그물망 위상의 P2P 네트워크를 활용한 파일 분리 분산 방안

이명훈* · 박정수* · 김진홍* · 조인준

The File Splitting Distribution Scheme Using the P2P Networks with The Mesh topology

Myoung-Hoon Lee* · Jung-Su Park* · Jin-Hong Kim · In-June Jo*

요 약

최근 들어 무선 단말장치의 소형화 및 파일크기의 대형화 추세로 소형 무선 단말에서 대형 파일처리의 문제점이 대두되고 있다. 또한 파일서버나 웹 서버에 수많은 파일의 집중화로 인한 과부하가 문제점으로 쟁점화 되고 있다. 그리고 데이터 처리가 단일 파일단위를 기반으로 함에 따라 보안상 여러 취약점을 지닌다.

본 논문에서는 이러한 문제점 해결을 위한 새로운 방안으로 그물망 위상의 P2P를 활용한 파일 분리 분산 방안을 제안하였다. 파일을 그물망 구조의 P2P를 활용하여 분리 분산시킴으로써 소형 단말에 적합한 파일을 생성할 수 있고, 파일이 특정 서버에 집중화 되지 않음에 따라 과부하를 예방하고, 한 파일이 여러 피어에 분산됨에 따라 보안상 취약점을 완화할 수 있다.

ABSTRACT

Recently, the small sized wireless terminals have problems of processing of large sized file because of the trends of a small sized terminals and a large sized files. Moreover, the web servers or the file servers have problems of the overload because of the concentration with many number of files to the them. Also, There is a security vulnerability of the data processing caused by the processing with a unit of the independent file.

To resolve the problems, this paper proposes a new scheme of file splitting distribution using the P2P networks with the mesh topology. The proposed scheme is to distribute blocks of file into any peer of P2P networks. It can do that the small sized wireless terminals can process the large size file, the overload problems of a web or file servers can solve because of the decentralized files, and, the security vulnerability of the data processing is mitigated because of the distributed processing with a unit of the blocks to the peers.

Keyword

File splitting, P2P Network, Small sized terminal, Large sized file, Wireless terminal

Ⅰ. 서 론

최근 들어 멀티미디어 데이터를 중심으로 대형화된 파일이 기하급수 적으로 증가되고 있다. 하지만, 이들을 처리하는 단말은 이동성 및 휴대성을 극대화시키는 방향으로 발전함에 따라 소형화 추세로 전개되고 있다 [4]. 또한 90년대에 보편화되어 발전해온 클라이언트/서버 컴퓨팅 환경은 서버에게 부하가 집중화됨에 따라

* 배재대학교

접수일자 : 2005. 10. 6

서 이에 대한 위기론이 제기되기 시작하였다. 최근 들어서는 이에 대한 대안으로 P2P 기반의 컴퓨팅 환경이 급부상하고 있는 추세이다[8]. 다음으로 보안상 측면에서 보면 물리적으로 한 단말기에서 하나의 단위로 파일이 저장되고 처리됨에 따라 중요 파일에 대한 보안상 취약성이 노출될 수밖에 없다. 즉, 단일 단말기에 저장된 중요 파일이 공격자에게 집중적인 공격목표가 되기 때문에 파일 탈취 및 변조 공격, 그리고 Brute Force 및 DOS공격[4]에 쉽게 노출될 수 있다.

본 논문에서는 이러한 문제점 해결을 위한 하나의 방안으로 파일 분리 분산처리 방안을 새롭게 제안하였다. 이를 위하여 최근에 대두되고 있는 P2P네트워크기술과 분리된 파일들의 안전성 확보를 위해 암호 및 해쉬 보안기술을 활용하였다.

본 논문은 II장에 관련연구, III장에 제안 시스템 구조, IV장에 제안 시스템의 동작 절차, V장에 제안 시스템의 고찰 및 검토 내용을 다루었고, 마지막으로 VI장에 결론을 맺었다.

II. 관련 연구

기존의 P2P 네트워크에서 파일 분리 분산 기술은 파일의 익명성 제공을 위한 방안으로 제안되었다. 그 예는 Mnet, Mojo, Freenet, GNUnet 등이다2,4,7]. Mnet[1]과 Freenet[6]은 8개의 블록으로 분리 분산하는 방안을 제시하고 있으나 그 구체적인 사양은 공개되지 않고 있다. Freenet도 분리 방안만을 제시하고 있을 뿐이다.

