디지털 콘텐츠 공유를 위한 개선된 Kerberos P2P 인증시스템 설계

김종우* · 한승조*

Design of an Advanced Kerberos P2P Authentication System to Share Digital Content

Jong-Woo Kim* · Seung-Jo Han*

본 연구는 산업자원부의 지역혁신 인력양성사업의 연구결과로 수행되었음.

요 약

본 논문에서는 상호 인증을 위한 알고리즘인 Kerberos를 개선하여 P2P 시스템에 적합한 알고리즘을 제안하였다. 제안된 알고리즘은 Kerberos의 기능은 유지하면서 서버의 부담을 최소화하기 위하여 티켓 승인 서버의 기능을 상대측의 Peer에 부여하였다. 이러한 방법을 이용하여 티켓 승인 서버를 위한 서버의 개수 증가를 막고, 인증을 위한 서버의 기능을 최소화하여 서버의 부담을 최소화하였다. 제안된 알고리즘은 상호 인증을 위하여 서버는 최소한의 역활만하고, Peer간에 인증의 역활을 부여하면서도 강력한 상호 인증을 할 수 있다. 또한 P2P 시스템에 맞도록 시간위주의 인증 만료 시간보다, 횟수 위주의 인증 제한 값을 주었다. 또한 본 논문에서는 이러한 알고리즘을 적용하여 P2P 인증 시스템을 설계하였다.

Abstract

In the paper, an algorithm fitted to P2P system was proposed by improving Kerberos which is an algorithm for mutual authentication. To keep the role of Kerberos and minimize load to server, the proposed algorithm imposed the server role of ticket recognition to the opposite peer. Using this method, the number of servers as ticket recognition server was averted and function of server for authentication was minimized, so that server load was minimized. The proposed algorithm enables the server to play the minimum of the role and to perform strong mutual authentication, while imposeing on the peers the role of authentication. To make suitable to P2P system, trial number oriented authentication limit was given, not time-oriented authentication expiration time. In the paper, a new P2P system was designed using this algorithm.

키워드

인증, Kerberos, P2P, 공유

Ⅰ. 서 론

최근 인터넷의 발달로 인하여 사용자가 전세계적으로 급격히 증가하고, 이에 따라 네트워크 기반의 컴퓨팅 기술은 급속한 발달이 이루어져 왔다. P2P(Peer to Peer)란 인터넷에서 서버에 의존하지 않고 정보의 요구자와 제공자간의 컴퓨터를 직접 연결시켜 데이터를 공유할 수 있게 해주는 기술과

*조선대학교

접수일자 : 2004. 5. 21.

그 기술을 응용해서 제공되는 서비스를 말한다[1].

P2P 서비스는 1995년 인터넷을 기반으로 시작된 SETI의 SETI@Home, 1996년 11월에 공개된 ICQ, 1999년 1월에 공개된 Napster, 2000년에 공개된 Groove 등이 개발되어 현재 많은 이용자를 확보하고 있다. 이러한 P2P 시스템은 트래픽 분산과 서버의 리소스 및 부하량 감소를 위해 개발되었다. P2P 서비스를 사용하기 위해 연결되는 가상의 공유 시스템은 개방 시스템이기 때문에 근본적으로 보안상의 문제점을 안고 있다[2][3]. 그래서 부정 사용자의 접근, 보안과 개인정보 노출, 컴퓨터 바이러스 확산, 비 공유 영역에 대한 침해(일종의 해킹)와 같은 부당한 침해에 대한 문제점이 심각하게 대두될 수 있다. 이러한 환경에서 안전한 콘텐츠 공유를 위하여 강력한 상호 인증 기능은 필수적이라 할 수 있다.

본 논문에서는 Kerberos를 P2P 시스템에 맞게 개선하여 보안 위협에 대하여 안전하고, 신뢰성 있는 데이터 공유를 위한 안전한 P2P 시스템을 구현하고자 한다. 제안된 알고리즘을 이용하여 Kerberos의 장점은 유지하면서, 최대한 서버의 부하량과 트래픽을 줄이면서 안전한 Peer간의 인증을 할 수 있다. 즉 P2P를 위한 강력하고 안전한 Peer 인증 시스템을 설계하고자 한다.

II. P2P 어플리케이션 및 암호 알고리즘

2.1 P2P 시스템 환경에서 정보보호

기업이나 개인이 P2P를 이용하는 것은 불특정다수의 이용자에 대해 자사의 PC의 자원이나 서비스의 이용을 가능하도록 하기 위해서이다. 따라서 이것에 의해 새로운 취약성이 발생한다. P2P 네트워크의 사용은 악의적인 소프트웨어를 전달하는 능력뿐만 아니라 악의적인 소프트웨어에 의해 통신하는 프로토콜의 사용까지도 허가한다.

