한국산학기술학회논문지
Vol. 11, No. 5 pp. 1653-1660, 2010

# 암호화된 데이터베이스에서 인덱스 검색 시스템 구현

신승수[1*], 한군희[2]
[1]동명대학교 정보보호학과, [2]백석대학교 정보통신학부

# The Implementation of the Index Search System in a Encrypted Data-base

**Seung-Soo Shin[1*] and Kun-Hee Han[2]**
[1]**Dept. of Information Security, College of Information & Communication, Tongmyong University**
[2]**Division of Information & Communication Engineering, Baekseok University**

**요 약** 데이터베이스에 저장된 고객 정보들에 대한 유출 사례가 빈번히 발생하고 있다. 악의적인 목적을 갖고 있는 내부 관리자나 외부 공격자로부터 정보를 막기 위해서는 정보를 암호화하여 DB에 저장하는 것이 가장 효율적인 방법 중 하나이다. 암호화를 해 놓고선 DB에 저장만 하여놓고 다른 어떤 활용도 할 수 없다면 차라리 파기하는 편이 나을 것이다. 암호화된 DB 검색시스템이 다양하게 발전하고 있고 여러 분야에서 활용되고 있다. 본 논문에서는 모바일 디바이스에서 신뢰할 수 없는 서버에게 사용자의 정보를 노출하지 않고 암호화된 문서를 검색할 수 있는 스킴을 구현하고 비교분석을 하였다. 구현 결과를 대칭키 기반의 DES, AES, ARIA별로 검색시간을 비교 분석하였다.

**Abstract** The user information stored in database have been leaked frequently. To protect information against malevolent manager on the inside or outside aggressor, it is one of the most efficient way to encrypt information and store to database. It is better to destruct information than not to use encrypted information stored in database. The encrypted database search system is developed variously, and used widely in many fields. In this paper, we implemented the scheme that can search encrypted document without exposing user's information to the untrusted server in mobile device. We compared and analyzed the result embodied with DES, AES, and ARIA based on symmetric key by searching time.

**Key Words :** Database, Index,Pseudo-random permutation functions, Pseudo-random functionsDes, Aes, ARIA

## 1. 서 론

최근에 기업들은 데이터베이스에서에서 고객 정보에 대한 유출사례가 빈번히 발생하고 있고 외부 저장 공간에 저장된 정보에 대한 보안 문제가 이슈가 되고 있다[1, 2, 9]. 정보화 사회에서 그 정보들의 저장소인 데이터베이스의 관리는 무엇보다 중요하다. 정보의 홍수라는 말처럼 시시때때로 새로운 정보를 접하게 되며 이런 정보를 접하면서 그것의 정확성이나 정당성에 대해 의심을 품게 되는 경우가 많이 발생하게 된다. DB 관리의 소홀 및 내부자의 의도적인 시도로 인한 프라이버시 침해 사례가 많이 접수되고 있다. 특히, 이런 내부 관리자 및 외부 공격자에 의한 정보유출을 막고 민감한 정보 및 개인정보를 보호하기 위한 가장 확실하고도 현실적인 방법은 이러한 정보를 저장하는 데이터베이스의 암호화이다[3].

데이터 마이이닝에서의 프라이버시 보호하기(PPD M, Privacy Preserving for Data Mining)라는 주제에 관한 연구는 오래전부터 진행되어져 왔다. 이런 PPDM 연구의 또 다른 방법의 하나로 2000년에 D. Song et al. 이 "Practical techniques for searches on encrypted data"를 발표하면서 데이터베이스에 암호화 기법을 적용한 검색 시스템에 관한 연구의 활로를 열어 놓았다[4]. 기능적인 면에서 D. Song et al.의 연구 발표 후, 수많은 연구가 다양한 방면으로 진행되어져 왔다. 문서 전체를 암호화하여

*교신저자: : 신승수(shinss@tu.ac.kr)
접수일 10년 03월 19일 수정일 10년 04월 23일 게재확정일 10년 05월 13일

서버에 저장해 두고 암호화된 키워드로 질의하여 문서 전체를 검색하는 D. Song et al.의 최초의 스킴 이후, Chang과 Mitzenmacher[5]의 미리 만들어둔 사전(pre-built dictionary)을 이용한 두 가지 인덱스 스킴과 Goh[6]가 제안한 의사난수함수와 블룸필터를 사용한 Z-IDX라는 안전한 인덱스 검색 스킴 등 문서 전체를 검색하지 않고 문서에 대한 일종의 인덱스 형태를 사용하여 검색하는 스킴으로 변화해 왔다. Golle et al.은 키워드 필드라는 형태의 인덱스를 사용하여 동시적 검색(conjunctive search)을 가능하게 하였다[7].

