KIPS Tr. Comp. and Comm. Sys.
Vol.5, No.2 pp.33~40 pISSN: 2287-5891

http://dx.doi.org/10.3745/KTCCS.2016.5.2.33

Design and Implementation of HDFS Data Encryption Scheme Using ARIA Algorithms on Hadoop

Youngho Song[†] · YoungSung Shin[††] · Jae-Woo Chang[†††]

ABSTRACT

Due to the growth of social network systems (SNS), big data are realized and Hadoop was developed as a distributed platform for analyzing big data. Enterprises analyze data containing users' sensitive information by using Hadoop and utilize them for marketing. Therefore, researches on data encryption have been done to protect the leakage of sensitive data stored in Hadoop. However, the existing researches support only the AES encryption algorithm, the international standard of data encryption. Meanwhile, Korean government choose ARIA algorithm as a standard data encryption one. In this paper, we propose a HDFS data encryption scheme using ARIA algorithms on Hadoop. First, the proposed scheme provide a HDFS block splitting component which performs ARIA encryption and decryption under the distributed computing environment of Hadoop. Second, the proposed scheme also provide a variable-length data processing component which performs encryption and decryption by adding dummy data, in case when the last block of data does not contains 128 bit data. Finally, we show from performance analysis that our proposed scheme can be effectively used for both text string processing applications and science data analysis applications.

Keywords : Hadoop Security, Data Encryption, HDFS Data Encryption, ARIA Encryption Algorithm, Hadoop Encryption Codec

하둡 상에서 ARIA 알고리즘을 이용한 HDFS 데이터 암호화 기법의 설계 및 구현

송 영 호[†] · 신 영 성[††] · 장 재 우[†††]

요 약

최근 소셜 네트워크 서비스(SNS)의 발전으로 빅데이터가 출현하였고, 이를 분석하기 위한 분산 병렬 플랫폼으로 하둡이 개발되었다. 하둡을 사용하는 기업은 개인적인 정보가 포함된 데이터를 분석하여 마케팅 등에 활용하고 있다. 이에 따라, 하둡에 저장된 센서티브(sensitive) 데이터의 유출을 방지하기 위한 데이터 암호화 연구가 수행되었다. 하지만 기존 데이터 암호화에 대한 연구는 국외 표준인 AES 암호화 알고리즘만을 지원하는 한계점이 존재한다. 한편 정부에서는 데이터 암호화 알고리즘으로 ARIA 알고리즘을 국내 표준으로 지정하였다. 본 논문에서는 하둡 상에서 ARIA 알고리즘을 이용한 HDFS 데이터 암호화 기법을 제안하였다. 첫째, 제안하는 암호화 기법은 하둡의 분산 컴퓨팅 환경에서 ARIA 암호화 및 복호화를 수행하는 HDFS 블록 분할 컴포넌트를 제공한다. 둘째, 제안하는 암호화 기법은 데이터의 마지막 블록이 128비트 단위의 데이터가 아닐 경우, 더미(dummy) 데이터를 추가하여 암호화 및 복호화를 수행하는 가변길이 데이터 처리 컴포넌트를 제공한다. 마지막으로 성능 평가를 통해, 제안하는 ARIA 기반 암호화 기법이 텍스트 문자열 처리 응용 및 과학 데이터 분석 응용에서 효과적으로 사용될 수 있음을 보였다.

키워드 : 하둡 보안, 데이터 보호, HDFS 데이터 암호화, ARIA 암호화 알고리즘, 하둡 암호화 코덱

1. 서 론

최근 스마트폰 보급 및 서비스 산업 고도화로 인해 소셜 네트워크 서비스(SNS)를 중심으로 사용자가 생성하는 데이터가 급증하고 있다. 해당 데이터는 다양한 형태의 멀티미디어 콘텐츠를 포함하고 있으며, 그 규모는 빅데이터(Bigdata) 수준으로 발전하였다. 한편 분산된 컴퓨팅 환경에서 빅데이터에 대한 작업을 병렬적으로 지원하기 위한 대표적인 작업 모델로 맵리듀스(MapReduce)연구[1]가 존재한다. 해당 모델에서는 사용자 정의가 가능한 맵(Map)과 리듀스(Reduce) 함수를 기반으로 병렬된 환경에서의 데이터 처리를 지원한다. 맵리듀스 기술을 구현한 대표적인 빅데이터 분산 처리 시스템으로 하둡(Hadoop)[2]이 존재한다. 하둡은 아마존, 페이스북 등 글로벌 기업에서 주요 프로젝트로 적극 활용됨으로써, 빅데이터 분산 병렬 처리 기법의 표준으로 인정받고 있다.

※ 이 논문은 2014년 교육부와 한국연구재단의 지역혁신창의인력양성사업의 지원을 받아 수행된 연구임(NRF-2014H1C1A1065816).
※ 이 논문은 2015년도 한국정보처리학회 추계학술발표대회에서 '하둡 상에서 ARIA 알고리즘을 이용한 HDFS 데이터 암호화 기법의 설계 및 구현'의 제목으로 발표된 논문을 확장한 것임.
† 준 회 원 : 전북대학교 컴퓨터공학부 박사과정
†† 비 회 원 : 전북대학교 컴퓨터공학부 박사과정
††† 종신회원 : 전북대학교 IT정보공학과 교수
Manuscript Received : December 22, 2015
First Revision : February 15, 2016
Accepted : February 22, 2016
* Corresponding Author : Jae-Woo Chang(jwchang@jbnu.ac.kr)

