클라우드 컴퓨팅의 분산저장서버를 고려한 XOR기반의 고성능 비밀분산 기법

김수현*, 홍인식**, 이임영*
순천향대학교 컴퓨터소프트웨어공학과
e-mail:[kimsh, ishong, imylee]@sch.ac.kr

High-Performance Secret Sharing Scheme based on XOR for Distributed Storage Server in Cloud Computing

Su-Hyun Kim, In-Sik Hong, Im-Yeong Lee
Department of Computer Software Engineering Soonchunhyang University*
Department of Computer Engineering Soonchunhyang University**

요 약

클라우드 컴퓨팅 환경에서는 사용자의 데이터를 수많은 분산서버를 이용하여 데이터를 암호화하여 저장한다. 구글, 야후 등 글로벌 인터넷 서비스 업체들은 인터넷 서비스 플랫폼의 중요성을 인식하고 자체 연구 개발을 수행, 저가 상용 노드를 기반으로 한 대규모 클러스터 기반의 클라우드 컴퓨팅 플랫폼 기술을 개발 활용하고 있다. 이와 같이 분산 컴퓨팅 환경에서 다양한 데이터 서비스가 가능해지면서 대용량 데이터의 분산관리가 주요 이슈로 떠오르고 있다. 한편, 대용량 데이터의 다양한 이용 형태로부터 악의적인 공격자나 내부 사용자에 의한 보안 취약성 및 프라이버시 침해가 발생할 수 있다. 이러한 문제점을 해결하기 위해 본 논문에서는 XOR기반의 효율적인 분산 저장 및 복구 기법을 제안하였다.

1. 서론

최근 국내외로 클라우드 컴퓨팅에 관한 관심이 높아지며 많은 연구가 진행되고 있다. 많은 기업들이 IT기술의 성장을 발판으로 다양한 분야로 확장 가능하고, 컴퓨팅 파워의 효율적인 사용이 가능한 클라우드 컴퓨팅에 관심을 가지고 있다. 구글, 야후 등 글로벌 인터넷 서비스 업체들은 인터넷 서비스 플랫폼의 중요성을 인식하고 자체 연구 개발을 수행, 저가 상용 노드를 기반으로 한 대규모 클러스터 기반의 분산컴퓨팅 플랫폼 기술을 개발 활용하고 있다. 대용량 데이터 처리 및 저장 관리가 필요한 대표적인 어플리케이션으로는 인터넷 서비스 분야 외에 예를 들면, 비즈니스 인텔리전스 등 다른 응용 영역으로 확대하여 클라우드 서비스로 제공하려는 비즈니스 모델이 제시되고 있다. 이와 같이 분산 컴퓨팅 환경에서 다양한 데이터 서비스가 가능해지면서 대용량 데이터의 분산관리가 주요 이슈로 떠오르고 있다. 한편, 대용량 데이터의 다양한 이용 형태로부터 악의적인 공격자나 내부 사용자에 의한 보안 취약성 및 프라이버시 침해가 발생할 수 있다. 이러한 문제점을 해결하기 위해 본 논문에서는 XOR기반의 효율적인 분산 저장 및 복구 기법을 제안하였다.

본 논문의 구성은 다음과 같다. 2장에서는 본 논문에서 제안하는 기법의 이해를 돕기 위한 관련 기술들을 소개하고, 3장에서는 클라우드 컴퓨팅 환경이 갖추어야 할 기본적인 보안 요구사항에 대하여 알아보며, 4장에서는 제안 방식에 대하여 설명한다. 5장에서는 제안 방식의 안전성을 분석한고, 마지막으로 6장에서는 결론 및 향후 연구 방향으로 마치도록 한다.

