# 데이터베이스 암호화 및 검색 시스템의 효율성에 관한 연구

이유정*, 박현아*, 변근덕*, 이동훈*

*고려대학교 정보보호대학원

# A Study for Efficiency of Database Encrypting and Searching Systems

YuJeong Lee*, HyunA Park*, KeunDuck Byun*, DongHoon Lee*

Graduate School of Information Security, Korea University

**요 약**

암호화 문서 검색 시스템은 사용자의 비밀키를 이용하여 데이터베이스 서버에 저장되어 있는 암호화 문서를 키워드를 이용하여 검색하는 시스템이다. 이러한 시스템은 데이터베이스 서버 관리자조차도 사용자들에 의해 저장되고 검색되는 문서에 대한 어떠한 정보도 노출시키지 않는 장점이 있다. 즉, 암호화 문서 검색 시스템은 일반 사용자들이 자신의 정보가 손쉽게 노출되는 것을 원치 않는 사회적 변화에 발맞추어 앞으로 지속적으로 발전할 수 있는 연구 주제이다. 하지만, 지금까지 제안된 대부분의 기법들은 이론적인 안전성에만 치우친 나머지 현실적인 적용 가능성이 고려되지 않았다. 이에 본 연구에서는 기존에 제안된 대표적인 기법들을 현실 상황에 맞게 구현하여 검색 시스템의 효율성에 대해 판단한다. 더불어 해당 결과를 바탕으로 앞으로 암호화된 검색 시스템이 나아가야 할 방향을 제시한다.

## I. 서론

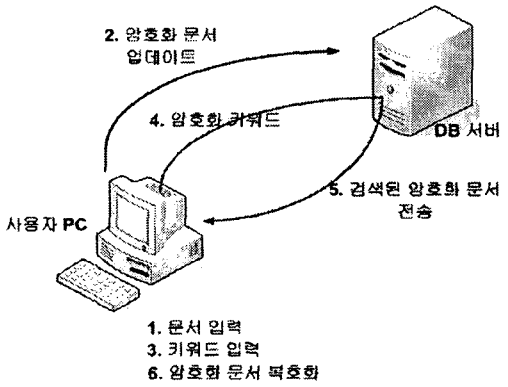
IT 기술 발전 속도가 급속해짐에 따라 점차 우리 사회도 유비쿼터스 사회로 변해가고 있다. 언제, 어디서나, 어떤 장비로도 손쉽게 네트워크에 접속하여 서비스를 제공받을 수 있는 유비쿼터스 시대는, 기존의 인터넷이나 도서관을 통해 제공받던 시절의 정보에 비해 양적이나 질적으로 많은 정보에 손쉽게 접근할 수 있는 이점이 있다. 이러한 경우, 사용자의 주민등록번호나 특정 웹 사이트의 사용자 계정과 비밀번호와 같은 민감한 정보가 기존의 환경에 비해 더욱 쉽게 악의적인 공격자 뿐 아니라 일반 사용자에게도 손쉽게 노출될 수 있다.

따라서, 다양한 정보가 저장되는 데이터베이스 자체의 보안에 대한 관심이 높아지고 있다. 이와 같은 맥락으로 현재 일부 데이터베이스에서 데이터의 암호화를 지원하고 있다. 하지만, 데이터베이스 관리자가 데이터의 내용을 모두 알 수 있는 문제점은 여전히 존재한다. 즉, 단순히 데이터베이스에서 지원되는 암호화 기능만으로 사용자들의 정보를 보호하는 것은 여전히 현실적으로 한계가 있다. 정당한 권한이 없는 사용자는 데이터베이스에 저장되는 문서 자체의 내용까지 알 수 없게 하는 데이터베이스 보안 기법이 필요하다.

## II. 데이터베이스 보안

데이터베이스 속에 저장된 사용자 정보 유출을 막는 가장 직접적이면서도 안정적인 데이터베이스 보안의 실현 방법은 데이터베이스에 저장될 문서/내용을 암호화하여 저장하고 필요할 때 마다 데이터베이스에서 해당 문서/내용을 찾아 복호화하여 문서를 보여주는 방법이다. 이러한 경우 단순히 문서를 암호화하여 저장시키는 작업 뿐 아니라 정확하고 효율적인 방법으로 암호화된 문서를 검색하는 작업과 해당 문서를 복호화하는 작업이 필요하다. 암호화 문서의 검색 시스템은 사용자가 자신의 개인키로 암호화

한 키워드를 사용자 자신의 비밀키를 이용하여 자료를 검색함으로써 데이터베이스 서버가 사용자의 정보를 알 수 없게 하는 [그림 1]과 같은 구조를 취하고 있다.




