Crystal : 클러스터 기반의 암호화 파일 시스템

황보준형[0] 서대화
경북대학교 전자공학과
bluesky@palgong.knu.ac.kr , dwseo@ee.knu.ac.kr

Crystal : Cryptographic File System Based On Clustering Environment

Jun-Hyung Hwangbo[0] Dae-Wha Seo

Dept. of Electronic Engineering, Kyungpook National University

요 약

하드웨어의 발달과 인터넷의 보편화로 점차 정보의 보안의 필요성이 대두되었다. 암호화 파일 시스템은 사용자의 기밀성을 요구하는 파일의 안전한 저장을 위해 제안되었다. 이 암호화 파일 시스템은 사용자에게 투명성을 제공하여 사용의 편리성을 제공한다. 또한 기존의 암호화 시스템이 사용자 영역에서 이루어져 문맥교환의 횟수가 많아 시스템의 성능이 떨어지는데 반해 암호화 파일 시스템은 커널레벨에서 암호화 서비스가 이루어지므로 시스템의 성능이 저하되는 것을 방지해준다. 하지만 암호화 서비스 자체가 큰 과부하가 되어 일반 파일 시스템에 비해 성능이 많이 떨어진다는 단점이 있다. 따라서 본 논문에서는 클러스터 기반의 파일 시스템을 통해 암호화 파일 시스템의 부하를 분산시켜 성능을 개선함과 동시에 암호화된 파일을 분산 저장하므로 보안성을 높여준다. 제안된 암호화 파일 시스템은 시스템이 확장되었을 경우 그와 비례해서 시스템의 성능이 개선됨을 알 수 있다.

1. 서 론

최근 통신기술과 하드웨어의 발달로 사용자간의 데이터 공유가 일반화 되어 사용자의 편의성을 제공하고 있다. 하지만 이런 발달에 따라 외부에서의 침입이 용이해져 개개인 기밀정보의 보안을 필요로 하게 되었다[3].

이를 위해 여러 가지 연구가 이루어 졌는데 그 대표적인 것으로 사용자에 대한 로긴 서비스와 기밀파일에 대한 암호화 기법이다. 하지만 사용자에 대한 로긴 서비스는 디스크의 도난이나 로긴 서비스를 거치지 않은 디스크에 대한 직접적인 액세스에는 정보의 보안을 유지 할 수 없다. 그리고 파일에 대한 암호화는 시스템의 보안보다는 우수한 보안성을 가지지만 사용자가 각각의 암호화 파일에 대한 키를 관리하여야 하며, 또한 각 파일의 암호화에 대해 사용자가 개입 하여야 하는 과부하가 있다.

따라서 사용자의 데이터를 암호화 하면서도 사용자에게 투명성을 제공하는 시스템 레벨의 암호화 파일 시스템의 연구가 필요하게 되었다. 이 암호화 파일 시스템은 커널 레벨에서 암호화 서비스가 이루어져 사용자 수준의 암호화 시스템에 비해 문맥교환의 횟수가 적어 뛰어난 성능을 기대 할 수 있으며, 커널 영역은 사용자 영역에 비해 접근이 어렵다는 점에서 보안성이 우수하다. 하지만 이 암호화 파일 시스템도 부가적인 암호화 서비스 루틴이 시스템에 들어가게 되므로 일반 파일 시스템에 비해서는 성능이 떨어진다. 또한 사용자로부터 과도한 서비스 요청이 온다면 시스템의 부하는 더욱 커지게 되어 성능이 떨어진다. 그래서 본 논문에서는 사용자의 기밀파일을 암호화 하여 보안하는 동시에 암호화 파일 시스템의 부하를 줄이고자 Crystal 분산 암호화 파일 시스템을 제안한다.

Crystal 분산 암호화 파일 시스템은 클러스터 환경의 파일 시스템을 통해 사용자의 파일 요청을 분산하여 암호화 시켜 기존 암호화 파일 시스템보다 향상된 성능을 기대할 수 있다. 또한 큰 파일에 대해서도 분산 파일 시스템의 스트라이핑 개념을 이용하여 뛰어난 성능을 가지는 암호화 서비스를 제공할 수 있으며 파일을 분산 저장하기 때문에 데이터 I/O에 의한 디스크 병목현상도 해결할 수 있다.