그런데, P2P 소프트웨어는 일반적으로 방화벽에 의해 방지되지 않는다. 왜냐하면 이것이 중앙 디렉토리 서비스 또는 다른 서밴트들과 송신연결을 하기 때문이다. 일단 송신 연결이 발생하면, 중앙 디렉토리 서비스 또는 서밴트는 클라이언트로 정보를 전달할 수 있다.

이 같은 방법은 P2P 네트워크를 불법적으로 사용하도록 수행되어진다. 예를 들면, 악의적인 위협은 P2P의 중앙 서버를 통해 등록될 수 있고 특정한 파일의 목록을 전달할 수 있다. 정보의 수집과 시스템의 제어는 방화벽을 우회하고 해커의 익명성을 보증하는 방법으로 수행된다.

2.2 Kerberos 인증방식

Kerberos는 MIT에서 Athena 프로젝트이 일환으로 개발된 인증 서비스이다[5]. Kerberos는 분산망에서의 인증 시스템으로서 프로세스인 클라이언트가 특정 사용자를 대신하여 검증자에게 사용자의 신분을 확인시켜 주기 위한 일련의 암호화된 메시지를 교환하는 과정이다.

Kerberos의 동작은 AS(Authentication Server)라는 인증 서버, TGS(Ticket Granting Server)라는 티켓 승인 서버, 그리고 클라이언트(C:Client)와 서버(S:Server)간의 티켓 발행 단계, 서비스 승인 단계를 각각 거쳐서 이루어지는 것이 기본적인 것으로, Kerberos V.4의 단계별 동작 설명은 다음과 같다.

(a) 인증 서비스 교환

[단계1](메시지1)

$$C \rightarrow AS : ID_c \| ID_{tgs} \| TS_1$$

[단계2](메시지2)

$$AS \rightarrow C : E_{K_c}[K_{c,tgs} \| ID_{tgs} \| TS_2 \| lifetime_1 \| Ticket_{tgs}]$$

$$*Ticket_{tgs} = E_{K_{tgs}}[K_{c,tgs} \| ID_c \| AD_c \| ID_{tgs} \| TS_2 \| lifetime_1]$$

(b) 티켓-승인 서비스 교환

[단계3](메시지3)

$$C \rightarrow TGS : ID_s,\ Ticket_{tgs},\ Authenticator_c$$

$$*Authenticator_c = E_{K_{c,tgs}}[ID_c \| AD_c \| TS_3]$$

[단계4](메시지4)

$$TGS \rightarrow C : E_{K_{c,tgs}}[K_{c,s} \| ID_s \| TS_4 \| Ticket_s]$$

$$*Ticket_s = E_{K_s}[K_{c,s} \| ID_c \| AD_c \| ID_s \| TS_4 \| lifetime_3]$$

(c) 클라이언트/서버 인증 교환

[단계5](메시지5)

$$C \rightarrow S ;\ Ticket_s,\ Authenticator$$

$$*Authenticator = E_{K_{c,s}}[ID_c \| AD_c \| TS_5]$$

[단계6](메시지6)

$$S \rightarrow C ;\ E_{K_{c,s}}[TS_5 + 1]$$

서버 S는 Tickets에서 복구된 ID_c, AD_c가 $K_{c,s}$를 이용하여 인증자로부터 복구된 ID_c, AD_c가 일치하는지 검사하여 클라이언트 C의 정당성을 확인한다. 정당한 클라이언트 C이면, 서버 S는 자신이 정당한 서버 S임을 증명하기 위해 메시지 6을 클라이언트 C에게 보내며, 클라이언트 C는 이를 복호화한 후, TS_5+1를 확인하여 서버 S의 정당성을 확인한다.

III. P2P를 위한 개선된 Kerberos 제안

P2P 시스템의 시장이 급속하게 성장함에 따라 보안의 위험성도 증가하고 있다. Peer간의 허가받지 않은 자의 접근 방지와 신뢰성 있는 상호 인증이 절실히 필요하다. 그러나 기존의 인증 방법을 P2P 시스템에 그대로 적용한다면, 클라이언트와 서버 중심의 인증 방식은 P2P 시스템에 잘 맞지 않을 뿐 아니라, 인증을 위한 서버의 부하량과 트래픽의 증가로 서버의 부하량과 트래픽 분산이라는 P2P 시스템의 개념에 맞지 않는다. 이에 본 논문에서는 인증시 서버의 부담을 최소화하고, 최대한 Peer를 활용할 수 있는 알고리즘을 제안한다.