이 중에서 GNUnet[3]은 구체적으로 1 KB블록으로 파일을 분리하여 분산시키는 방안을 제시하였다. 하지만 파일 분리 분산 목적이 파일 익명성 보장에 있기 때문에 모든 블록을 고정 1 KB 블록으로 하여 모든 블록을 피어(peer)에게 분산시키는 방안을 사용하고 있다.

본 논문에서는 고정된 피어에서 하나의 파일을 196개의 블록으로 분리 분산하고, 이를 논리적으로 하나의 파일처럼 검색처리 되는 방안을 새롭게 제시한 것이다. 즉, 파일 분리 분산이 익명성 보장 목적이 아니다. 소형 무선 단말에서 대형 파일 처리를 가능케 하고, 서버에 집중된 파일을 분산시키고, 파일 분리 분산에 의한 보안성 강화를 위한 새로운 제안이다.

III. 제안시스템 구조

그림 1과 같이 제안 시스템은 파일 분산에 동의한 피어들이 그물망 위상의 P2P 네트워크를 통해서 서로 연결된 집합으로 구성된다. 이는 파일 분산 및 재구성 기능의 효율성 증대를 위해 그물망 위상의 오버레이 네트워크 구조를 선택한 것이다. 따라서 각 노드는 파일 분산에 참여하는 모든 피어들의 IP주소 목록을 저장하는 데이터 구조체를 가진다. 그리고 분산된 파일을 저장하는 디스크 영역이 할당된다.

각 구성요소의 기능 및 역할을 정리하면 다음과 같다. 각 피어의 역할은 첫째, 파일을 분리하여 분배하는 역할을 행한다. 이때 다음 장에서 설명하는 파일 분리 인코딩 법칙에 따라 파일을 블록으로 분리하고 분리된 블록을 랜덤 방식으로 선택된 피어에게 분산 시키는 기능을 한다. 둘째, 사용자가 요구한 파일 탐색기능을 한다. 파일 탐색 요구를 받은 피어는 자신의 루트 블록 저장 파일로부터 해당 루트 블록을 추출하여 제안 시스템에 산재되어 있는 데이터 블록을 질의하여 이에 대한 응답으로 수신된 결과들을 순차적으로 결합하여 원형 파일을 복원한다.

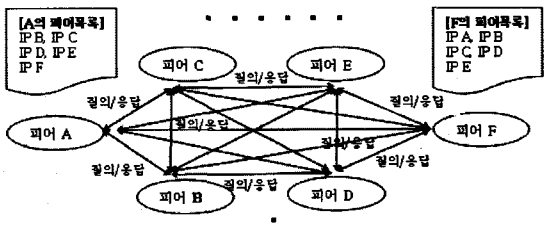


그림 1. 6개의 피어로 구성될 경우 제안 시스템의 구조
Fig 1. The structure of proposed system

IV. 제안 시스템의 동작절차

1. 파일 분리 인코딩 및 블록 분산 절차

제안 시스템은 파일을 가변길이 블록으로 인코딩하여 분리하고, 랜덤하게 선택된 2개의 피어에게 이를 전송하여 블록을 분산시키는 방안이다. 이들 블록은 3가지 유형으로 나누었다. 즉, D(Data)블록, I(Indirection)블록, R(Root)블록이다. 이들 블록의 구체적인 생성 및

분산 절차는 다음과 같다.

(1) *D* 블록 생성

이는 원형 파일을 196개로 분리하여 생성된 데이터 블록(즉, *D* 블록)이다. 이의 생성과정은 다음과 같다.

step1) 원형파일 F 를 196개 블록으로 분리한다.
F = {D1, D2 …… D196},
D블록 길이 = (file size)KB / 196

step2) 분리된 각 블록 Di 를 160비트 RIPE MD[2,5]로 해쉬한다.
{H(D1), H(D2), …,H(D196)}

step3) 상기 해쉬값을 키로 각 D 블록을 암호화하여 최종 D 블록 생성을 완료한다.
{EH(D1)(D1), EH(D2)(D2),……. EH(D196(D196}

(2) *I* 블록 생성 및 *D* 블록 분산

I 블록은 196개로 분리된 D 블록을 색인하기 위해 만들어진 우회블록을 의미한다. 이들의 생성과정은 다음과 같다.