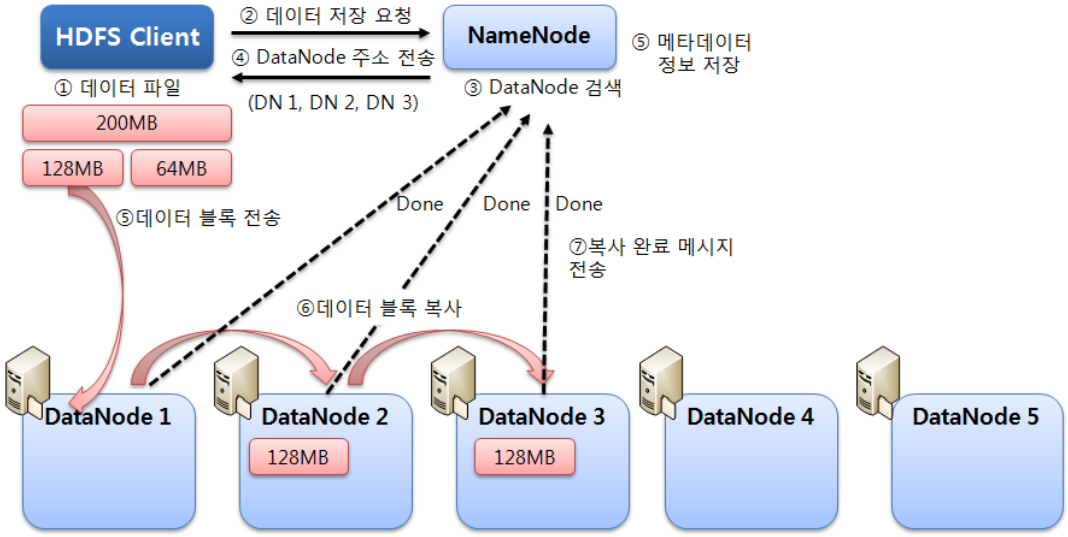
2. 관련연구

본 장에서는 본 논문에서 제안하는 기법의 이해를 돕기 위한 관련 기술과 기존 제안방식들을 소개한다.

2.1 HDFS

HDFS(Apache Hadoop Distributed File System)는 기성 하드웨어에서 실행 가능하도록 제작된 파일 시스템으로 기존의 분산 파일 시스템과 많은 유사점을 가지고 있다. 하지만 많은 차이점도 보이는데 높은 장애복구 기능과 저가의 하드웨어에 적용이 가능하도록 설계되었다. HDFS는 현재 Amazon, IBM, Yahoo 등과 같은 글로벌 IT 기업들의 클라우드 컴퓨팅 플랫폼의 기반이 되는 분산 파일 시스템으로 가장 널리 활용이 되고 있다[1].

HDFS의 설계와 구현을 위해 도출된 사항들을 살펴보면



(그림 1) 데이터 분산 저장 과정

GFS와 대부분 동일하며 플랫폼 간의 손쉬운 이식성을 보장하기 위해 자바를 사용하여 구현되었다는 점이 크게 다르다. 높은 이식성을 지닌 자바 언어의 사용은 자바를 지원하는 다양한 서버들에서 HDFS가 구동할 수 있다는 장점을 지닌다.

HDFS클라이언트는 사용자의 데이터를 128MB 블록 단위로 분할하여 NameNode에 저장 요청을 하게 된다. NameNode는 데이터 블록이 저장될 DataNode의 주소를 파악하여 다시 클라이언트에게 보내주게 된다. 3개의 DataNode 주소를 받은 클라이언트는 첫 번째 DataNode에 직접데이터를 전송하게 되고, 첫 번째 DataNode는 데이터 수신과 동시에 복사본을 다음 DataNode에 송신하게 된다. 이렇게 하나의 블록은 데이터의 손실이나, DataNode의 고장 및 오류에 대비하기 우해 총 3개의 DataNode에 백업되어 저장된다(그림 1).

2.2 GFS

구글에서 사용하는 GFS(Google File System)은 대용량의 데이터에 적합하도록 개발되었으며 핵심 데이터 저장 및 검색 엔진에 최적화 되어 있다[2].