본 논문은 서론에 이어 2장에서는 기존 암호화 파일시스템에 대해 알아보고 3장에서는 본 논문이 제안하는 분산 암호화 파일시스템에 대한 디자인에 대해 설명하고 4장에서는 제안된 시스템을 기존 시스템의 성능과 비교 및 분석 하였다. 마지막으로 5장에서는 실험 결과를 바탕으로 결론을 맺는다.

2. 관련연구

데이터의 암호화는 사용자의 기밀 파일을 안전하게 저장하는 수단으로 사용자 레벨의 암호화 방식과 시스템 레벨에서의 암호화 파일 시스템이 있다. 사용자 수준의 암호화 방식은 문맥교환 같은 과부하가 있어 성능의 저하가 있을 뿐 아니라 사용자가 각 파일에 대한 키를 관리해야 하는 문제가 있다. 이에 대한 대안으로 시스템 수준에서 암호화 하는 암호화 파일 시스템이 제시 되었다. 이는 사용자에게 투명성을 제공해줄

뿐 아니라 시스템 레벨에서 암호화가 이루어져 성능의 개선을 얻을 수 있다. 대표적인 암호화 파일 시스템으로는 CFS, TCFS, Cryptfs, StegFS등이 있다.

CFS(Cryptographic File System)는 NFS 기반의 암호화 파일 시스템으로 AT&T Bell 연구소의 Matt Blaze에 의해 개발되었다[1]. 이는 유닉스 기반의 파일 시스템을 이용하여 디렉토리를 암호화 하는 파일 시스템이다. 하지만 암호화되는 디렉토리마다 키를 사용자가 관리해야 하므로 사용자에게 완벽한 투명성을 제공하지 못한다는 단점이 있다.

TCFS(Transparent Cryptographic File System)는 Matt Blaze의 CFS를 개선한 암호화 파일 시스템으로 커널레벨에서의 암호화 서비스를 제공하며 키를 시스템에서 관리하여 암호화 서비스와 파일 시스템에 대한 더 깊은 통합을 제공한다[2]. 그리고 TCFS는 각각의 파일과 디렉토리에 대해서 암호화가 가능하다. 하지만 모든 파일들은 같은 암호화 알고리즘으로 암호화 되고, 사용자 키는 사용자의 로그인 패스워드로 이루어져 보안성이 떨어진다는 단점이 있다.

Cryptfs는 Erez Zadok에 의해 Stackable Vnode Layer loadable kernel module로써 설계 및 구현된 파일 시스템이다[3]. 그리고 제공되는 암호화 서비스가 커널레벨에서 이루어져 모든 응용에 대해 일관된 암호화를 제공해주며, 문맥교환의 횟수를 줄여 사용자 수준의 암호화 시스템에 비해 뛰어난 성능 향상을 가진다. 이용되는 키는 사용자 ID와 세션 ID를 통하하여 생성되므로 뛰어난 보안성을 제공한다. 하지만 단일 사용자를 위한 암호화 파일 시스템이라서 암호화 파일에 대한 공유가 어렵다는 단점이 있다.

StegFS는 데이터를 암호화 하는 것만 아니라 steganography(정보은닉)의 특성까지 추가하여 보안성을 강화시킨 암호화 파일 시스템이다[4]. 하지만 데이터의 암호화 외에 별도의 정보은닉의 서비스를 제공해야 하므로 전체적인 성능의 저하가 불가피하다.

3. Crystal

3.1 Crystal 개요

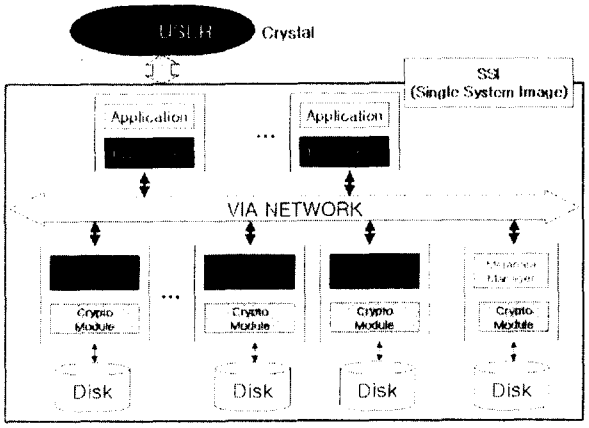

그림 1 Crystal 구조

CRYSTAL은 클러스터 파일 시스템을 기반으로 하는데, 크게 파일서버, 블록서버, 메타데이터서버로 구성되어진다. 그림 1은 CRYSTAL에 대한 구조를 보여준다.