D. Song et al.이 제안한 문서전체 검색 스킴은 데이터의 기밀성을 만족하면서 검색 기능을 지원하는 방법으로 비신뢰적인 서버에게 어떠한 정보도 노출시키지 않는 암호화 기법을 사용하는 것에 관한 시발적인 논문이다. 이것은 증명 가능한 안전성을 제공하며 제한된 검색, 은닉 검색, 질의 분리 기술을 가능하게 한다. 이러한 특징들은 향후 연구되어지는 암호화된 검색 시스템의 안전성에 관한 기술을 제공해 준다. 검색 시 서버가 문서 전체를 계산하는 양과 검색한 문서를 복호화 할 때 사용자의 계산양이 많이 필요하기 때문에 문서의 주요 키워드들을 뽑아 만든 인덱스나 키워드 필드를 사용하는 것이 훨씬 효율적이다[8, 10].

Song 기법에서는 문서의 양과 키워드의 수에 따라 데이터베이스의 저장 공간이 크게 증가하게 되는 문제점이 있다. 이를 보완하고자, Goh가 블룸필터(Bloom Filter)를 사용하는 검색 기법을 제안하였고, Chang은 서버에 사용자의 정보를 노출시키지 않으면서 서버로부터 저장된 데이터를 검색할 수 있는 모바일환경의 검색 기법을 제안하였다.

본 문에서는 모바일 디바이스에서 사용자의 정보를 노출하지 않고 암호화된 문서를 검색할 수 있는 스킴을 구현하고 실험을 하였다.

본 논문의 구성은 다음과 같다. 2장에서는 기존에 연구된 인덱스 검색스킴에 대해 알아보고, 3장에서는 인덱스 기반의 암호화된 문서에 대한 데이터베이스 스킴을 구현 및 실험을 하고, 4장에서는 대칭키 기반인 DES, AES, ARIA 암호모듈을 적용하여 암/복호화 속도와 검색속도를 비교분석 하였다. 끝으로, 5장에서 결론을 맺는다.

## 2. 관련 연구

비밀키를 가진 사용자가 자신의 데이터를 암호화해서 서버에 저장하여 검색하는 환경으로 대부분의 대칭키 암호를 이용한 검색기법이 여기에 속한다. 이 모델은 2000년에 Song 등에 의해 처음으로 연구가 시작되었다. 사용자가 자신의 문서들을 암호화하여 원격 파일 서버에 저장하고, 이후 언제 어디서나 특정 키워드를 포함하고 있는 암호화된 문서들을 효과적으로 검색한다[5]. 이 기법은 문서 설정단계와 문서 검색단계로 구성된다. 본 논문에서는 사용되는 표기법은 다음과 같다.

**[표 1]** 표기법

| 기호 | 정의 |
|---|---|
| $t$ | 안전성 파라미터 |
| $[n]$ | {1,2,…,n}, 만약 i∈[n] ⇔ 1≤i≤n |
| $I[i]$ | 인덱스 $I$ 의 $i$번째 비트 |
| $K \in \{0,1\}^t$ | 의사난수 집합 |
| $P_k(x)$ | $\{0,1\}^d \rightarrow \{0,1\}^d$ (의사난수치환함수) |
| $F_k(x)$ | $\{0,1\}^d \rightarrow \{0,1\}^t$ (의사난수함수) |
| $G_k(x)$ | $\{0,1\}^d \rightarrow \{0,1\}$ (의사난수비트생성기) |
| $w_\lambda$ | 키워드 |
| $M_j$ | 인덱스 |
| $E_j(m_j)$ | 암호화된 문서 |

Chang 프로토콜은 문서 설정단계와 문서 검색단계로 이루어지며, 각 단계별은 다음과 같다.

### 2.1 문서 설정단계

문서 설정단계는 사용자가 암호화된 문서를 서버에 저장하는 단계이다.

① 값, 값 $\in_R \{0,1\}^t$, P(), F(), G(), Dic = (갱, $w_i$), 갱 $\in [2^{\text{값}}]$를 계산한다. 사용자는 $s$, $r$을 $\{0,1\}^t$에서 비밀키를 선택한다.