[그림 1] 암호화 문서 검색 시스템

지금까지 암호화 문서 검색 시스템에 관한 연구는 이론적인 안전성 및 증명가능한 안전성에 연구가 집중되어 있다. 하지만, 암호화 데이터베이스 및 검색 시스템의 필요성이 날로 증대하고 있는 지금, 기존의 이론적 안전성에 근거한 시스템들이 현실적으로 적용가능한지에 대한 효율성 검증은 이루어지지 않고 있다. 이에 본 연구에서는, 기존 연구 결과들을 실제 구현하고 현실적인 환경 및 제약사항을 고려하여 구현하고 해당 결과를 분석한다. 그리고 이를 통해 해당 기법들이 현실적으로 적용가능한지를 판단해본다. 그리고 이러한 결과를 통해 암호화된 데이터베이스 및 검색 시스템이 현실 상황에 적용되기 위해서 가져야하는 조건을 파악하고 효율적인 암호화된 데이터베이스 검색 시스템 설계를 위한 방향을 제시하고자 한다.

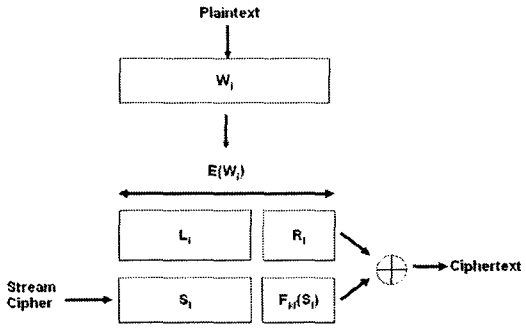
## III. 기존 시스템 분석

본 절에서는 효율성 분석을 위해 사용한 기존에 제안된 시스템들에 대해 간략히 살펴본다.

### 3.1 Song et al.의 기법[1]

해당 기법은 데이터베이스 서버를 신뢰하지 않는 환경을 고려하여 사용자에게 데이터의 무결성(integrity)과 안전성(security) 부여를 목적으로 한 최초의 암호화 문서 검색 기법이다.

제안된 기법은 순차적으로 문서 전체를 스캔하여 검색을 수행하는 방법을 취하며, 주어진 문서를 암호화하여 데이터베이스에 저장할 때 [그림 2]의 방법을 사용한다. 각각의 문서를 가장 긴 길이의 단어를 기준으로 하여 $n$개의 $W_i(|W_i| = mbit)$단어 블록으로 나눈다. 그리고 각 블록을 대칭키로 암호화한 뒤 $< L_i, R_i >$로 나눈다. 이때 $|L_i| = k, |R_i| = m - k$bit이다.




[그림 2] Song et al.의 문서 업데이트 방법

제안된 기법은 다음과 같은 순서로 검색을 실행한다. 먼저 해당 기법의 사용자는 검색하기 원하는 키워드가 특정 문서의 특정 위치에 나타날 것인지 미리 예측하여 검색을 수행한다. 검색을 원하는 키워드를 문서를 [그림 2]와 동일한 방법을 사용하여 암호화를 수행한다. 이 경우, 이미 사용자는 해당 키워드가 어느 위치에 나타날 것인지를 알기 때문에, 동일한 의사난수 수열을 생성할 수 있다. 그리고 특정 위치의 블록들을 해당 암호문과 비교하여 동일한 블록이 존재하는 문서들을 가져와서 사용자에게 복호화하여 보여준다.

### 3.2 Golle et al.의 기법[2]

Golle et al.의 기법은 한 번에 여러 개의 키워드를 포함한 암호화된 문서를 검색하는 방법(conjunctive search)과 키워드 필드(keyword field) 개념을 제안한 최초의 기법이다. 특히, 해당 기법은 비신뢰적인 메일 서버에 대해 이메일을 암호화하여 해당 서버를 이용하고, 필요할 때마다 사용자의 이메일을 검색을 통하여 사용

자에게 제공하는 환경을 가정하였다. [표 1]은 이메일이라는 환경을 최대한 고려하여 키워드 필드를 구성한 예이다. 문서 $D_i$는 $i$번째 문서의 식별자로서, 해당 문서는 서로 다른 $m$개의 키워드 필드로 구성되어있다. $W_{i,j}$는 $i$번째 문서 $D_i$의 $j$번째 키워드를 의미한다.