3.1 제안된 알고리즘의 동작 과정

본 논문에서 제안하는 개선된 Kerberos 인증 알고리즘의 단계별 동작은 다음과 같다. P2P 시스템은 메인 서버와 Peer A, Peer B로 이루어져 있다. 메인 서버는 모든 Peer들의 ID와 Password를 관리하고 있고, 인증 서버(AS)의 역할을 한다. Peer A는 파일이나 정보를 필요로 하고, 요청하는 사용자이다. 그리고, 여기서 Peer B는 정보나 파일을 제공하는 Peer로써 티켓 승인 서버(TGS)와 서버의 역할을 같이하고 있다.

[단계1](메시지1)

$$Peer\ A \rightarrow AS:\ ID_a \| ID_b \| TS_1$$

Peer A가 Peer B에 접근하여 파일을 다운로드 하고자 할때는 ID_a, ID_b, TS_1를 AS에 보내어 서비스를 요청한다. 여기서, ID_a는 사용자의 신분을 알리기 위한 Peer A의 ID이고, ID_b는 엑세스를 원하는 Peer B(TGS B)의 ID이다. Peer B로부터 티켓을 구하기 때문에, 직접 ID_b를 사용하였다. TS_1는 Timestamp 정보로 현재 서비스를 요구하는 시각을 나타낸다. TS_1를 이용하여 AS는 AR(Authenticative Request) 정보가 시기적절한가를 확인한다. TS_1 정보는 replay 공격을 방지할 수 있다.

[단계2](메시지2)

$$AS \rightarrow Peer\ A:\ E_{K_a}\ [K_{a,\,tgsb} \| ID_b \| TS_2 \| lifetime_1 \| Ticket_{tgsb}]$$

$$AS \rightarrow Peer\ B(TGS\ B):\ E_{K_b}[K_{tgsb} \| ID_a \| TS_2]$$

$$*Ticket_{tgsb} = E_{K_{tgsb}}\ [K_{a,\,tgsb} \| ID_a \| AD_a \| ID_b \| TS_2 \| lifetime_1]$$

AS는 클라이언트와 Peer B(TGS B)간의 세션키($K_{a,tgsb}$)와 $Ticket_{tgsb}$를 각각 생성하여 Peer A의 패스워드를 일방향 해쉬함수에 적용하여 구한 사용자 키 K_a로 암호화하여 Peer A에게 전달한다. $K_{a,tgsb}$는 Peer A와 Peer B(TGS B)간의 안전한 암호통신을 위한 일회용 세션키로 AS에서 생성한다. $K_{a,tgsb}$는 K_a로 암호화된 메시지 내부에 있기 때문에 Peer A만이 $K_{a,tgsb}$를 알 수 있다. Ka는 Peer A의 패스워드로부터 유도된 비밀암호키로서 패스워드를 일방향 해쉬 함수 MD_5를 이용하여 구한다. TS_2는 $Ticket_{tgsb}$가 발행된 시간이며, $lifetime_1$는 $Ticket_{tgsb}$의 유효시간을 나타낸다. 여기서, AD_a는 $Ticket_{tgsb}$를 사용할 Peer B의 IP 주소를 사용한다.

또한 동시에 AS는 Peer B(TGS B)만을 위해 생성한 일회용 키 K_{tgsb}와 ID_b, TS_2를 Peer B(TGS B)의 패스워드를 일방향 해쉬함수에 적용하여 구한 사용자 키 K_b로 암호화하여 Peer A에게 전달한다.

[단계3](메시지3)

$$Peer\ A \rightarrow Peer\ B(TGS\ B):\ Ticket_{tgsb},\ Authenticator_a$$

$$*Authenticator_a = E_{K_{a,\,tgsb}}\ [ID_a \| AD_a \| TS_3]$$

Peer A는 AS로부터 수신하여 구한 $K_{a,tgsb}$를 이용하여 생성한 Authenticatora와 $Ticket_{tgsb}$를 Peer B(TGS B)에게 전송한다. 여기서 $Authenticator_a$는 $Ticket_{tgsb}$의 유효성을 검증하기 위한 정보로 Peer A와 $Ticket_{tgsb}$간의 세션키 $K_{a,tgsb}$로 ID_a, AD_a, TS_3를 암호화한 암호문이다. TS_3는 인증자가 생성된 시간을 나타내는 정보로서, 재생 공격을 방지하기 위하여 매우 짧은 유효시간을 준다.

