논문 2008-45CI-3-4

암호화 데이터베이스에서 영역 질의를 위한 기술

(Technique of Range Query in Encrypted Database)

김 천 식*, 김 형 중**, 홍 유 식***

(Cheonshik Kim, Hyoung Joong Kim, and You-sik Hong)

요 약

최근 들어 개인 정보 보호의 중요성에 대한 인식이 매우 높아지고 있다. 많은 국가에서 개인정보 보호에 관련한 법규를 새로이 제정하고 있으며, 이제 개인 신상에 관련된 데이터를 보호하는 것은 단순한 기업 이미지 관리 차원이 아닌 법적인 의무가 되었다. 대부분의 기업의 세일즈 데이터베이스에는 예외 없이 고객의 이름, 주소, 신용카드번호와 같은 정보가 저장되어 있다. 개인 정보는 개인 신상에 관한 민감한 정보이고, 또한 기업의 전략적인 자산이며, 따라서 기업은 개인 정보를 사전 예방차원에서 안정적으로, 그리고 포괄적으로 보호하기 위한 모든 노력을 다해야 한다. 그러나 만일 데이터베이스 관리자의 패스워드 보안이 뚫린다면 속수무책일 수밖에 없다. 이를 위해서 데이터베이스 암호화는 반드시 필요하다. 그러나 암호화의 결과 데이터베이스의 성능에 문제가 될 수 있고, 또한, 암호화로 인하여 기존의 SQL 언어에서의 영역(range) 질의에 제한이 있다. 따라서 이와 같은 문제를 해결하여 암호화된 데이터를 효과적으로 질의하기 위한 방법을 본 논문에서 제안하였다.

Abstract

Recently, protection of personal information is getting more important. Many countries have legislated about the protection of personal information. Now, the protection of relevant personal information is required not for a simple image of enterprises but law obligation. Most databases in enterprises used to store customers' names, addresses and credit card numbers with no exceptions. The personal information about a person is sensitive, and this asset is strategic. Therefore, most enterprises make an effort to preserve personal information safely. If someone, however, hacks password information of DBMS manager, no one can trust this system. Therefore, encryption is required based in order to protect data in the database. Because of database encryption, however, it is the problem of database performance in terms of computation time and the limited SQL query. Thus, we proposed an efficient query method to solve the problem of encrypted data in this paper.

Keywords: encryption, data, database, query, AES

Ⅰ. 서 론

최근에 개인 정보가 대량으로 수집·활용됨에 따라, 수집된 개인 정보의 불법적인 접근 및 유출 등에 대한 우려가 증가하고 있다. 하지만 현재의 개인정보 이용 환경은 데이터 접근 시에 사용자의 질의 내용과 그에 대한 결과가 관리자에게 그대로 노출되어 사용자의 프라이버시가 침해되는 치명적인 문제로 지적되고 있다. 한국의 엔씨소프트는 2004년 5월 11일 '리니지2'를 업데이트하면서 사용자의 게임정보를 담은 로그 파일을 암호화하지 않아 8,500명의 게임 이용자 개인정보가 노출되어 소송이 진행되고 있으며, 2005년 리니지 게임 대규모 명의 도용사건과 관련해 28만 명의 명의가 도용되어 사고 책임자가 입건되었다[14]. 최근에 중국에서 국내의 게임회사 및 투자회사를 대상으로 빈번하게 해킹을 하고 있다. 이들은 국내의 개인정보가 돈이 되는 것을 알고 데이터베이스에 저장된 데이터를 획득하여 기업을 대상으로 거래를 시도하고 있다. 이와 같은 문제를 방

* 정회원, 안양대학교 디지털미디어공학전공
(Major in Digital Media Engineering, Anyang University)
** 정회원, 고려대학교 경영정보대학원
(Graduate School of Information Management and Security, Korea University)
*** 정회원, 상지대학교 컴퓨터공학과
(Dept. of Computer Science, Sangji University)
접수일자: 2008년4월26일, 수정완료일: 2008년5월6일

지하기 위해서 데이터베이스에 저장된 데이터를 암호화된 형태로 저장하는 것에 대한 많은 연구가 진행되고 있다[1~2, 4~5, 9]. 암호화 방식은 DBMS 내 주요 데이터마다 암호를 설정한다. 침입자가 네트워크를 뚫고 DBMS에 들어와도 암호를 풀지 못하면 데이터를 가져갈 수 없고, 혹 가져간다 해도 열람할 수 없게 한다는 것이다. 이와 같은 장점에도 불구하고 암호화 방식이 시스템 성능을 저하시킨다는 문제로 많은 논쟁거리가 되고 있다[1~2, 8]. 이것은 DBMS 자체에 모듈을 탑재하기에는 과부하 문제가 될 수 있다고 업체들은 말하고 있다. 특히 데이터를 빨리 검색하기 위해 설치된 인덱스에도 암호화가 걸리면서 프로세스가 느려지는 문제가 발생한다는 지적이 있다[9].

데이터베이스 암호화는 크게 다음과 같이 분류할 수 있다.

■ 모든 데이터베이스에 대한 파일 수준 암호화: 데이터베이스 파일들을 도난으로부터 보호한다. 이 연구는 도입이 간단하고 경제적이지만 많은 약점을 가지고 있다. 파일시스템에 암호화가 적용될 때, 어떤 인증된 사용자라도 모든 데이터를 볼 수 있다. 또, 파일 수준 암호화는 아무리 간단한 데이터베이스 처리 요청이라도 암호화나 복호화가 이루어져야 하기 때문에 성능에서 중대한 문제를 가지고 있다. 파일 수준 암호화는 단지 저장된 데이터만을 보호한다[2, 10].

■ 어플리케이션 수준 암호화: 데이터를 암호화하여 데이터베이스에 저장하고 암호화 키 관리와 접근관리가 어플리케이션에 의해 이루어진다. 데이터베이스에 질의한 후 결과 값은 암호화된 데이터로 반환되어 어플리케이션에서 복호화 된다. 이 연구는 데이터가 획득에서 저장까지 안전한 데이터에 대한 역할 기반 접근을 가능하게 한다. 이 연구의 가장 주된 결점은 비용의 문제이다[1, 3~7, 9].

