상세 접근 통제와 안전한 데이터 관리를 위한 데이터베이스 보안 시스템

(A Database Security System for Detailed Access Control and Safe Data Management)

조 은 애 [†] 문 창 주 [††] 박 대 하 [†††] 홍 성 진 [††††] 백 두 권 [†††††]

(Eun-Ae Cho) (Chang-Joo Moon) (Dae-Ha Park) (Sung-Jin Hong) (Doo-Kwon Baik)

요 약 최근 데이터베이스의 보안 취약성으로 인해, 내부의 비인가자 또는 인가자의 데이터 접근에 대한 통제 정책이 제대로 이루어지지 않아 정보 유출 사고가 발생하고 있다. 현재의 데이터베이스 권한부여 방식은 관리자가 데이터베이스 오브젝트에 접근할 수 있는 권한을 사용자에게 부여하는 방식이다. 그러나 이러한 방법은 다양한 사용자 접근을 통제하기 위한 정책을 데이터베이스에 적용할 수 없다. 또 다른 데이터베이스 보안 방법인 데이터 암호화는 데이터의 인덱싱이 어렵다는 단점이 있다. 본 논문에서는 다양한 보안 정책을 반영하기 위해, 클라이언트에서 데이터베이스 서버로 요청되는 네트워크상의 패킷 분석을 통한 데이터베이스의 접근 통제 시스템을 제안한다. 제안된 보안 시스템에서는 특정 일자 및 시간, SQL에 포함되어 있는 특정 문자열, 결과 데이터 수, 레벨에 따른 컬럼 제한 등의 통제 정책을 적용할 수 있을 뿐만 아니라 사용자 정보 및 SQL의 위변조를 방지하기 위해서 공개키 인증과 메시지 인증코드 교환으로 무결성을 확보할 수 있다.

키워드 : 데이터베이스 보안, 프라이버시, 접근 통제, 데이터 관리

Abstract Recently, data access control policies have not been applied for authorized or unauthorized persons properly and information leakage incidents have occurred due to database security vulnerabilities. In the traditional database access control methods, administrators grant permissions for accessing database objects to users. However, these methods couldn't be applied for diverse access control policies to the database. In addition, another database security method which uses data encryption is difficult to utilize data indexing. Thus, this paper proposes an enhanced database access control system via a packet analysis method between client and database server in network to apply diverse security policies. The proposed security system can be applied the applications with access control policies related to specific factors such as date, time, SQL string, the number of result data and etc. And it also assures integrity via a public key certificate and MAC (Message Authentication Code) to prevent modification of user information and query sentences.

Key words : Database Security, Privacy, Access Control, Data Management

· 본 과제는 한국소프트웨어진흥원의 SW공학 요소기술 개발과 전문인력 양성사업의 결과물임을 밝힙니다.
· 이 연구에 참여한 연구자(의 일부)는 '2단계 BK21사업'의 지원비를 받았음

† 학생회원 : Purdue University 박사후과정
cho98@purdue.edu
†† 정 회 원 : 건국대학교 항공우주정보시스템공학과 교수
cjmoon@konkuk.ac.kr
††† 정 회 원 : 한국디지털대학교 디지털정보학과 교수
summer69@kdu.edu
†††† 정 회 원 : (주)웨어밸리 기술연구소 책임연구원
jindigit@gmail.com
††††† 종신회원 : 고려대학교 정보통신대학 교수
baikdk@korea.ac.kr
(Corresponding author임)

논문접수 : 2008년 8월 18일
심사완료 : 2009년 7월 10일

정보과학회논문지: 데이타베이스 제36권 제5호(2009.10)

1. 서 론

사회의 급속한 정보화는 정보의 가치를 빠르게 상승시키고 있다. 개인정보를 포함한 여러 가지 정보들은 데이터베이스에 점차 더 많이 저장되고 있으며[1], 각 조직들은 그에 따라 많은 수의 데이터베이스를 관리하고 있다. 그리고 조직의 정보 이용자들은 데이터베이스에 있는 정보에 접근하려는 다양한 접근 방식을 요구하고 있다. 데이터베이스의 정보에 대한 다양한 접근은 정보의 활용성을 높일 수 있지만, 그와 비례하여 정보를 노출시킬 가능성도 많아지기 때문에 악의적인 대량 수집이나 불법적인 이용의 확률을 높인다. 따라서 데이터베이스에서 정보의 안전한 관리는 필수적이라고 할 수 있다.

데이터베이스 보안 방법은 크게 데이터베이스의 입출력 경로를 감시하는 접근 통제(access control)/감사(audit)와 데이터베이스 내부의 데이터 자체를 암호화(encryption)하는 것으로 분류된다[2]. 그래서 현재 시판되고 있는 국내외 데이터베이스 보안 솔루션 역시 접근 통제 및 감사 방식과 데이터 암호화 방식을 사용하고 있다. 이들은 여러 가지 장점들을 가지고 있기도 하지만 접근 통제/감사 방식은 데이터 자체에 대한 보안 미흡의 문제점이 있고 데이터 암호화 방식은 성능 저하와 같은 문제점이 있다. 국내외 DBMS 시장의 대다수를 점유하고 있는 오라클 DB의 경우에도 8i 버전 이상에서는 암호화 모듈이 기본적으로 내부에 장착되어 있으나, 이것을 이용하는 경우 성능저하가 매우 심하여 극히 제한적으로 적용되고 있다. 따라서 이러한 환경에서 내부자 또는 외부에서의 정보 불법 획득을 원천 차단할 수 있는 정보 보호 관련 기술 개발이 시급히 요구된다[2].

위의 솔루션들과 함께 데이터베이스 보안을 위해 접근을 통제하고 데이터를 관리하기 위한 연구도 활발히 진행되고 있다. 그러나 이러한 여러 가지 연구에서도 문제점이 존재한다. 접근 통제 모델을 사용한 기존의 연구들은 상세하고 다양한 접근 요구를 수용하지는 못하고, 사용자의 보안 요구 사항이 수시로 변할 경우 변경이 용이하지 않다.

위와 같은 사항으로 인하여 데이터베이스의 정보가 유출되는 사고가 여전히 발생하게 되고, 이를 해결하기 위해서는 데이터를 보호하면서 사용자 요구들을 만족하는 새로운 데이터베이스 보안 방법이 필요하다. 따라서 본 논문은 데이터 자체에 대한 보안, 접근 통제에 대한 효율적인 질의·응답의 수행, 이질적인 데이터베이스 정책들의 통합적 관리, 다양한 접근 요구 사항(특정 컬럼, 특정 시간, 특정 SQL 등)에 따른 상세 접근 통제, 인덱싱이 가능하도록 데이터 처리 등을 목적으로 한다.