현재 하둡을 활용하는 기업에서는 다양한 분야에서 수집된 개인정보, 사용자 위치 데이터 등 민감한 데이터를 분석하여 마케팅, 고객 유치 모델 구축 등에 활용하고 있다. 따라서 보호되지 않은 센서티브(sensitive) 데이터의 유출이 발생할 경우, 회사 운영에 막대한 악영향을 끼칠 가능성이 존재한다. 이러한 문제점을 해결하기 위해, 하둡 보안에 대한 연구[3-6]가 활발히 진행되고 있다. 첫째, 네트워크 공격을 통한 데이터 유출을 방지하기 위한 연구로, 케르베로스(Kerberos)[3, 4]가 존재한다. 하둡은 케르베로스 시스템을 통해, 사용자 인증을 이용한 네트워크 보안을 지원하여 외부의 악의적인 공격자의 접근을 차단한다. 둘째, 하둡 내부의 데이터 보호를 위해 HDFS(Hadoop Distributed File System)내 데이터를 압축하는 코덱(Compression Codec) 기능을 이용하여 데이터 암호화를 수행하는 연구[5, 6]가 존재한다. 먼저 Liu Yi의 연구[5]는 기존 압축 코덱이 입력으로 주어진 데이터를 하나의 맵에서 압축 처리하여, 데이터 분산 암호화를 수행할 수 없는 문제점을 해결하기 위해, Splittable-CompressionCodec을 제안하였다. Splittable-CompressionCodec은 HDFS 블록과 맵을 1대1로 매칭하여 맵 하나가 HDFS 블록 하나를 처리하기 때문에, 여러 블록으로 나누어진 HDFS 데이터에 대한 압축 및 압축 해제 기능을 병렬적으로 지원한다. 아울러 박선영의 연구[6]는 Splittable-CompressionCodec과 자바의 Cipher 라이브러리를 이용하여 HDFS 데이터의 AES 암호화를 지원하였다. 하지만 두 연구 모두 미국에서 표준으로 채택된 AES 데이터 암호화만을 지원하는 한계점이 존재한다.

현재 국내에서는 ARIA 암호화 알고리즘[9, 10]을 표준으로 지정하여 사용을 권장하고 있으며, 공공기관에 납품되는 소프트웨어는 ARIA 알고리즘을 탑재할 것을 의무화 하고 있다. ARIA 암호화 알고리즘은 AES와 마찬가지로 128비트의 블록을 기반하여 데이터 암호화를 수행하기 때문에, AES 암호화 알고리즘을 대체할 수 있는 특징이 존재한다. 그러나 현재 하둡에서는 데이터 암호화를 위한 ARIA 알고리즘을 지원하지 못하는 문제점이 존재한다.

따라서 본 논문에서는 HDFS 코덱 기반 ARIA 암호화 기법을 제안한다. 제안하는 암호화 기법은 첫째, 데이터를 HDFS 블록 데이터로 나누어 암호화 및 복호화를 수행하기 위해, 하둡의 분산 컴퓨팅 환경에서 ARIA 암호화 및 복호화를 수행하는 HDFS 블록 분할 컴포넌트를 제공한다. 둘째, 기존 AES 기반 하둡 암호화 기법은 암호화 블록 단위인 128비트 단위의 데이터만을 암호화/복호화를 수행한다. 그러나 제안하는 암호화 기법은 데이터의 마지막 블록이 128비트 단위의 데이터가 아닐 경우, 더미(dummy) 데이터를 추가하여 128비트가 아닌 데이터를 암호화/복호화를 수행하는 가변길이 처리 컴포넌트를 제공한다. 아울러 본 논문에서는 제안하는 ARIA 기반 하둡 암호화 코덱뿐만 아니라 기존 AES 암호화 코덱을 선택적으로 활용할 수 있도록 코덱 변환 기능을 제공한다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 AES 기반 HDFS 암호화 기법에 대해 소개한다. 3장에서는 제안하는 ARIA 알고리즘을 이용한 HDFS 데이터 암호화 기법을 제시한다. 4장에서는 제안하는 ARIA 기반 암호화 기법의 성능평가를 수행한다. 마지막으로 5장에서는 연구의 결론 및 향후 연구 방향에 대해서 기술한다.

2. 관련 연구

2.1 암호화 알고리즘

AES 암호화 알고리즘[7, 8]은 미국 표준 기술 연구소(NIST)에서 연방 정보처리 표준으로 발표한 대칭키 기반 암호화 알고리즘으로, 현재 국외에서는 프로그램에서 관리하는 데이터에 암호화가 필요할 경우, AES 암호화 알고리즘을 보편적으로 활용하고 있다.

ARIA 알고리즘[9, 10]은 국가 보안 기술 연구소 주도로 학계, 연구소, 정부기관 등의 암호 기술 전문가들이 개발한 국내 기술 암호화 알고리즘이다. 128비트의 데이터 블록을 처리할 수 있으며, AES와 동일한 128/192/256비트의 암호화 키를 사용한다. 대부분의 연산은 XOR과 같은 비트 연산으로 구성되어 있으며, 키 크기에 따라 12/14/16 라운드로 암호화를 수행하여 데이터 보호 복잡도를 시스템 성능 요구에 맞추어 제공할 수 있다. 아울러 ARIA는 AES와 동일한 128비트 블록 비트 기반의 데이터 암호화를 수행하기 때문에 국내에서 AES 암호화 알고리즘을 대체할 수 있는 특징이 존재한다. Table 1은 AES와 ARIA를 비교한 것이다.

Table 1. Comparison of the AES and ARIA

	AES	ARIA
Period	1999	2004
Main Agent	National Institute of Standards and Technology	National Security Research Institute
Target	National service Private service	Administrative service
Characteristic	Standard of domestic and international encryption	AES alternative domestic technology

2.2 AES 암호화 알고리즘 기반 HDFS 데이터 암호화 기법

Liu Yi의 연구[5]는 기존 압축 코덱이 입력으로 주어진 데이터를 하나의 맵에서 압축 처리하여 데이터의 분산 암호화를 수행할 수 없는 문제점을 해결하기 위해, Splittable-CompressionCodec을 제안하였다. Splittable-Compression Codec은 HDFS 블록과 맵 함수를 1대1로 매칭하여 각각의 맵 함수가 HDFS 블록들을 처리하기 때문에, 여러 블록으로 나누어진 HDFS 데이터에 대한 병렬적인 압축 및 압축 해제 기능을 지원한다. 현재 하둡 1.x 버전 및 2.x 모두 Splittable-Compression Codec을 지원하고 있기 때문에 호환성 문제가 없으며, 설정 파일에 클래스 명을 기입하는 것으로 사용 여부를 결정할 수 있다.