어플리케이션의 암호화 방법의 문제점은 근사 질의와 영역 질의가 가능하지 못하다는 것이다. 예를 들면 'tom'으로 시작하는 모든 사람의 정보를 열람한다고 가정하자. 현재는 이름 필드가 암호화 되어 있기 때문에 이름필드에 정확한 이름을 암호화 하여 비교하거나 반대로 이름 필드에 저장된 값을 복호화 하여 텍스트와 비교를 하는 방법 외에는 없다[9]. 따라서 본 논문에서는 어플리케이션 수준의 암호화에서의 이와 같은 문제점을 해결하기 위해서 해시함수를 이용한 영역질의 방법을 제안하여 이와 같은 문제를 해결하고자 한다.

Ⅱ. 관련 연구

대부분은 기업들은 자신의 중요한 데이터를 웹 호스팅 회사의 서버에 보관한다. 이것은 안전한 데이터 보관 방법이라고 볼 수 없지만 기업에서 큰 비용을 들여서 데이터를 관리하기 어려운 상황에서는 많이 이용하는 방법이다. 이 경우에 데이터를 안전하게 지키기 위해서 데이터를 암호화 하고 암호화된 데이터를 데이터베이스에 보관한다. 암호화된 데이터를 검색하기 위해서 주로 사용하는 방법이 암호화 데이터를 복호화 하여 찾고자 하는 데이터와 비교한 다음 질의 결과를 보내주는 방법이 주로 사용되고 있다. 이 방법은 시간이 많이 소요될 뿐만 아니라 프로그램 코드를 알면 쉽게 데이터를 도난당할 우려가 있다.

암호화 데이터의 데이터베이스에 저장과 검색에 대한 최초의 시도는 Song 등[2]이 제안하였다. 이 연구에서는 대칭키(symmetric key) 암호화 알고리즘을 사용하였다. 이 방법으로 데이터를 검색할 경우 필요이상의 많은 복호화 과정으로 인하여 데이터를 검색하는데 많은 시간이 요구됨을 알 수 있다.

Hacigumus[5]는 일치 검색의 경우, 정확하게 일치하는 평 문 값을 알기 위해 추가 필터링이 필요하다. 또한, 범위 검색의 경우에 범위 내의 값이 포함된 버킷을 검색하고, 이어서 버킷 내의 값을 복호화 하는 재 필터링 과정이 필요하므로 실질적으로 범위검색이 된다고 볼 수 없다. [3, 5]는 bucket-based index 방법과 유사하며 범위 검색을 지원하지 않는다.

데이터베이스 암호화가 중요하고 필요하다는 것은 누구나 공감하지만 실제 현장에서 데이터베이스를 다루는 사람들의 입장에서는 회의(懷疑)를 가질 수 있다. 왜냐하면 그들은 데이터베이스에 접속하기 위해서는 관리자의 암호를 알아야 하고, 뿐 만 아니라 뛰어난 시스템과 네트워크 보안 체계와 보안 프로그램을 가동 등으로 특별히 해커의 침입에 대비하고 있기 때문일 수 있다. 결국, DB 암호화 기능은 궁극적인 DB 보안 솔루션이 아니라, 다음과 같은 경우에 효과적으로 이용될 수 있는 보안 구성 요소이다[11].

- DB 파일 도난 예방
- 내부 관리자의 기밀 데이터 접근 제한

DB 암호화는 다음과 같은 단점이 있다.

■ 인덱스나 키 필드를 암호화하는 경우 성능에 큰

악영향을 미치게 된다.

■ 응용 계층에서 수정을 해야 하는 경우, 비용이 추가될 수 있다.

■ 범위나 부분 검색을 요청하는 경우, 모든 암호 데이터가 복호화 되어야 하기 때문에 심각한 성능 저하가 생기게 된다.

이와 같이 데이터베이스 암호화는 아직은 초보적인 단계에 있다. 그 결과 대부분의 기업들에서는 아직 데이터를 암호화하여 데이터베이스에 저장하고 질의하는 기법이 보편화되어 있지 않다. 이것은 데이터베이스에 대한 암호화의 요구가 적어서 라기 보다는 아직은 기능적인 면과 비용적인 면에서 개선할 점이 있기 때문일 것이다. 우리는 본 논문에서 데이터베이스에 저장된 데이터를 보다 안전하게 관리하기 위해서 AES[13]로 암호화 하여 데이터를 저장하고 저장된 데이터를 보다 효과적으로 검색할 수 있는 영역 질의 방법에 대해서 연구하였다. 또한, 일치 질의 및 영역 질의 방법을 개선하여 효율성을 높이는 방법을 제안하고자 한다.

Ⅲ. 데이터베이스 암호화

1. 암호화 모델 구조

본 논문에서는 중소업체의 사용자 및 개인 사용자의 귀중한 자료를 웹 호스팅 형태나 혹은 개인 서버 형태로서 자료를 관리 할 경우에 발생할 수 있는 자료의 도난과 해커들의 불법적인 행동으로 인한 자료 유출 시에 방어하기 위한 방법의 하나로서 자료를 암호화된 형태

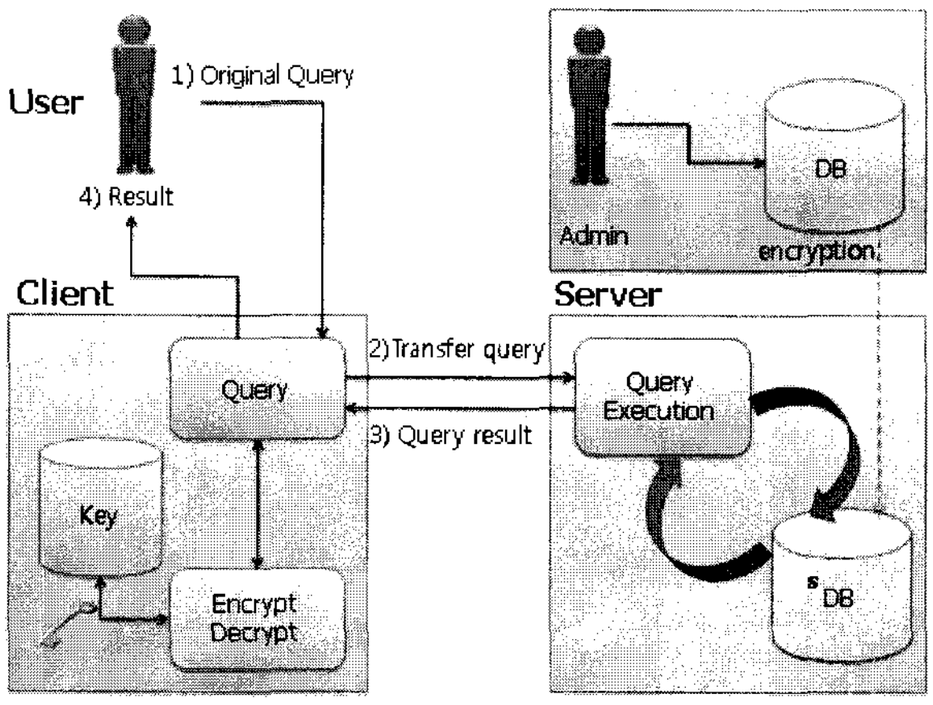


그림 1. 관계형 DB에서 암호화 모델 구조
Fig. 1. Model of encryption structure in relational database.