본 논문의 나머지 부분은 다음과 같다. 먼저, 2장에서는 관련연구로 기존의 데이터베이스의 보안방식과 그에 따른 문제점을 나열하고 그에 대한 해결방법을 살펴보며, 3장에서는 시스템 혹은 정책과 관련된 표시 및 기호를 사전 정의하고 4장에서는 정보 이용자들의 요구사항을 도출하여 새로운 데이터베이스 보안 시스템의 구조를 제안한다. 5장에서는 이에 대한 구현된 결과를 확인하고, 6장에서는 보안 모델의 적용 전, 후에 대해 비교 및 평가를 한다. 마지막으로 7장에서는 제안된 데이터베이스 보안 시스템에 대한 결론 및 향후 연구로 마무리한다.

2. 관련 연구

현재 시판되고 있는 국내외 데이터베이스 보안 솔루션은 접근 통제 및 감사 방식을 사용하는 Chakra, Middleman, DBSafer, DB-i, dGriffin과 데이터 암호화 방식을 사용하는 Cube One, D'AMO, SafeDB, Secure.Data(외국산) 등이 있다.

아래의 표 1에서 볼 수 있는 바와 같이 접근 통제 및 감사 방식은 DBMS의 성능을 보장할 수 있고 운영이 간편한 장점을 가지는 반면 데이터 자체에 대한 보안이 되어있지 않기 때문에 데이터를 도난당했을 경우 데이터가 그대로 노출되는 단점을 가진다.

반대로 데이터 암호화 방식[3]은 데이터가 유출된다 하더라도 암호화 되어 있기 때문에 안전하다는 장점이 있지만 DBMS에 직접 설치되어 운영되기 때문에 DBMS의 성능을 저하시키게 되는 단점을 가진다.

또한 위의 시판 솔루션 이외에, 데이터베이스 보안을 위해 접근을 통제하고 데이터를 관리하기 위한 연구가 접근 통제 모델[4](MAC, DAC, RBAC, etc.)을 이용한 정책 기반 연구[5]와 데이터 암호화[3,6] 기반 연구 등으로 진행되고 있다. 그러나 이러한 여러 가지 연구에서도 다음과 같은 장단점이 존재한다. 먼저, 접근 통제 모델

표 1 데이터베이스 보안 솔루션 혹은 연구 별 장단점

	장점	단점	관련 시판 솔루션	관련 연구
접근 통제 및 감사 방식	• DBMS 성능 보장 • 편리한 운영 • 사용자 접근 관리가 가능	• 데이터 자체에 대한 보안이 불가능 • 특정 항목에 대한 상세접근이 불가능 • 다양한 접근 요구를 수용하지 못함	Chakra, Middleman, DBSafer, DB-i, D-그리핀	[5,7]
데이터 암호화 방식	• 데이터 자체에 대한 안전한 보안	• DBMS 성능의 저하 • 비효율적인 인덱싱	Cube One, D'AMO, SafeDB, Secure.Data	[3,6]

을 사용한 기존의 연구들은 사용자의 접근은 관리하지만 특정 SQL에 대한 접근 통제 혹은 특정 컬럼에 대한 접근 통제와 같은 상세하고 다양한 접근 요구를 수용하지는 못한다. 예를 들어, 임의의 두 사용자가 특정 테이블의 동일한 속성에 접근하는 정책은 적용이 가능하지만, 그 속성 각각의 튜플별로 권한을 주고자 할 경우, 기존 모델들에서는 적용이 어렵다[7]. 또, 사용자의 보안 요구 사항이 수시로 변할 경우 변경이 용이하지 않다. 즉, 임의의 데이터에 대한 사용자의 접근 권한이 수시로 변동되는 상황에 효율적으로 대처하지 못하고, 뿐만 아니라 하나의 정보에 대하여 각 사용자별 접근 통제를 제공하지 못한다. 예를 들어, 특정 데이터에 대한 접근 권한을 가진 사용자가 그 데이터에 대한 권한을 갖지 않은 사용자에게 그 데이터 중 일부에 대한 접근 권한을 부여하는 상황이 요구될 수 있는 것이다.

위에서 언급한 기존에 연구된 데이터베이스 보안 시스템 중 최근에 연구된 다양한 크기의 데이터 그룹을 위한 시스템은 다음 그림 1과 같다[7].

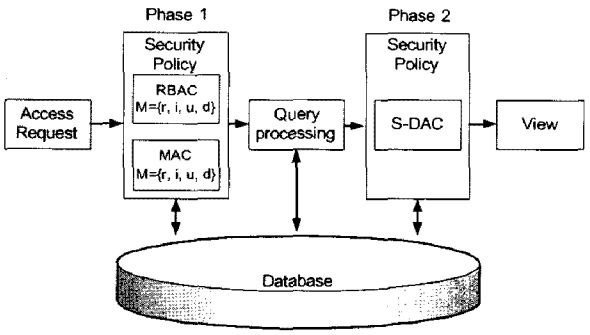

그림 1 다양한 크기의 데이터를 위한 데이터베이스 보안 시스템

위의 그림 1과 같이, 이 시스템은 MAC, 변경한 DAC (Selective-DAC: S-DAC), RBAC 등 여러 가지 접근 통제 정책을 활용하여 데이터를 보호한다. 먼저 Phase1에서 접근하고자 하는 사용자 ID를 검사하여 접근 요청이 SQL 문장인지 혹은 역할에 의한 것인지를 파악하고 각각에 따른 대응을 한다. 또한 다양한 크기의 데이터 그룹에 따라 선택적 접근 통제가 가능하도록 S-DAC 정책과 RBAC 정책을 이용하여 보안 등급과 역할에 따라 접근 통제를 수행한다. 그런 뒤, Phase2에서는 Phase1에서 수행된 결과를 다시 S-DAC 정책을 이용하여 다양한 크기의 데이터 항목들이 선택적으로 필터링 되도록 하고, 최종적으로 모든 보안 정책들을 만족하는 데이터 항목만을 사용자에게 보여주도록 한다.