② 문서 ; $m_1$, $m_2$, ..., 갯수 , ..., $m_n$, $1 \le$ 갱 $\le 2^{\text{값}}$에 대해서, 사용자는 $2^{\text{값}}$ 비트 인덱스 스트림 $I_j$를 계산한다.

$$I_j = \begin{cases} 1 & \text{if } w_i \in m_j \\ 0 & \text{if } w_i \notin m_j \end{cases}$$

여기서, $m_j$가 키워드 $w_i$를 갖고 있는 경우 사용자는 $i$번째 키워드 $w_i$의 인덱스 $i$를 치환시킨 값 $P_s(i)$가 $j$번째 문서에 존재하면 $I_j[P_s(i)] = 1$로, 그렇지 않은 경우 $I_j[P_s(i)] = 0$으로 정의한다.

③ 사용자는 $M_j[i] = I_j[i] \oplus G_{r_i}(j)$ 계산한다.

$r_i (= F_r(i))$는 $w_i$의 인덱스 $i$를 의사난수함수

$F$ 에 대응시켜 나온 값 $F_r(i)$이다.

④ 사용자는 서버에게 $M_j$와 $E_j(m_j)$를 보낸다.

⑤ 사용자는 비밀키 $r$, $s$ 그리고 사전을 자신의 모바일디 바이스에 복사해둔다.

사용자가 문서를 서버에 저장하기 위해 문서 설정단계 ③에서 계산된 $M_j$와 $E_j(m_j)$를 서버에 보내고 사용자는 비밀키 s, r을 모바일 디바이스에 저장한다.

### 2.2 문서 검색단계

문서 검색단계는 사용자가 서버에 저장한 문서 중에 특정 키워드를 가지고 있는 문서를 검색하는 단계이다.

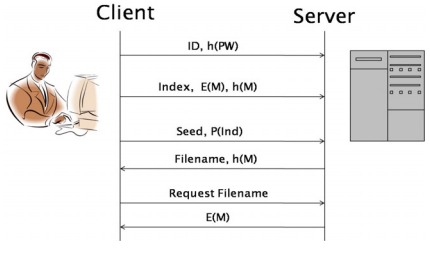
① 키워드 $\lambda$를 가진 파일을 검색하기 위해서 $p = P_s(\lambda)$와 $f = F_r(p)$를 이용해서 사용자의 사전으로부터 $\lambda$에 대응하는 파일을 찾아 보내준다.

② 서버는 $I_j[p] = M_j[p] + G_f(j)$를 각 문서마다 계산해서 $I_j[p] = 1$이면, 서버는 사용자에게 $E_j(m_j)$를 보낸다.

사용자가 저장한 문서를 검색하기 위해 문서 검색단계 ①에서 계산된 p, f를 서버에게 전송하면 서버는 문서 검색단계 ②와 같이 인덱스를 검증하여 검색된 암호 문서를 사용자에게 전송한다.

## 3. 구현 및 실험

인덱스 기반의 암호화된 데이터베이스 검색 기법은 개인정보 등록 단계, 로그인 단계, 파일 저장 단계, 파일 검색 단계로 이루어진다. 이러한 프로토콜에 대한 전반적인 흐름을 그림 1에 나타냈다.




[그림 1] 전체 프로토콜

### 3.1 개인정보 등록단계

개인정보 등록단계는 사용자가 서버를 사용할 때 한번만 수행하는 단계이다. 사용자는 서버에게 개인정보를 등록하기 위해 ID, Password, name, e-mail 등을 서버에게 보낸다. 이 때, Password는 네트워크에 노출시키지 않기 위해 해쉬함수를 적용한다. 개인정보 등록단계는 서비스를 이용하기 위해 사용자가 서버에게 개인정보를 등록하는 단계이며, 개인정보 등록단계는 다음과 같은 과정에 의해 서버에 등록한다.

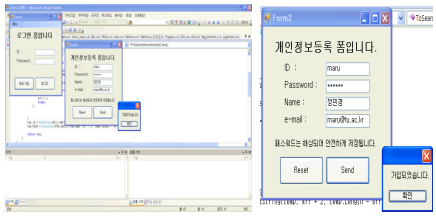
① 사용자는 ID, Password, name, e-mail을 입력한다.

② 서버는 사용자로부터 받은 ID의 중복여부를 확인한다.

③ 사용자는 Client 프로그램으로 h(PW)를 계산하고 서버에게 전송하면, 서버는 사용자의 개인정보를 등록한다.

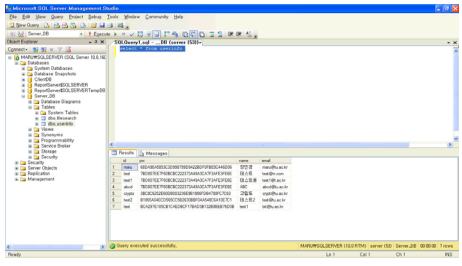
④ 서버는 사용자에게 가입여부를 알려준다.

