공되어야 한다. 각각의 voting에 점수를 두어 그의 합산으로 전체 reputation을 비교하는 것이다. uploader들은 자신의 reputataion을 공개할 때, table과 미리 계산해 놓은 evaluation 값을 동시에 제공하는 것으로 한다.

Evaluation의 방법으로 여러 가지를 생각할 수 있다. 가장 기본적인 것으로는 각각의 voting에 -2(아주 나쁨), -1, 0, 1, 2(아주 좋음)의 점수를 주는 것을 들 수 있겠다. 이러한 것을 점수로 환산하여, 그 총합으로 사용자의 reputation을 평가한다. 또는 0, 1, 2, 3, 4로 두는 것도 고려해 볼 수 있다. 이 evaluation의 경우는 사용자들의 전체 점수가 시간이 지남에 따라서 절대 감소하지 않고 증가하게 되어있다. 사람들은 자신의 점수가 줄어드는 것을 싫어할 것이므로, 그러한 심리적인 면을 고려한 evaluation 방법이라고 할 수 있다.

Evaluation 방식을 만들 때에 가장 중요하게 고려해야 할 것은, voting을 받지 못한 경우에 대한 처우이다. 왜냐하면, 나쁜 voting을 받은 것을 voting을 받지 못한 것처럼 꾸밀 수 있기 때문이다. 이것은 메시지만은 지워버림으로서 쉽게 위장할 수 있다. 위의 프로토콜에서는 이러한 것을 막을 수는 없고, 따라서 이러한 위조가 결코 도움이 될 수 없는 evaluation방법을 고려하여야 한다. 공란으로 비워져 있는 경우의 점수를 최저, 예를 들면 -2 정도로 고려하는 것이 일반적일 듯 하다. 하지만, 대개의 경우 사용자들은 '좋음'의 경우 voting을 잘 하지 않고, '나쁨'의 경우는 voting을 적극적으로 한다라는 사실도 고려하여야 할 것이다.

마지막으로 DL의 voting table에서의 evaluation도 잘 설계하여야 할 것이다. 나쁜 voting을 한 경우, UL는 ACK 메시지를 보내지 않을 확률이 높을 것으로 보이므로, 이러한 경우에 대해서 DL이 큰 피해를 보지 않도록 voting table에서의 공란은 UL의 reputation table에서와는 좀 다른 방법으로 점수를 주어야 할 것이다. 두 table의 weight를 다르게 두는 것을 일반적으로 생각해 볼 수 있다.

이 방식에서는 repu.table의 평가에서는 -2, -1, 0, 1, 2 의 점수를, voting table의 평가에는 0, 1 의 점수를 주는 것으로 제안한다. 그리고 두 테이블 모두 공란으로 비워져 있을 경우는 최저점을 주는 것이 합당할 것이다.

4.4 Platform Independence

Platform independence라는 것은 평판 관리 시스템이 어떠한 P2P 네트워크에 적용되든지 같은 프로토콜을 통해서 동작할 수 있다는 것을 의미한다.

P2P 네트워크는 크게 query, response, download의 과정을 거친다고 할 수 있다. 위의 프로토콜을 P2P에 적용시키면, response 과정에서 각각의 사용자는 자신의 reputation을 답변의 일부로 해서 query를 한 사용자(DL)에게 보내주게 된다. DL은 response에 동반된 reputation을 보고 그 중 가장 좋은 reputation을 가진 사용자(UL)를 선택한다. 그 이후의 과정은 DL 과 UL 사이의 직접연결에 의한 자료의 전송이다. DL은 UL로부터 필요한 자료를 전송받게 되고, 그 자료를 통해 평판을 검증해 본 이후에, voting을 할 필요가 있으면, 자료를 전송 받을 때, 사용한 UL의 IP주소를 이용하여, request 메시지를 보내고, 그 이후의 평가단계를 수행한다. 즉, query, response, comparing

reputation, download로 이루어지고, voting은 선택사항이 된다.

결국, P2P 네트워크와는 별개로 평판을 참조하고, 투표하는 과정이 이루어지므로 위의 시스템은 Platform independent 하다고 할 수 있다.