GFS는 대용량의 데이터를 저장하기 위해 수많은 저비용 스토리지 서버를 사용하여 분산 저장하게 된다. GFS의 구조는 마스터 서버와 청크 서버로 구성이 되며, 데이터는 다수의 청크 서버에 64메가바이트 단위로 나눠진다. 마스터 서버에는 각 청크 서버의 생성 시점에 고유의 64비트 레이블을 할당하고, 논리적 매핑을 이용해 청크 서버와 연결을 유지한다. 하지만 다수의 청크 서버를 사용함으로써 빈번하게 발생할 수 있는 장애를 대처할 수 있는 방안을 필요로 하는 단점이 존재한다.

2.3 AONT

OAEP(OAEP(Optimal Asymmetric Encryption Padding)

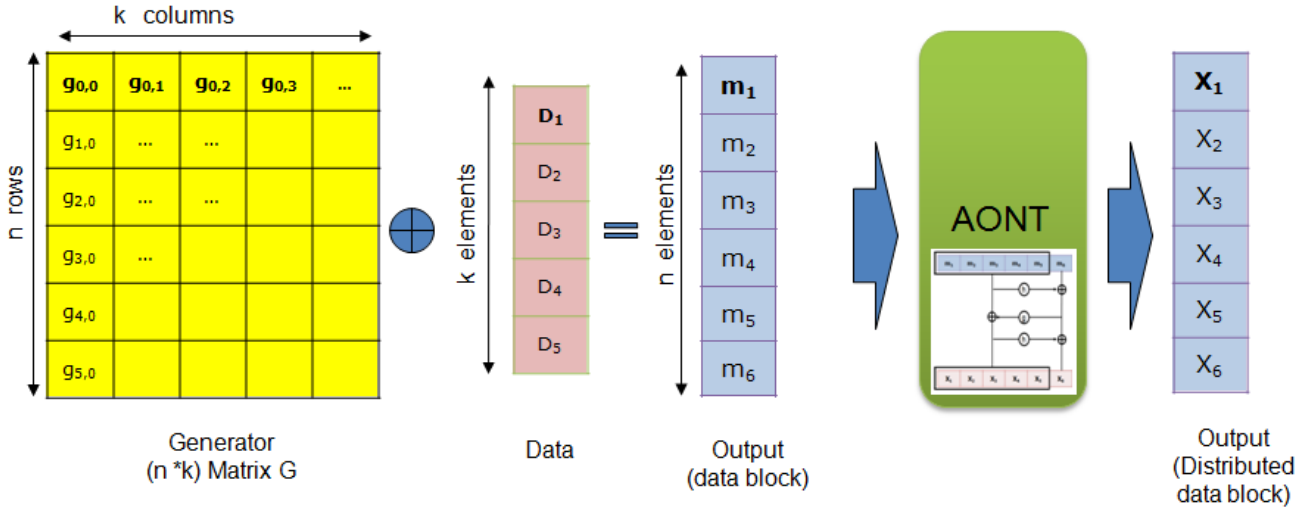
(그림 2) AONT 방식의 비밀분산 개념

에 근거한 AONT(All Or Nothing Transform) 방식으로서 독자적으로 고속처리를 가능하게 하는 구현방식이고, 특히 대용량 데이터의 처리에 적합하다. 비밀분산 라이브러리도 본 방식을 구현한 것으로서 특징은 분산데이터의 개수 및 크기를 유연하게 설정 가능하며 모든 분산 데이터의 크기의 합계가 원 데이터의 크기와 거의 동일하게 된다[3](그림 2).

AONT(All-or-Nothing Transform)은 RSA암호방식의 발명자인 Rivest에 의해 제안된 개념으로서 원래는 RSA 암호의 강도를 높이기 위해 OAEP라는 Preprocessor에 이용된 방식이다. AONT는 원래의 데이터에 대해서 연산을 수행, 원래의 데이터와 거의 동일한 크기의 출력데이터를 얻을 수 있다. 출력데이터의 모든 비트가 모아지면 용이하게 원 데이터를 복원할 수 있는데, 어느 정도 수 이상의 비트를 곱하면 원래의 데이터의 복원이 불가능하게 되는 특징이 있다. 이 특성으로부터 출력 데이터를 복수의 데이터로 분할하는 것으로 분산 데이터조각이 모두 모아지지 않으면 원래의 데이터를 복원할 수 없는 성질을 갖는 비밀분산 기법의 일종이다.