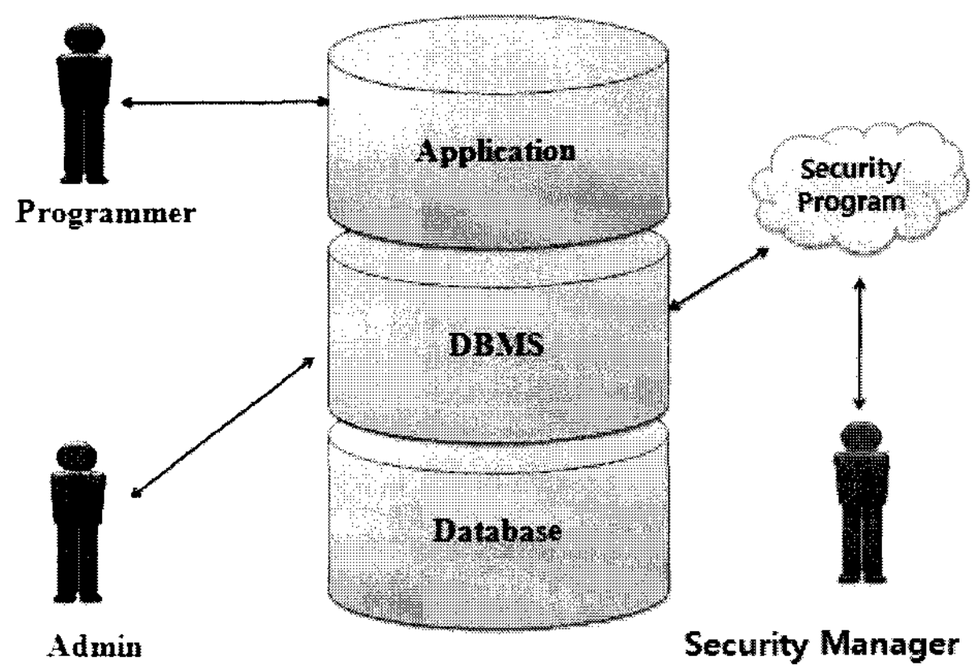


그림 2. 보안의 역할 분담
Fig. 2. division of role in security part.

로 보관하는 방법을 구현하고자 한다. (그림 1)은 이와 같은 시스템을 구현하기 위한 구조를 보인 것이다. 이 시스템에는 사용자(user), 클라이언트(client), 서버(server), 데이터 소유자(data owner)등의 요소로 구성된다.

사용자는 클라이언트에 질의(original query)를 보낸다. 클라이언트 질의 프로세서(query processor)에 의해서 사용자의 질의 내용을 암호화한다. 암호화된 질의는 서버에 보내진다.

서버는 질의를 DBMS에 의해 DB에 질의를 보내고 질의 결과를 서버가 받아서 클라이언트에 보낸다. 클라이언트는 질의 결과를 복호화 한 다음 유저에게 보낸다. (그림 2)는 기존의 관리자와 보안 관리자의 역할과 의무를 나타낸 것이다.

전통적인 데이터베이스 관리 방법은 데이터베이스 관리자가 보안 관리까지를 책임지는 구조이다. 그러나 전문적인 지식 및 네트워크를 통한 보안 능력이 필수적이므로 최근에는 보안 전문가가 데이터베이스 보안을 담당하는 구조가 향후 필요할 것이다.

3.2 해싱(Hashing) 알고리즘

해싱은 검색할 키 값을 비교하지 않고 검색할 수 있는 방법으로써 번지를 이용한 정렬방식과 유사한 방식이다. 해싱은 해시 테이블을 사용하여 단 한 번의 접근으로 원하는 레코드를 검색할 수 있다. 해시 테이블은 파일 레코드의 키 값에 대응하는 해시 주소와 레코드를 저장하는 공간인 버켓으로 구성되어 있다. 포인터를 사용하여 구현하는 경우에는 실제의 레코드 대신에 레코드가 저장되어있는 메모리 포인터를 저장한다. 파일 내

```
def self.sdbm( str, len=str.length )
  hash = 0
  len.times{ |i|
    hash = str[i] + ( hash << 6 ) + ( hash <<
     16 ) - hash
  }
  hash & SIGNEDSHORT
end
```

그림 3. 해시 알고리즘 (sdbm)

Fig. 3. Hash algorithm (sdbm).

의 키 값에 해시 함수를 적용하여 해시 주소를 생성한다. 해시함수는 임의의 길이를 갖는 메시지를 입력으로 하여 고정된 길이의 해시값 또는 해시 코드라 불리는 값을 출력한다. 보다 엄밀히 말하면, 해시 함수 h는 임의의 길이의 문자열을 고정된 길이를 갖는 n비트 문자열로 대응시킨다.

본 논문에서는 암호화된 데이터를 검색하는 용도로서 해시를 사용한다. 왜냐하면 기존의 암호화된 데이터를 데이터베이스 질의 언어의 근사질의 및 영역질의가 가능하지 않기 때문이다. 해시 함수 (SDBM 해시 알고리즘)[12]는 다음과 같이 정의된다.

이 알고리즘은 좋은 배치를 가지는 일반적인 해시 함수이다. 실질적인 함수는 hash(i) = hash(i - 1) * 65599 + str[i]. (그림 3)은 gawk에서 사용되는 빠른 형태의 수정된 버전이다.