박선영의 연구[6]는 자바의 Cipher 라이브러리를 이용하여

HDFS 데이터의 AES 암호화를 지원하였다. Fig. 1에서 Cipher 클래스는 자바에서 제공하는 암호화 관련 라이브러리로써, 설정에 따라 AES, DES, RSA 등 여러 암호화 알고리즘을 상속받을 수 있다. 아울러 이를 Splittable-CompressionCodec에 오버라이딩하여 선택한 암호화 알고리즘에 따라 데이터의 암호화 및 복호화를 수행한다.

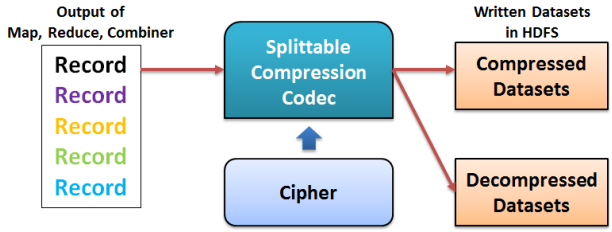


Fig. 1. The way of Encryption using Hadoop Compression function

Liu Yi의 연구[5]와 박선영의 연구[6]는 국외 표준인 AES 암호화만을 지원하고 있다. 따라서 기존 연구는 첫째, 국내 암호화 표준으로 채택된 ARIA 를 지원하지 못하는 문제점이 존재한다. 둘째, AES 암호화의 경우, 기존 연구는 블록 단위인 128비트 단위의 데이터에 대해서만 암호화를 지원하고 그렇지 않을 경우 스트림 에러를 발생시키는 문제점이 존재한다.

3. 제안하는 ARIA 암호화 기법의 설계

3.1 시스템 구조

개인정보보호법이 시행됨에 따라 정부에서는 국내에서 개발한 블록 암호화 알고리즘인 ARIA 암호화 알고리즘을 표준으로 지정하였다. 그러나 대표적인 빅데이터 분석 처리 플랫폼인 하둡을 활용하는 국내 연구기관이나 기업을 위한 ARIA 데이터 암호화 알고리즘 연구는 존재하지 않는 실정이다. 따라서 본 논문에서는 ARIA 암호화 알고리즘을 이용한 HDFS 데이터 암호화 기법을 제안한다. Fig. 2는 하둡 상에서 제안하는 HDFS 데이터 암호화 기법을 수행하기 위한 전체 시스템 구조를 보인다. 그림에서 네임노드(NameNode)의 로컬 파일 시스템에 저장된 데이터에 대한 암호화를 수행할 때의 순서는 다음과 같다. i) 사용자는 제안하는 데이터 암호화 기법을 호출한다. 호출된 암호화 기법은 사용자 설정에 따라 AES 및 ARIA를 선택할 수 있다. ii) 선택된 암호화 기법에 따라 암호화를 수행한다. iii) 암호화된 데이터는 각 데이터 노드의 HDFS 블록 단위(기본 64MB)로 나누어 데이터 노드별로 분산 저장한다. 복호화 과정은 암호화 과정의 역순으로 이루어진다.

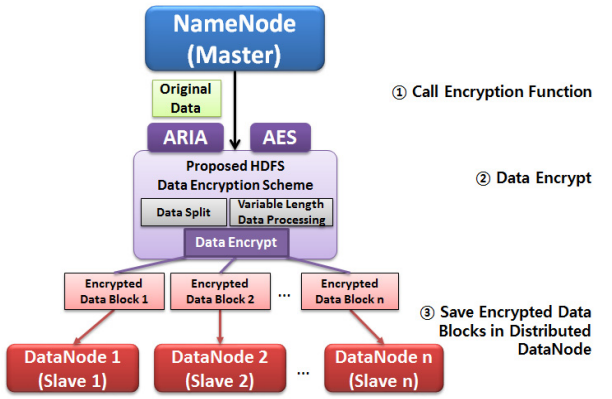


Fig. 2. Structure of entire system for encrypt HDFS data

한편 하둡에서 암호화된 데이터를 처리하는 맵리듀스 수행과정은 Fig. 3과 같다. 그림에서 사용자가 정의한 맵과 리듀스를 수행할 때의 순서는 다음과 같다. i) 맵 함수를 실행하기 전, 맵 함수의 입력 데이터로 들어온 암호화 데이터에 대한 복호화를 수행하고 맵 함수를 실행한다. ii) 맵 함수가 종료되면 출력 결과를 다시 암호화하여, HDFS에 저장한다. iii) 리듀스 함수 실행 전, 리듀스 함수의 입력 데이터인 임시 암호화 데이터를 복호화 하여 리듀스 함수를 실행한다. iv) 리듀스 함수 종료 후, 최종 결과 데이터를 암호화하여 HDFS에 저장한다. 한편, 사용자 설정에 따라 ii) 및 iii) 과정에서 암호화 및 복호화의 생략이 가능하다.

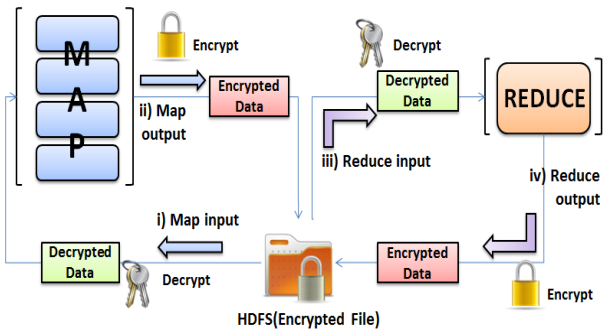


Fig. 3. Processing of Hadoop Mapreduce on encrypted HDFS

3.2 제안하는 기법의 암호화 과정

제안하는 HDFS 암호화 기법에서 암호화 과정은 Fig. 4와 같다. i) 사용자는 제안하는 ARIA 코덱과 기존 AES 코덱 중 하나를 선택하여 암호화 호출 컴포넌트를 호출한다. 암호화 호출 컴포넌트는 암호화를 위해 HDFS 블록 분할 컴포넌트를 호출한다. ii) HDFS 블록 분할 컴포넌트는 HDFS 블록 단위(64MB)로 나눈 후 가변길이에 대한 암호화를 수행하기 위해 가변길이 처리 컴포넌트를 호출한다. iii) 가변길이 처리 컴포넌트는 데이터를 암호화 블록 단위로 나눈 후 가장 마지막 블록이 암호화 블록 단위인 128비트 단위인지 확인한다. 128비트 단위가 아닌 데이터일 경우 더미 데이터를 추가한 후 각각의 블록별로 암호화 컴포넌트를 호출한다. iv) 암호화 컴포넌트는 각각의 블록에 대해 호출된 코덱(ARIA, AES)을 통한 암호화를 수행한다. 암호화된 블록들은 가변길이 처리 컴포넌트에서 하나로 병합한 후 HDFS 블록 분할 컴포넌트를 통해 저장된다.