4.5 Self-Reputation에서의 평판

Self-Reputation에서 생각하는 평판은 P2P 환경에서 사용되는 가명(pseudonym)에 주어지는 평판이고, 이것은 긴 사용주기를 가지는 것으로 생각한다. 즉 현실 세계에서의 평판을 며칠 사이에 쌓을 수 없는 것처럼, 오랜 기간을 가지고 쌓아가는 평판을 고려한다. 따라서 자신의 가명에 평판을 쌓아가는 것은 상당한 노력이 필요한 것이고, 따라서 악의적인 사용자들에 의해서 쉽게 평판이 나빠지는 것을 막아낼 수 있어야 한다. 이 시스템에서는 투표를 일방적으로 진행하는 것이 아니라, UL가 token을 주어야만 투표를 할 수 있기 때문에 악의적인 사용자들이 모여서 한 사용자의 가명의 평판을 악화시키는 것을 어느 정도는 차단할 수 있다고 본다.

4.6 Storage Clearing

시간이 지남에 따라, 모든 사용자들의 table의 크기가 커지게 되므로, 다른 사람의 table을 살펴보는데 많은 시간과 자료의 전송이 필요하게 된다. 따라서, table을 시간이 지남에 따라서 관리할 필요성이 생겨난다. 이 문제에 대해 한 가지 해결 방법은, 적당한 시간 또는 voting의 개수를 설정한 후 그 기준을 경과한 voting에 대해서는 삭제할 수 있도록 하는 것이다. table의 가장 윗 부분의 voting은 삭제를 하여도 table의 무결성에 아무런 영향을 주지 않는다는 점을 이용한 것이다.

이 방식에서는 최근 6개월 동안의 기간을 기준으로 정하여 그에 해당하는 voting만을 저장한다. 이것은 평판이 좋은 사용자라 할 지라도 최근 활동이 없는 경우는 좋은 자료를 제공하기 힘들 것이라는 상식과도 부합된다.

V. Xrep와의 비교

5.1 알려진 공격에 대한 분석

P2P 시스템의 공격 방법들 중에서 중간자 공격(man in the middle attack)과 자기복사공격(self replication attack)에 대해서 살펴보기로 하자. 중간자 공격이란 정직한 사용자들의 중간에서 악의적인 사용자가 모든 작업을 대치하여 행동하게 되는 것을 말한다. 그리고, 자기복사는 악의적인 사용자가 모든 질의에 대하여 답하며 악의적인 내용의 파일이나 내용을 배포하는 것을 의미한다. 이 절에서는 이러한 공격방법에 대해서 Xrep와 위의 제안된 시스템의 대처방안을 비교해 본다.

중간자 공격은 query에 대한 query hit message를 공격자(A)가 위조하여, 자신이 DL이 원하는 자료를 가지고 있는 것처럼 위조한 후, 거짓 자료를 보내주는 것을 말한다. Xrep에서는 query hit message가 위조되어 있더라도, A의 IP에 대한 평판을 조사하면 A의 평판이 나쁘다고 가정할 때, A를 최종 UL로 선정하지 않게 된다. 혹은 A의 평판이 나쁘지 않다고 하더라도, 전송받은 자료가 조작된 것이라면, 알려진 파일의 digest(hash value)와는 다른 digest값을 가지고 있게 되므로 버려지게 된다.(Xrep에서는 파일 마다 고유한 file id를 가지고 있다고 가정한다. file id가 바로 file의 hash value 이다.)

Self-Reputation에서는 기본적으로 서명을 이용하여 메시지를 보안하므로 중간자가 끼어들 여지가 없다. 중간자 공격의 상황은 중간에 파일을 가로채어 다른 파일을 보내는 것과, 다른 사람의 평판을 자신의 평판인 것처럼 위조하여 사용하는 것, 크게 두 가지를 생각해 볼 수 있다.

