JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지(J. Korea Inst. Inf. Commun. Eng.) Vol. 19, No. 4 : 848~854 Apr. 2015

# 선택적 요소 암호화 방식에 대한 성능 분석

양설[1] · 김지홍[2*]

# The performance analysis of the selective element encryption method

**Xue Yang[1] · Ji-hong Kim[2*]**

Department of Information and Communication, Semyung University, Jecheon 390-711, Korea

## 요 약

최근 데이터베이스 보안을 위한 다양한 암호화 방법이 제안되고 있다. 이러한 암호화 방법들은 사용자의 민감한 데이터를 효과적으로 보호할 수 있지만 데이터베이스 질의어에 대한 검색 성능이 저하된다.

본 논문에서는 이러한 단점을 보완하기 위하여 블룸필터에서의 선택적 요소 단위의 암호화 방식을 제안하였다. 그리고 다수의 질의어를 통하여 기존에 제안된 튜플 암호화 방식과 성능을 비교하였다. 결과적으로 민감한 정보에 대하여 요소 암호화 방법을 적용하고, 이에 대한 인덱스로서 블룸필터를 사용한 제안 방식이 기존에 제안된 방식보다 검색 시간이 매우 빠르게 나타남을 알 수 있다.

## ABSTRACT

There are a lot of encryption methods to secure database proposed recently. Those encryption methods can protect the sensitive data of users effectively, but it deteriorates the search performance of database query.

In this paper, we proposed the selective element encryption method in order to complement those drawbacks. In addition, we compared the performance of the proposed method with that of tuple level encryption method using the various queries. As a result, we found that the proposed method, which use the selective element encryption with bloom filter as a index, has better performance than the other encryption method.

**키워드** : 데이터베이스 암호화, 암호화된 데이터의 검색, 요소 암호화, 블룸필터

Key word : Database Encryption, Encrypted data search, Element encryption, Bloom filter

**Received** 06 January 2015, **Revised** 29 January 2015, **Accepted** 16 February 2015

*** Corresponding Author** Ji-hong Kim(E-mail: jhkim@semyung.ac.kr, Tel:+82-43-649-1310)
Department of Information and Communication, Semyung University, Jecheon 390-711, Korea

Open Access http://dx.doi.org/10.6109/jkiice.2015.19.4.848 print ISSN: 2234-4772 online ISSN: 2288-4165

## Ⅰ. 서 론

최근 현대사회는 네트워크와 인터넷의 발달로 의해 정보화 사회로 빠르게 변화하고 있다. IT 서비스가 점점 다양화되고 개인 맞춤형 서비스로 발전함에 따라, 사용자의 주민등록번호와 같은 민감한 정보가 더 쉽게 노출될 수 있다.

개인정보 사고를 방지하고 차단하기 위한 가장 확실한 방법은 데이터베이스내의 정보에 대한 암호화 방법이다. 최근 데이터베이스 보안을 위한 다양한 암호화 방법이 제안되고 있다. 그런데 이러한 암호화 방법들은 사용자의 민감한 데이터를 효과적으로 보호할 수 있지만 데이터베이스 질의어에 대한 검색 성능이 저하된다는 단점이 있다[1].

데이터베이스 암호화 방식은 주로 튜플 단위와 요소 단위의 암호화 방식으로 구분할 수 있다. 튜플 단위의 암호화 방식은 해당 튜플 전체를 복호화하고, 다시 필요한 부분만을 선택하여야 한다는 단점을 가지고 있다. 튜플 단위의 암호화 방식의 대표적인 것은 버킷기반 방식이다. 버킷기반(Bucker Based Index) 방식[2]은 Hacigumus et al이 제안한 방법으로서, 튜플 전체를 암호화하고, 각 필드의 속성을 버킷 단위로 분류하여 처리하는 방식이다. 검색 처리할 때는 버킷단위로 검색결과를 복호화하고 다시 분석하여야 버킷 안에서 원하는 결과를 얻을 수 있기 때문에 검색 성능이 저하된다는 단점이 있다[3].