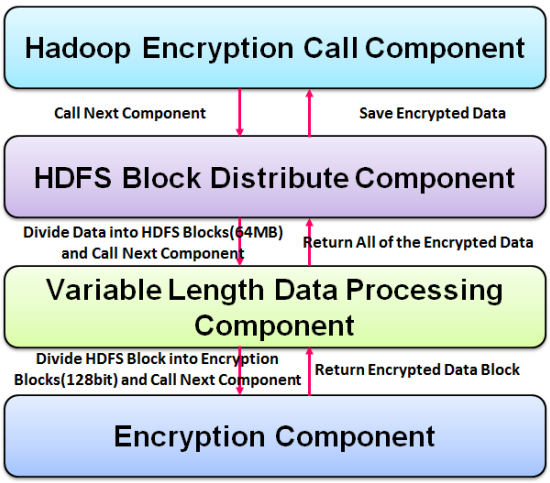


Fig. 4. Encryption process

1) 하둡 암호화 호출 컴포넌트(Hadoop Encrpytion Call Component)

사용자는 하둡 암호화 호출 컴포넌트를 통하여 실제 데이터의 암호화를 위한 알고리즘 선택 및 결과 데이터를 반환받는다. Fig. 5는 하둡 암호화 호출 컴포넌트의 세부 구조를 나타낸다. 그림에서 사용자는 알고리즘 선택 모듈을 통해, ARIA 및 AES 암호화 알고리즘 중 하나를 선택하여 암호화를 위한 코덱에 오버라이딩한다. 한편, 암호화 알고리즘이 오버라이딩된 코덱은 HDFS 블록 분할 컴포넌트를 통해 생성된 데이터 블록에 적용하여, 암호화된 결과데이터를 사용자에게 반환한다.

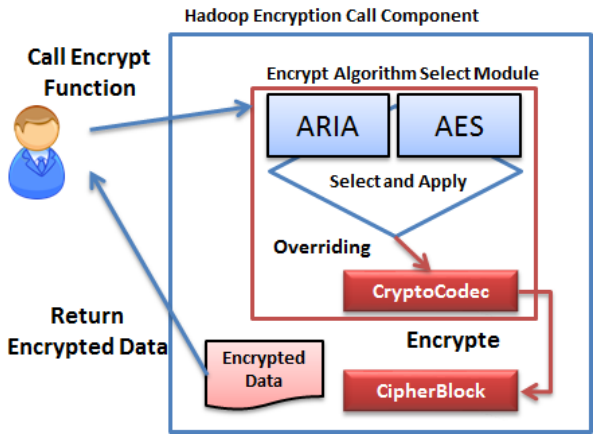


Fig. 5. Hadoop Encryption Call Component

2) HDFS 블록 분할 컴포넌트(HDFS Block Distribute Component)

기존 ARIA 암호화 알고리즘은 데이터 전체에 대한 암호화를 지원하기 때문에, 하둡에서 분산 컴퓨팅 환경을 지원하기 위한 분산 저장 시스템인 HDFS에서 암호화를 수행할 수 없다는 문제점이 존재한다. 이 문제점을 해결하기 위해 본 논문에서는 Liu Yi가 제안한 Splittable-Compression Codec[5]을 적용하여 데이터를 HDFS 블록 단위(64MB)로 나눈 후 동일한 암호화 키를 사용하여 암호화를 수행하도록 함으로써 하둡의 분산된 환경에서도 암호화를 지원한다. Fig. 6은 HDFS 블록 분할 컴포넌트를 이용한 ARIA 암호화 과정을 보인다.

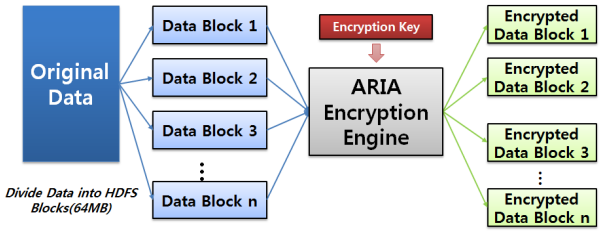
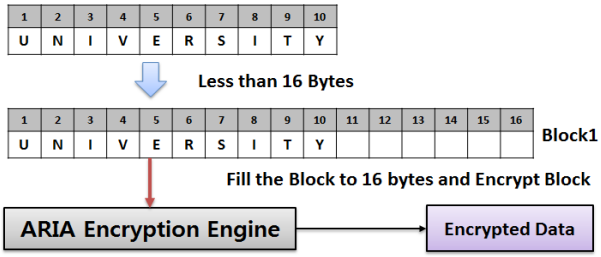


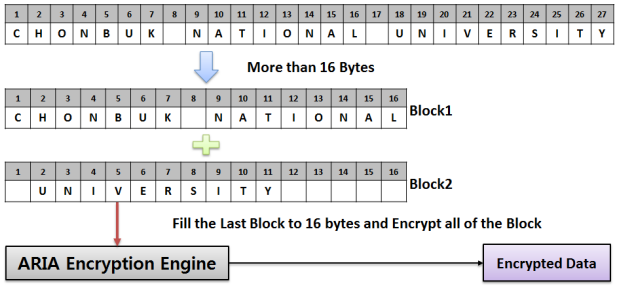
Fig. 6. HDFS Block Distribute Component

3) 가변길이 처리 컴포넌트(Variable Length Data Processing Component)

AES와 ARIA 알고리즘은 암호화 블록인 128비트(16바이트) 단위의 데이터만 암호화를 지원한다. 따라서 하둡 기반 AES 알고리즘과 기존 ARIA 알고리즘은 16바이트 미만이거나 16바이트 단위가 아닌 데이터는 암호화를 지원하지 못하는 문제점이 존재한다. 이 문제점을 해결하기 위해 본 논문에서는 16바이트 배수의 아닐 경우 더미(Dummy) 데이터를 자동으로 추가하여 암호화를 수행함으로써 16바이트 단위가 아닌 다양한 크기의 데이터에 대해서도 암호화를 지원하도록 가변길이 처리 컴포넌트를 제안한다. 아울러 해당 컴포넌트는 기존 AES 코덱에도 적용하여 128비트 단위가 아닌 데이터에 대한 AES 암호화 또한 지원한다. Fig. 7(a)와 (b)는 각 가변길이 데이터의 처리과정을 나타낸다.