그러나 이 데이터베이스 보안 시스템에서는 데이터 크기와 관련된 요구사항에 초점을 맞추고 있기는 하지만 몇 가지 문제점이 있다[7]. 첫째, 다양한 크기의 데이터에 대해 접근 통제는 하고 있지만 데이터 자체의 보안은 보장하지 못하고 있다. 둘째, 데이터 그룹을 정의하는 과정에서 대량의 레코드 키값을 구하기 위한 오버헤드가 발생할 수 있다. 셋째, 2개의 Phase를 통하여 RBAC, MAC, S-DAC 모델을 모두 사용하기 때문에 새로운 접근 통제가 필요할 경우 정책의 추가 생성에 따라 시스템의 성능을 저하시킬 우려가 있으며, 넷째, 같은 사용자에 따른 정책들의 중복현상이 나타날 수 있다. 마지막으로 다섯째, 여러 가지 종류의 데이터베이스를 사용할 경우 보안정책들이 각각 적용되기 때문에 정책들을 통합적으로 관리할 수 없다.

위와 같이 기존에 연구된 다양한 크기의 데이터들을 위한 데이터베이스 보안 시스템은 언급된 요구사항들을 만족하지 못한다. 따라서 이런 문제점을 해결하기 위하여 제안하는 보안 시스템은 패킷 분석과 참조 모니터를 통하여 엄격한 보안정책을 기반으로 하는 데이터베이스 접근 통제를 구현한다. 앞의 서론에서 언급한 바와 같이 본 논문의 목적은 다음과 같다.

(1) 데이터 자체에 대한 보안
(2) 접근 통제에 대한 효율적인 질의·응답의 수행
(3) 이질적인 데이터베이스 정책들의 통합적 관리
(4) 다양한 접근 요구 사항에 따른 상세 접근 통제
(5) 인덱싱이 가능하도록 데이터 처리

이 목적들에서 (1)은 기존 데이터베이스 보안 시스템의 첫 번째 문제점을 해결해줄 수 있고, (2)는 두 번째 문제점에 대응할 수 있다. (3)은 세 번째, 네 번째, 다섯 번째 문제점을 해결할 수 있고, 덧붙여 (4)와 (5)는 정보 이용자들의 요구사항을 만족시킬 수 있다. 먼저, (1)에서 데이터를 보호하는 동시에 (5)에서 인덱싱을 가능하도록 하기 위해서 본 논문에서는 데이터 마스킹 방법을 적용한다. 또한 데이터 마스킹 방법을 이용하면 사용자가 데이터베이스에 접근하여 데이터를 획득하는데 걸리는 시간을 효과적으로 관리할 수 있으므로 (2)도 만족할 수 있다. 이때, 본 논문에서는 모니터링을 통한 패킷 분석을 기반으로 하여 (4)를 만족할 수 있도록 하였다. 그리고 이 모든 과정들은 (3)에서 언급한 '서로 다른 정책들을 사용하고 있는 데이터베이스'들을 효율적으로 통합 관리하기 위하여 참조 모니터 개념을 도입하여 데이터베이스에 대한 모든 접근을 모니터링 하도록 설계 및 구현하도록 한다.

따라서 제안하는 시스템은 사용자의 접근 자체에 대한 상세 통제가 가능하고, 마스킹이 되어야 하는 데이터를 결정하는 모듈과 마스킹을 실행하는 모듈이 분리되어 있기 때문에 정책 적용에 대한 시간 지연을 감소시킬 수 있다. 또한 참조모니터를 통해 사용자가 실행하는

애플리케이션에서 발생하고, 데이터베이스로 가는 모든 요청을 관리하도록 하여, 정의되지 않은 접근으로 인한 공격으로부터 데이터를 보호한다[4].

3. 사전 정의

사용자 U_i는 정보 접근의 주체(subject)이고 데이터베이스의 사용자를 의미한다. 데이터는 정보의 집합을 나타내는 객체(object)이며, 하나의 데이터 또는 데이터 그룹으로 정의할 수 있다. 보안 등급(Security Level) SL은 사용자 보안 등급(Use Security Level) USL과 데이터 보안 등급(Data Security Level) DSL을 모두 포함하며, 각 조직 및 시스템의 정책에 따라 정의될 수 있다. 이와 관련된 정의들은 아래 표 2와 같다.

표 2의 기호들을 이용한 기본 전제는 다음의 식과 같이 표현된다.

- $R(U_i) \subset R_j$
- $SL(R_i) \subset USL_s$
- $SL(C_t) \subset DSL_s$
- $SL_1 \geq SL_2 \geq \cdots \geq SL_s$
- $1 \leq N(USL_s) \leq i,\ 1 \leq N(R_j) \leq i$

각각의 사용자들은 역할의 한 멤버가 된다. 각 역할들은 사용자 보안 등급을 가지고, 데이터의 컬럼들은 데이터 보안 등급을 가진다. 또한 사용자 보안 등급과 역할의 종류는 반드시 하나 이상이어야만 한다. 그러나 데이터 보안 등급은 ø (공집합)을 포함할 수 있다.

다음으로 특정 조건에 대한 정책 작성 시 나타낼 수 있는 내용은 다음 표 3과 같이 정의한다.

표 3의 항목들은 상세 접근 통제를 위해 정책을 정의할 때 사용될 수 있는 것이며, 여기에서 정의된 내용은 '5. 데이터베이스 보안 시스템의 구현'에서 반영할 수 있다.

4. 데이터베이스 보안 시스템의 설계

본 절에서는 클라이언트에서 데이터베이스로 작업이 요청될 때, 제안하는 보안 시스템이 어떤 방식으로 보안을 수행하는지를 보여주기 위해 패킷을 취득하고, 분석하여 접근 통제를 적용하는 과정을 설계한다.

표 3 상세 접근 통제 정책 기호의 정의

표시 및 기호	설명
IP	IP 주소
Host	호스트
DbUser	데이터베이스 사용자
DbTable	데이터베이스 테이블
DbInst	데이터베이스 인스턴스
DbSess	데이터베이스 세션
DbSql	데이터베이스 SQL
Time	시각
Date	날짜
App	애플리케이션
LoginUser	로그인 사용자
Cmd	명령어

4.1 보안 요구사항

본 논문에서 제안하는 데이터베이스 보안 시스템은 서론에서 언급한 목적에 따라 다음과 같은 요구사항을 만족하도록 한다.

요구사항 1. 데이터를 보호하는 동시에 인덱싱이 가능하도록 한다.

데이터 암호화를 하는 경우 인덱싱이 불가능 하거나 성능이 저하되는 현상이 생겼는데, 이것을 막기 위해, 보호되어야 하는 데이터를 암호화하는 대신 마스킹하여 감추는 방법을 사용한다.