개인정보 등록 폼에서 ID, Password, name, e-mail을 입력하면, 서버는 ID의 중복여부를 체크한다. ID가 중복되지 않으면 사용자는 ID, h(PW), name, e-mail를 서버에게 전송하고, 서버는 가입승인 여부에 관한 과정을 C#을 이용해서 실행한 결과를 그림 2에 나타냈다.




[그림 2] 개인정보 등록 단계

사용자들이 보낸 ID와 h(PW)등의 개인정보는 서버의 DB에 저장되는데, 이 때 사용자의 Password는 그림 3과 같이 Password의 값이 저장되는 것이 아니라, Password를 해쉬함수를 취한 값인 Hash Code가 서버의 데이터베이스에 저장되므로 비신뢰적인 서버나 네트워크상에서 노출되지 않기 때문에 안전하다.

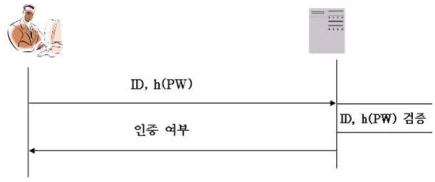


[그림 3] 서버의 개인정보 DB

### 3.2 로그인 단계

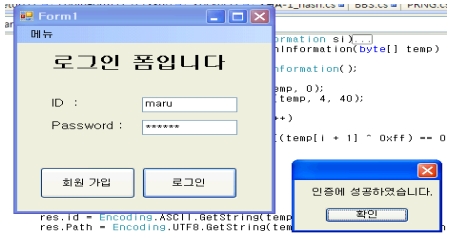
다음은 사용자가 서비스를 이용하기 위해 서버에 로그인하는 단계이다. 사용자는 그림 4와 같은 과정을 통해 서버로부터 인증을 받는다.

① 사용자는 ID와 PW를 입력한다.
② 사용자의 Client 프로그램은 h(PW)를 계산하고, 서버에게 ID, h(PW)를 전송한다.
③ 서버는 ID, h(PW)를 검증한다.
④ 서버는 사용자에게 인증 여부를 알려준다.

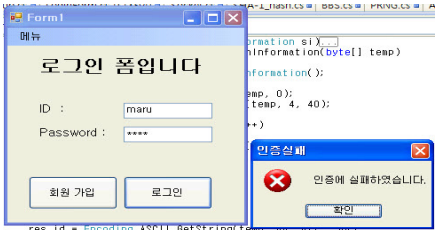



**[그림 4]** 로그인 단계

다음은 C#을 이용한 인증 성공 여부를 보여주는 과정이다. 먼저, 사용자는 로그인 폼에 ID와 Password를 입력하여 서버에게 전송한다. 서버는 ID와 Password를 검증하여 인증에 성공하면 그림 5와 같이 성공 메시지를 띄우고, 인증에 실패하면 그림 6과 같이 실패 메시지를 띄운다.




**[그림 5]** 인증 성공




**[그림 6]** 인증 실패

### 3.3 파일 저장단계

파일 저장단계는 사용자가 서버에게 암호화된 파일을 저장하는 단계이다.

① 사용자는 여러 문서 $m_j$를 만든다.
② 각 문서마다 대표적인 키워드 $w_i$ (1~3개)를 정하고, 키워드 목록을 사용자 DB에 저장한다.
③ 각 문서 $m_j$의 인덱스 $M_j[i] = I_j[i] \oplus Gr_i(j)$를 계산한다.
④ 각 문서 $m_j$를 암호화한 $E_j(m_j)$과 ③에서 계산된 인덱스 $M_j$를 서버에 전송한다.
⑤ 서버는 인덱스 $M_j$와 암호화된 문서 $E_j(m_j)$를 DB에 저장한다.

#### 3.3.1 Index 생성

인덱스 생성 과정은 다음과 같다.

① 치환 배열을 만든다.
② 소수와 초기값을 정한다.
③ 각 문서를 키워드 목록과 비교하여 비트열을 생성한다.

$$I_j = \begin{cases} 1 & \text{if } w_i \in m_j \\ 0 & \text{if } w_i \notin m_j \end{cases}$$

④ 생성된 비트열 $I_j$에 의사난수 치환함수 P( )를 적용한다. 의사난수 치환함수는 $P_k(x) : \{0, 1\}^d \rightarrow \{0, 1\}^d$ 이다.
⑤ 4단계의 값을 비트생성기 G( )를 이용하여, 최종 인덱스값 $G_{r_i}(j)$을 생성한다. 비트생성기 $G_k(x) : \{0, 1\}^d \rightarrow \{0, 1\}$ 이다.