step1) 생성된 D 블록을 160 RIPE MD로 해쉬하여 D 블록의 저장소를 생성한다.
{H(EH(D1)(D1)), H(EH(D2)(D2)),…….
…. H(EH(D196)(D196))}

step2) 각 D 블록에 대해 평문 D 블록의 해쉬값과 암호화된 D 블록의 해쉬값을 선택하고 이들이 분산될 피어의 IP주소 2개를 자신의 피어 목록에서 선택하여 D--blocks-idx를 생성한다.
D-blocks-idx =
{(H(D1), ip-add1(D1), ip-add2(D1), H(EH(D1)(D1))),
(H(D2), ip-add1(D2), ip-add2(D2), H(EH(D2)(D2)))…
…(H(D196), ip-add1(D196), ip-add2(D196),
H(EH(D196)(D196)))}

step3) 'step2)'의 196개의 D 블록에 대해 선택된 피어의 IP주소로 해당 D 블록을 전송한다. 이를 수신한 각 피어는 전송 받은 암호화 D 블록을 대상으로 암호화된 D 블록의 해쉬값을 파일이름으로 하여 저장한다.

step4) 'step2'에서 생성된 D blocks idx 를 구성하는 196개의 쌍을 대상으로 한 14개의 쌍을 한 묶음으로 14개 그룹을 생성한다.
D-blocks-idx1 =
{(H(D1),ip-add1(D1),ip-add2(D1),H(EH(D1)(D1))),
(H(D2),ip-add1(D2),ip-add2(D2),H(EH(D2)(D2)))
,……,
(H(D14),ip-add1(D14),ip-add2(D14),H(EH(D14)(D14)))}
,……,
D-blocks-idx14 =
{(H(D183),ip-add1(D183),ip-add2(D183),H(EH(D1)(D183))),(H(D184),ip-add1(D184),ip-add2(D184),H(EH(D184)(D184))),……,
(H(D196),ip-add1(D196),ip-add2(D196),H(EH(D196)(D196)))}

step5) 'step4)'에서 생성된 14개의 D-blocks-idx1 의 각각에 대해 해쉬값을 계산하여 다음의 I 블록을 생성한다.
I_1 = H(D-blocks-idx1)|D-block-idx1
I_2 = H(D-blocks-idx2)|D-blocks-idx2
……
I_{14} = H(D-blocks-idx14)|D-blocks-idx14

step6) 생성된 14개의 I 블록 암호화하여 최종 I 블록을 완성한다.
I 블록 = {I_1 , I_2, I_3,……, I_{14},}
- 각 블록 Ii를 160비트 RIPE MD로 해쉬한다.
{H(I_1), H(I_2), ….H(I_{14})}
- 상기 해쉬값을 키로 하여 각 I 블록을 암호화한다.
{$E_{H(I1)}$(I_1), $E_{H(I2)}$(I_2) ,……. $E_{H(I14)}$(I_{14})}

(3) 루트 *I* 블록 생성 및 *I* 블록 분산

루트 I 블록은 14개로 분리된 I 블록을 색인하기 위해 만들어진 우회 블록을 의미한다. 이들의 생성과정은 다음과 같다.

step1) 14개의 암호화 I 블록을 160 RIPE MD로 해쉬하여 암호화 I 블록의 저장소를 생성한다.
{H($E_{H(I1)}$(I_1)),H($E_{H(I2)}$(I_2)),… H($E_{H(I14)}$(I_{14}))}

step2) 각 I 블록에 대해 평문 I 블록의 해쉬값과 암호화된 I 블록의 해쉬값을 선택하고 이들이 분산될 피어의 IP주소 2개를 자신의 피어 목록에서 선택하여 I-blocks-idx를 생성한다.
I-blocks-idx =
{(H(I_1),ip-add1(I_1),ip-add2(I_1),H($E_{H(I1)}$(I_1))),
(H(I_2),ip-add1(I_2),ip-add2(I_2), H($E_{H(I2)}$(I_2)))…
(H(I_{14}),ip-add1(I_{14}),ip-add2(I_{14}),H($E_{H(I14)}$(I_{14})))}

step3) 'step2)'의 14개의 I 블록에 대해 선택된 피어의 IP주소로 해당 I 블록을 전송한다. 이를 수신한 각 피어는 전송 받은 암호화 I 블록을 암호화된 I 블록을 해쉬한 값을 파일이름으로

하여 저장한다.

step4) 'step2)'에서 생성된 I-blocks-idx 의 해쉬값을 계산하여 다음의 루트 I 블록을 생성한다.