3. 보안요구사항

클라우드 컴퓨팅의 핵심 메커니즘 중 하나는 바로 대용량 데이터의 효율적인 관리이다. 대용량 데이터의 다양한 이용 형태로부터 악의적인 공격자나 내부 사용자에 의한 보안 취약



(그림 3) 데이터 분산 저장 과정

성 및 프라이버시 침해가 발생할 수 있기 때문에, 다음과 같은 보안 요구사항을 필요로 하게 된다[4].

· 기밀성 : 데이터 저장 서버와 클라이언트 단말기 간의 통신 데이터는 정당한 개체만이 확인 가능해야 한다.
· 인증 : 데이터 저장 서버가 정당한 개체인지 검증 가능해야 하고, 정당한 사용자만이 데이터에 대한 접근을 가능하게 해야 한다.
· 가용성 : 대용량의 데이터를 전송 시 가용성을 보장하기 위해 인증 및 기밀성이 빠른 속도로 이루어 져야 한다.
· 연산효율성 : 수시로 이루어지는 데이터 전송 시 클라이언트 단말기 및 클라우드 서버의 오버헤드를 줄이기 위해 최소한의 연산만을 보장해야 한다.

3. 제안방식

3.1 시스템 모델 및 가정

전체적인 클라우드 컴퓨팅 환경은 Apach의 HDFS를 바탕으로 설계되었다. 클라우드 컴퓨팅 환경에서는 대용량 데이터 저장을 위해 128MB 블록 단위로 나누어 분산 저장하게 된다. 각각 분산 저장서버에 저장된 데이터는 암호화 되지 않은 평문상태로 저장되어 있어 공격자에 의해 분산 저장 서버가 탈취되는 경우 데이터 일부가 그대로 노출되는 문제점이 발생한다. 이를 방지하기 위해 암호화를 사용자에게 권고하지만, 시스템 자체적으로는 제공되지 않고 있다. 따라서 본 논문에서는 분할된 데이터의 모든 블록이 모아져야만 원본 데이터가 복구 가능한 XOR기반의 데이터 복구 기법을 제안한다.

3.2 시스템 계수

· $h: 0,1^{l(s-1)} \rightarrow 0,1^{l}$
· $g: 0,1^{l} \rightarrow 0,1^{l(s-1)}$
· s : 평문 블록수
· $m_1, m_2, ..., m_s (m_i \in 0,1^{l})$: 평문 블록
· $x_1, x_2, ..., x_s$: 의사평문블록

3.3 행렬기반 XOR 분산 저장 과정

사용자가 저장하고자 하는 데이터는 HDFS 클라이언트를 통해 k개의 조각으로 나누어지게 된다. HDFS 클라이언트에서는 사용자의 데이터가 나누어진 조각의 개수 k를 확인하여 n×k행렬의 매트릭스 G를 생성한다. 매트릭스 G와 k 조각의 데이터는 각각 행렬간의 XOR연산이 이루어진다. XOR 행렬 연산 결과로 n조각의 데이터 블록이 생성된다(그림 3).

$$m_n = \bigoplus_{t=2}^{k} (g_{n-1,t-2} \oplus D_{t-1})$$

3.4 AONT 분할 과정

n개의 조각으로 나누어진 데이터 블록은 분산 저장 시 데이터의 안전성을 증가시키기 위하여 AONT 분할 과정을 거친다. AONT변환의 안전성은 h(hash function), g(pseudo-random number generator)함수의 안전성을 전제로 한다.

Step 1. AONT 변환을 위해 평문 데이터 m을 $m_1, m_2, ..., m_s (m_i \in 0,1^{l})$로 분할한다. 그리고 h함수를 이용하여 μ_s를 계산한다.
- $\mu_s = h(m_1 || m_2 || ... || m_{s-1})$

Step 2. 계산된 μ_s와 m_s 그리고 g함수를 이용하여 다음과 같이 계산한다.