본 논문은 이러한 단점을 보완하기 위하여 요소 단위의 암호화 방식을 바탕으로 민감한 데이터에 대하여 블룸필터에서의 선택적 요소 단위의 암호화 방식을 제안하였다. 그리고 제안하는 방식에 대한 성능 분석을 위해서 3가지 경우에 대해서 기존의 튜플 방식[3]과 성능을 비교하였다.

본 논문의 구성은 다음과 같다. 2장에서는 블룸필터와 튜플 암호화 방식에 대해 살펴보고, 3장에서는 데이터베이스 보호하며 암호화된 데이터에 대해 더 효율적으로 검색하기 위한 방법을 제시한다. 4장에서는 제안한 방식에 대한 성능을 비교하고 마지막 5장에서 결론으로 마무리한다.

## Ⅱ. 관련연구

### 2.1. 블룸필터(Bloom Filter)

블룸필터는 전체 도메인 U의 어떤 집합 S={$s_1, s_2, \cdots, s_n$}의 키 값을 비트벡터 V로 표현하기 위한 방법으로 1970년 Burton Bloom에 의해 개발되었다[4].

블룸필터는 다음과 같이 동작한다. 처음 비트벡터 V의 모든 비트는 0으로 초기화 되어있다. 블룸필터는 k개의 해시함수 $h_1, h_2, \cdots, h_k$를 이용하여 전체 도메인 U의 키 값을 {1,2,⋯,m}으로 사상시킨다. s∈S인 각 요소는 해시함수를 사용하여 비트벡터v의 $h_1$(s),$h_2$(s),⋯, $h_k$(s)위치를 1로 바꾼다. 만약 t∈U인 요소가 t∈S인지를 검사하기 위해서는 비트벡터V의 $h_1$(t),$h_2$(t),⋯,$h_k$(t)위치의 비트를 확인한다. 그 비트들 중 하나라도 0이면 t∉S임이 분명하다. 하지만 t∉S라 할지라도, 모든 비트가 1이 될 수 있는데 이것은 블룸필터에 존재하는 오류로서 긍정오류(False Positive)라고 한다. 긍정오류는 실제로 존재하지 않는 요소 t에 대하여 $h_1$(t),$h_2$(t),⋯,$h_k$(t)비트가 1로 설정되어 존재하는 것으로 판정되었기 때문이다[5].

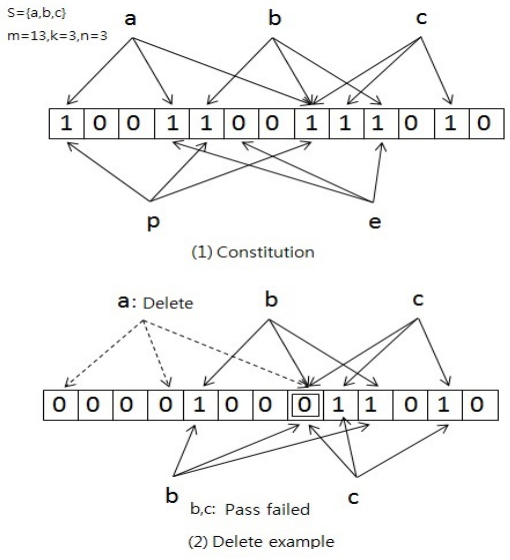


**그림 1.** 블룸필터
**Fig. 1** The bloom filter

그림 1의 (1)는 블룸필터의 구조를 보여주고 있다. 요소 집합 S={a,b,c}이고, 블룸필터의 크기 m=13, 해쉬함수의 개수 k=3일 때, 요소 e는 블룸필터에 존재하지 않는 것으로 판단되며 요소 p는 p∉S지만 블룸필터 검사를 통해 존재하는 것으로 판단되기 때문에 긍정오류가 발생된다. 그러나 블룸필터를 설계할 때 블룸필터 크기를 증가시켜 긍정오류의 비율을 조절할 수 있다[6]. 또한 다수의 테이블에 존재하는 각각의 블룸필터 값은 OR 연산으로 한 개의 블룸필터로 구성될 수 있으므로 테이블 간의 조인에 활용될 수 있다.