(a) Encrypt data that is less then 16 bytes



(b) Encrypt data that is more then 16 bytes

Fig. 7. Encrypt variable length data

가변길이 처리 컴포넌트를 통해 암호화 블록 단위의 암호화를 수행하는 알고리즘은 Fig. 8과 같다. 알고리즘은 입력으로 들어온 파일이 끝날 때까지 16바이트 단위의 데이터를 읽어 암호화 코덱 블록에 할당하고 암호화를 수행한다. 첫째, 읽은 데이터가 마지막 블록일 경우 읽은 마지막 블록이 16바이트 단위의 데이터인지 확인하고 그렇지 않으면 공백문자를 넣어 16바이트 단위의 블록이 되도록 한다(Line 3~8). 둘째, 읽은 데이터가 마지막 블록이 아닐 경우 입력 파일에서 16바이트를 읽어 암호화 블록에 할당한다(Line 9~12). 최종적으로 블록을 암호화하기 위하여, 암호화 컴포넌트를 호출한다(Line 13).

```
Algorithm 1. Variable Length Processing Algorithm
 input : Input File
 1 : CurrentOff = 0
 2 : while CurrentOff >= EOF do
 3 :     if CurrentOff + 16 >= EOF then
 4 :         CodecBlk = Read InputData to EOF
 5 :         CurrentOff = CurrentOff + Read Bytes
 6 :         if CodecBlk.size % 16 != 0 then
 7 :             Fill CodecBlk
 8 :         end if
 9 :     else
 10:         CodecBlk = Read InputData 16 Bytes
 11:         CurrentOff = CurrentOff + 16
 12:     end if
 13:     encrypt(CodecBlk, OutputStream)
 14: end while
 15: return OutputStream
End Algorithm
```

Fig. 8. Variable Length Processing Algorithm

4) 암호화 컴포넌트(Encryption Component)

전체 데이터에 대한 암호화를 위하여 가변길이 처리 컴포넌트에서 생성된 중간 데이터에 대한 암호화를 수행한다. 암호화를 수행하는 과정에는 두 가지 유형이 존재한다. 첫째, 사용자가 ARIA 코덱을 선택한 경우, 입력받은 암호화 Key 값을 ARIA 엔진을 적용하여 암호화를 수행한다. 이후 암호화된 데이터를 가변길이 처리 컴포넌트, HDFS 블록 분할 컴포넌트 및 하둡 암호화 호출 컴포넌트까지 반환하여 사용자에게 암호화된 데이터를 반환한다. 둘째, 사용자가 AES 코덱을 선택한 경우 역시 입력받은 암호화 Key 값을 AES 코덱에 적용하여 암호화를 수행하고, 앞서 설명한 순서로 사용자에게 암호화된 데이터를 반환한다.

3.3 제안하는 기법의 복호화 과정

복호화 과정은 Fig. 9와 같다. i) 사용자는 제안하는 ARIA 코덱과 기존 AES 코덱 중 하나를 선택하여 복호화 호출 컴포넌트를 호출한다. 복호화 호출 컴포넌트는 복호화를 위해 HDFS 블록 컴포넌트를 호출한다. ii) HDFS 블록 컴포넌트는 HDFS에 분산 저장되어 있는 암호화된 데이터를 병합한 후 가변길이 처리 컴포넌트를 호출한다. iii) 가변길이 처리 컴포넌트는 암호화 블록 데이터(128bit) 단위로 나눈 후 복호화 컴포넌트를 호출한다. iv) 복호화 컴포넌트는 각각의 블록에 대해 호출된 코덱(ARIA, AES)을 통해 복호화를 수행한다. 복호화된 데이터 블록들은 가변길이 처리 컴포넌트에서 하나로 병합 후 HDFS 블록 분할 컴포넌트를 통해 저장된다.