#### 3.3.2 파일 저장 단계

사용자는 파일 저장 단계에서 각 문서의 키워드 목록을 뽑아내고, 치환배열을 통해서 서버에 보낼 인덱스를 다음과 같은 과정을 통해서 보여준다.

1) 구성요소

사용자 데이터베이스에 있는 문서로부터 키워드를 선택하여 키워드 목록에 저장하고, 만약 키워드 목록이 8개이면, 문서 $m_1$에 있는 키워드를 8자리의 비트로 나타내고자 할 때, $w_i$에 속하면 1로 나타내고, 속하지 않으면 0으로 나타낸다. 8자리의 비트를 치환배열을 통해 치환시킨다.

2) Index 생성

① 문서 m1이 포함한 키워드를 계산한 값은 10100000이다.

② 치환배열을 이용하여 1)에서 계산된 값을 치환한 값은 11000000이다.

③ BBS를 이용하여 키워드의 수만큼 비트열을 다음과 같이 생성하였다. 먼저, 서로 다른 소수 2개를 p=7, q=11 선택하고, 초기값은 2로 설정하였다. 다음은 키워드 목록의 개수만큼 의사난수 비트를 생성한 것이다. 의사난수 비트는 (초기값 * 초기값 mod p*q) mod 2 = 0 또는 1 와 같이 계산하고, 그 결과는 00110011로 생성된다.

④ 3)에서 생성된 비트열과 2)에서 계산된 값을 XOR 비트 연산하여 Index 계산하면, 11000000 ⊕ 00110011 = 11110011 이다.

3) 사용자는 ID, 11110011, E(m1), h(m1)를 서버에게 전송한다.

### 3.4 파일 검색 단계

사용자는 파일 검색 단계에서 찾고자 하는 문서의 키워드를 키워드목록을 이용하여 비트열을 생성하고 치환배열을 적용하여 서버에게 전송한다. 서버는 받은 중간 인덱스 값과 BBS정보를 이용하여 인덱스를 계산하는 과정을 다음과 같이 보여준다.

① 구성 요소

· 서버 DB : ID, 11110011, E(m1), h(m1)

· 키워드 목록 : w1: 정보보호, w2: 보안, w3: 암호, w4: 공개키, w5: RSA, w6: 시스템, w7: 전자서명, w8: 대칭키.

· 치환 배열 : ( 3 1 4 5 2 8 6 7 )

② 사용자는 키워드를 입력하고 인덱스 중간값을 계산한다.

1) 사용자가 '암호'라는 키워드를 검색하면, 키워드 목록을 비교하여 8자리의 비트열 00100000를 얻을 수 있다.

2) 1)에서 계산된 비트열을 치환배열을 이용하여 비트열을 치환하면 10000000를 얻을 수 있다.

3) 사용자는 10000000, n=77과 x=2를 서버에게 전송한다.

③ 사용자는 서버에게 uID, 10000000과 seed(77,2)를 전송한다.

④ 서버는 uID, 10000000과 seed(77,2)를 이용하여 다음과 같은 과정으로 문서를 검색한다.

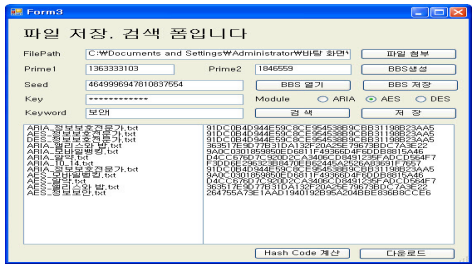
1) BBS계산

2) XOR 연산 : 11110011 ⊕ 0000000 = 11110011

3) 문서 m1은 '암호'라는 키워드를 포함하고 있다.

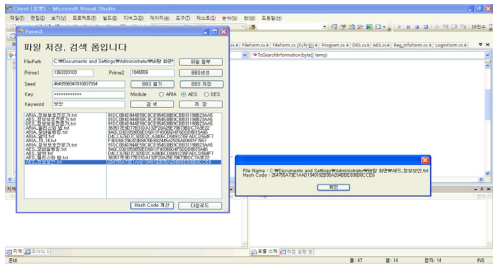
⑤ 서버는 사용자에게 검색된 암호화된 문서 E(m1)와 사용자가 저장했던 문서의 Hash Code인 h(m1)을 전송한다.

그림 7은 사용자가 검색 폼에서 키워드를 사용하여 검색하면 서버는 중간 인덱스 값을 계산하여 키워드를 포함하는 파일과 파일에 대한 Hash Code를 사용자에게 전송하는 과정을 보여준다.