Peer B(TGS B)는 단계2에서 받은 일회용 비밀키인 K_{tgsb}를 이용하여 $Ticket_{tgsb}$를 복호화하여 $K_{a,tgsb}$, ID_a, AD_a, ID_{tgs}, TS_2, $lifetime_1$를 복구한다. 그 다음 $K_{a,tgsb}$를 사용하여 인증자를 복호화한 후, Peer B(TGS B)는 $Ticket_{tgs}$에서 복구된 ID_a, AD_a와 인증

자로부터 복구된 ID_a, AD_a가 서로 일치 하는지 확인함으로써, Peer A의 정당성을 확인하게 된다.

[단계4](메시지4)

$$Peer\ B(TGS\ B) \rightarrow Peer\ A : E_{K_{a,tgsb}}[K_{a,b} \| TS_4 \| Ticket_b]$$

$$*Ticket_b = E_{K_b}[K_{a,b} \| ID_a \| AD_a \| ID_b \| TicketNum \| downloadcount]$$

Peer B(TGS B)는 $Ticket_{tgsb}$로부터 구한 $K_{a,tgsb}$를 이용하여, Peer B와 Peer A와의 세션키 $K_{a,b}$, 그리고 $Ticket_b$를 암호화하여 Peer A에게 보낸다. 여기서 ID_b는 Peer B의 ID를, TS_4는 $Ticket_b$가 발행된 시간을 나타낸다. $Ticket_b$는 $K_{a,b}$, ID_a, AD_a, ID_b, TicketNum, downloadcount를 Peer B와 AS가 공유하고 있는 비밀키인 K_b(Peer B의 패스워드를 일방향 해쉬함수에 적용하여 구한 사용자 키)로 암호화한 암호문이다. $Ticket_b$내의 TicketNum는 $Ticket_b$의 고유번호로써 Peer B의 $Ticket_b$의 유효기간까지 메모리에 저장된다. downloadcount는 파일을 다운받을 수 있는 개수를 의미한다. P2P 시스템에서는 파일이나 정보를 받기 위하여 많은 시간이 소요될 수도 있다. 이러한 P2P의 특성상 Ticket의 유효시간보다 다운로드 횟수를 제한하는 것이 바람직하다. downloadcount 값은 Peer가 임의로 설정할 수 있다(여기서는 Peer B).

$Ticket_s$의 유효시간을 의미하며, TGS와 서버 S가 사전에 공유하고 있는 비밀키인 K_s로 암호화한 암호문이다.

[단계5](메시지5)

$$Peer\ A \rightarrow Peer\ B\ ;\ Ticket_b,\ Authenticator$$

$$*Authenticator = E_{K_{a,b}}[ID_a \| AD_a \| TS_5]$$

Peer A는 Peer B(TGS B)와의 세션키 $K_{a,tgsb}$를 이용하여 암호문으로부터 Peer A와 Peer B용의 세션키 $K_{a,b}$를 복구한 후, 이를 이용하여 구한 인증자 Authenticator를 $Ticket_b$와 함께 서버 S에 전송한다. Authenticator는 $K_{a,b}$로 ID_a, AD_a, TS_5를 암호화한 값이다. $Ticket_b$는 Peer A가 AS에 의해 인증 받았음을 확인하기 위한 정보로, 재사용이 가능하다. ID_a는 $Ticket_{tgsb}$의 정당한 소유자의 ID이고, AD_a는 Peer A의 IP주소로서 다른 주소를 갖는 Peer로부터의 $Ticket_{tgsb}$의 사용을 방지하기 한다. 인증자 Authenticator는 $Ticket_b$를 제시하고 있는 Peer A가 $Ticket_b$를 발행받은 정당한 사용자임을 입증하기 위한 정보이며, TS_5는 인증자가 생성된 시간이다.

[단계6](메시지6)

$$Peer\ B \rightarrow Peer\ A\ ;\ E_{K_{a,b}}[TS_5+1]$$

Peer B는 $Ticket_b$에서 복구된 ID_a, AD_a가 $K_{a,b}$를 이용하여 인증자로부터 복구된 IDa, ADa가 일치하는지 검사하여 Peer A의 정당성을 확인한다. 정당한 Peer A이면, Peer B는 자신이 정당한 Peer B임을 증명하기 위해 메시지 6을 Peer A에게 보내며, Peer A는 이를 복호화한 후, TS_5+1를 확인하여 Peer B의 정당성을 확인한다.

3.2 알고리즘의 효율성과 분석

본 논문에서 제안한 알고리즘은 P2P 시스템에 적합하게 개선된 Kerberos 알고리즘을 제안하고 있다. 그러나 서버의 부담을 분산시키기 위한 개념으로 발달된 P2P 시스템에서는 인증을 위하여 Kerberos를 그대로 적용시키는 것은, 티켓 발행과 인증으로 인한 서버의 부담을 가중시키는 결과를 초래한다. 또한 티켓 승인 서버(TGS)라는 또 하나의 서버가 필요로 하게 된다.