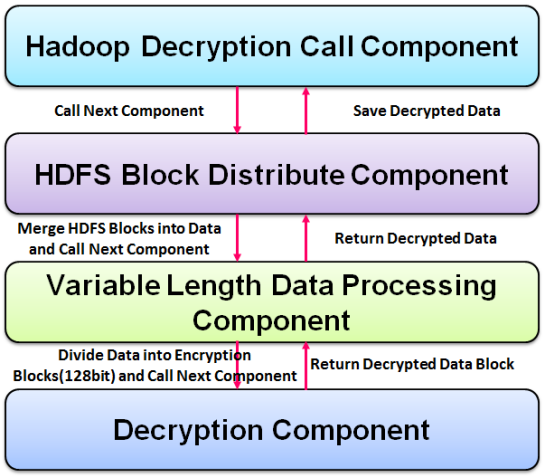


Fig. 9. Description process

3.4 제안하는 기법의 상세설계

본절에서는 제안하는 ARIA 알고리즘을 이용한 HDFS 데이터 암호화 기법의 상세설계를 기술한다. 제안하는 ARIA 코덱의 컴포넌트 구조 및 호출 흐름도는 Fig. 10과 같다. 그림에서 i) AriaParallelCryptoCodec 클래스는 하둡 암호화 호출 컴포넌트에 해당하며, AriaCrypto OutputStream을 호출한다. ii) AriaCryptoOutputStream은 원본 데이터를 입력받아 암호화된 데이터를 출력하는 클래스로 입력받은 원본 데이터를 HDFS 블록 단위(64MB)로 데이터를 분할한 후 AesEncrypt를 호출한다. 따라서 AriaCryptoOutputStream은 HDFS 블록 분할 컴포넌트에 해당한다. iii) AriaEncrypt 클래스는 AriaCryptoOutputStream으로부터 입력받은 HDFS 블록 데이터를 암호화 블록 단위(16Byte)로 나눈 후 가장 마지막 블록이 16바이트 단위인지 확인한다. 아울러 16바이트 단위가 아닐 경우, 블록을 16바이트 단위로 채운 후 각 블록 데이터에 대한 암호화를 수행하는 클래스로써 가변길이 처리 컴포넌트에 해당한다. iv) ARIAEngine은 Aria-Encrypt 클래스에서 호출받은 메소드를 통해 실제 암호화를 수행하는 클래스로 암호화 컴포넌트에 해당한다. 복호화 또한 암호화와 같은 호출 흐름을 가진다. 여기서 Aria-Constant 클래스는 암호화 키, 블록 사이즈 등을 관리하는 암호화 상수 값 관리 클래스이다. 복호화 클래스 호출 흐름 또한 암호화 클래스 호출 흐름과 동일하다.

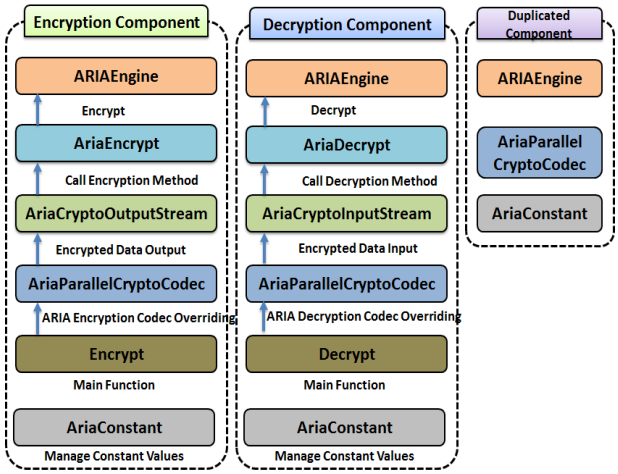


Fig. 10. Detailed design of ARIA-based HDFS data encryption scheme

4. 성능 평가

4.1 실험 데이터 및 성능평가 환경

제안하는 ARIA 기반의 HDFS 암호화 기법을 Java 1.8.24 버전 상에서 개발도구는 Eclipse Luna를 사용하여 구현하였다. 아울러, JAR 파일로 패키징(packaging)하여 사용자가 하둡 내부를 수정할 필요 없이, 하둡 어플리케이션에서 ARIA 암호화 알고리즘 코덱을 적용할 수 있도록 지원하였다.

아울러 ARIA 알고리즘을 이용한 HDFS 데이터 암호화 기법에 대한 성능평가는 문자열 데이터를 처리하는 두 개의 대표적인 응용 및 과학 빅데이터 처리를 수행하는 두 개의 대표적 응용에 대해 성능을 측정하였다. 단어계산 응용은 문자열 데이터를 처리하는 가장 기본적인 응용이며, 정렬 응용은 단어 계산보다 연산량이 많은 응용에 대해 암호화 알고리즘의 처리성능을 확인할 수 있다. 아울러, 과학 빅데이터 분야의 k-Means 은 군집화 문제를 해결하는 기본적인 알고리즘이다. 또한 계층적 클러스터링 응용은 k-Means 보다 복잡한 군집화 연산을 수행하는 대표적인 응용이다. Table 2는 성능평가에 사용된 4개의 응용 및 선정이유를 나타낸다.

Table 2. Application of performance evaluation

Application	Reason of select
Word Count	Most basic application that can understand the operation process of Mapreduce framework
Sort	String types bigdata processing Application that has large number of arithmetic operations
k-Means	The basic algorithm for solving Clustering problem
Hierarchical Clustering	Application that perform complex clustering amount of calculation than k-Means

네 개의 응용에 사용한 각 데이터는 Table 3과 같다. Table 2에서 위키피디아 영문 덤프 데이터 및 메타위키 토막글 덤프 데이터는 타임스탬프가 포함된 XML 데이터 셋이며, MODIS AQUA 데이터는 해양 과학 분야에서 사용하는 연/월 간 해조류 정보를 포함하는 데이터이다.

성능평가 환경은 동일한 환경을 가진 노드 3대에서 암호화, 복호화 기능을 수행하였다(Table 4). Table 4에서 CPU는 Intel(R) Core i3-3240 3.40GHz의 성능을 가지며, Memory는 12GB, OS는 Linux Ubuntu 12.04.2 버전을 사용하였다. 하둡은 2.6.0 버전을 사용하였다.

Table 3. Data for application

Application	Type of data	Size
WordCount	Dump of English Wikipedia in 25.07.2015	86GB
Sort	Dump of English Metawiki Stub in 10.08.2015	6.45GB
k-Means	MODIS AQUA data in 10.2015	368.06MB 15 million records
Hierarchical Clustering	MODIS AQUA data in 2012	1.34GB 75 million records

Table 4. Evaluation environment

Item	Performance
CPU	Intel(R) Core i3-3240 CPU @ 3.40GHz
Memory	12GB
OS	Ubuntu 12.04.2

4.2 단어계산 성능평가

단어계산 응용의 성능평가 결과는 Fig. 11과 같다. 암호화를 수행하지 않은 NoEncrypt는 평균 12307.8초의 성능을 보였고, AES의 경우 평균 13190.4초의 성능을 보였다.

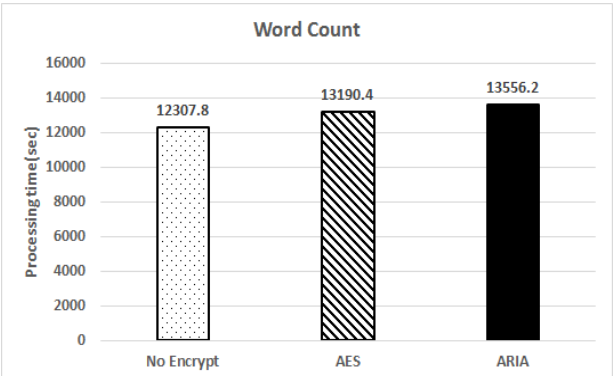


Fig. 11. Result of Word Count Application

본 논문에서 제안하는 ARIA 알고리즘은 13556.2초의 성능을 보여 암호화를 하지 않았을 경우보다 약 10%, AES 암호화를 수행했을 경우보다 약 3% 느린 성능을 보였다.