파일서버는 사용자에게 클러스터 암호화 파일 시스템 접근을 위한 인터페이스를 제공하며, 사용자의 암호화 파일 요청을 분산된 블록서버로 분산하여 암호화 서비스를 제공하게 된다. 이 때 파일 서버는 사용자의 인증을 위해 메타데이터서버와 통신을 하여 인증을 받게된다.

블록서버는 파일서버의 논리적인 암호화 파일 요청을 커널레벨의 암호화 모듈을 이용하여 처리하게 된다. 또한 파일의 암호화를 위한 키생성을 위해 메타데이터서버로부터 통신하여 키를 얻게 된다.

메타데이터서버는 전체적인 시스템 구성을 관리하며, 파일 시스템에서 이용되어지는 파일의 분산 정보를 관리한다. 또한 사용자에 대한 적절한 인증정보를 관리하며, 키 생성을 위한 키에 대한 정보를 넘겨주게 된다.

CRYSTAL은 큰 파일일 경우에도 파일을 분산 저장함으로써 얻을 수 있는 이점과 비교적 큰 부하를 가지는 암호화 서비스를 분산하여 실행하므로 전체적인 시스템 성능향상은 뛰어나다고 할 수 있다. 그리고 파일을 분산 암호화 하였을 경우 각 블록서버당 키가 다르므로 파일의 일부분의 정보를 알아냈다 하여도 다른 부분의 정보를 알 수 가 없으므로 기존 암호화 파일 시스템에 비해 높은 보안성을 제공한다.

3.2 서비스 절차

그림 2는 암호화 파일 시스템의 암호화 서비스 절차를 나타낸다.

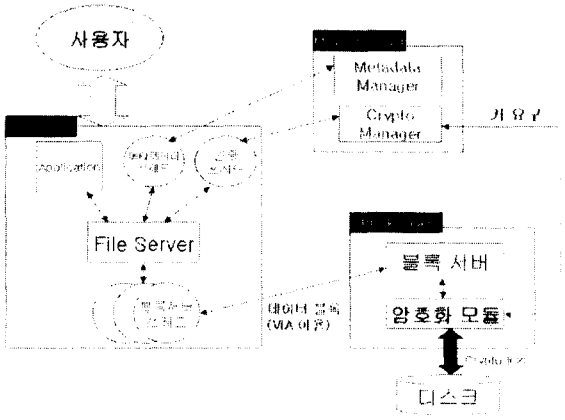

그림 2 Crystal 서비스 절차

먼저 사용자의 파일 요청이 들어오면 파일 서버는 인증 쓰레드를 이용하여 메타데이터 서버로부터 사용자 인증을 받는다. 이렇게 인증작업이 확인되면 파일 서버는 메타데이터 쓰레드를 이용하여 요청 작업의 종류에 따라 메타데이터 서버로부터 시스템 구성의 정보와 요청된 파일의 저장정보를 주고받는다. 이렇게 모든 초기화 작업이 끝나게 되면 블록서버 쓰레드를 이용하여 블록서버와 요청된 파일을 주고받게 된다.

블록서버는 이렇게 파일 서버로부터 받은 파일요청을 암호

화 모듈을 통해 디스크에 저장 또는 읽어오게 된다. 이때 암호화에 이용되는 키는 메타데이터 서버로부터 받은 세션키와 사용자 ID를 통해 생성된다.

3.3 키매니저먼트

키에 대한 전체적인 관리는 메타데이터서버에서 이루어진다. 먼저 인증된 사용자가 서비스를 요청하면 메타데이터서버는 각 블록서버에게 파일의 암호화 서비스를 위한 세션키를 제공하게 된다. 블록서버는 이렇게 받은 세션키와 사용자 ID를 이용하여 키를 생성하게 된다. 이렇게 한번 생성된 키는 세션이 유지되는 동안에는 블록서버의 커널메모리에서 관리하게 되므로 메타데이터서버와는 더 이상의 통신을 필요로 하지는 않는다. 커널메모리는 사용자 메모리에 비해 액세스가 어렵다는 점과 키는 사용자 ID뿐만 아니라 세션ID와의 조합으로 이루어진다는 점에서 시스템은 우수한 보안성을 제공하게 된다.

이렇게 인증된 사용자에게 분배되는 세션키의 보안을 위해서 메타데이터서버에서 암호화 하여 저장되게 된다. 다음에 인증된 사용자 키를 필요할 때 메타데이터서버는 복호화 하여 제공하며, 이 세션키는 키를 소유하는 사용자에 의해서만 키가 변경될 수 있다.