I_{root} = $H(I\text{-}blocks\text{-}idx)|I\text{-}blocks\text{-}idx$

step5) 생성된 루트 I 블록 암호화하여 최종 루트 I 블록을 완성한다.

- 루트 I 블록을 160비트 RIPE MD로 해 쉬한다.

$\{H(I_{root})\}$

- 상기 해쉬값을 키로 하여 루트 I 블록을 암호화한다.

$\{E_{H(Iroot)}(I_{root})\}$

(4) *R* 블록 생성 및 루트*I* 블록 분산

R 블록은 루트 I블록을 색인하고 파일의 전체적인 기술을 위해 구성요소로 파일기술 내용, 파일 총 길이 정보, 루트 *I* 블록을 탐색하기 위한 인덱스(즉, *I-root-idx*)를 지닌다. 이들의 생성과정은 다음과 같다.

step1) 파일기술 내용 및 파일 *ID*를 생성한다.

step2) 1개의 암호화 루트 *I* 블록을 160 RIPE MD로 해쉬하여 암호화 루트 *I* 블록의 저장소를 생성한다.

$\{H(E_{H(Iroot)}(I_{root})\}$

step3) 루트 *I* 블록에 대해 평문 루트 *I* 블록의 해쉬값과 암호화된 루트 *I* 블록의 해쉬값을 선택하고 이들이 분산될 피어의 *IP*주소 2개를 자신의 피어 목록에서 선택하여 *I-root-idx*를 생성한다.

$I\text{-}root\text{-}idx$ =

$\{H(I_{root}),\ ip\text{-}add1(I_1),\ ip\text{-}add2(I_1),\ H(E_{H(Iroot)}(I_{root}))\}$

step4) 'step3)'의 루트 *I* 블록에 대해 선택된 피어의 IP주소로 해당 루트 *I* 블록을 전송한다. 이를 수신한 각 피어는 전송받은 암호화 루트 *I* 블록을 암호화된 루트 *I* 블록의 해쉬값을 파일 이름으로 하여 저장한다.

step5) 'step3)'에서 생성된 *I-root-idx* 의 해쉬값을 계산하여 다음의 *R* 블록을 생성한다.

$R = H(I\text{-}root\text{-}idx)|I\text{-}root\text{-}idx$

step6) 생성된 루트 R블록을 암호화하여 최종 루트 R 블록을 완성한다.

- *R* 블록을 160비트 RIPE MD로 해쉬한다.

$\{H(R)\}$

- 상기 해쉬값을 키로 하여 루트 *R* 블록을 암호화한다.

$\{E_{H(R)}(R)\}$

(5) *R* 블록 저장 및 탐색 키워드

파일을 생성한 노드가 유일하게 유지하는 *R*블록은 다음과 같은 절차에 따라 저장된다.

step1) 기밀 키워드 *Kj* 를 선택하여 160비트 RIPE MD로 해쉬하여 다음의 해쉬값을 계산한다.

$H(K_j),\ H(H(H(K_j)))$

step2) 첫번째 해쉬값 $H(Kj)$으로 다음을 암호화한다.

$E_{H(Kj)}\ (H(R)\ |E_{H(R)}(R))$

step3) 두번째 해쉬값 $H(H(H(K_j)))$을 파일이름으로 하여 'step 2)'에서 생성된 암호화된 R 블록을 저장한다.

상기와 같은 절차에 따라 생성된 파일 분리 인코딩 구조도는 그림2와 같다.

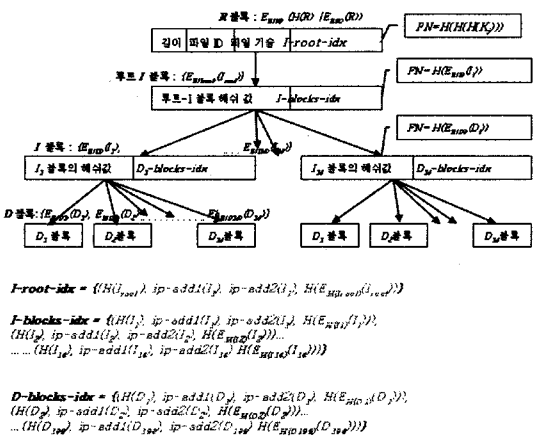


그림 2. 제안시스템의 파일분리 인코딩 구조도
Fig 2. File encoding scheme of proposed system.