| | 송신인 | 수신인 | 날짜 | 제목 | … | 본문 |
|---|---|---|---|---|---|---|
| $D_1$ | 철수 | 영희 | 6/10 | 일정 | … | … |
| $D_2$ | 길동 | 둘리 | 6/12 | Null | … | … |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| $D_n$ | $W_{n,1}$ | $W_{n,2}$ | … | … | … | $W_{n,m}$ |

[표 1] Golle et al.의 키워드 필드 구성 예

제안된 기법은 기본적으로 검색을 수행할 때마다 전체 데이터베이스 필드를 다 검색하지 않고, 데이터베이스에서 저장 된 특정 값($g^{a_i}$)을 가져와서 검색 질의어(query) $R_i$를 생성한다. 그리고 $h(R_i)$와 $h(g^{a_i S})$가 동일하면 검색이 성공한 것으로 간주하여 해당 문서를 복호화하여 사용자에게 보여준다.

3.3 Waters et al.의 기법[3]

제안된 기법은 분산 환경의 Audit Log 파일을 안전하게 저장하기 위한 환경을 고려하였다. Audit Log는 특정 사용자가 시스템에 접근 한 시간 및 사용자의 시스템 사용 내역과 같은 정보를 쉽게 알 수 있는 특징이 있다. 즉, 사용자 보호를 위하여 권한이 없는 사용자들이 해당 파일에 손쉽게 접근할 수 없어야한다. 이에 제안하는 기법은 Audit log를 암호화하여 서버에 저장한 후, 권한이 있는 사용자에게만 검색할 수 있는 권한을 주어 암호화된 Audit log를 검색하고 복호화 할 수 있게 한다. 대칭키 기반과 공개키 기반의 두 가지 기법이 제안되었으며, 본 연구에서는 대칭키 기반 기법만을 살펴본다.

| $E_{k_i}(D_i)$ | $r_i$ | $H_{a_1}(r_i)$ | $H_{a_2}(r_i)$ | … | $H_{a_n}(r_i)$ |
|---|---|---|---|---|---|
| $E_{k_1}(D_1)$ | $r_1$ | $H_{a_1}(r_1)$ | $H_{a_2}(r_2)$ | … | $H_{a_n}(r_n)$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |
| $E_{k_n}(D_n)$ | $r_n$ | $H_{a_1}(r_n)$ | $H_{a_2}(r_n)$ | … | $H_{a_n}(r_n)$ |

[표 2] Waters et al.의 데이터베이스 테이블 구성

해당 기법은 크게 초기 설정(setup) 단계, 암호화 단계, 그리고 검색 및 복호화 단계로 나누어 살펴볼 수 있다.

▪ 초기 설정 단계 : Audit Escrow agent는 각각 서로 다른 $t$개의 비밀 값 $S_1, S_2, \cdots, S_t$를 생성한다. 여기서 $S_j$는 $j$번째 서버의 비밀 값이다.

▪ 암호화 단계 : 서버는 암호화된 audit log를 다음과 같이 생성한다.

$$< E_k(m), r, c_1, c_2, \cdots, c_n >$$

▪ 검색 및 복호화 단계 : audit log를 검색하고 싶은 사용자는 우선 audit escrow agent에게 정당한 사용자인지를 검증받는다. 사용자가 검색을 원하는 키워드 $w$를 audit escrow agent에 전송하면, agent는 해당 키워드에 맞는 capability인 $d_w = < H_{S_1}(w), \cdots, H_{S_t}(w) >$를 생성한다. $d_w^j = H_{S_j}(w)$는 $j$번째 서버의 capability를 나타낸다. 검색을 원하는 사용자는 agent로부터 넘겨받은 capability를 이용하여 $H_{d_w^j}(r)$을 계산하고 해당 값을 이용하여 복호화과정을 수행한 후 원하는 문서를 획득하게 된다.

## IV. 시스템 설정 및 실험 결과

본 절에서는 간략히 살펴본 기법들을 현실성을 고려하여 실제 구현한 과정과 그 결과에 대해 논한다.

4.1 시스템 설정

실험에 사용한 PC 환경과 실험 데이터 구성은 다음과 같다.