첫 번째의 경우는 DL이 UL의 평판을 받아볼 때, 원하는 파일에 대한 해쉬 값인 $h(file)$ 의 값을 동시에 받게 된다. 이것은 Xrep에서의 file id와 같은 역할을 하여 원하지 않은 파일을 전송받는 것을 막을 수 있다. 만약 중간자가 UL의 평판 table을 미리 복사해 두고, UL 인 것처럼 행세하려고 하면, 즉, UL이 예전에 서명했던 $sign_{UL}(T_{rep}, h(file), nonce)$ 의 값을 기억하고 그 값을 이용하려고 하면, DL이 같은 nonce의 값을 선택하지 않는 한 불가능하다. 물론, nonce의 값은 임의로 선택되어지므로 같은 nonce를 선택하는 확률은 무시할 수 있다.

자기복사공격에 대해서 살펴보자. 자기복사공격의 대표적인 예로 Mandragore라는 Gnutella worm을 들 수 있다. 이 worm은 P2P 네트워크 상의 모든 query에 대해서 query hit를 보내고, 원하는

파일 대신에 자신의 복제를 보내는 방법으로 퍼져나간다. Xrep에서의 대응방법은 다음과 같다. 이러한 worm이 퍼져나감에 따라 worm의 file id 는 점점 나쁜 평판을 얻게 되고 결국 그 파일을 전송하는 사용자는 줄어들게 된다.

제안된 시스템의 경우에서는 worm이 P2P상의 Peer, 즉 가명을 만들어서 worm을 퍼트리려고 한다고 가정해 보자. 이 경우 worm은 가명은 쉽게 만들 수 있지만, 평판 테이블은 만들 수 없다. 따라서 평판이 없는 상태가 되고, 그러한 가명을 가진 사용자에게서 파일을 전송받는 사용자는 거의 없을 것이다.

이러한 공격 방법 이외에 자기 복제공격에 대한 약한 방어를 제공해 주기는 하지만, Xrep나 제안된 시스템의 경우 모두 완벽하게 막아내기는 힘들다. 이 경우의 사용하는 PKI system의 안정성에 의존한다.

5.2 효율성 비교

위의 프로토콜을 앞 절에서 설명한 Xrep 프로토콜과 간략하게 비교해 보겠다. Xrep는 Gnutella 등과 같이 자료를 검색하는 단계부터 정의가 되어 있지만, 여기서는 일반적인 P2P 프로토콜에서 다루는 부분을 제외한, 사용자의 평판을 관리하는 부분만을 비교해 본다.

우선 Xrep에서 사용자의 평판을 알아보는 과정을 간단하게 살펴보자. 먼저 DL이 P2P 네트워크를 통해서 UL(uploader)에 대해서 voting을 해줄 사람들을 모집하게 된다. 이 과정이 vote polling 단계이다. UL에 대해서 voting을 할 의사가 있는 사용자들이 DL에게 답변을 보내면, 그 사용자들에 한해서, DL은 true vote 메시지와 true vote reply 메시지를 서로 주고 받게 된다. DL은 모아진 true vote reply 메시지를 종합하여 UL에 대해서는 평판을 평가하게 된다. 이 과정에서 통신횟수를 생각하면, 처음 vote polling 과정에서 메시지를 broadcast하는 과정이 한 번 필요하고, 그 다음에는 답변한 사용자의 수에 비례하여 통신량이 증가한다. 답변한 사용자의 수를 n이라고 할 때, 통신량은 3n + M(M은 임의의 큰 수)가 된다. 계산량의 경우, true vote reply를 주고받을 때, 한번의 암호화와 그에 따른 한 번의 복호화가 필요하므로, n(Enc +Dec) 이라고 할 수 있다. 마지막으로 저장하는 자료의 양을 생각하면, 각각의 사용자가 다른 많은 사용자들의 ID 와 평가점수를 직접 저장하고 또, 필요에 따라 업데이트를 하면서 관리를 하고 있어야 한다.