3.4 암호화 모듈

CRYSTAL은 암호화 모듈을 이용하여 암호화 서비스를 제공하게 된다. 이 모듈은 stackable한 Vnode인터페이스를 제공한다. 즉 사용자로부터 요청된 파일 서비스는 이 stackable한 암호화 파일 시스템을 이용하여 디스크에 암호화 되어 저장되거나 복호화 되어 읽혀진다. 여기에 이용된 암호 알고리즘은 128bit의 키를 가지는 blowfish를 이용하였다. 이 blowfish는 키가 자주 변하지 않는 용량이 큰 데이터 암호화에 유리하다.

3.5 노드들간의 통신

CRYSTAL에서 노드들간의 통신은 CLAN환경의 VIA(Virtual Interface Architecture)를 이용한다. VIA는 커널영역으로의 복사가 일어나지 않아 기존의 TCP/IP에 비해 뛰어난 성능향상을 가져온다.

그리고 CRYSTAL은 노드들간의 통신에서 암호화 프로토콜을 이용하지 않는다. CRYSTAL은 클러스터 환경의 암호화 파일 시스템으로 외부 사용자 또는 침입자에게는 단일 이미지(SSI, Single System Image)를 제공하게 된다. 따라서 클러스터 내부에서의 통신은 사용자나 침입자에게는 보이지 않으므로 별도의 암호화 프로토콜이 필요하지 않게 된다.

3.6 파일의 분산정책

CRYSTAL은 클러스터 환경의 파일시스템을 이용하여 과도한 암호화 서비스요청을 분산하여 저장한다. 이때 파일의 크기가 작을 경우에는 각 블록서버의 노드 부하를 계산하여 가장 유휴한 노드에서 저장하게 된다. 하지만 큰 파일의 경우에는 병렬 파일 시스템의 스트라이핑 기법을 이용하여 분산 저장하게 된다.

따라서 작은 파일의 많은 암호화 요청이 들어와도 적절히 부하를 분산시켜 암호화 서비스를 제공하므로 성능 개선을 얻을 수 있으며, 큰 파일의 경우에도 파일을 스트라이핑 기법을 이용하여 분산 하므로 성능개선을 얻을 수 있다.

If (file_size < a)

로드밸런싱 기반의 가장 유휴한 노드에 암호화 저장

else

스트라이핑 기반하에 분산 암호화 저장

4. 결론

점차 인터넷과 네트워크의 발달로 외부 공격에 노출된 사용자의 기밀정보를 보호하기위해 암호화 파일 시스템은 사용자에게 투명성을 제공하며 안전한 파일의 암호화 서비스를 제공해준다. 이는 기존의 암호화 시스템이 사용자 레벨에서 이루어져 문맥교환으로 인한 성능 저하되는 문제를 커널레벨에서 암호화 서비스를 제공하여 성능개선을 보여준다. 하지만 암호화 서비스 자체의 부하는 여전히 문제가 된다. 본 논문에서는 이 암호화 서비스의 부하를 분산시켜 암호화 저장하여 암호화 파일 시스템의 성능을 향상시키는 분산 암호화 파일 시스템을 제안한다.

이 분산 암호화 파일 시스템은 사용자의 과도한 요청이 들어와도 클러스터 기반의 파일 시스템을 이용하여 적절히 부하를 분산하여 암호화 저장하므로 뛰어난 성능개선을 보여준다. 또한 큰 파일의 암호화도 스트라이핑 기법을 이용하여 분산 저장하므로 병렬 I/O를 통한 디스크 I/O의 문제를 해결함과 동시에 암호화 서비스의 부하를 분산하여 계산하므로 효과적인 성능개선을 보여줌을 알 수 있다.

5. 참고문헌

[1] M. Blaze, "A Cryptographic File System for Unix," *Proceedings of the first ACM Conference on Computer and Communications Security (Fairfax, VA)*, ACM, November, 1993.

[2] Transparent Cryptographic File System http://tcfs.dia.unisa.it/

[3] E. Zadok *et al*, "Cryptfs: A Stackable Vnode Level Encryption File System," *Technical Report CUCS-021-98*, Computer Science Department, Columbia University, 28 July 1998. Available at http://www.cs.columbia.edu/~library/

[4] D. Andrew *et al*, "StegFS: A Steganographic File System for Linux," *IH'99, LNCS 1768*, pp. 463-477, 2000.