### 2.2. 튜플 암호화 방식

버킷 인덱스와 블룸필터를 이용하는 암호화 방식은 튜플 암호화 방식의 대표적인 것으로 요즘 많이 사용되고 있다. 이 방식[4]는 버킷인덱스와 블룸필터를 이용하는 암호화 방식을 제안한 것이다. <그림 2>와 같이 데이터베이스내의 민감한 숫자 데이터에 대한 검색방안으로서, 숫자 데이터와 범위형 문자 데이터를 일정한 크기로 분리되는 버킷을 설정할 때 먼저 구간별로 버킷을 설정하고, 해쉬함수를 통해 블룸필터에 입력한다. 반면에 문장형 문자데이터의 경우에는 해당 키워드만을 추출하여, 해쉬함수를 통해 블룸필터에 적용함으로서, 존재유무를 확인할 수 있도록 한다.

그림 2는 버킷 인덱스와 블룸필터를 이용하는 암호화 방식 예시로 보여 주고 있다.

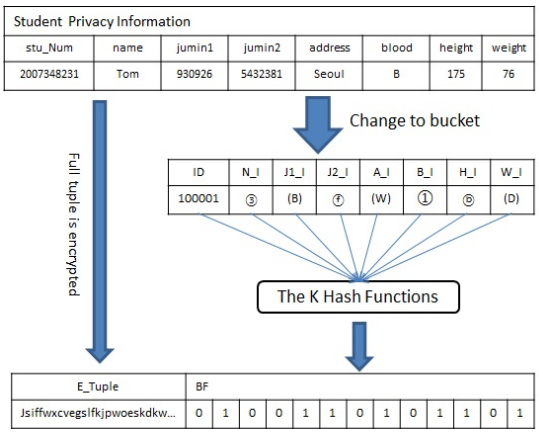


**그림 2.** 버킷 인덱스와 블룸필터를 이용하는 암호화 방식
**Fig. 2** The encryption method using bucket Index and bloom filter

## Ⅲ. 암호화 기법

본 장에서는 민감한 정보를 위한 블룸필터에서의 선택적 요소 단위의 암호화 방식을 제안하고자 한다.

제안방식을 설명하기 위하여 그림 3과 같은 요소를 포함된 학생 개인정보 테이블과 학생 성적정보 테이블로 구성된 학생 데이터베이스를 이용한다.

Student Privacy Information

| stu_Num | name | jumin1 | jumin2 | address | blood | height | weight |
|---|---|---|---|---|---|---|---|
| 2007348231 | Tom | 930926 | 5432381 | Seoul | B | 175 | 76 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

Student Performance Information

| stu_Num | year | dept. | grade | course_n |
|---|---|---|---|---|
| 2007348231 | 3 | computer | 85 | 3 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

**그림 3.** 학생 데이터베이스 구성
**Fig. 3** The configuration of the student database

### 3.1. 제안 암호화 방식의 특징

우선 각 요소의 특징에 따라서 민감한 요소만을 암호화시켜 저장하고 암호화 된 데이터를 포함한 몇 개의 키워드를 블룸필터에 설정함으로써 튜플방식의 암호화에 비해 다양한 쿼리문을 보다 빠르게 수행할 수 있어 검색 성능을 향상시킬 수 있다는 장점을 가지고 있다. 우선 요소 특징에 따라서 암호화 필요할 요소를 분석한다.

표 1은 학생 개인정보 테이블에 있는 요소의 특징에 대한 분석 결과를 보여준다.

**표 1.** 학생 개인정보 테이블 안의 요소 특징
**Table. 1** The feature of the elements in student personal Information table

| Student Privacy Information | | | |
|---|---|---|---|
| Element Name | Encryption Required? | Element Name | Encryption Required? |
| stu_num | ○ | address | ○ |
| name | ○ | blood | △ |
| jumin1 | △ | height | × |
| jumin2 | ○ | weight | × |

(○: BF+ 암호문, △: BF, ×: 평문)

표 1과 같이, 학생 개인정보 테이블에서, 학번, 이름, 주민번호 뒷자리(주민2)와 주소는 민감한 개인정보라서 암호화해서 블룸필터에 기록한다. 주민 앞자리(주민1)와 혈액형은 블룸필터를 사용할 수 있지만 본 논문에서는 평문으로 이용한다. 키와 몸무게는 중요하지 않은 정보이기 때문에 평문으로 이용한다. 표 2는 학생 성적정보 테이블에 있는 요소의 특징에 대한 분석 결과를 보여준다.