[그림 7] 파일 검색 단계

다음은 사용자가 서버에 암호화되어 저장된 문서 중에서 찾고자 하는 문서를 다운 받은 후, 문서가 변경이 되었는지 아닌지를 확인하기 위해서는 사용자가 다운 받은 문서의 Hash Code를 계산한 값과 서버에 암호화되어 저장된 문서의 Hash Code값을 비교하면 받은 문서가 변경되거나 손실되었는지를 그림 8과 같이 확인할 수 있다.



[그림 8] Hash Code 검증

## 4. 구현 결과 및 고찰

### 4.1 구현환경

인덱스 기반의 암호화된 데이터베이스 검색 시스템을 암호모듈 DES, AES, ARIA별로 암호화 속도, 복호화 속도, 검색 속도의 효율성을 비교 분석하기 위해서 데이터

베이스를 구축하고, 100개의 문서와 32개의 키워드를 만들었다. 구현 방법은 등록 단계, 로그인 단계, 문서 저장 단계, 문서 검색 단계로 나누어진다. 등록 단계는 사용자가 개인정보를 입력하고 인덱스 기반의 암호화된 데이터베이스 검색시스템을 구현하기 위해서 PC 환경은 다음과 같다. Local Server 운영체제는 Win-XP, Database 서버는 MS SQL Server 2008, PC 사양은 Client - Pentium(R) Dual T2370, Server - Pentium(R) 4 3.00GHz을 사용하였다.

먼저, 문서를 암호화하기 위해서 각 문서들은 대칭키 기반 DES-ECB-64, AES-ECB-128와 ARIA-ECB- 128을 이용하여 암호화하였다.

본 구현에서는 각 문서 당 1~3개의 키워드를 포함한다고 가정하였다. 구현에 사용된 전체 키워드 수는 32개, 전체 문서 수는 100개이다. 표 2는 인덱스 기반의 암호화된 데이터베이스에서 검색 기법을 구현하기 위한 구현 환경이다.

**[표 2]** 구현 환경

| | Server | Client |
|---|---|---|
| OS | Windows XP sp3 | Windows XP sp3 |
| RAM | 1.00GB | 1.75GB |
| Process | Pentium(R) 4 3.00GHz | Pentium(R) Dual T2370 |
| Database | Microsoft SQL 2008 | Microsoft SQL 2008 |
| Framework | C# Framework 3.5 sp1 | C# Framework 3.5 sp1 |

## 4.2 구현 결과 비교

본 논문에서 구현한 암호화된 데이터베이스 검색기법은 서버, 사용자와 같이 두 개체로 나누어져 있으며, 사용자는 모바일 디바이스를 사용하는 모델이다. 그러나 모바일 디바이스는 일반적인 컴퓨터에 비해 제한적인 8비트의 경량화된 시스템이다. 따라서 본 논문에서는 AES, DES, ARIA와 같은 암호모듈을 8비트의 경량화된 시스템에 맞게 구현하고 각각에 대한 암·복호화에 대한 시간 분석과, 검색시간을 측정한다. 본 실험에서 암·복호화는 사용자가 문서를 저장하고 문서를 수신하는 과정에서 암·복호화 연산시간만을 각각 파일에 대해 적용하여 실험한 측정값들 중에 다수의 비슷한 시간을 추출하여 평균값을 측정하였다. 검색시간은 여러 사용자를 등록하여 각 사용자에 대하여 파일을 분산시키고, 로그인한 사용자가 키워드를 이용해 검색하고자 할 때, 서버의 데이터베이스에 저장된 모든 사용자들의 문서의 수를 다르게 하였다. 검색시간은 사용자가 키워드를 입력하고 서버로부터 검색된 암호문서의 목록을 수신하는 과정까지 포함한다.

### 4.2.1 모듈별 암호화 분석

저장단계 중에서 DES, AES, ARIA 암호 모듈을 사용하여 암호화하는 과정을 문서의 용량별로 효율성을 비교하였다. 각 암호모듈의 암호화 시간을 비교하기 위해 저장단계에서 각 문서를 암호화하는 시간만을 측정하며, 사용자가 서버로 파일을 송신하는 과정은 암호화 분석시간에 포함하지 않았다.

1kb의 문서에서는 세 모듈의 속도가 비슷하였고, 50kb 이상의 문서에서는 ARIA의 암호화 속도가 가장 효율적이었다. 표 3은 크기별 각각 5번 실행한 결과를 세 모듈별로 분석한 결과로서, DES는 문서 크기가 50kb당 약 250ms가 증가하였고, AES는 약 400ms가 증가하였다. 반면에, ARIA는 약 120ms가 증가하여 암호화된 DB 검색기법의 저장단계에서 가장 효율적이다.