▪ 서버 및 클라이언트 : Intel Pentium Ⅳ 2.66GHz, 512RAM

▪ 서버 및 클라이언트 운영체제 : Win-XP

▪ 데이터베이스 : MS SQL 서버 2000

4.2 실험 데이터 집합 생성 및 설정

본 연구에서 효율성을 확인해보기 위해 실험하는 기법들은 모두 대칭키 암호 기반이다. 따라서 실험에서 사용하는 데이터 집합(data set)은 동일한 길이를 가진 23개의 블록(block)으로 이루어지도록 생성하였다. 데이터 집합 생성에 사용된 각각의 블록은 32byte(=32×16=256bits) 길이를 가진 난수이다. (즉, 23×32=736byte 길

이의 난수 23개를 각 문서는 포함한다.) 그리고 해당 문서를 검색하기 위한 키워드(검색어)는 편의를 위해 각 문서에서 처음부터 나타나는 블록 7개로 설정하였다. (각 문서가 몇 개의 검색어를 포함할 것인지는 해당 시스템의 정책에 따라 달라질 수 있다. 본 연구에서는 각 문서당 7개의 검색어를 포함한다고 가정하였다.) 실험에 사용된 전체 문서 수는 10,000개이다.

현실적으로 대부분의 검색 시스템에서 검색의 대상으로 여기는 문서는 동일한 키워드가 존재할 가능성이 있다. 따라서, 실험을 위한 데이터 집합 생성 시 10,000개의 블록을 순차적으로 사용함으로써, 전체 10,000개의 문서 안에 약 $435 = (10{,}000 \div 23)$번마다 중복되는 키워드가 포함되도록 설정하였다.

### 4.3 사용 암호화 알고리즘

본 연구에서 대상으로 하는 각각의 기법들은 Keyed Hash 함수, 대칭키 암호 알고리즘, 그리고 스트림 암호만을 사용하고 있다. 이에 본 연구에서는 현실적으로 가장 빠르고 안전하다고 알려진 AES-CBC-128을 사용하였다. 기본적인 연산 알고리즘은 OpenSSL[4]을 사용하였다. 또한 데이터베이스와 통신하고 문서를 전송하고 검색하기 위해 MS-SQL에서 제공되는 DB library for C[5]를 사용하였다.

### 4.4 데이터베이스 업데이트 시간 비교

기존의 기법들을 구현하여 암호화 문서와 검색을 위한 키워드를 데이터베이스에 전송하는데 걸린 시간을 살펴본다. 데이터베이스에 전송하는 문서의 수를 2,500개, 5,000개, 7,500개, 10,000개로 변화를 주면서 전송에 소요된 시간을 측정한 결과는 다음 [표 3]과 같다.

| 문서 수 | Song | Golle | Waters |
|---|---|---|---|
| 2,500 | 25,641 | 62,797 | 23,851 |
| 5,000 | 43,140 | 113,375 | 35,164 |
| 7,500 | 57,967 | 182,106 | 48,046 |
| 10,000 | 78,109 | 211,719 | 60,501 |

[표 3] 데이터베이스 업데이트 시간 비교(단위: ms)

Song et al.의 기법과 Waters et al.의 기법에 비해 Golle et al.의 기법이 문서를 데이터베이스에 업데이트하는데 상대적으로 매우 많은 시간이 소요된 것을 알 수 있다. 이는, Golle et al.의 기법이 다른 기법들에 비해 많은 지수연산을 포함하고 있기 때문으로 판단된다.

### 4.5 검색 시간 비교

데이터베이스에 저장된 문서를 키워드를 이용하여 검색할 경우 소요되는 시간을 살펴본다. 데이터베이스에 저장된 문서의 수를 2,500개, 5,000개, 7,500개, 10,000개로 변화를 주면서 검색에 소요된 시간을 측정한 결과는 다음 [표 4]과 같다. 이를 통해 검색을 위한 연산에서도 지수 연산이 존재하였던 Golle et al.의 기법이 가장 많은 시간이 소요됨을 알 수 있다. Song et al.의 기법과 Waters et al.의 기법은 상대적으로 Song et al.의 기법이 적은 시간이 소요된 것을 알 수 있다. 이것은 Song et al.의 기법이 검색을 하는 사용자는 특정 키워드가 나타날 위치를 알고 있다고 가정하였기 때문에 나타나는 결과로 판단된다. 하지만, 현실적으로 사용자가 데이터베이스에 저장한 모든 문서의 키워드 위치를 아는 것은 불가능하기 때문에, 현실적인 제약 사항이 감안된다면, Song et al.의 기법은 더 많은 시간이 소요될 것으로 예상된다.