3.3 AES를 이용한 데이터베이스 암호화

1990 년대 들어 DES(Data Encryption Standard)암호의 해독의 가능성이 높아지고, 1998년을 기점으로 DES는 표준 기한이 만료됨에 따라, 미국 NIST(National Institute of Standards and Technology)에서 1997년 9월에 암호 키의 길이가 128 비트 이상인 새로운 블록암호인 AES(Advanced Encrytion Standard)를 공모하였다. 총 21개가 응모하여 그 중 1998년 8월에 15개의 1차 후보가 올라, 1999년 4월 5개의 후보로 압축하여, 최종적으로 2000년 10월 2일에 벨기에에서 제안해 만든 Rijndael(Rijmen & Daemen)이 AES로 채택되었다. Rijndael은 다른 AES 후보 기술보다 보안성, 성능, 효율성, 구현 용이성, 유연성 등의 항목에서 가장 우수한 기술로 평가받았고, 또한 이 기술은 서로 다른 다양한 컴퓨터 환경에서도 우수한 성능을 보여주고 메모리를 적게 차지해 스마트카드 등 메모리 용량이 적은 장치에서 손쉽게 사용될 수 있다는 점이 특징이다[13].

표 1. 고객 스키마

Table 1. Customer Schema.

cust_no	cust_name	cust_ssn	cust_addr
1	Tomato	860919-*******	Seoul
2	Christopher	871014-*******	Pusan
3	James	860715-*******	Daegu
4	Jenna	870815-*******	Incheon

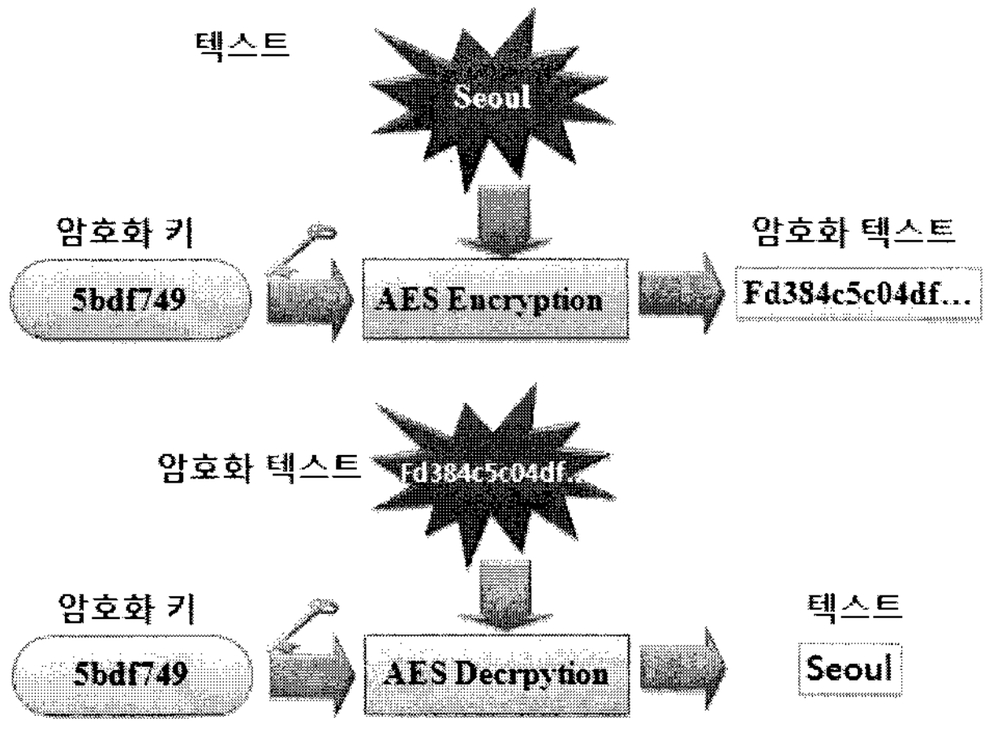


그림 4. AES를 이용한 암호화 과정

Fig. 4. Procedure of encryption using AES.

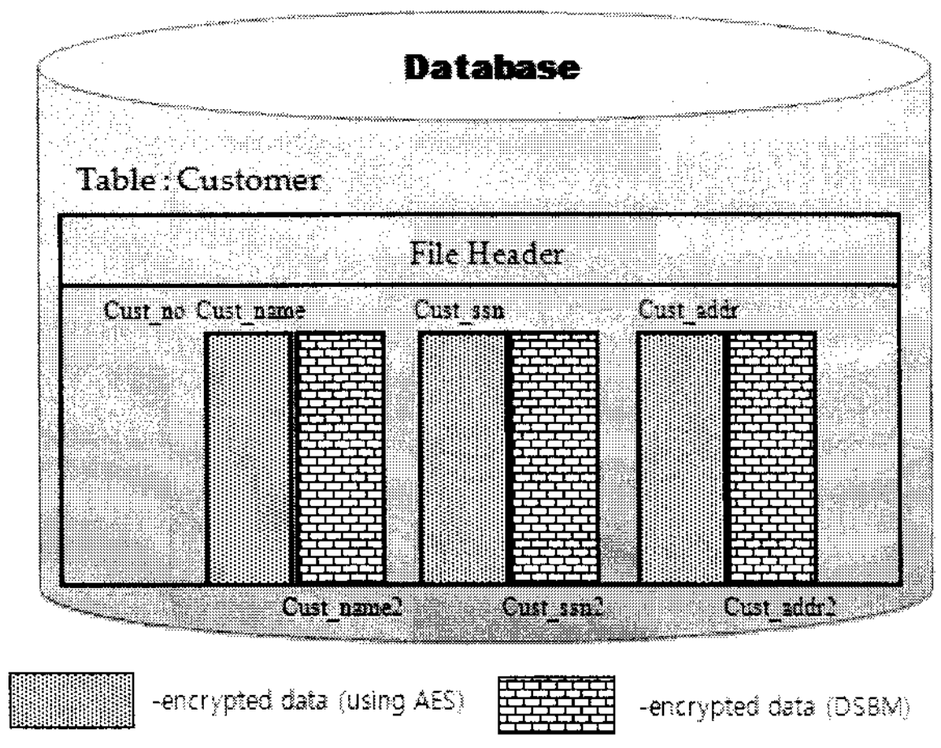


그림 5. 암호화 데이터 저장을 위한 Customer 테이블의 구조

Fig. 5. Structure customer table for storing encryption data.