본 논문에서 제안한 알고리즘은 P2P의 특성상 서버와 클라이언트의 기능을 모두 가질 수 있는 Peer에게 티켓 승인 서버의 기능을 부여함으로써 이를 해결하였다. 하지만 또 하나의 인증을 위한 서버인 티켓 승인 서버를 사용하지 않음으로써 생길 수 있는 인증의 문제는 [단계2]에서 다음과 같은 메시지를 상대측(Peer B)으로 보냄으로써 해결할 수 있다.

$$AS \rightarrow Peer\ B(TGS\ B) : E_{K_b}[K_{tgsb} \| ID_b \| TS_2]$$

K_{tgsb}라는 서버와 상대측(Peer B)만이 알 수 있는 일회용 키를 사용함으로써 상대측은 티켓 승인 서버의 기능을 할 수 있다. 이러한 메시지의 추가로 인한 또 다른 이점이 있다. 티켓 발행의 위하여 K_{tgsb}라는 일회용 키를 사용함으로써 재전송(replay) 공격에 강력하고 견고하다. 이러한 K_{tgsb}는 외부에 노출이 되더라도, 일회용 키로써 재사용이 불가하다.

또한, [단계4]에서 티켓의 고유번호인 Ticket Num와, 실행횟수 제한값인 downloadcount를 추가함으로써 P2P 시스템의 특성상 시간적 인증보다는 횟수의 인증과 시간적 인증을 겸하였다.

한번 인증을 받은 Peer는 재접속을 위하여 인증을 받을 필요가 없다. 티켓에 허용된 횟수만큼은 언제든지 [단계5]와 [단계6]의 반복만으로 상호 인

증이 가능하다. 또한 제안된 알고리즘은 상호 인증을 위해 인증 서버(메인 서버)가 수행해야할 단계는 [단계1], [단계2]뿐이다. 그러므로 제안된 알고리즘은 P2P 시스템에서 인증을 위한 메인서버의 기능을 최소화할 수 있다.

표 1. 기존 Kerberos와 제안된 알고리즘과의 비교 분석

	Kerberos	제안된 알고리즘	비 고
총 시스템 개수	4(AS, TGS, C, S)	3(AS, Peer A, Peer B)	
서버의 총개수	2(AS, TGS)	1(AS)	서버의 개수 감소
티켓 발행	중앙처리	분산처리	서버 부하량과 트래픽 분산
티켓 발행시 서버이용 단계	4(AS→2, TGS→2)	2	서버 부하량과 트래픽 분산
Ticket 발행을 위한 키	고정 비밀키	일회성 비밀키	replay 공격에 강력하고, 일회성 비밀키는 노출이 되어도 재사용 불가
Ticket 발행	TGS	Peer	상대측의 Peer에서 Ticket을 발행함으로써 Ticket 발행 부담 분산
Ticket 인증 유효 기간	lifetime(시간)	downloadcount (다운 횟수)	P2P 환경에 맞도록 인증 유효 시간이 아닌 다운 횟수를 인증 유효 설정 값으로 사용

IV. 보안 P2P 시스템 설계

4.1 Peer 프로그램 설계

4.1.1. 리스너(Listener)

리스너는 피어의 접속 요청이 들어오는 것을 처리한다. 리스너의 중요한 역할은 서버에 로그인 하는 것이다. 리스너는 서버에게 자신의 존재유무를 알리고, 자신의 공유된 자원을 선언한 후, 파일과 폴더의 이름을 위치와 함께 전송한다.

리스너는 로그인 페이지가 성공적으로 호출되면, 서버로부터 응답을 받게 되고 로그인 인증 여부를 확인하기 위해 이 응답을 파싱한다. 로그인시 사용되는 패스워드는 해쉬함수 MD_5를 이용하여 128bit의 키로 변환되어 로그인된다. 또한 리스너가 자신의 컨텐츠나 정보를 읽기 전용 혹은 쓰기 가능뿐만 아니라, 읽기 가능한 상태로 지정할 수 있다.

본 논문에서 리스너는 다수의 접속을 처리하기 위하여, 모든 접속 요청에 대하여 Thread를 생성하도록 설계하였다. 모든 접속 요청에 대한 감시는 Thread하에서 이루어진다. 리스너는 수신된 모든 요청이 유형에 따라서 적절한 응답이 생성되고 브라우처측의 네트워크 스트리밍에 응답을 기록하여 브라우저가 읽을 수 있도록 한다.

이러한 리스너가 Peer 시스템에서 티켓 발행 서버(TGS)의 기능하도록 설계하였다. 상대측 Peer이 브라우저에서 접속하여 다운로드 권한을 얻기 위한 인증을 요구하면, 제안된 알고리즘에 의해서 단계2~단계6을 차례로 실행한다.