2. 파일 탐색 절차

앞 절에서는 원형 파일 F 가 각각의 블록으로 분리되는 인코딩 방안과 분리된 블록이 어떻게 피어들에게 분산 되는지를 살펴보았다. 이 절에서는 사용자가 분리 분산된 블록을 어떻게 수집하여 원형 파일로 복원하여 사용하는 지에 대해 살펴보면 다음과 같다.

step1) K_j 를 선택하여 160비트 RIPE MD로 해쉬하여 해쉬값 $H(H(H(K_j)))$을 계산하여 루트 블록 R 을 탐색한다.

step2) H(Kj))를 계산하여 루트 블록 R을 복호시킨다.

$$D_{H(Kj)}(E_{H(Kj)}(H(R)|E_{H(R)}(R))) = H(R)|E_{H(R)}(R)$$

$$D_{H(R)}(E_{H(R)}(R)) = R$$

step3) 위에서 얻어진 R 블록의 *I-root-idx*항의 내용으로 루트 I 블록을 얻기 위해 목적지 주소를 '*ip-add1*(I_1), *ip-add2*(I_1)', 루트 I 블록이 저장된 파일이름으로 '$H(E_{H(Iroot)}(I_{root}))$'을 포함하는 질의문을 생성하여 해당 피어에게 전송한다.

step4) 'step3)'의 질의문에 대한 응답으로 암호화된 루트 I 블록이 수신되면, 자신의 *I-root-idx*항의 내용 중 '$H(I_{root})$'으로 루트 I 블록을 복호하고, 해쉬값을 계산, 루트 I 블록의 변조 유무를 검사한다.

$$I\text{-root-블록} = D_{H(Iroot)}(E_{H(Iroot)}(I_{root}))$$

step5) 'step4)'에서 얻어진 루트 I 블록의 *I-blocks-idx* 항의 내용으로 I 블록을 얻기 위해 목적지 주소를 '*ip-add1*(I_i), *ip-add2*(I_i)', 블록이 저장된 파일이름으로 '$H(E_{H(Ii)}(I_i))$),'을 포함하는 질의문을 생성하여 해당 피어에게 전송한다.

step6) 'step5)'의 질의문에 대한 응답으로 암호화된 I 블록이 수신되면, 자신의 *I-blocks-idx*항의 내용 중 '$H(I_i)$'으로 I 블록을 복호하고 해쉬값을 계산 I 블록의 변조 유무를 검사한다. 한다.

$$I_i = D_{H(Ii)}(E_{H(Ii)}(I_i))$$

상기의 'step5)'와 'step6)'를 14번 반복하면 $\{I_1, I_2, I_3, \cdots\cdots\cdots I_{14}\}$을 취득할 수 있다.

step7) 상기에서 얻어진 I_i 블록의 *D-blocks-idx* 항의 내용으로부터 D 블록을 얻기 위해 목적지 주소를 '*ip-add1*(I_i), *ip-add2*(I_i)', D 블록이 저장된 파일이름으로 '$H(E_{H(Di)}(D_i)$'을 포함하는 질의문을 생성하여 해당 피어에게 전송한다.

step8) 'step7)'의 질의문에 대한 응답으로 암호화된 D 블록이 수신되면, 자신의 *D-blocks-idx*항의 내용 중 '$H(D_i)$'으로 D 블록을 복호하고, 해쉬값을 계산, D 블록의 변조 유무를 검사한다.

$$D_i = D_{H(Di)}(E_{H(Di)}(D_i))$$

상기의 'step7)'와 'step8)'를 14개의 I 블록 각각에 대해 14번 반복하면 196개의 D블록을 취득하여 원형 파일을 복원할 수 있다.