**표 2.** 학생 성적정보 테이블 안의 요소 특징
**Table. 2** TThe feature of the elements in student grade table

| Student Performance Information | | | |
|---|---|---|---|
| Element Name | Encryption Required? | Element Name | Encryption Required? |
| stu_num | ○ | grade | △ |
| year | △ | course_n | × |
| dept | ○ | | |

(○: BF+ 암호문, △: BF, ×: 평문)

표 2와 같이, 학생 성적정보 테이블에서, 학번과 학과는 민감한 개인정보라서 암호화해서 블룸필터에 기록하고 학년과 성적은 블룸필터를 사용할 수 있지만 본 논문에서는 평문으로 이용한다. 그리고 과목수는 중요하지 않은 정보이기 때문에 평문으로 이용한다.

3.1.1. 제안 암호화 방식의 구성

분석된 요소의 특징에 따라 민감한 데이터를 암호화해서 저장하고, 이를 검색하기 위한 방안으로 블룸필터에서 검색한다. 평문은 테이블이 분리되어 있기 때문에 조인문(JOIN)을 사용함으로써 검색을 하게 되고, 암호화한 데이터들은 한 테이블에 모두 저장하여 검색하는 것으로 한다. 즉, 본 제안방식에서는 블룸필터간의 OR 특성을 이용하여 두 개의 테이블을 한 개의 테이블로 조인한 후에 검색을 실시한다.

그림 4는 학생 개인정보 테이블과 학생 성적정보 테이블을 블룸필터에 적용한 결과를 보여준다.

그림 4에서, E_I 필드는 학번을 암호화한 것이고 E_N 필드는 이름을 암호화한 것이다. 그리고 E_J2 필드는 주민번호 뒷자리를 암호화한 것이고 E_A 필드는 주소를 암호화한 것이다. 마지막 E_D 필드는 학과를 암호화한 것이다.

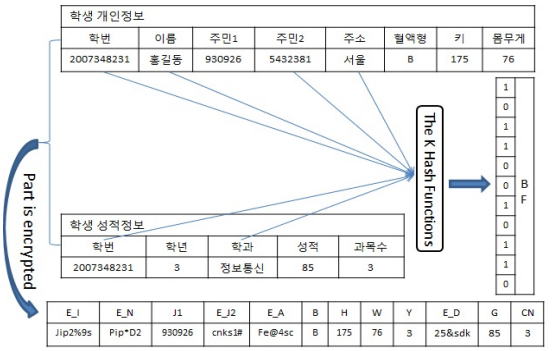


**그림 4.** 블룸필터 방식 적용한 학생 개인정보 테이블과 학생 성적정보 테이블
**Fig. 4** The student Personal Information table and student grade table using bloom filter method

## Ⅳ. 제안 방식에 대한 성능 비교

본 논문에서 제안한 방식의 효율성을 평가하기 위하여 블룸필터의 32, 64, 128 비트의 블룸필터를 기준으로 튜플 암호화 방식과 비교하였다. 본 실험에서 사용된 데이터는 10만 명의 임의의 학생 데이터를 사용하였다. 튜플 암호화 방식과 제안한 방식의 비교를 위하여 평문, 튜플 암호화 방식과 제안 방식(요소 암호화 방식)인 각각 경우의 검색을 10번씩 수행한 시간의 평균으로 검색시간을 확인하였다. 그리고 제안방식과 튜플 암호화 방식을 비교하기 위해 3 가지의 경우에 대한 검색조건을 가지고 각각의 검색순서와 데이터를 확인하였다.

a) 암호문(학번, 이름, 주민2, 주소, 학과)을 가진 검색 조건을 이용하여 평문(주민1, 혈액형, 키, 몸무게, 학년, 성적, 과목수)을 가진 결과를 검색하는 경우.

질의어1: 간호학과이면서 키가 150 이상인 학생을 검색하라.

그림 5와 표 3은 질의어1에 대한 제안 방식을 이용한 검색 과정과 이에 대한 검색시간을 비교한 것이다.

암호문을 가진 검색 조건을 이용하여 평문을 가진 결과를 검색하는 첫 번째 쿼리문의 경우에서, 제안방식은 그림 5와 같이, 간호학과를 블룸필터에 검색하면서 동시에 키(H)가 150 이상인 학생을 검색하게 된다.