AES 알고리즘은 Intel사의 AES-NI와 같은 성능 최적화[11] 및 암호화 기법의 꾸준한 최적화로 인해 가장 좋은 성능을 나타내는 암호화 알고리즘이다. 한편 ARIA 자체의 암호화 알고리즘이 AES 알고리즘 보다 늦기 때문에, ARIA 기반 암호화 기법이 약 3% 느린 암호화 성능을 가짐을 나타낸다.

4.3 정렬 알고리즘 성능평가

정렬 응용의 성능평가 결과는 Fig. 12와 같다. 암호화를 하지 않은 상태인 NoEncrypt는 평균 1947.6초의 성능을 보였고, AES의 경우 평균 2016.6초의 성능을 보였다. 본 논문에서 제안하는 ARIA 알고리즘의 경우 평균 2081.6초의 성능을 보여 암호화를 하지 않았을 경우보다 약 7%, AES 암호화를 수행했을 경우보다 약 3% 느린 성능을 보였다. 한편, 단어계산 성능평가와 비교하였을 때 AES 기반 암호화 기법과 비교하여 전체 성능이 줄어들지 않은 결과를 확인할 수 있다. 이는 문자열 데이터에 대한 연산량이 AES 및 ARIA 암호화 알고리즘의 성능적 차이에 큰 영향을 주지 않는 것으로 분석할 수 있다.

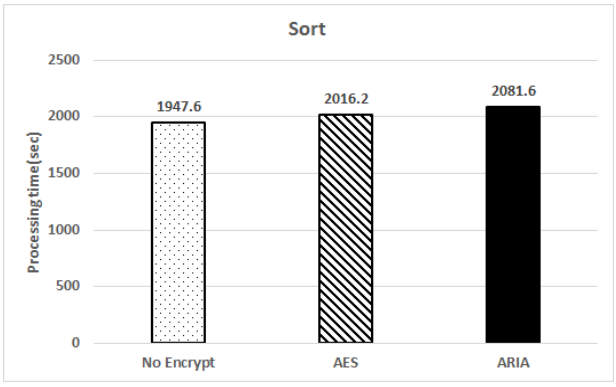


Fig. 12. Result of Sort Application

4.4 k-Means 성능평가

k-Means 클러스터링 응용의 성능평가 결과는 Fig. 13과 같다. 암호화를 하지 않은 상태인 NoEncrypt는 평균 440.0초의 성능을 보였고, AES의 경우 약 462.4초의 성능을 보여 암호화를 하지 않았을 경우보다 약 5% 느린 성능을 보였다.

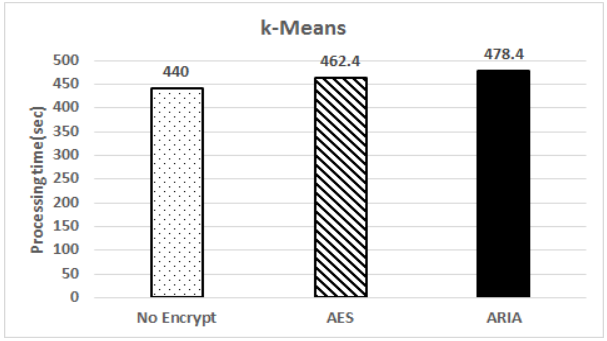


Fig. 13. Result of k-Means Application

본 논문에서 제안하는 ARIA 알고리즘의 경우 평균 478.4초의 성능을 보여 암호화를 수행하지 않았을 경우보다 약 9%, AES 암호화를 수행했을 경우보다 약 3% 느린 성능을 보였다. 단어계산 및 정렬 응용과 마찬가지로, AES 기반 암호화 기법과 비교하여 약 3% 느린 성능을 보였다. k-Means의 경우 단어계산, 정렬과 비슷한 성능 차이를 보이고 있는데, 이는 다른 군집화 응용에 비해 비교적 적은 연산으로 군집화를 수행하기 때문에 연산 오버헤드보다 암호화 및 복호화 오버헤드가 큰 것으로 분석할 수 있다.

4.5 계층적 클러스터링 성능평가

계층적 클러스터링 알고리즘의 성능평가 결과는 Fig. 14와 같다. 암호화를 하지 않은 상태인 NoEncrypt는 평균 1460초의 성능을 보였고, AES의 경우 평균 1470.2초의 성능을 보였다. ARIA 알고리즘은 1490.6초의 성능으로 암호화를 하지 않았을 경우보다 약 2%, AES 암호화를 수행했을 경우보다 약 1% 느린 성능을 보였다. 다른 응용들에 비해 연산 시간이 많은 계층적 클러스터링 응용에서는 ARIA 기법 암호화 기법은 암호화를 수행하지 않았을 경우와 비교하여 성능이 늦어진 것이 큰 폭으로 감소하였다. 따라서 데이터의 암호화 및 복호화 시간보다 데이터 연산 시간이 많은 응용일 경우, AES 및 ARIA 기반 암호화 기법의 성능은 암호화/복호화에 따른 시간에 덜 의존적인 것으로 분석되었다.