| 문서 수 | Song | Golle | Waters |
|---|---|---|---|
| 2,500 | 47 | 8,328 | 79 |
| 5,000 | 62 | 16,422 | 157 |
| 7,500 | 109 | 24,453 | 204 |
| 10,000 | 147 | 32,454 | 297 |

[표 4] 검색 시간 비교 (단위: ms)

### 4.6 안전성 비교

본 절에서는 해당 기법들이 제시하는 안전성을 살펴본다. 안전성 평가 항목은 Song et al.에 의해 제시된 기법에 나타난 항목들이다.

| | Song | Golle | Waters |
|---|---|---|---|
| Model | 대칭키 | 대칭키 | 대칭키 |
| Controlled Search | ○ | ○ | ○ |
| Hidden Search | ○ | ○ | ○ |
| Query Isolation | X | ○ | X |
| Provable Security | PRG | Semantic Security | X |

[표 5] 안전성 비교

Controlled Search는 데이터베이스 서버는 사용자의 허락없이 검색을 실행할 수 없어야 한다는 것을 의미한다. Hidden Search는 데이터베이스 서버는 사용자가 던진 쿼리의 내용이 무엇인지 알 수 없어야 한다는 것이다. Query Isolation은 데이터베이스 서버가 검색 결과 외에 임의의 문서에 대한 어떠한 정보도 알 수 없어야 한다는 것이다.

## V. 효율성 분석 및 결론

본 절에서는 암호화 데이터베이스 검색 기법의 효율성을 높일 수 있는 방안에 대해 살펴본다.

본 연구에서 실험한 기존의 기법들은 평문 대상의 데이터베이스 검색 시스템에 비해 상대적으로 높은 안전성을 제공한다. 특히, Golle et al.의 기법의 경우 모든 안전성 항목을 만족하는 것을 알 수 있었다. 하지만 해당 기법의 경우, 암호화 문서의 업데이트 및 검색에 소요되는 시간이 다른 기법들에 비해 매우 길다는 사실 역시 알 수 있다. 이는 이론적인 안전성 추구에만 집중하여 현실적인 사항이 고려되지 않았기 때문으로 판단된다.

본 연구에서 실험을 통해 효율성을 살펴본 각각의 기법들은 현재 사용되고 있는 데이터베이스 정규화(Database normalization)를 적용하기 힘든 구조이다. 이것은 결국 데이터베이스 자체가 제공하는 기능을 전혀 이용하지 못하게 하는 결과를 가져오게 되어, 많은 검색 시간이 소요되게 된 것으로 판단된다.

암호화 문서 검색 시스템이 연구된 시간에 비하면, 평문 데이터베이스의 검색 시스템에 관한 연구는 오랜 시간에 걸쳐 이루어졌다. 본 연구를 통하여, 두 검색 시스템이 서로 다르다는 생각으로 접근하기 보다는 평문 검색 시스템에서 사용되었던 효율적인 데이터베이스 스키마들을 적용할 수 있는 암호화 문서 검색 시스템이 연구되어야 한다는 결론을 유출할 수 있다. 더불어, 기존의 효율적인 평문 검색 시스템의 데이터베이스 스키마를 최대한 사용할 수 있으면서도 안전성 측면을 보장할 수 있다면 해당 시스템은 현실적으로 가장 효율적이면서도 안전한 시스템이 될 것으로 예상된다.

## [참고문헌]

[1] Dawn Xiaodong Song, David Wagner, Adrian Perrig, Practical Techniques for Searches on Encrypted Data, Proceeding of IEEE Symposium on Security and Priavacy, pp.44-55, 2000

[2] Phlippe Golle, Jessica Staddon, and Brent Waters, Secure Conjunctive Keyword Search Over Encrypted Data, Proceedings of the Second International Conference on Applied Cryptography and Network Security (ACNS'2004), Lecture Note in Computer Science Vol.3089, pp.31-45, 2004

[3] Brent R. Waters, Dirk Balfanz, Glenn Durfee, and D.K.Smetters, Building an encrypted data searchable audit log, Proceedings of the 11th Network and Distributed System Security (NDSS) Symposium, pp.205-214, 2004

[4] http://www.openssl.org

[5] http://msdn.microsoft.com