데이터베이스에 고객 자료를 저장하기 위한 스키마를 (표 1)과 같이 정의하였다. 일반적인 이 구조는 암호화가 되지 않은 상태의 구조이다.

즉, $R(A_1, A_2, \ldots, A_n)$를 서버에 저장된 상태를 나타낸 것이다.

(표 1)의 저장된 데이터의 암호화를 위해서 AES 암

표 2. 암호화된 고객 레코드
Table 2. Encrypted Customer record.

cust_no	cust_name	cust_ssn	cust_addr
1	b4cc344d25..	7502ffbae..	fd38499..
2	5447aa4026..	c34e8996..	2c66e79..
3	d52e323a47..	f93eade81..	545f950..
4	106068fc25..	8e1681e5..	7afedf25..

호화 알고리즘을 사용한다. (그림 4)는 AES 알고리즘의 사용 모습을 보인 것이다.

텍스트인 'Seoul'을 암호화하기 위해서 암호화키와 텍스트를 AES 암호화 알고리즘에 적용한다. 적용결과 그림과 같이 암호화된 텍스트를 얻었다. 반대로 암호화된 키를 복호화 하기 위해서 암호화키를 사용하면 암호화 했던 텍스트를 얻을 수 있다.

(그림 5)는 고객(Customer) 테이블에 고객의 정보를 저장하기 위한 구조를 그림으로 도식화 한 것이다. 여기에서 고객의 아이디(Cust_no)는 암호화 하지 않았다. 고객의 이름(Cuter_name), 고객의 주민번호(Cust_ssn), 고객의 전화번호(Cust_tel)는 AES 알고리즘으로 암호화했다. (표 2)는 고객의 정보를 AES로 암호화 한 것을 나타낸 것이다.

(표 2)를 이용해서 간단한 질의를 가정하면 다음과 같다.

질의1 : 이름이 "Tomato"인 사람의 주소를 검색하라.

Π AES_DECRYPT(cust_addr, key) (σ cust_name = AES_ENCRYPT ('Tomato' ,'key') (customer))

질의2 : 이름이 "Tom"으로 시작하는 모든 사람의 주소를 검색하라.

Π AES_DECRYPT(cust_addr, key) (σ cust_name = AES_ENCRYPT ('Tom%' ,'key') (customer))

$\sigma_c(R)$는 릴레이션 R로부터 조건 c에 해당하는 레코드를 가져오라는 의미의 데이터베이스의 *SELECT* 연산자이다.

Π는 관계형 데이터베이스의 *PROJECTION* 연산자로 사용된다. AES_ENCRYPT는 텍스트 데이터를 암호화 하는데 사용되는 AES 함수이다. AES_DECRYPT는 텍스트 데이터의 복호화에 사용되는 함수이다.

(질의 1)에서와 같이 Customer 테이블에 저장된 데

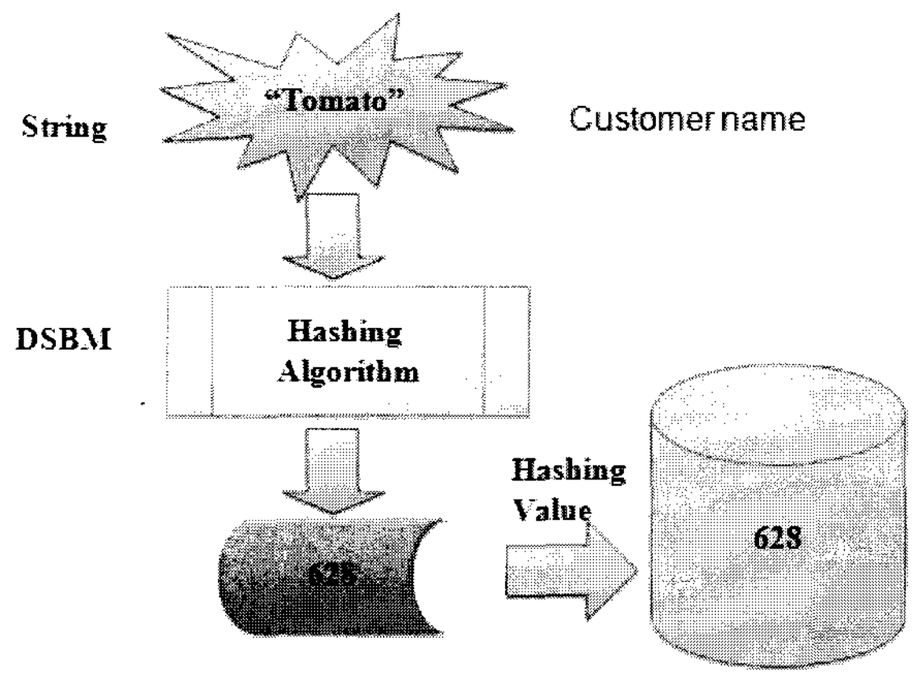


그림 6. 해시코드 생성과정
Fig. 6. Procedure of a hash code generation.

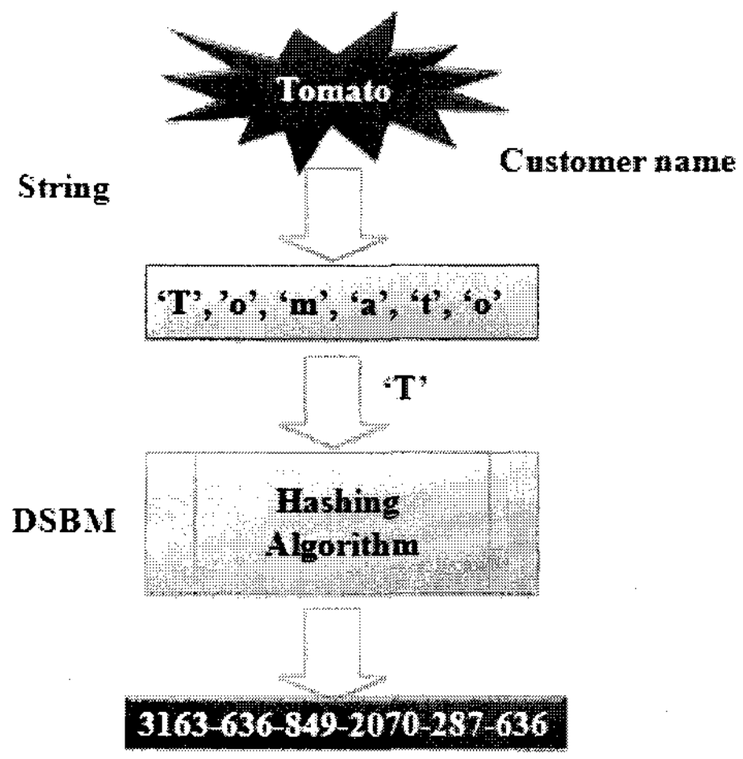


그림 7. 개선된 해시코드 생성과정
Fig. 7. Innovation of a hash code generation.