요구사항 2. 데이터 처리 시 성능이 저하되지 않도록 한다.

데이터베이스의 성능을 저하시키는 요소로 데이터 암호화를 이미 언급한 바가 있다. 따라서 제안하는 시스템은 데이터 암호화를 하는 대신 데이터 마스킹을 통하여 데이터를 보호하도록 한다. 또한 데이터의 효율적인 처리를 위하여 사용자 역할 보안 레벨과 컬럼 보안 레벨을 참조 모니터에 정의하고, 이를 이용해서 마스킹을 처

표 2 접근 통제 정책 기호의 정의

표시 및 기호	설명
U_i	사용자(User) 집합, $U = \{U_1, U_2, U_3, \cdots, U_i\}$
i	사용자 번호
R	사용자 역할(User role), $R = \{R_1, R_2, R_3, \cdots, R_j\}$
j	역할 번호
SL	보안 등급(Security Level) $SL = \{SL_1, SL_2, \cdots, SL_s\}$ * User security level $USL = \{USL_1, USL_2, USL_3, \cdots, USL_s\}$ * Data security level $DSL = \{DSL_1, DSL_2, DSL_3, \cdots, DSL_s\}$
s	보안 등급 번호
C	컬럼(Column) 집합, $C = \{C_1, C_2, C_3, \cdots, C_t\}$
t	컬럼 번호
N	사용자의 총 수

리하도록 하여 보호되어야 하는 데이터의 검색 및 처리 시간을 단축시키도록 한다.

요구사항 3. 데이터베이스에 따라 서로 다른 정책의 통합 관리가 가능하도록 한다.

각각의 데이터베이스에서 정의하고 있는 여러 가지 정책들을 참조 모니터와 연동하여 정책 저장소에 저장하도록 한다. 데이터베이스에 접근하는 모든 패킷들은 참조 모니터를 거쳐서 가야만 하고, 참조 모니터는 데이터베이스에 저장된 정책에 따라 접근의 허가 여부를 결정하도록 하여, 데이터베이스들의 통합적인 관리가 가능하도록 한다.

요구사항 4. 세부적인 접근 통제가 가능하도록 한다.

[세부 사항 4-1] 컬럼 단위의 권한 부여,

[세부 사항 4-2] 특정 날짜 및 시간에 대한 접근 통제,

[세부 사항 4-3] 특정 SQL에 대한 접근 통제

세부적인 접근 통제는 다양한 사용자 및 데이터베이스 관리자에 의해서 요구될 수 있다. 먼저, 컬럼 단위의 권한 부여를 포함하여 데이터 그룹에 대한 권한 부여가 가능하도록 하고, 특정한 날짜, 시간, 혹은 특정한 SQL 문이나 호스트에 대한 접근 통제를 정책으로 정의하여 정책 저장소에 저장하여 세부적인 접근 통제가 가능하도록 구현한다.

4.2 제안한 데이터베이스 보안 시스템

위의 '4.1 보안 요구사항'에서 언급한 사항들을 고려하여 클라이언트와 데이터베이스 사이에 위치하면서 요청하는 모든 패킷을 감시 및 분석하여 정책을 적용하는 접근 통제 모델을 제안한다. 본 논문은 상용 DBMS인 오라클을 대상으로 데이터베이스 접근 통제를 수행하며, SQL*Plus[8] 혹은 TOAD(Tool for Oracle Application Developers)[9] 등을 이용하여 데이터베이스에 접속한다[10].

제안하는 구조는 물리적으로 데이터베이스 혹은 데이터베이스 팜의 바로 앞단에 위치하도록 하여 바이패스(bypass)가 불가능하도록 접근을 통제한다. 접근 통제 시스템의 정책은 데이터베이스 권한과는 서로 독립적으로 동작하며, 클라이언트에서 데이터베이스로 가는 작업 요청 패킷과 데이터베이스에서 클라이언트로 전달되는 결과 패킷을 분석하여 데이터베이스로 가기 전에 통제 정책을 적용한다.

제안하는 보안 시스템은 크게 클라이언트 인증 에이전트(Client Authentication Agent: CAA), 접근 통제 서버(Access Control Server: ACS)로 구성되며 전체 구조는 그림 2와 같다.

애플리케이션 서버에 있는 CAA는 클라이언트에서 애플리케이션 서버를 거쳐 데이터베이스로 가는 패킷으로부터 이더넷 프레임 헤더, TCP 헤더, IP 헤더를 제거하고, 메시지를 추출한다[11]. 무결성 검증을 위하여 추출된 메시지로부터 인증코드를 생성하여 보관하고 패킷을 데이터베이스로 보낸다. ACS는 네트워크를 감시하고 있다가 데이터베이스로 가는 패킷을 붙잡고, 이 중 TCP 데이터 부분을 취득, 분석하여 데이터를 처리한다[12,13]. 취득한 패킷은 모든 접근을 관리하는 ACS의 참조모니터를 먼저 거치게 된다.

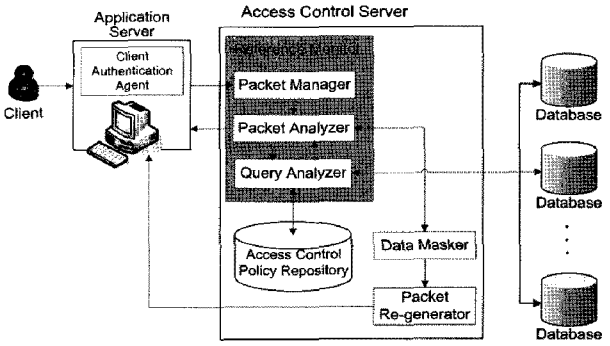

그림 2 Access Control Server의 전체구조

참조 모니터에서는 패킷 분석기를 통해 패킷의 이더넷 프레임 헤더, TCP 헤더, IP 헤더로부터 IP, Mac Address를 추출한 후, 사용자 인증을 거쳤는지 확인한다. 일방향 해쉬함수, 메시지 인증코드, 세션키 암호화 등으로 사용자 인증과 데이터의 무결성을 검증하고, 패킷에 대한 접근 통제 정책을 접근 통제 정책 저장소(Access Control Policy Repository: ACPR)를 통하여 확인하고 쿼리 분석기에서 확인한 것을 적용한다. 적절하지 않은 사용자에게는 데이터에 대한 접근을 거절하고, 정의된 정책에 의해 숨겨져야 하는 데이터베이스의 데이터들은 Data Masker에서 마스킹을 과정을 통하여 보호되도록 한다.