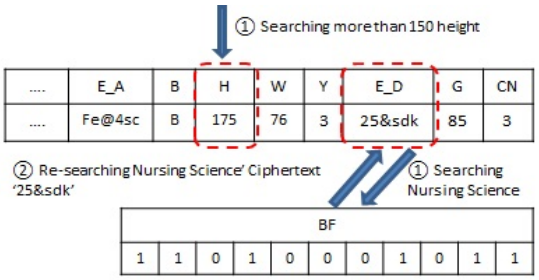


**그림 5.** 질의어1에 대한 제안 방식을 이용한 검색 과정
**Fig. 5** The search process using the proposed method for the queries 1

**표 3.** 질의어 1에 대한 검색시간
**Table. 3** The search time of the queries 1

| Bit | Classify | Search time (Unit: Sec.) |
|---|---|---|
| | Plaintext | 0.042 |
| 32bit | Tuple | 0.672 |
| | Element | 0.099 |
| 64bit | Tuple | 0.396 |
| | Element | 0.086 |
| 128bit | Tuple | 0.335 |
| | Element | 0.084 |

이 과정에서 긍정오류확률이 발생하게 되지만 정확한 데이터를 걸러내기 위해 간호학과를 암호화 하여 암호화된 값과 저장된 데이터의 값을 비교함으로써 최종적인 데이터를 확인 할 수 있다. 반면 튜플 암호화 방식에서는 간호학과를 블룸필터에서 검색하는 것은 동일하지만 키에 대한 정보가 없기 때문에 키를 확인하고자 튜플 암호화 된 학생 성적정보 테이블을 전부 복호화 해야 한다. 이 데이터 역시 긍정오류를 포함하고 있기 때문에 복호화 한 데이터 중에서 간호학과와 키가 150인 정보를 통해 재검토를 해야 한다.

간호학과와 키를 동시에 검색한 후 간호의 암호화 문장을 재검색하여 정확한 데이터를 확인한 제안 방식에 비해 블룸필터에서 간호만 검색하여 해당 튜플을 가져온 후 복호화 하여 조건을 재검색해야 하는 튜플 방식 암호화는 검색된 수가 많고, 전부 복호화 해야 되는 차이 때문에 검색 시간이 더 걸리게 되는 것을 확인하였다. 또한 블룸필터의 크기가 클수록 블룸필터에 검색되는 키워드의 긍정오류 수가 적어지기 때문에 검색 시간이 줄어드는 것을 확인하였다.

b) 암호문(학번, 이름, 주민2, 주소, 학과)을 가진 검색조건을 이용하여 암호문(학번, 이름, 주민2, 주소, 학과)을 가진 결과를 검색하는 경우.

질의어2: 서울에 살면서 전기학과인 학생을 검색하라.

그림 6과 표 4는 질의어2에 대한 제안 방식을 이용한 검색 과정과 이에 대한 검색시간을 비교한 것이다.

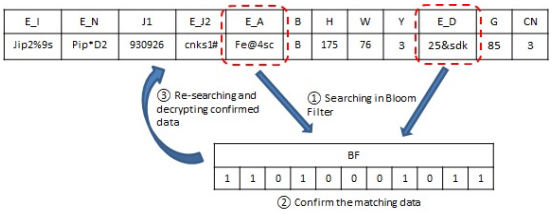


**그림 6.** 질의어2에 대한 제안 방식을 이용한 검색 과정
**Fig. 6** The search process using the proposed method for the queries 2

**표 4.** 질의어 2에 대한 검색시간
**Table. 4** The search time of the queries 2

| Bit | Classify | Search time (Unit: Sec.) |
|---|---|---|
| | Plaintext | 0.052 |
| 32bit | Tuple | 0.422 |
| | Element | 0.422 |
| 64bit | Tuple | 0.335 |
| | Element | 0.337 |
| 128bit | Tuple | 0.311 |
| | Element | 0.312 |

암호문 두 개에 대한 검색결과를 검색하는 두 번째 쿼리문의 경우에서, 제안 방식은 그림 6과 같이 서울과 전기학과에 대한 정보는 블룸필터에 입력했기 때문에 두 키워드를 블룸필터에서 먼저 검색하고, 그 후 나온 결과에 대해 두 키워드의 암호화된 값과 비교하여 재검색을 하게 된다.