이터는 암호화 및 복호화 과정과 질의 과정을 결합해서 원하는 정보를 얻을 수 있다. 그러나 (질의2)는 이름이 "Tom"으로 시작하는 모든 사람의 주소를 검색하라는 질의를 (질의 2)와 같이 하면 결과를 얻을 수 없다. 결국 원하는 결과를 얻기 위해서 cust_name 필드를 복호화 한 후 그것이 Tom으로 시작하는지를 비교해야만 한다. 이렇게 되면 결과적으로 프로그램에 의해서 복호화 과정이 발생하고 보안된 데이터의 안전을 보장하기 어렵다. 이와 같은 문제를 해결하기 위해서 해시함수를 이용하여 영역 값을 찾는 방법을 개선하였다. (그림 6)은 고객의 이름을 해시 함수를 이용해서 해시 값을 얻는 과정을 그림으로 나타낸 것이다. 첫 번째 고객의 이름인 "Tomato"를 암호화 한 결과 "628"를 얻었다. 638을 customer 테이블의 cust_name2라는 필드에 저장하고 Tom으로 시작하는 사람을 검색한다고 가정하면, 이 역시 근사 질의가 가능하지 않다.

(그림 7)은 고객의 이름 "Tomato"를 문자열로 한 번에 코드화 하는 것이 아니라 Tomato의 T, o, m, a, t, o의 각각에 대해서 해시코를 그림과 같이 생성한 다음, 이것을 문자열로 변환하여 customer 테이블의 cust_name2에 저장한다.

(질의 2)는 유사검색에 관한 질의이다. 이를 개선된 방법으로 다음과 같이 질의가 가능하다.

① "Tom"의 해시코드가 고객 테이블의 cust_name2에 저장되어 있다고 가정한다.

② 프로그램에 의해서 "Tom"에 대해서 Hash코드를 생성한다.

$cust_name2 = 3163-636-849

③ 질의는 다음과 같다.

Π AES_DECRYPT(cust_addr,key) (σ cust_name2 in { *Π* ***cust_name2*** (σ **cust_name2 = $cust_name2 (customer)**) } (customer))

위의 단계를 거치면 결과적으로 like 질의가 되어 "Tom"으로 시작하는 모든 cust_name를 얻을 수 있다.

sdbm hash알고리즘은 문자열 데이터와 관련하여 해시 값을 생성하였다. 이 알고리즘을 매우 효과적으로 이용하였다. 그러나 숫자 값의 근사 비교에는 효과적이지 못했다. 따라서 (그림 8)의 K_hash 알고리즘을 고안했다. 즉, 특정 숫자를 비교하는데 이 숫자보다 큰지 혹은 작은지 등의 비교를 위해서 해시 알고리즘을 설계했다. K_hash를 이용한 영역 질의를 가정하면 다음과 같다.

```
def K_hash(number, key   )

  hash = 0
    hash = number << key % 10
  hash & LONG

end
```

그림 8. K_hash 알고리즘
Fig. 8. K_hash algorithm.

질의1: 나이가 20세 이하인 사람들의 이름을 검색하라.

Π AES_DECRYPT(cust_name, key) (σ cust_age <= K-hash (20, KEY) (customer))

질의2: 나이가 20세 이상 30세 미만인 사람들의 이름을 검색하라.

Π AES_DECRYPT(cust_name, key) (σ cust_age >= K-hash (20, KEY) and cust_age < K-hash (30, KEY) (customer))

(질의 1)(질의 2)와 같이 K_hash를 이용하여 영역 범위 값을 구할 수 있다.

3.4 암호화된 데이터를 위한 질의 구현

Hi는 릴레이션에서 속성이며 해시 값을 의미한다. $H(T)$는 평문 T를 해시 함수에 의해서 해시 코드로 바꾸는 것이다. $A.conform(T)$는 해시 코드로 바뀐 속성을 정의한 것이다. Si는 암호화된 속성을 나타낸다.

$$A.conform(T) = \{Hi \in R | H(T) \leq Hi\}$$

Select Query : 릴레이션(R)으로부터 조건(P: Predicate)에 해당하는 모든 데이터를 가져온다.

$$Q1 = \sigma_P(R)$$

Q1의 암호화된 질의 형태는 다음과 같다. 암호화된 질의에서는 검색을 위한 해싱 비교와 복호화 과정이 필요하다.

$$Q1 = decryption(\sigma_P(\pi_{A.conform}(T)(R)))$$

Select-Project Query: 릴레이션(R)로부터 조건(P)에 해당하는 모든 데이터들 중에서 특정 도메인에 해당하는 데이터를 나타내고자 한다. πA는 도메인에서의 특정 필드의 속성 값을 의미한다. Q2는 릴레이션(R)에서 조건에 맞는 필드 값을 가져오라는 질의이다.

$$Q2 = \pi_A(\sigma_P(R))$$

Q2의 암호화된 질의 형태는 다음과 같다.

$$Q2 = decryption(\pi_{Si}(\sigma_P(\pi_{A.conform}(T)(R))))$$

Select-Project-Join Query : 2개의 관계형 데이터베이스 테이블을 조인하고, 질의를 하기위한 것이다.

$$Q3 = \pi_{Ra_1.Aa,\, Ra_2.Ab}(\sigma_P(Ra_1 \times Ra_2))$$

Q3의 암호화된 질의 형태는 다음과 같다.

$$Q3 = decryption(\pi_{Ra_1.Aa,\, Ra_2.Ab}(\sigma_P(\pi_{A.conform}(T)(Ra_1 \times Ra_2))))$$

3.5 실험 및 분석

우리는 본 논문에서 제안한 암호화 방법을 이용하여 근사 질의와 범주 질의를 실험하였다. 실험에 사용된 데이터는 UCI 기계학습 저장소[15]로부터 자동차 평가 데이터를 실험에 사용하였다. 이 데이터는 1,728개의 레코드에 6개의 속성으로 구성되어 있다. 이 데이터의 속성은 (buying, maint, doors, persons, lug_boot, safety)이다. 질의 성능을 측정하기 위해서 5개의 질의를 만들어 시간을 측정했다.