여기서 ACS와 데이터베이스 서버간의 네트워크 보안은 안전하다고 가정한다. ACS의 세부 구성요소에 대한 자세한 설명은 다음 표 4와 같다.

표 4에서 참조모니터에 있는 ACPR에는 보안 등급 정책 테이블이 필수적인 요소로 들어가야 한다. 정책에 대한 정의는 일반적인 표현 방식에 따라 다음과 같이 나타낼 수 있다.

Condition → {subject, object, permission(or operation)}

보안 등급 정책 테이블은 효율적인 중요 데이터의 검색 및 처리를 위하여 사용자를 관리하기 위한 사용자 보안 등급 테이블(USL_Table), 데이터를 관리하기 위한 데이터 보안 등급 테이블(DSL_Table)로 정의하여 관리하도록 한다. 이것을 위한 정책은 다음과 같다.

정책 1. 기본적인 보안 레벨 정책

$$[\{USL_k \geq DSL_i\} \cap \{R_i \in USL_k\} \cap \{C_t \in DSL_i\}] \rightarrow \{R_i,\ C_t,\ \{R\}\}$$

사용자 보안 레벨이 데이터 보안 레벨을 지배(dominate)

표 4 접근통제서버(ACS)의 구성요소

이름		설명
Reference Monitor	Packet Manager	· 클라이언트로부터 데이터베이스로 가는 모든 접근을 관리
	Packet Analyzer	· 해쉬값으로 사용자 인증 · 메시지 무결성 검사 · 분석된 데이터베이스의 결과 패킷을 통해 마스킹이 필요한 데이터를 결정
	Query Analyzer	· ACPR에서 접근 통제 정책 가져오기 · 가져온 정책에 따라 패킷 내용의 수행 허용 및 차단 · ACPR에 로그를 기록
Access Control Policy Repository		· 전체 데이터베이스의 통합 관리를 위한 보안 등급 테이블 저장 · 사용자 요구에 의해 정의된 상세 접근 통제 정책을 저장
Data Masker		· Packet Analyzer에서 결정된 데이터를 마스킹 하기
Packet Re-generator		· 패킷 재생성 · 질의에 대한 결과 전달

하면 그 사용자 보안 레벨에 속하는 역할 R은 데이터 보안 레벨을 가지는 데이터 C_t를 read(R)할 수 있다. 권한은 R이 가지는 속성에 따라 데이터를 read(R)할 수 있을 뿐만 아니라 insert(I), update(U), delete(D)도 할 수 있다. 이때는 permission 부분을 '{R_i, C_t, {R, I, U, D}}'와 같이 나타낼 수 있다.

정책 2. 상세 접근 통제를 위한 특정 사용자와 데이터 컬럼에 대한 정책

[2-1] 상세 접근 통제의 표현식 : 항목 = 설정 값

[2-2] 마스킹할 상세 데이터 보호 항목의 표현식 : *columns = (테이블명$_1$: 컬럼명$_1$, 컬럼명$_2$, … 컬럼명$_n$), (테이블명$_2$: 컬럼명$_3$, 컬럼명$_4$, … 컬럼명$_m$)*

상세 접근 통제는 표 3에서 정의한 바와 같이, 주소(IP), 호스트(Host), 데이터베이스 사용자(DbUser), 데이터베이스 테이블(DbTable), 시각(Time)뿐만 아니라, 날짜(Date), 애플리케이션(App), 로그인 사용자(Login-User), 명령어(Cmd), 데이터베이스 인스턴스(DbInst), 데이터베이스 세션(DbSess), 데이터베이스 SQL(DbSql) 등에 대한 정책을 설정할 수 있으며, 마스킹을 할 컬럼은 테이블 별로 구분하여 정책을 설정할 수 있다.

다음 그림 3은 위의 정책들을 적용하여 데이터를 마스킹하기까지 접근 통제 서버의 여러 구성요소들뿐만

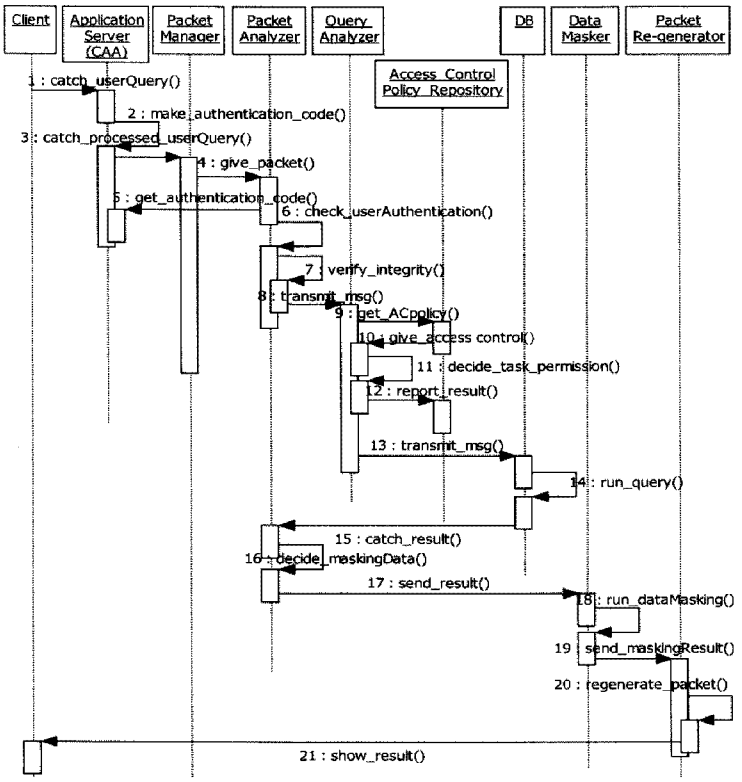

그림 3 제안하는 접근 통제 시스템의 시퀀스 다이어그램

아니라 클라이언트, 데이터베이스를 포함한 전체적인 동작 순서를 UML(Unified Modeling Language)의 시퀀스 다이어그램[14]으로 나타낸 것이다.

ACS는 그림 3의 과정 1~7과 같이 CAA로부터 전달받은 패킷에 대해 사용자 인증과 메시지 인증코드, 사용자ID, 세션고유코드를 가지고 인증받은 접근인지 확인한다. 메시지 무결성 검증이 완료되면, 과정 8에서 쿼리분석기를 통해 쿼리를 분석한다.