$x_1 \| x_2 \| ... \| x_{s-1} = (m_1 \| m_2 \| ... \| m_{s-1}) \oplus g(\mu_s \oplus m_s)$

Step 3. 다음으로 x_s를 아래와 같이 계산하여 의사평문블록을 생성한다.

$x_s = (\mu_s \oplus m_s) \oplus h(x_1 \| x_s \| ... \| x_{s-1})$

3.5 데이터 복원 과정

분산 저장서버에 저장된 모든 데이터 블록을 모은 뒤, AONT 분할 과정의 역순으로 n개의 데이터를 다시 생성한다. 그 후, n개의 데이터 블록 간 XOR연산으로 매트릭스 G를 생성할 수 있게 되고, 매트릭스 G와 XOR 행렬 연산을 통해 원본 데이터 블록을 복구 할 수 있다.

1) $m_1 \oplus m_2 \oplus m_3 \oplus m_4 \oplus m_5 \oplus m_6 = G_n^k$

$D_k = G_n^k \oplus m_i$

$D_i = D_1 \| D_2 \| ... \| D_k$

4. 제안방식 분석

4.1 기밀성

사용자의 데이터는 마스터 서버를 통하여 분산서버에 여러 조각으로 나뉘어 저장된다. 사용자로부터 데이터 요청이 들어왔을 경우 분산되어 있는 데이터를 수집하는 과정이 필요하다. 이 때, 분산 서버는 마스터 서버와 물리적, 논리적으로 분리되어 있기 때문에, 각각의 분산서버에 대한 인증과 조각 데이터 전송 시 통신로 상에서의 기밀성을 제공해야 한다. 기존의 공개키 암호 시스템을 사용할 경우, 비용적 측면에서 매우 비효율적이다.

4.2 가용성 및 연산효율성

본 논문에서 제안한 시스템은 전체 시스템의 과부하를 줄이기 위해 별도의 데이터 암호화 과정 없이 추측 불가능한 데이터 조각을 나누어 정당한 사용자만이 복구 가능하다. 또한 암호화 방식에 비해 처리속도 측면에서 고속으로 분산, 복원이 가능하다.

5. 결론 및 향후 연구 방향

본 논문에서는 대용량 데이터의 다양한 이용 형태로부터 악의적인 공격자나 내부 사용자에 의한 보안 취약성 및 프라이버시 침해를 방지하기 위해 XOR기반의 효율적인 분산 저장 및 복구 기법을 제안하였다. 전체 시스템의 과부하를 줄이기 위해 별도의 데이터 암호화 과정 없이 추측 불가능한 데이터 조각을 나누어 정당한 사용자만이 복구 가능하도록 제안하였다. 이를 바탕으로 제안한 XOR 기반의 비밀 분산・복원 구조는 데이터량 불변 AONT 암호화 모드를 사용하여 안전성을 향상시키고 효율성 측면에서 XOR 연산에 기반하였다. 제안 프로토콜은 기밀성이 높은 데이터, 사용자의 개인정보를 포함하는 다양한 대용량 데이터를 안전하고 효율적으로 분산 관리하는 구조로서 클라우스 컴퓨팅 환경에서 보다 효율적으로 사용될 것으로 기대된다.

참고문헌

[1] Apache Hadoop, 2009, http://hadoop.apache.org/.

[2] Sanjay Ghemawat , Howard Gobioff , Shun-Tak Leung, "The Google file system," ACM SIGOPS Operating Systems Review, v.37 n.5, December 2003

[3] R. L. Rivest, "All-or-nothing encryption and the package transform," Fast Software Encryption FSE '97, Lecture Notes in Computer Science, vol.1267, pp.210－218, 1997.

[4] D. Hubbard and M. Sutton, "Top threats to cloud computing," in Cloud Security Al liance,Mar.2010.