튜플 암호화 방식에서는 제안방식과 마찬가지로 두 키워드가 블룸필터에 입력되어 있기 때문에 먼저 블룸필터에서 검색을 하지만, 그 후 결과로 나온 튜플을 복호화 하여 키워드에 맞는 조건으로 재검색하게 된다. 이 쿼리문에서는 두 과정 모두 블룸필터에서 두 키워드를 먼저 검색하기 때문에 긍정오류의 수를 포함한 같은

검색수가 나오며, 두 키워드를 암호화하여 재검색하는 제안방식과 튜플을 복호화하여 두 키워드를 재검색하는 튜플 암호화 방식의 최종 검색시간이 거의 비슷한 것을 확인하였다. 또한 첫 번째 쿼리문과 마찬가지로 블룸필터의 크기가 클수록 블룸필터에 검색되는 키워드의 긍정오류 수가 적어지기 때문에 검색 시간이 줄어드는 것을 확인하였다.

c) 평문(주민1, 혈액형, 키, 몸무게, 학년, 성적, 과목수)을 가진 검색조건을 이용하여 평문(주민1, 혈액형, 키, 몸무게, 학년, 성적, 과목수)을 가진 결과를 검색하는 경우.

질의어3: 과목수가 4개이면서 몸무게 50kg 이상인 학생을 검색하라.

표 5는 과목수가 4개이면서 몸무게 50kg 이상인 학생을 검색한 결과를 보여준다.

**표 5.** 질의어 3에 대한 검색시간
**Table. 5** The search time of the queries 3

| Bit | Classify | Search time (Unit: Sec.) |
|---|---|---|
| | Plaintext | 0.055 |
| 32bit | Tuple | 3.764 |
| | Element | 0.038 |
| 64bit | Tuple | 3.763 |
| | Element) | 0.039 |
| 128bit | Tuple | 3.764 |
| | Element | 0.038 |

평문을 가진 검색조건을 이용하여 평문을 가진 결과를 검색하는 세 번째 쿼리문의 경우에서, 제안 방식은 두 키워드 모두 평문으로 되어 있으므로 블룸필터에서 확인할 필요 없이 바로 확인할 수 있다.

반면 튜플 암호화 방식에서는 확인할 수 없기 때문에 두 가지 튜플 암호화 방식의 모든 데이터를 복호화 한 후 키워드를 검색해야 한다.

제안 방식은 조인문을 사용할 필요없이 블룸필터의 OR 특성을 이용하여 조인된 한 개의 테이블에서 검색하기 때문에 평문보다 조금 더 빨리 검색된 것을 확인할 수 있었으며, 튜플 암호화 방식에 비해 매우 빨리 검색된 것을 알 수 있었다.

## Ⅴ. 결 론

IT 서비스가 점점 다양화되고 개인 맞춤형 서비스로 발전함에 따라, 사용자의 주민등록번호와 같은 민감한 정보가 더 쉽게 노출될 수 있다. 개인정보 사고를 방지하고 차단하기 위한 핵심 방법으로 요즘 정보보안 분야에서 가장 주목을 받는 데이터베이스 보안 기술에 집중되고 있다. 최근 데이터베이스 보안을 위한 암호화 방법들이 점점 많아지고 있으며 이러한 암호화 방법들은 사용자의 민감한 데이터를 효과적으로 보호할 수 있지만 데이터베이스 질의어에 대한 검색 성능이 저하된다.

본 논문에서는 이러한 단점을 보완하기 위하여 블룸필터에서의 요소 단위의 암호화 방식을 제안하였다. 이와 더불어 두 개의 테이블간의 조인연산에 블룸필터 간의 OR 특성을 이용하여 한 개의 테이블로 변환한 후에 3 가지 형태의 특정 질의어를 검색함으로써 튜플 암호화 방식과 성능을 비교하였다. 암호문을 가진 검색조건을 이용하여 평문을 가진 결과를 검색하는 경우, 제안방식은  질의어 1과 같이 간호학과와 키를 동시에 검색한 후 간호의 암호화 문장을 재검색하여 정확한 데이터를 확인하기 때문에 검색 시간이 더 빠르다는 장점이 있다.