**[표 3]** 암호화된 시간 비교

(단위 : ms)

| 모듈 / 크기 | DES | AES | ARIA |
|---|---|---|---|
| 1kb | 10 | 13 | 16 |
| | 10 | 9 | 13 |
| | 6 | 14 | 5 |
| | 7 | 12 | 3 |
| | 7 | 11 | 7 |
| 50kb | 254 | 404 | 118 |
| | 242 | 412 | 131 |
| | 244 | 409 | 119 |
| | 251 | 403 | 121 |
| | 242 | 406 | 122 |
| 100kb | 487 | 836 | 237 |
| | 505 | 808 | 242 |
| | 487 | 810 | 240 |
| | 489 | 807 | 241 |
| | 482 | 812 | 239 |

### 4.2.2 모듈별 복호화 분석

검색단계 중에서 세 가지 암호모듈을 사용하여 복호화한 과정을 문서 용량별로 효율성을 비교하였다. 각 암호모듈의 복호화 시간을 비교하기 위해 검색단계에서 각 문서를 수신하고, 복호화하는 시간만을 측정한다. 이때, 사용자가 문서를 요청하고 서버로부터 문서를 수신하는 과정은 복호화 분석시간에 포함하지 않았다.

1kb의 문서에서는 세 모듈의 속도가 비슷하였고, 50kb 이상의 문서에서는 ARIA의 암호화 속도가 가장 효율적이었다. 표 4는 크기별 각각 5번 실행한 결과를 세 모듈

별로 분석한 것으로서, DES는 문서 크기가 50kb당 약 260ms가 증가하였고, AES는 약 430ms가 증가하였다. 반면에, ARIA는 약 130ms가 증가하여 암호화된 DB 검색 기법의 검색단계에서 가장 효율적이다.세 모듈을 분석한 결과 DES, AES, ARIA 암호모듈은 암호화보다 복호화의 속도가 약 10~20ms 떨어졌다.

**[표 4]** 복호화된 시간 비교

(단위 : ms)

| 모듈 / 크기 | DES | AES | ARIA |
|---|---|---|---|
| 1kb | 2 | 3 | 3 |
| | 3 | 2 | 2 |
| | 2 | 4 | 2 |
| | 3 | 9 | 4 |
| | 7 | 11 | 4 |
| 50kb | 278 | 432 | 122 |
| | 265 | 432 | 124 |
| | 244 | 432 | 120 |
| | 245 | 427 | 121 |
| | 243 | 425 | 140 |
| 100bk | 485 | 851 | 240 |
| | 485 | 858 | 239 |
| | 489 | 851 | 237 |
| | 491 | 865 | 238 |
| | 482 | 865 | 254 |

#### 4.2.3 모듈별 검색 분석

사용자가 세 모듈을 사용해 암호화된 문서들을 각각 10개, 50개, 100개씩 저장하고, 서버로부터 암호화된 문서를 검색한 시간을 비교하였다. 검색시간은 사용자가 키워드를 입력하고 서버로부터 문서의 목록을 수신하는 과정과 오류제어를 모두 포함한다. 모바일 디바이스를 이용하였을 경우, 모바일 디바이스는 일반적인 컴퓨터 클라이언트 환경에 비해 무선 통신이 취약하기 때문에, 서버와의 라우팅 거리나 모바일 디바이스의 무선 통신환경에 따라 검색시간에 차이가 있을 수 있다.

분석 결과는 표 5에서처럼 저장된 문서가 10개일 때 약 4ms, 50개일 때 약 6ms, 100개일 때 약 8ms가 걸리는 것으로 나타났다. 암호모듈은 검색 시간에 영향을 주지 않으며, 데이터베이스에 저장된 문서 수가 많을수록 검색시간이 더 걸린다. 그러나 검색단계에서는 검색한 키워드가 포함된 문서가 적을수록 서버와 클라이언트 간의 송수신 데이터가 적기 때문에 검색시간이 줄어든다. 표 5에서 1이라는 숫자는 키워드가 문서에 포함되지 않았을 때를 의미한다.

**[표 5]** 검색 시간 비교

(단위 : ms)

| 모듈 / 문서 수 | DES | AES | ARIA |
|---|---|---|---|
| 10개 | 4 | 4 | 5 |
| | 5 | 5 | 5 |
| | 2 | 3 | 4 |
| | 3 | 4 | 3 |
| | 4 | 4 | 4 |
| 50개 | 4 | 4 | 122 |
| | 5 | 5 | 124 |
| | 8 | 7 | 120 |
| | 1 | 1 | 1 |
| | 3 | 3 | 3 |
| 100개 | 15 | 7 | 7 |
| | 9 | 8 | 11 |
| | 8 | 7 | 7 |
| | 5 | 4 | 5 |
| | 1 | 1 | 1 |

## 5. 결론

암호화된 DB에서 대칭키 기반 검색 기법의 효율적인 구현을  DES, AES, ARIA 암호모듈을 적용한 저장단계 및 검색단계를 비교분석하였다.