리스너의 제안된 인증 알고리즘을 위한 동작 순서는 다음과 같다.

1) 입력 패킷이 티켓값($Ticket_b$, Authenticator)인지, 티켓 요청값($Tickettgs_b$, Authenticatora)인지 판단한다.
   - 만약 Ticketb, Authenticator 이면, 4)단계로 간다.
   - 만약 Tickettgsb, Authenticatora 다음 단계를 실행한다.
2) $Ktgs_b$를 이용하여 $Ticket_{tgsb}$를 복호화한다. 즉, $DK_{tgsb}[Ticket_{tgsb}]$를 실행하여, $K_{a,tgsb}$를 구하고, Authenticatora를 복호화한다. $Ticket_{tgsb}$와 Authenticatora를 복호화하여 구한 ID_a, AD_a가 서로 일치하는지 확인한다. 일치하면 다음 단계를 실행하고, 일치하지 않으면 에러 메시지를 보낸다.
3) TicketNum와 downloadcount를 포함한 티켓 $Ticket_b$를 생성하여, 상대측 Peer에 보낸다. TicketNum와 downloadcount는 메모리에 임시 저장하고, 상대측 Peer의 logout시 메모리에서 삭제한다. downloadcount값은 옵션으로 설정할 수 있도록 설계 하였다.
4) 상대측 Peer로부터 $Ticket_b$, Authenticator를 받으면, 자신의 비밀키를 MD_5를 이용하여 만들어진 128bit의 키 Kb를 이용하여 $Ticket_b$를 복호화한다. 즉, $DK_b[Ticket_b]$를 실행하여, $K_{a,b}$를 구하고, 이 키를 이용하여 Authenticator를 복호화한다. $Ticket_b$와 Authenticator를 복호화하여 구한 ID_a, AD_a가 서로 일치하는지 확인한다. 일치하면 다음 단계를 실행하고, 일치하지 않으면 에러 메시지를 보낸다.

상대측 Peer는 downloadcount값이 만료될 때까지 따로 서버(AS)에 티켓을 요구하지 않고, $Ticket_b$와 Authenticator값만을 이용하여 인증을 받는다. 이러한 방법은 매번 파일을 다운로드할 때마다 서버(AS)의 인증을 받아야하는 불편함을 덜고, 티켓-

승인 티켓을 만들기 위한 트래픽과 부하량을 감소시킬 수 있다.

5) TS_5값을 TS_5+1의 값을 만들어 서로 Peer간의 비밀키 $K_{a,b}$로 암호화한 값 $EK_{a,b}[TS_5+1]$를 상대측에 보낸다. 티켓의 유효횟수가 남아있다면 5)과 6)단계의 반복만으로 상호 인증을 한다.

4.1.2 브라우저 설계

브라우저는 사용자와 컴퓨터 사이에 인터페이스 역할을 한다. 브라우저를 통하여 요청을 전송하고 또한 리스너의 응답을 수신한다. 브라우저는 로그인한 리스너들의 목록을 보여주고, 모든 요청을 리스너에게 전송한다. 제안된 인증 알고리즘을 위하여 다음과 블록이 추가되었다.

1) 접속하고자 하는 Peer의 티켓값($Ticket_b$, Authenticator)이 존재하는지 체크한다.
- 티켓이 존재하면, 상대측의 Peer에 전송한다. 회신되는 데이터가 $EK_{a,b}[TS_5+1]$이면, 5)단계로 넘어간다.
- 티켓이 존재하지 않거나 그 외의 회신 데이터가 수신되면, 인증 서버(AS)에 IDa, IDtgsb, TS1 값을 보내어 서비스를 요청한다.

2) AS에서 수신되는 데이터를, Peer 사용자의 패스워드를 해쉬함수 MD_5에 적용하여 구한 128bit의 키값 K_a을 이용하여 복호화한다. 그리하여 티켓 발행을 위한 임시 키값 $K_{a,tgsb}$와 티켓
- 승인 티켓 $Ticket_{tgsb}$를 구한다.

3) 임시 키값 $K_{a,tgsb}$을 이용하여 ID_a, AD_a, TS_3를 암호화하여 생성한 Authenticatora와 $Ticket_{tgsb}$를 상대측 Peer에 전송한다.

4) 상대측의 Peer로 부터 받은 데이터를 키값 $K_{a,tgsb}$로 복호화하여, Peer 양측만이 공유하는 키값 $K_{a,b}$와 $Ticket_b$를 구한다.