질의 1 : *SELECT buying FROM cars WHERE persons >= 4*

질의 2 : *SELECT buying FROM cars WHERE doors >= 2 and doors <=4*

질의 3 : *SELECT maint FROM cars WHERE bug_boot like 'sma%'*

질의 4 : *SELECT safety FROM cars WHERE buying like '%high'*

질의 5 : *SELECT buying FROM cars WHERE safety = 'high'*

(질의 1)은 4명 이상 승차 가능한 차량의 가격을 질의 한 것이다. (질의 2)는 차량의 문 개수가 2개 이상이고, 동시에 4개 이하인 차량의 가격을 질의 한 것이다. (질의 1),(질의 2)는 유사한 질의로서 질의 시간에 차이가 없음을 알 수 있다.

(질의 3)의 트렁크의 크기가 'sma'로 시작하는 것의 유지비를 구하라는 질의이다. (질의 4)는 차량의 가격에 'high'가 포함된 것의 safety를 구하라는 질의이다. (질의 3), (질의 4)는 각각 like가 포함된 질의로서 암호화된 질의 값을 구하기 위해서는 복호화하고 비교를 해야 하지만 본 논문에서는 질의 값을 해시로 변환한 후 비교하는 방식이므로 많은 시간이 필요하지 않다.

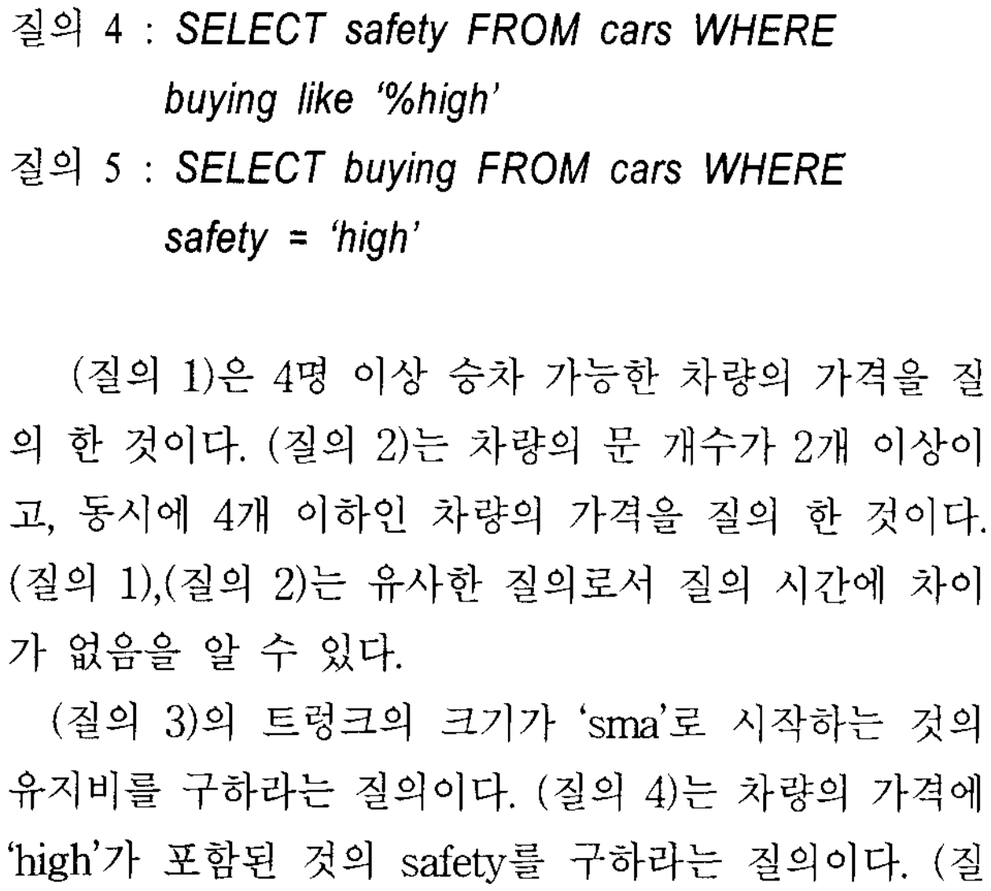


그림 9. 질의 성능 측정
Fig. 9. Time query performance.

(질의 5)는 차량의 안전성(safety)이 'high'에 해당하는 차량의 가격을 구하라는 질의 이다. 이 질의는 (질의 1)과 (질의 2)에 비해서 상대적으로 약간의 차이를 보이고 있는데 (질의 1), (질의 2)의 경우는 정수 데이터이므로 상대적으로 검색시간이 빠른 것으로 분석된다. (그림 9)에서 AES 복호화 시간은 측정에 포함하지 않았다.

Ⅳ. 결 론

최근에 인터넷이 급속히 발전하면서 개인 정보를 요구하는 인터넷 사이트가 많이 있다. 대부분이 회원으로 가입을 해야 정보를 이용하거나 아니면 서비스를 이용하도록 된 경우가 많다. 우리도 모르지만 우리의 개인 정보는 불법적으로 거래되고 있다. 이와 같은 대부분의 사건은 우리의 정보가 인터넷과 데이터베이스에 노출되기 때문이다. 따라서 꼭 필요한 경우가 아니면 개인 정보는 절대로 노출되어서는 안 된다. 그러나 현재의 시스템에서는 개인 정보가 노출될 수밖에 없다. 이와 같은 문제를 근본적으로 해결하기 위해서 우리는 자료를 저장할 때 암호화해서 데이터베이스에 저장할 필요가 있다. 그러나 확실한 상적업인 용도로 쓰기에는 아직도 많은 문제를 갖고 있다. 예를 들면 근사질의 문제를 아직도 완벽하게 실용화 하고 있지 못하고, 또한 암호화된 데이터베이스 시스템의 성능도 암호화 하지 않은 시스템에 비해서 성능이 좋지 않다.