과정 9~10에서 ACS는 ACPR로부터 접근 통제정책을 수집 후, 과정 11에서 분석된 쿼리로부터 통제 정책을 적용하여 결과에 따라 작업허용 및 차단을 수행한다. 허가되지 않은 접근을 시도한 경우, 관리자에게 통보된다. 일련의 작업들은 과정 12에서 ACPR에 기록된다.

ACS로부터 작업이 완료된 패킷은 과정 13에서 클라이언트에서 전달받은 원본 그대로 네트워크를 통해 데이터베이스에 전달된다.

과정 14에서 데이터베이스가 작업처리를 하고 결과를 클라이언트에 전달할 때, ACS는 과정 15와 같이 네트워크를 감시하고 있다가 패킷을 취득하여, 패킷 분석기를 통하여 분석한다. 패킷으로부터 데이터 마스킹에 필요한 정보들을 추출한 후, 과정 16에서 해당 결과에 마스킹이 필요한 데이터가 있는지 결정하고, 과정 17~19에서 데이터 마스킹 작업을 수행한다. 마스킹이 필요가 없는 데이터는 원본 그대로 전달된다.

데이터 마스킹 작업이 완료된 패킷은 과정 20에서 변경된 정보를 가지고 패킷을 재생성하고, 마지막으로 과정 21에서 클라이언트에 그 결과를 전달한다.

이러한 과정을 통해서 제안한 모델은 데이터베이스에 접근한 사용자의 질의결과에서 보호되어야 하는 데이터의 노출을 방지할 수 있다.

4.3 상세 접근 통제 적용 시나리오

앞의 '4.2'에서 언급한 정책을 적용하는 다음과 같은 시나리오를 가정해 볼 수 있다. 사용자 10명에 대해서 5개의 역할이 정의되어있다고 하자. 이러한 경우 사용자는 그림 4와 같이 각각의 역할에 한 멤버가 된다.

그리고 이에 해당하는 보안 등급을 다음 그림 5와 같이 가정한다.

앞의 4.2에서 언급하였던 보안 정책에 따라, 사용자 보안 등급이 대상 컬럼의 보안 등급보다 높으면, 즉, USL이 DSL을 지배(dominate)하면, 접근을 허용하도록 한다.

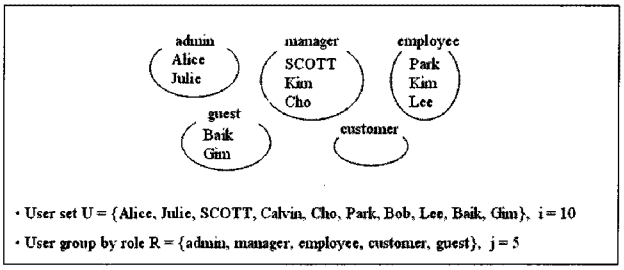

그림 4 사용자 역할 할당의 예

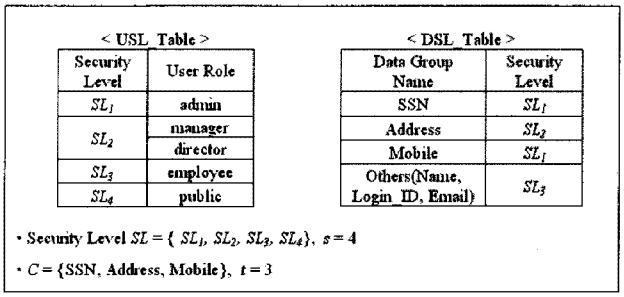

그림 5 사용자 보안 등급과 데이터 보안 등급 테이블의 예

예를 들어 데이터베이스에 다음 표 5와 같은 테이블이 있다고 가정해 보자.

사용자 'SCOTT'이 질의를 통해서 이 테이블이 가지고 있는 컬럼인 Name, SSN, Email을 결과로 받기를 원할 경우, 데이터베이스의 자체의 정책 상 세 컬럼 모두가 결과로 온다고 하더라도, 제안하는 시스템을 통하면 이 컬럼들 중 'SSN'은 마스킹이 된 상태로 사용자에게 보여진다. 왜냐하면 'SCOTT'의 역할이 'manager'이므로 다음 식에서 볼 수 있는 바와 같이 정책 1)을 만족하지 못하기 때문이다.

$$R(SCOTT) = manager,$$
$$USL(manager) = SL_2,\ DSL(SSN) = SL_1$$
$$\therefore DSL(SSN) > USL(manager)$$

또 다른 예로, 만약 사용자가 데이터베이스에 접속하는 과정에서 애플리케이션 서버와 제안 시스템의 인증 코드가 일치하지 않는 경우, 무결성을 만족하지 못하기 때문에 접근 자체가 차단되어 데이터베이스에 접근할 수 없다.

앞에서 언급한 바와 같이 위에서 정의된 정책들은 각각의 데이터베이스에서 적용하는 정책과는 독립적으로 사용된다. 따라서 접근의 전후 처리로 시간 지연이 생기는 문제점이 발생할 수 있지만 마스킹 방법으로 데이터를 보호함으로써 데이터의 암·복호화보다는 지연 시간을 단축시킬 수 있다.

표 5 대상이 되는 테이블의 스키마 예제

Id	Name	SSN	Login_Id	Address	Mobile	Email
1	HONG, Gildong					
2	PARK, Seonghwa					

다음 예제로, 상세 접근 통제를 생각해 보자. 위의 표 5에 대하여 이미 정의되어 있는 사용자 보안 등급 혹은 데이터 보안 등급 이외에 상세 접근 통제를 위한 요구 사항을 다음과 같이 가정한다.

"오전 9시 부터 오후 6시 사이에 사용자 'SCOTT'이 IP '192.168.XXX.XXX'의 컴퓨터 'SJHONG001'을 통하여 CUSTOMER 테이블에만 접속할 수 있도록 한다. 또, 사용자 질의에 대한 결과에서 'SSN', 'ADDRESS', 'MOBILE_NO' 컬럼의 내용을 마스킹하기를 원한다."

위와 같은 상세 접근을 원할 경우, 다음 그림 6과 같은 표현식을 이용하여 정책을 정의할 수 있다.

```
• Protection Expression:
    IP = 192.168.XXX.XXX and
    Host = SJHONG001 and
    DbUser = SCOTT and
    DbTable = CUSTOMER and
    Time = 09:00~18:00
• Masking Columns:
    Columns = (CUSTOMER: SSN, ADDRESS, MOBILE_NO)
```

그림 6 상세 접근을 위한 정책 정의의 예

위의 그림 6과 같이 세부 정책을 정의 했을 때, 사용자 SCOTT이 CUSTOMER에 접근하였고, 사용자 보안 등급 USL이 데이터 보안 등급 DSL과 비교하여 같거나 높다면 SCOTT은 질의한 결과 중 Masking Columns, 즉 SSN, ADDRESS, MOBILE_NO은 마스킹된 상태로 데이터들을 보게 되고 나머지 컬럼들은 실제 데이터들을 볼 수 있다.