5) Peer 양측만이 공유하고 있는 키값 $K_{a,b}$를 이용하여 ID_a, AD_a, TS_5를 암호화하여 생성한 Authenticator와 $Ticket_{tgsb}$를 상대측 Peer에 전송한다.

6) 상대측의 Peer로 부터 받은 데이터를 키값 $K_{a,b}$를 이용하여 구한 값이 TS_5+1인지를 확인한다. 일치하면 인증이 종료된 것이고, 일치하지 않는다면 에러 메시지와 함께 1)단계부터 다시 시작한다.

4.1.3 서버 설계

본 논문에서는 Hybrid형 P2P 시스템으로 서버는 아주 최소한의 기능만 하도록 설계 되었다. 서버는 정보의 검색기능과 인증기능, 메시지의 일시적 보관기능 등을 수행한다. 데이터베이스(DB)를 위하여 SQL Server 2000을 사용하였다.

서버에서 인증을 위한 블록은 다음과 같다.

1) 다른 Peer에 접속하고자 하는 Peer의 요청 정보를 받아 Peer의 ID들을 확인한다.

2) 인증서버는 티켓-승인 티켓 $Ticket_{tgs}$와 서비스-승인 티켓 발행을 위한 일회용 임시 키값 $K_{a,tgsb}$와 K_{tgsb}를 생성한다. 요청하고자 하는 Peer측의 패스워드를 해쉬함수 MD_5에 적용한 키값 K_a를 이용하여, $K_{a,tgsb}$, ID_b, TS_2, $lifetime_1$, $Ticket_{tgsb}$를 암호화하여 요청자측의 Peer에 전송한다.

또한 동시에 K_{tgsb}, ID_a, TS_2를 K_b값으로 암호화하여 피요청자측의 Peer에 전송한다.

이와 같이 Peer의 상호 인증을 위한 메인 서버의 역활은 매우 간단하다.

4.2 제작된 보안 P2P 프로그램

- 보안 P2P 프로그램 메인 화면

첫 번째 탭에는 전송상태 목록을 나타낼 수 있다. 두 번째 탭에서는 다른 Peer 접속자 및 목록 보기, 세 번째 탭에서는 자신의 파일의 공유설정할 수 있는 공유폴더 설정, 마지막으로 로그기록과 티켓을 보기 위한 사용자관리 탭이 있다.

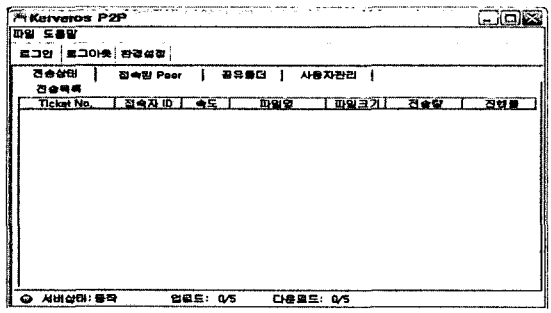

그림 1. Kerberos P2P 시스템

- Login 화면

그림 2는 로그인 창으로 사용자의 비밀번호를 해쉬함수 MD_5를 이용하여 구한 키값을 이용하여 로그인 한다.

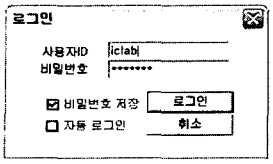

그림 2. Login 화면

－환경 설정

환경 설정에서 포트번호는 8090,8091을 고정으로 사용하고, Ticket의 다운로드 허용 갯수 downloadcount를 설정할 수 있는 옵션이 있다.

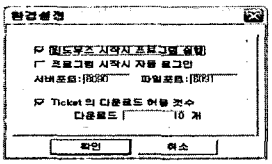

그림 3. 환경 설정

－접속된 Peer 목록

그림 4는 접속된 Peer 목록 및 공유 파일 목록을 보여 준다. 오른쪽에는 Online 상태인 Peer의 ID 목록과 왼쪽에는 현재 접속 중인 Peer의 공유 목록들이 나타난다. 파일을 선택하고, 다운로드를 클릭하면 인증을 시도한다.

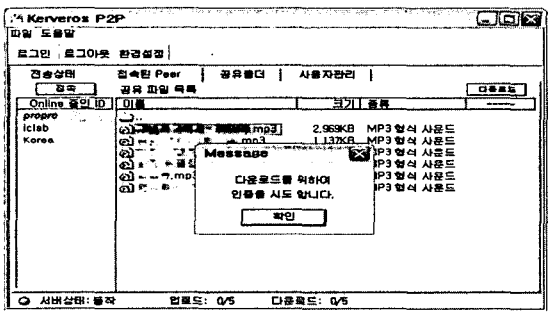

그림 4. 접속된 Peer 목록 및 인증시도

－티켓 승인 메시지

티켓 값이 올바르게 승인이 되면 다음과 같은 메시지가 뜬다.