$$F = \{D_1, D_2, D_3, \cdots\cdots\cdots D_{196}\}$$

V. 제안 시스템의 검토 및 고찰

제안 시스템은 4가지 측면에서 검토 및 고찰 될 수 있다. 먼저, 제안시스템의 기반구조에 대한 고찰이다. 제안시스템은 하나의 파일을 여러 블록으로 분리하여 물리적인 P2P 네트워크의 피어들에게 분산시킨다. 그럼에도 불구하고 논리적으로는 하나의 파일처럼 처리된다. 이를 위해서는 크게 2가지 조건이 만족 되어야 한다. 첫째, 지리적인 분산이 이루어지기 때문에 매우 효율적인 P2P네트워크가 사용되어야 한다. 즉, 블록의 분산과 재구성이 제안 시스템의 성능 좌우 요인이다. 따라서 본 제안 시스템에서는 그물망 위상의 P2P네트워크를 활용하였다. P2P네트워크는 TCP 상위계층에 형성된 논리적인 오버레이 네트워크이기 때문에 각 피어는 다른 모든 피어의 IP주소를 유지함으로써 그물망 위상의 네트워크를 형성한다. 이를 활용함으로써 블록의 분산 및 재구성의 성능을 제고하고 특정 통신채널에 집중화된 과부하 문제를 해결하였다. 둘째, 각각의 블록이 P2P네트워크의 피어들에게 분산됨에도 불구하고 이들이 하나의 파일처럼 다루어지기 위해서는 각 피어내의 블록들이 보호되어야 한다. 이를 위해 본 제안에서는 모든 피어는 자기가 생성하지 않은 블록에 대해서는 암호화된 블록을 저장한다. 그리고 파일을 요구한 피어가 블록을 송신 받았을 경우에는 파일의 변조 유무를 검사하는 해쉬 보안기술을 적용하였다. 따라서 파일이 블록 단위로 암호화되고, 파일 생성 피어가 블록의 해쉬값을 생성하여 유지한 후 P2P네트워크의 피어들에게 분산하기 때문에 하나의 파일처럼 관리된다.

다음으로 제안 시스템의 3가지 응용분야 문제점 해결에 대한 검토 및 고찰이다. 즉, 소형단말에서 대형파일처리 문제, 서버의 과부하 완화 문제, 파일 보안의 취약성 완화 문제 등을 위한 것이다. 따라서 이러한 3가지 측면에서 검토 고찰하면 다음과 같다.

첫째, 파일 크기의 대형화에 따라 소형 무선 단말기가 현재로서는 여러 개의 대형 파일을 저장하여 처리한다는 것은 불가능하다. 이는 소형 무선 단말기의 저장용량, 컴퓨팅 파워, 배터리, 그리고 용이한 휴대 등의 한계성들 때문이다. 이러한 문제점 해결책으로 제안한 시스템을 활용하면 저장장치가 없는 초소형 무선 단말기를 제작하여 대형파일을 원활히 처리할 수 있다. 즉, 대형 파일을 196개의 D 블록으로 분리하여 그 물망 위상의 P2P 네트워크를 통해 유선 서버에 분산시킨 다음, 초소형 무선 소형 단말기가 이 파일 요구 시 분산된 블록들을 하나로 재구성하여 원 파일로 복구시킴으로써 가능하다.

둘째, 클라이언트/서버로 구성된 포탈시스템을 운영하는 기관은 한결같이 특정 서버에 집중화된 파일의 과부하 문제를 제기하고 있다. 이 문제 해결을 위해 특정서버의 H/W 저장자원을 계속적으로 증설할 수만은 없다. 이러한 환경에서 제안 시스템을 활용하면 저장장치가 최소화된 서버구축이 가능하다. 즉, 그물망 위상의 P2P네트워크로 구성된 서버(즉, 피어)들을 활용하여 파일을 분리 분산시킴으로써 가능하다. 또한 특정 서버에 집중화된 파일들을 다른 서버로 분산시킴으로써 저장장치의 균형을 유지할 수 있다.

셋째, 하나의 파일이 하나의 서버에 존재하면 공격자가 이를 공격 목표로 하기가 용이하다. 제안 시스템은 하나의 파일을 196개의 블록으로 나누어 여러 피어에게 분산시켜 저장하기 때문에 공격목표를 다중화 시켜 이를 완화시키는 효과를 보인다. 분산된 블록은 그 블록의 해쉬값으로 암호화되어 있기 때문에 이를 저장한 피어도 이의 내용을 알 수 없다. 단지 파일의 기밀키워드만을 소지한 피어만이 분산된 블록들을 질의하여 원형파일을 복원하는 기능을 한다.

그리고 196개의 블록으로 파일을 인코딩하는 문제에 대한 검토 및 고찰 내용이다. 앞에서 살펴 본 바와 같이 제안 시스템의 응용분야에 따라 파일 크기가 매우 다양하다. 하지만, 제안 시스템은 모든 파일을 196개로 분리하는 정적인 인코딩 방안을 사용하고 있다. 이는 극히 크거나 작거나 하는 파일에 적응적이지 못하다. 따라서 다양한 파일크기에 일반적으로 적응될 수 있는 파일 분리 인코딩 방안의 일반화가 필요하다.