5. 데이터베이스 보안 시스템의 구현

데이터베이스 구성 환경에서 클라이언트와 ACS는 Windows 2000과 Oracle Client Release 9버전을 사용하였고, ACPR와 통제 대상 데이터베이스는 Unix와 Oracle9i Enterprise Edition Release 9를 사용하였다. 또한 개발 구성 환경은 Windows 2000과 Visual C++6.0을 사용하였다. 자세한 구현 환경은 다음 표 6, 7과 같다.

네트워크상에서 패킷 취득 및 데이터 마스킹을 하기 위해서, Vadim V.Smirnov가 개발한 'Windows Packet Filter Kit'을 이용한다. 'Windows Packet Filter Kit'은 윈도우의 TCP/IP Layer와 Network Adapter 사이의 NDIS(Network Driver Interface Specification) 후킹을 통하여 패킷을 취득할 수 있는 라이브러리이다[15,16]. 또한 취득된 SQL의 파싱을 위해서 'sqlparser.com'의 'General SQL Parser'를 이용한다[17].

5.1 패킷 분석

본 논문에서는 네트워크상에서 클라이언트로부터 통제 대상 데이터베이스로 흐르는 패킷 중 TCP 데이터 영역을 수집하여 분석한다[18]. 그 다음, 수집된 정보를 ACPR의 사용자에게 할당된 접근 통제정책과 비교하여, 해당 데이터베이스로 접근할 수 있는지 판단한다. 따라서 패킷 분석의 과정이 필수적이다.

서버에는 클라이언트 프로그램과 통신하기 위해서, 네트워크 프로토콜(TCP/IP)을 사용한 외부사용자의 접근과 Bequeath protocol(BEQ)를 이용한 내부 사용자의 접근을 나누어 적용하지만 본 논문에서는 데이터베이스에서 외부 사용자의 접근인 TCP/IP에 대해서만 초점을 맞추며, 쿼리를 수행할 때 모니터링한 패킷 내용은 Ethereal[19]을 이용하여 분석한다. 분석한 패킷의 내용은 관련 샘플 100개를 기반으로 분석한 결과이며, 결과 응답 패킷은 SQL 쿼리 결과로 컬럼 정보를 가지고 있는 select문을 기반으로 분석하였다.

다음 그림 7은 연결된 프로세스를 통하여 사용자 SQL 질의를 요청하고 응답받는 패킷을 Ethereal을 통하여 모니터링 한 화면이다.

그림에 표시된 부분에서 알 수 있듯이 그림 7은 사용자가 다음과 같은 쿼리를 요청한 것이다.

```
SELECT id, name, jumin_no, login_id, address,
        mobile_no, email_addr
FROM    Customer
```

또, 그림 8은 그림 7에서 질의한 내용에 대한 결과를

표 6 개발 구성 환경

구분	Library	OS	Complier
CAA	Windows Packet Filter Kit (Packet capture & modify library)	Windows 2000	VC++ 6.0
ACS	General SQL Parser (SQL Parser)		

표 7 데이터베이스 구성 환경

구분	OS	Oracle Server Version	Oracle Client Version
클라이언트/접근 통제 서버(ACS)	Windows2000	-	Release 9.2.0.1.0
접근 통제 정책 저장소(ACPR)	Unix	Oracle9i Enterprise Edition Release 9.2.0.1.0	
통제 대상 데이터베이스			