제안된 시스템의 경우를 살펴보자. 이 프로토콜에서는 지금까지 모아진 reputation을 참조하는 것과 실제 voting을 하는 것이 별도로 주어져 있는데, 실제 voting을 하는 과정을 비교하겠다. DL과 UL이 서로 두 번씩 메시지를 주고 받으므로, 통신량은 4c가 된다(c는 한 번 통신할 때 기본적으로 필요한 통신량을 나타내는 상수라고 하자). 계산량에 있어서는 두 번의 서명과 두 번의 검증단계가 요구되고, token의 확인 단계에서 2번씩의 pairing의 계산이 필요하므로 2(Sig + Ver) + 4 pairing이라고 할 수 있다. 마지막 저장량에서는 지금까지 자신에게 voting한 사용자의 (ID, 서명)과 자신이 voting한 사용자들의 (ID, 서명)을 저장하게 된다.

두 프로토콜을 비교해 볼 때, 통신량과 계산량에 있어서 현격한 차이가 드러남을 알 수 있다. 사용자의 평판은 가급적 많은 다른 사용자들에게서 모아야만 더 정확한 평가를 내릴 수 있다는 것을 생각하면, 즉 n이 어느 정도 크기를 넘어야만 한다는 것을 생각하면 더욱 그러하다. 저장량에 있어서는 Xrep의 경우에도 사용자들이 자신이 한 번 평가를 한 사용자에 대해서는 점수를 저장하고 있어야 하므로, Self reputation의 경우와 비교했을 때, 저장해야 하는 ID의 수는 비슷할 것이라 예측할 수 있다.

계산 시간에 있어서 pairing의 계산과 Weil pairing의 계산은 비슷하고, Tate pairing의 계산에 다음과 같은 시간이 필요하다는 것이 알려져 있다.

표 3. 효율성비교

	통신횟수	계산횟수	저장용량
Xrep	3n+M	n(Enc+Dec)	(ID,점수)
RCert	4c+α	n(ver)	(ID,서명,TS)
제안 시스템	4c	2(sig+ver) +4pairing	(ID,서명)

표 4. P3 1GHz에서 Tate pairing의 계산

Underlying field	time(ms)
$F_{3^{97}}$	26.2
$F_{2^{251}}$	23.0
F_p (p: 512 bit)	20.0
F_p with preprocessing	8.6

Ⅵ. 결 론

P2P 프로토콜은 현재 파일 공유 목적으로 많이 사용되고 있으며, 앞으로는 파일 공유 뿐만 아니라 분산계산, 분산저장 등의 분야에서 더욱 활발하게 사용되어질 것으로 기대된다. 하지만, 보안상의 많은 문제점이 드러나고 있고, 현재까지는 이러한 보안 문제점에 대해서 적절한 해결책이 제시되어 있지는 못한 실정이다. 이에 대한 연구 결과로 Xrep 와 같은 평판관리시스템이 제안되었으나, 평판 수렴과정에서의 통신량이 가중되는 등의 문제점을 지니고 있고, 또한 앞으로의 P2P 의 주류가 될 것으로 예상되어지는 DHT 모델 등에서의 적용은 힘들어 보인다.

이에 우리는 이 논문에서 새로운 평판 관리 시스템인 Self-Reputation을 제안하였고, 그 요구조건을 만족시키는 시스템을 제시하였다. 이 시스템은 기존의 Xrep과 비교했을 때, 평판을 참조하는 과정에서 Broadcasting 기법 대신 1 대 1 통신을 이용하고 있어서 통신량을 감소시켰다. 그리고, 어떠한 p2p환경에서도 P2P의 특성에 영향을 받지 않고 제안한 프로토콜을 사용할 수 있기 때문에, 하나의 가명을 서로 다른 P2P 네트워크들에서 이용하는 것 등의 활용방안을 지니고 있다.

앞으로는 이러한 시스템을 실제로 구현해 보고, P2P 네트워크에서 사용할 때 발생할 수 있는 문제점들을 해결하기 위한 연구가 좀 더 필요할 것으로 생각되어 진다.

참 고 문 헌

[1] The Gnutella protocol specification v.0.4, Availible at http://www.clip2.com/GnutellaProtocol04. pdf, 2001.

[2] E. Damiani, S. Vimercati, S. paraboschi, P. Samarati, F. Violante, "A Repuatation-Based Approach for Choosing Reliable Resources in Peer-to-Peer Network," Conference on Computer and Communications Security archive Proceedings of the 9th ACM *conference on Computer and communications security*, pp. 207-216, 2002.