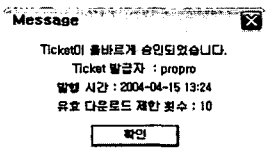

그림 5. 티켓 승인 메시지

－발행된 티켓을 관리

그림 6은 사용자 관리 탭을 이용하여 접속자 목록 및 접속한 서버 목록을 확인할 수 있다. 또한 각각 접속자에 따른 발행된 티켓을 확인할 수 있다. 티켓의 발급자, 발급 대상, Ticket 값, 발급 시간, 다운로드 제한 값, 사용 횟수 등을 확인할 수 있다.

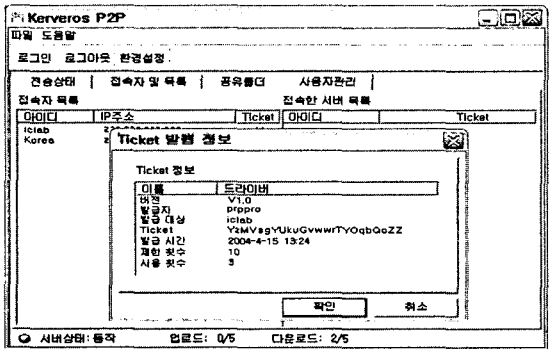

그림 6. Ticket 발행 정보

V. 결 론

본 논문에서는 상호 인증을 위한 알고리즘인 Kerberos를 개선하여 P2P 시스템에 맞는 알고리즘을 제안하였다. 제안된 알고리즘은 서버의 부담을 최소화하기 위하여 티켓 승인 서버의 기능을 상대측의 Peer에 부여하였다. 이러한 방법을 이용하여 티켓 승인 서버를 위한 서버의 개수 증가를 막고, 인증을 위한 서버의 기능을 최소화하여 서버의 부담을 최소화하였다. 제안된 알고리즘은 상호 인증을 위하여 서버는 최소한의 역할만하고, Peer간에 인증의 역할을 부여하면서도 강력한 상호 인증을 할 수 있다. 또한 P2P 시스템에 맞도록 시간위주의 인증 만료 시간보다, 횟수 위주의 인증 제한 값을 주었다. 이러한 알고리즘을 이용하여 P2P 시스템을 설계하여 직접 적용하여 보았다.

또한 이러한 인증과정에서 생기는 키값을 이용한 암호화 통신은 기밀성을 보장한다. 향후 연구과제로는 P2P 시스템에서 부인 방지 기술 및 컨텐츠 보호 기술의 적용이 되어야 할 것이다.

참고문헌

[1] David Barkai, Peer-to-Peer Architecture

Group, Microcomputer Research Lab and Intel Corporation, "An Introduction to Peer-to-Peer Computing", Intel®Devloper-UPDATE Magazine, pp.1-7, February 2000.

[2] I. Foster, C. Kesselman, G. Tsudik, S. Tuecke. "A Security Architecture for Computational Grids." Prpc. 5th ACM Comference on Computer and Communications Security Conference, pp. 83-92, 1998

[3] Azzedin, F.; Maheswaran, M.; "Towards trust-aware resourece management in Grid computing systems," Cluster Computing and the Grid 2nd IEEE/ACM International Symposium CCGRID2002, 21-24, pp. 419-424, May 2002

[4] Kant, K., Iyer, R., and Tewari, V.;"A framework for classifying peer-to-peer technologies", Cluster Computing and the Grid 2nd IEEE/ACM International Symposium CCGRID2002, 21-24, pp. 338-345, May 2002.

[5] J. Kohl and C. Neuman, "The Kerberos Network Authentication Service(V5)," RFC 1510, September, 1993

저자소개

김종우(Jong-Woo Kim)

1998년 2월 조선대학교 전자공학과 학사
2000년 8월 조선대학교 대학원 전자공학과 석사
2004년 현재 조선대학교 대학원 전자공학과 박사과정
※관심분야 : 통신보안시스템설계, 네트워크 보안, DRM, 무선 네트워크 보안

한승조(Seung-Jo Han)

1980년 2월 조선대학교 전자공학과 학사
1982년 2월 조선대학교 대학원 전자공학과 석사
1994년 2월 충북대학교 대학원 전자계산학과 박사
1986년 6월~1987년 3월 Univ. of New Oleans 객원교수
1995년 2월~1996년 1월 Univ. of Texas 객원교수
1997년~현재 조선대학교 정보통신공학과 교수
※관심분야 : 통신보안시스템설계, 네트워크 보안, ASIC설계, DRM