마지막으로 제안시스템은 파일이 블록단위로 분리되어 물리적인 네트워크를 통해 분산됨에도 불구하고 논리적으로는 하나의 파일로 처리되는 기본방안을 제시한 것이다. 따라서 이의 응용분야에 따라 제안한 인코딩 및 라우팅 방법이 다소 변경될 필요가 있다. 따라서 특정 응용에 따라 보다 구체적인 인코딩 및 라우팅 방안이 연구개발 되어야 한다.

VI. 결 론

제안 시스템은 하나의 파일을 블록으로 분리하여 그물망 구조로 된 P2P네트워크에 물리적으로 분산되지만, 논리적으로 하나의 파일처럼 처리할 수 있는 방안을 제시한 것이다.

이들에 대한 안전성 확보를 위해 물리적으로 분산된 모든 블록들이 암호화 되어 저장되도록 하였고, 블록들이 통신망을 통함에 따라 그 과정에서 위조 및 변조를 탐지하기 위해 해쉬 기술을 사용하였다.

본 논문에서 새롭게 제안된 방안은 "하나의 파일이 하나의 장소에 의존적이다."는 기존의 파일 의 위치 제약성을 극복한 것이다. 따라서 소형 무선 단말에서 저장장치 없는 대형파일 처리가 가능한 것과 같은 다양한 응용분야에 적용이 가능하다.

향후 과제로는 어떤 파일 크기에도 일반적으로 적용될 수 있는 표준화된 파일 분리 인코딩 방안을 연구개발을 들 수 있다. 이는 파일 분리 분산을 통한 파일 보안 강화 및 기밀키 분리 분산을 통한 안전한 키 관리기술에서 중요한 핵심기술이기 때문이다.

참고문헌

[1] B. Wilcox-O'Hearn, Experiences Deploying a Large-Scale Emergent Network IPTPS02 LNCS 2429

[2] CW I, Amsterdam. RIPE Integrity Primitives – Final Report of RACE Integrity Primitives Evaluation (R1040), June 1992.

[3] Dennis Kugler, "An Analysis of GNUnet and the

Implications for Anonymous, Censorship-Resistant Networks", 2003

[4] Frank Stajano, "Security for Ubiquitous Computing", Wiley, 2002

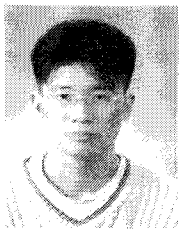
[5] Hans Dobbertin, Antoon Bosselaers, Bart Preneel "RIPEMD-160: A Strengthened Version of RIPEMD", 1996

[6] I. Clarke, O. Sandberg, B.Wiley, and T. Hong. "Freenet: A Distributed Anonymous Information Storage and Retrieval System", 2001

[7] Jianning Yang, "APTPFS: Anonymous Peer-to-Peer File Sharing", April, 2005.

[8] 팀 오라이리 외 24인, Peer-to-Peer 차세대 인터넷 P2P, 한빛미디어 출판사, 2001

저자소개



이명훈(Myoung-Hoon Lee)

2001년 배재대학교 컴퓨터공학과 학사
2003년 배재대학교 컴퓨터공학과 석사
2005년 ~ 현재 배재대학교 컴퓨터공학과 박사과정

※관심분야 : 네트워크 보안, 차세대 네트워크, 모바일 IPv6

박정수(Myoung-Hoon Lee)

2004년 2월 배재대학교 정보통신공학부 공학사
2004년 3월~현재 배재대학교 컴퓨터공학과 석사과정

※관심분야 : 정보보호, 컴퓨터

김진홍(Jin-Hong Kim)

2005년 배재대학교 컴퓨터공학과 학사
2005년 ~ 현재 배재대학교 컴퓨터공학과 석사과정

※관심분야 : 정보보호, P2P 시스템, 컴퓨터 네트워크

조인준(In-June Jo)

1982년 전남대학교 계산통계학과 공학사
1985년 전남대학교 전자계산학과 공학석사
1999년 아주대학교 컴퓨터공학과 공학박사

1983년~1994년 한국전자통신연구원 선임연구원
1994년~현재 배재대학교 컴퓨터공학과 교수
※관심분야 : 정보보호, 컴퓨터네트워크, 전산조직응용