따라서 본 논문에서는 AES와 해시 알고리즘에 의해서 영역 및 근사 질의를 할 수 있도록 알고리즘을 제안하고 구현하였다. 본 논문에서 제안한 해시 알고리즘은 다소 단순하기 때문에 얼마든지 복호화가 가능한 문제점이 있기 때문에 반드시 보안이 필수적인 부분은 근사 질의를 사용하지 않고 일치하는 값만 검색하도록 AES 알고리즘 만 사용하고, 상대적으로 보안이 덜 중요한 필드에 대해서 본 논문에서 제안한 방법을 사용한다면 효과적인 시스템의 운용이 될 것으로 기대한다. 향후 보안과 질의 면에서 보다 뛰어난 해시 알고리즘을 개발해서 AES 알고리즘을 대치할 계획이다.

참 고 문 헌

[1] B. Iyer, S. Mehrotra, E. Mykletun, G. Tsudik, and Y. Wu. *A framework for efficient storage security in RDBMS*. Lecture Notes in Computer Science, vol. 2992, pp. 147-164, 2004.

[2] D. X. Song, D. Wagner, and A. Perrig. *Practical techniques for searches on encrypted data*. IEEE Symposium on Security and Privacy, pp. 44-55, 2000.

[3] E. Damiani, S. De C. Vimercati, S. Jajodia, S. Paraboschi, and P. Samarati. *Balancing confidentiality and efficiency in untrusted relational dbmss*. In CCS, 2003.

[4] G. Ozsoyoglu, D. Singer, and S. Chung. *Anti-tamper databases: Querying encrypted databases*. In Proc. of the 17th Annual IFIP WG 11.3 Working Conference on Database and Applications Security, 2003.

[5] H. Hacigumus, B. R. Iyer, C. Li, and S.d Mehrotra. *Executing SQL over encrypted data in the database-service-provider model*. In SIGMOD, 2002.

[6] H. Hacigumus, B. R. Iyer, and S. Mehrotra. *Providing database as a service*. In ICDE, 2002.

[7] H. Hacigumus, B. R. Iyer, and S. Mehrotra. *Efficient execution of aggregation queries over encrypted relational databases*. In DASFAA, 2004.

[8] Z. Wang, J. Dai, W. Wang, and B. Shi, *Fast Query Over Encrypted Character Data in Database*, Communications in Information and Systems, Vol. 4, No. 4, pp. 289-300, 2004.

[9] Z. Yang, S. Zhong, and R. N. Wright, *Privacy-Preserving Queries on Encrypted Data*, Proceedings of the 11th European Symposium On Research In Computer Security, 2006.

[10] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. *Order preserving encryption for numeric data*. In SIGMOD, 2004.

[11] R. Mogull, "*When and How to Use Database Encryption,*" Gartner, 6 Feb 2004.

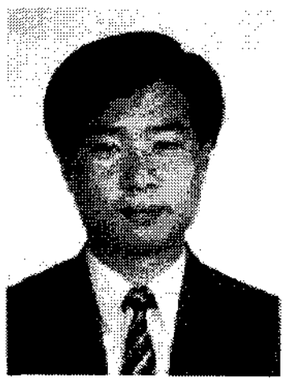
[12] Yigit. sdbm - substitute dbm. In http://search.cpan.org/src/NWCLARK/perl-5.8.4/ext/SDBM_File/sdbm/.

[13] Wikipedia, http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

[14] 디지털 타입즈, http://www.dt.co.kr/contents.htm?article_no=2006031302019922601028

[15] UCI, http://archive.ics.uci.edu/ml/index.html

저 자 소 개

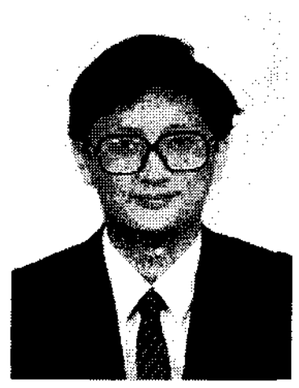


김 천 식(정회원)

1997년 한국외국어대학교 컴퓨터 및 정보통신공학과 (공학석사)
2003년 한국외국어대학교 컴퓨터 및 정보통신공학과 (공학박사)
2000년~2003년 경동대학교 정보통신공학부 교수
2004년~현재 안양대학교 교수
2007년~현재 대한전자공학회 컴퓨터소사이어티 분과위원장
2008년~현재 인터넷 방송통신 TV학회 상임이사
2006년~현재 인터넷 정보학회 학회편집위원
2006년~현재 대한교통학회 정회원
2005년~현재 한국데이터베이스학회 정회원
<주관심분야: 데이터베이스, 데이터마이닝, 유비쿼터스, 텔리매틱스, TPEG, DMB, 홈네트워크, e-Learning>

김 형 중(평생회원)

1978년 서울대학교 전기공학과 (공학사)
1986년 서울대학교 제어계측공학과 (공학석사)
1989년 서울대학교 제어계측공학과 (공학박사)
1992년~1993년 USC 방문교수
1989년~2006년 강원대학교 교수
2006년~현재 고려대학교 정보경영공학부 교수
<주관심분야: 멀티미디어보안, 분산처리, 콘텐츠 공학 등>



홍 유 식(정회원)

1984년 경희대학교 전자공학과 (학사)
1989년 뉴욕공과대학교 전산학과 (석사)
1997년 경희대학교 전자공학과 (박사)
1985년~1987년 대한항공(N.Y.지점 근무)
1989년~1990년 삼성전자 종합기술원 연구원
1991년~현재 상지대학교 컴퓨터공학부 교수
2000년~현재 한국 퍼지 및 지능시스템학회 이사
2004년~현재 대한전자공학회 ITS 분과위원장
2001년~2003년 한국정보과학회 편집위원
2001년~2003년 한국컴퓨터교육산업학회 이사, 편집위원
2004년~현재 건설교통부 ITS 전문심사위원
2004년~현재 원주 시 인공지능신호등 심사위원
2005년~현재 정보처리학회 이사
2005년~현재 인터넷 정보학회 이사
2005년~현재 정보처리학회 강원지부 부회장
2006년~현재 인터넷 방송통신 TV학회 상임이사
<주관심분야: 퍼지 시스템, 전문가시스템, 신경망, 교통제어>