저장단계에서는 문서를 송·수신을 제외하고 각 암호모듈을 적용하여 문서를 암호화하는데 걸린 시간만을 측정하여 암호 모듈의 암호화 시간을 분석했다. 분석 결과 ARIA의 암호화 시간이 가장 효율적이었다. 그러나 경량화된 모바일 디바이스에서는 모바일 디바이스의 환경과 구조에 따라 다소 차이가 있을 수 있다. 검색단계에서는 서버와 사용자의 문서 송·수신을 제외하고 사용자가 문서를 수신하여 문서를 복호화하는 시간만을 측정하였다. 세 암호 모듈은 암호화에 비해 복호화가 10~20ms정도 시간이 지연되었으며, ARIA의 복호화 시간이 가장 효율적이었다. 검색시간은 사용자가 키워드를 입력하고 문서목록을 수신하는데 걸리는 시간을 측정하였으며, 모바일 디바이스의 특성에 따라 클라이언트를 무선 암호통신으로 구축하였다. 무선 암호통신은 송·수신하는 파일이 커질수록 오류를 제어하는 시간이 길어지고 검색시간이 달라진다. 따라서 모바일 디바이스를 이용하여 무선 암호통신을 할 경우, 서버와 모바일 디바이스의 라우팅 거리와 무선 통신환경에 따라 검색시간의 다소 차이가 날 수 있다.

암호화된 데이터베이스에서의 검색기법은 이메일, 검색엔진, 개인정보관리 등과 같은 정보보호를 필요로 하는 다양한 응용분야에서 폭넓게 사용될 수 있다.

## 참고문헌


[1] Byunghee Lee, yunho Lee, Seokhyang Cho, Seungjoo Kim, Dongho Won, "A Study on the Keyword Search on Encrypted Data using Symmetric Key Encryption," 한국정보보호학회 정보보호학술대회논문집, 2006.

[2]「민관겸용 블록 암호 ARIA 알고리즘 명세서」, 2004.

[3] 김선영, 서재우, 이필중, "검색 가능 암호 기술의 연구 동향", 한국정보보호학회, 2009.

[4] D. Song, D. Wagner, and A. Perrig, "Practical techniques for searches on encrypted data," In Proceedings of IEEE Symposium on Security and Privacy, pp. 44-55. IEEE, May 2000.

[5] Y. C. Chang and M. Mitzenmacher, "Privacy preserving keyword seaches on remote encrypted data," Cryptology ePrint Archive, Report 2004/051, Feb 2004.

[6] P. Golle, J. Staddon, B. Waters, "Secure Conjunctive Keyword Search over Encrypted Data," Proc. of the 2004 Applied Cryptography and Network Security Conference, 2004.

[7] 이동훈, "계층적 그룹 DB에서 암호화 기법을 이용한 효율적인 검색 시스템의 설계 및 구현", 정보통신산업진흥원, 2006.

[8] 노건태, 정익래, 이동훈, 변진욱, "암호화된 데이터에서의 연산 기술 분류 및 최근 동향 연구", 한국정보과학회 학술발표논문집, 2008.

[9] 조상록,"하나로텔ㆍ옥션 개인정보 유출사건, 집단분쟁조정 개시", http://www.itdaily.kr/news/articleView.html?idxno =16119 , IT데일리, 2008.

[10] 정미경, 송희정, 신승수, 한군희, "암호화된 DB에서 대칭키 기반 검색기법 구현", 한국산학기술학회 춘계학술대회발표논문, 2009.



**신 승 수(Seung-Soo Shin)** **[정회원]**

- 2001년 2월 : 충북대학교 수학과 (이학박사)
- 2004년 8월 : 충북대학교 컴퓨터공학과 (공학박사)
- 2005년 3월 ~ 현재 : 동명대학교 정보보호학과 교수

<관심분야>
암호프로토콜, 무선 PKI, 네트워크 보안, USN, 스마트카드,

**한 군 희(Kun-Hee Han)** **[종신회원]**

- 2008년 8월 ~ 현재 : 백석대학교 정보통신학부 교수

<관심분야>
암호프로토콜, 네트워크 보안, 영상처리