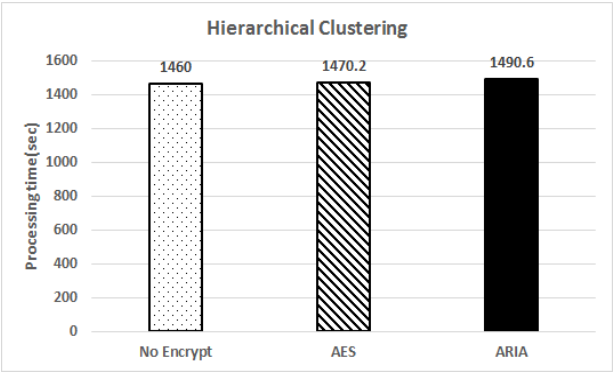


Fig. 14. Result of Hierarchical Clustering Application

5. 결론 및 향후 연구

본 논문에서는 국내 표준으로 채택된 ARIA 암호화 알고리즘을 하둡 기반의 응용에 사용하기 위해, 하둡 상에서 ARIA 알고리즘을 이용한 HDFS 데이터 암호화 기법을 제안하였다. 제안하는 기법은 데이터를 HDFS 블록 단위(64MB)로 나누어 암호화를 수행함으로써 하둡 분산 병렬 처리 환경에서 암호화 및 복호화를 지원하는 HDFS 블록 분할 컴포넌트를 제공하고, 암호화 블록 단위인 128비트 단위의 데이터뿐만 아니라 가변 길이의 데이터 암호화를 제공하는 가변길이 데이터 처리 컴포넌트를 제공한다. 아울러 국내외 표준을 지원하기 위한 ARIA/AES 코덱 변환 기능을 추가적으로 제공

한다. 성능평가에서는 단어계산, 정렬, k-Means, 계층적 클러스터링 응용에 대해 XML 데이터 및 과학 빅데이터를 사용하여 다양한 응용에 대한 ARIA 암호화를 지원할 수 있음을 보였다. 제안하는 ARIA 기반 암호화 기법은 AES 기반 암호화 기법과 비교하여 성능이 2-3% 늦은 것으로 나타났으나, AES 와 동일한 암호화 Key 비트 적용을 통해 같은 수준의 데이터 보호도를 지원하고, 가변 길이 블록을 지원하는 장점을 지니고 있다.

향후 연구로는 위치 정보, 금융 정보 등 실생활에서 사용되는 데이터에 대해 본 연구에서 제안하는 암호화 기법을 적용하여 보안을 지원하는 것이다. 아울러 더미 데이터를 공백문자로 채웠을 때, 산술 데이터를 처리할 경우에는 정수형이나 복소수형의 변수 값으로 변환할 때 가장 뒤에 채워진 공백문자를 무시하기 때문에 원본 데이터에 대한 손실 우려가 없다. 그러나 문자열 데이터를 처리할 경우, 복호화하였을 때 원본 데이터에 있던 공백 문자열이 더미 데이터로 판단될 수 있기 때문에 데이터 손실이 우려되는 문제점이 존재한다. 이러한 문제점을 해결하기 위해 공백문자뿐만 아니라 사용자가 지정한 문자열이나 바이트를 더미데이터에 삽입할 수 있게 하는 확장이 필요하다.

References

[1] J. Dean and S. Ghemawat, “MapReduce: Simplified data processing on large clusters,” *Communications of the ACM*, Vol.51, Issue.1, pp.107-113, 2008.

[2] Hadoop [Internet], http://hadoop.apache.org.

[3] S. Narayanan, “Securing Hadoop : Implement robust end-to-end security for your Hadoop ecosystem,” 1st Vol, PACKT Publishing, 2014.

[4] So Hyeon Park and Ik Rae Jeong, “A Study on Security Improvement in Hadoop Distributed File System Based on Kerberos,” *Journal of the Korea Institute of Information Security and Cryptology*, Vol.23, Issue.5, pp.803-813, 2013.

[5] Liu Yi, Hadoop Crypto Design [Internet], https://issues.apache.org/jira/secure/attachment/12571116/Hadoop Crypto Design.pdf.

[6] Seonyoung Park and Youngseok Lee, “A Performance Analysis of Encryption in HDFS,” *Journal of KISS: Databases*, Vol.41, Issue.1, pp.21-27, 2014.

[7] Byeong-yoon Choi. “Design of Cryptographic Processor for AES Rijndael Algorithm,” *The Journal of The Korean Institute of Communication Sciences*, Vol.26, Issue.10, pp. 1491-1500, 2001.

[8] Yong Kuk Cho, Jung Hwan Song, and Sung Woo Kang, “Criteria for Evaluating Cryptographic Algorithms based on Statistical Testing of Randomness,” *Journal of the Korea Institute of Information Security and Cryptology*, Vol.11, Issue.6, pp.67-76, 2001.

[9] ARIA Development Team, Block Encryption Algorithm ARIA [Internet], http://glukjeoluk.tistory.com/attachment/ok110000000002.pdf.

[10] Korea Internet & Security Agency, 민관겸용 블록 암호 알고리즘 ARIA 알고리즘 명세서 [Internet], http://seed.kisa.or.kr/iwt/ko/bbs/EgovReferenceDetail.do?bbsId=BBSMSTR_000000000002&nttId=39&pageIndex=1&searchCnd=&searchWrd=.

[11] Jeffrey Root, Intel® Advanced Encryption Standard Instructions(AES-NI), https://software.intel.com/en-us/articles/intel-advanced-encryption-standard-instructions-aes-ni.

송 영 호

e-mail : songyoungho@jbnu.ac.kr

2014년 전북대학교 IT정보공학과(공학사)

2015년 전북대학교 컴퓨터공학부(공학석사)

2016년~현 재 전북대학교 컴퓨터공학부 박사과정

관심분야 : 빅데이터 프로세싱, 분산 병렬 처리 시스템, 클라우드 컴퓨팅

신 영 성

e-mail : twotoma@jbnu.ac.kr

2010년 전북대학교 컴퓨터공학부(공학사)

2012년 전북대학교 컴퓨터공학부(공학석사)

2013년~현 재 전북대학교 컴퓨터공학부 박사과정

관심분야 : 데이터베이스, 클라우드 컴퓨팅, SNS, 빅데이터 병렬 프로세스, 협업 모델, 하둡, 맵리듀스

장 재 우

e-mail : jwchang@jbnu.ac.kr

1984년 서울대학교 전자계산기공학과 (공학사)

1986년 한국과학기술원 전산학과(공학석사)

1991년 한국과학기술원 전산학과(공학박사)

1996년~1997년 Univ. of Minnesota, Visiting Scholar

2002년~2004년 Penn State Univ., Visiting Scholar

1991년~현 재 전북대학교 IT정보공학과 교수

관심분야 : 공간 네트워크 데이터베이스, 하부저장구조, 클라우드 컴퓨팅, 데이터베이스 아웃소싱 등