[3] Dan S. Wallach, "A Survey of Peer-to-Peer Security Issues," *International Symposium on Software Security* (Tokyo, Japan), 2002.

[4] D. Boneh, B. Lynn and H. Shacham, "Short Signatures from the Weil Pairing," In *Advances in Cryptology - Asiacrypt 2001. Lecture Notes in Computer Science 2248*, Springer, pp. 514-532, 2002.

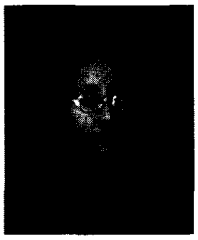
[5] P. Barreto, H. Kim, B. Lynn, and M. Scott, "Efficient *Algorithms for Pairing-Based Cryptosystems," in Advances in Cryptology - Crypto 2002. Lecture Notes in Computer Science 2442*, Springer, pp. 354-368, 2003.

[6] B. Ooi, C. Liau, K. Tan, "Managing Trust in Peer-to-Peer Systems Using Reputation-Based Techniques", *The 4th International Conference on Web Age Information Management (WAIM), August 2003. Keynote Paper.*

〈著者紹介〉

조 남 수 (Nam Su Jho)
1999년 8월: 한국과학기술원 수학과 학사
2001년 3월~현재: 서울대학교 수학과 석·박사 통합과정
〈관심분야〉 공개키 암호 이론, 정보보호

김 우 환 (Woo Hwan Kim)
1998년 2월: 서울대학교 수학과 학사
2000년 2월: 서울대학교 수학과 석사
2000년 3월 ~ 현재: 서울대학교 수학과 박사과정
〈관심분야〉 공개키 암호 이론, 정보보호

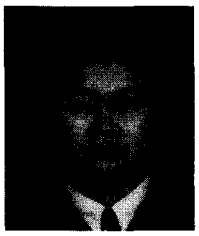


윤 효 진 (Hyo Jin Yoon)
1999년 2월: 서울대학교 수학교육과 학사
2001년 8월: 서울대학교 수학과 석사
2001년 9월~현재: 서울대학교 수학과 박사과정
〈관심분야〉 공개키 암호 이론, 정보보호

이 인 석 (In-Sok Lee)
1979년 2월: 서울대학교 수학과 졸업
1986년 5월: Yale 대학교 수학과 박사
1986년 9월~현재: 서울대학교 수리과학부 교수
〈관심분야〉 표현론, 암호학, 코딩이론

천 정 희 (Jung Hee Cheon) 정회원
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 조교수
〈관심분야〉 응용정수론, 암호론, 응용암호론

김 태 성 (Tae-sung Kim) 정회원
1999년 2월: 동국대학교 전자계산학과 학사
2001년 2월: 동국대학교 컴퓨터공학과 석사
2001년 3월~현재: 한국전자통신연구원 정보보호연구본부 인증기반연구팀
〈관심분야〉 ID 관리, 정보보호



진 승 헌 (Seung-hun Jin) 정회원
1993년 2월: 숭실대학교 전자계산학과 학사
1995년 2월: 숭실대학교 전자계산학과 석사
2000년 3월~현재: 충남대학교 컴퓨터과학과 박사 과정
1994년 12월~1996년 4월: 대우통신 종합연구소
1996년 5월~1999년 5월 : 삼성전자 통신연구소
1999년 6월~현재: 한국전자통신연구원 정보보호연구본부 인증기반연구팀장
〈관심분야〉 컴퓨터/네트워크 보안, 정보보호(PKI)

추 경 균 (Kyung-kyun Choo) 정회원
1982년 2월: 숭실대학교 전자계산학과 학사
1988년 2월: 숭실대학교 전자계산학과 석사
1998년 3월~현재: 숭실대학교 전자계산학과 박사과정
1998년8월~현재: 행정자치부 행정정보화담당관실 서기관
〈관심분야〉 전자정부, 공개키기반구조(PKI), 정보보호, DRM