반면 튜플 암호화 방식은 블룸필터에서 간호만 검색하여 해당 튜플을 가져온 후 복호화 하여 조건을 재검색해야 하는 튜플 방식 암호화는 검색된 수가 많고, 전부 복호화 해야 되는 차이 때문에 검색 시간이 더 걸린다. 하지만, 암호문을 가진 검색조건을 이용하여 암호문을 가진 결과를 검색하는 경우에서, 두 과정은 모두 블룸필터에서 두 키워드를 먼저 검색하기 때문에 긍정오류의 수를 포함한 같은 검색수가 나오며, 두 키워드를 암호화하여 재검색하는 제안방식과 튜플을 복호화하여 두 키워드를 재검색하는 튜플 암호화 방식의 최종 검색시간이 거의 비슷한 것을 확인하였다. 또한 위의 두 경우에서는 블룸필터의 크기가 클수록 블룸필터에 검색되는 키워드의 긍정오류 수가 적어지기 때문에 검색 시간이 줄어드는 것을 알 수 있다. 평문을 가진 검색조건을 이용하여 평문을 가진 결과를 검색하는 경우에서, 제안 방식은 조인문을 사용할 필요없이 한 테이블에서 검색하기 때문에 평문보다 조금 더 빨리 검색된다는 장점이 있다.

튜플 암호화 방식에서는 두 키워드를 확인할 수 없기 때문에 두 가지 튜플 암호화 방식의 모든 데이터를 복호화 한 후 키워드를 검색해야 하기 때문에 제안 방식에 비해 검색 시간이 매우 느리다는 단점을 있다. 그리고 제안 방식은 요소 단위로 암호화하는 방식이기 때문에 하나의 요소를 노출하더라도 다른 요소를 쉽게 노출할 수 없다는 장점이 있다.

따라서 본 논문에서 제안하는 방식은 민감한 데이터에 대해 보안기능을 유지할 수 있으며 검색성능을 향상시킬 수 있다. 향후에는 제안하는 방식을 클라우드 시스템에 있는 다양한 민감한 데이터에 대해 활용할 수 있도록 기대한다.


## 감사의 글

본 연구는 2014학년도 세명대학교 교내학술연구비 지원에 의해 수행된 연구임.


## REFERENCES


[1] B. LEE, “Comparison and Performance Evaluation of the Database Encryption Schemes”, *M.S. dissertation*, Seoul National University, 2013.

[2] H. Hacigü.mü.s, B. Iyer, Chen Li, and Shrad Mehrotra, “Executing SQL over encrypted data in the database service provider model”, *In Proc. of the ACM SIGMOD*, pp.45-49, 2002.

[3] J. Kim, T. Sahama, S. Kim, “*A* Performance test of Query Operation on Encrypted Database”, *LNCS 235*, pp 801-810, 2013.

[4] A. Broder and M. Mitzenmacher. Network applications of Bloom filters: A survey. *Internet Mathematics*, 1(4): 485-509, 2004.

[5] B. Chazelle, J. Kilian, R. Rubinfeld, and A. Tal, “The bloomier filter : an efficient data structure for static support lookup tables”, *in SODA '04: Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, (Philadelphia, PA, USA), pp.30-39, Society for Industrial and Applied Mathematics, 2004.

[6] S. Lee, H. Ahn, “An Efficient Group Key Management Scheme using Counting Bloom Filter in VANET”, *Journal of Convergence Security*, vol. 13, no. 4, pp. 47-52, 2013.



**양설(Xue Yang)**

2009년 세명대학교 정보통신학과 학사
2011년 세명대학교 전산정보학과 석사
2011년 ~ 세명대학교 정보통신학과 박사과정
※관심분야 : 정보보호, 데이터베이스 보안, 네트워크보안

**김지홍(Ji-hong Kim)**

1982년 한양대학교 전자공학과 학사
1984년 한양대학교 전자통신학과 석사
1996년 한양대학교 전자통신학과 박사
1991년 ~ 세명대학교 정보통신학부 교수
※ 관심분야 : 네트워크 보안, 데이터베이스 보안