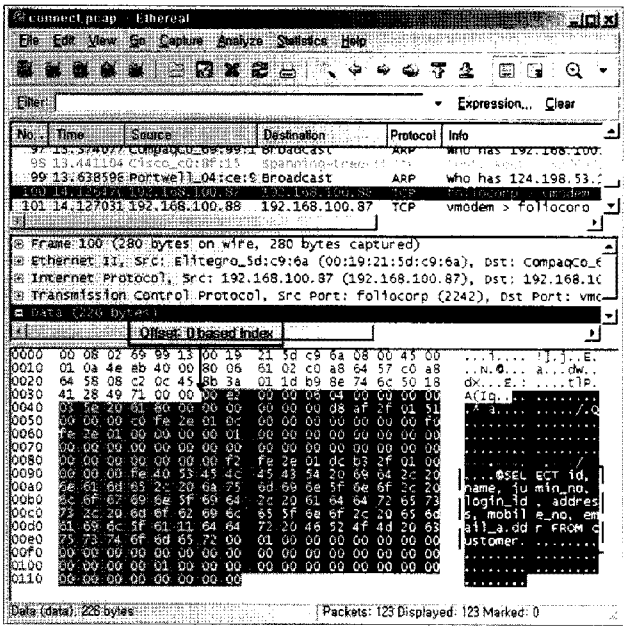

그림 7 사용자 SQL 쿼리 요청 패킷

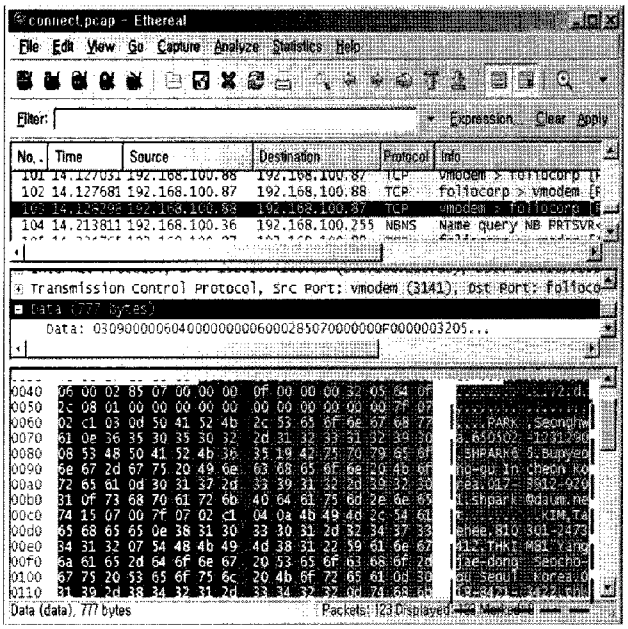

그림 8 서버의 쿼리 결과 전송 패킷

표 8 서버 쿼리 결과 전송 패킷 분석의 중요 부분

Offset (시작위치)	길이 (byte)	설명
offset + 0	2	TCP Payload의 전체 길이. 데이터가 긴 경우, IP Layer에서 데이터가 fragmentation됨.
offset + 4	2	데이터베이스 서버에서 클라이언트로 응답을 보낼 시, 'offset+4'가 '0×06'이고, 'offset+5'가 '0×04'이면, 클라이언트의 쿼리 실행 결과를 보냄.
offset + 41	1	[SELECT 쿼리 시] 쿼리 결과에 대한 컬럼 수를 가짐. [DML, DDL일 경우] 0×00의 값을 가짐.
offset + 46	1	**컬럼정보 시작코드** 0×01의 값을 가짐. 각각의 컬럼정보는 0×01로 시작.
offset + 85	1	컬럼명길이 L => len(colname)
offset + 86	L	컬럼명
offset + 85 + L + 9	1	다음 컬럼 정보 시작 코드 0×01의 값을 가짐.

클라이언트로 보내주는 패킷을 나타낸 부분이다. 서버의 결과를 전송하는 패킷은 아직 정책이 적용되지 않은 상태이므로 위의 그림 8과 같이 결과에 해당하는 모든 데이터들을 포함하고 있다. 표 8은 위의 그림에 해당하는 패킷 분석 내용 중 중요 부분을 나타낸 것이다.

그림 8과 표 8에서 볼 수 있는 바와 같이 클라이언트와 서버 사이에서 패킷에서 질의에 대한 결과를 분석할 수 있으며, 본 논문에서는 접속한 사용자의 보안 등급에 맞지 않는 데이터를 보호하기 위해 '4.2 제안한 데이터베이스 보안 시스템'에서 이미 언급한 바와 같이 컬럼 단위로 마스킹을 수행한다.

5.2 제안 시스템의 구현 및 실험

데이터의 보안 방법은 '데이터 마스킹'을 이용한다. 이 방법은 데이터베이스 서버의 수행 결과를 분석하여, 마스킹이 필요한 데이터인지를 판단한다. 그런 다음, 마스킹이 필요한 데이터의 경우, 첫 번째 문자만 '*'로 처리하고, 나머지는 'Null' 스트링으로 처리하는 방법으로 '데이터 암호화' 방법과는 다르다. 데이터 암호화의 경우, 암호화된 데이터는 암호화 알고리즘에 따라 원래 데이터 길이보다 길어지는 등의 경우가 발생하게 되는데, 이는 데이터베이스 서버에서 구조화한 프로토콜의 내용 변경을 의미하고, 암호화로 인해서 길이가 변경된다면, ACS 에서 오라클의 프로토콜에 맞도록 패킷 자체를 완전히 새로 만들어야 하는 작업이 필요하다. 또한 TCP 헤더의 페이로드 길이정보도 변경되어야 한다. 따라서 제안하는 시스템에서는 보호가 필요한 데이터에 대해 단순히 첫 문자만을 '*'로 처리하고 나머지는 'Null' 스트링 처리하는 마스킹 방법이 처리 비용 면에서 보다

효율적이며, 컬럼에 대한 접근권한이 없는 사용자에게 단순히 데이터를 숨기는 목적에 더욱 적합하다.

그림 9는 '4.3 상세 접근 통제 적용 시나리오'에서 언급한 시나리오에 기반하여 접근에 대한 상세 접근 통제 정책과 마스킹 컬럼들을 설정하는 화면이다.

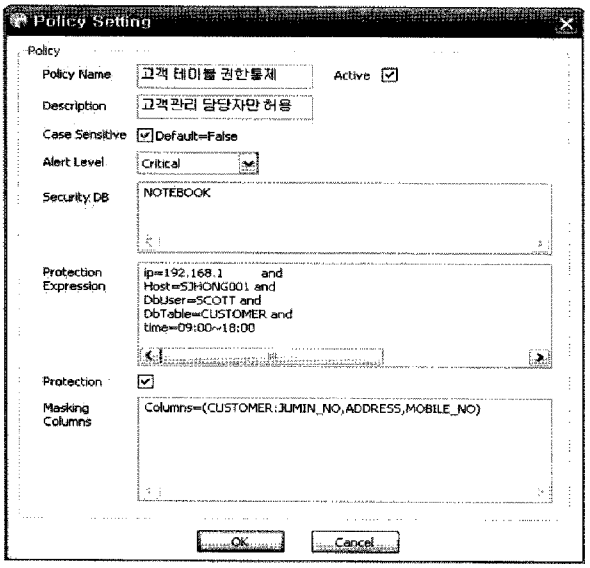

그림 9 접근 통제 정책 설정 화면

그림 9에서 볼 수 있듯이 통제하고자 하는 부분을 'Protection Expression'에 설정할 수 있으며, '3. 사전 정의'에서 언급한 표 3의 모든 항목에 대해서 통제 정책을 설정할 수 있다. 위의 정책 설정 화면에서는 결과데이터 중 JUMIN_NO, ADDRESS, MOBILE_NO에 대해서 마스킹하도록 설정하였다. 다음 그림 10, 11은 위의 정책에 대해서 결과를 수행한 화면이다.

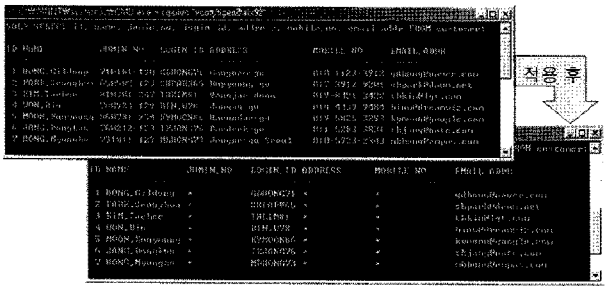

그림 10 SQL*Plus 컬럼마스킹 적용 전, 후 수행결과

그림 10과 그림 11은 제안된 보안모델 적용한 뒤, SQL*Plus와 TOAD version 9.0을 이용하여 각각 클라이언트에서 SQL을 수행한 테스트 결과이다. 쿼리 결과로 JUMIN_NO, ADDRESS, MOBILE_NO 컬럼 데이터가 '*'로 처리되었음을 확인할 수 있다.

다음 그림 12는 구현된 CAA와 ACS가 서비스 형태로 실행된 모습이다.

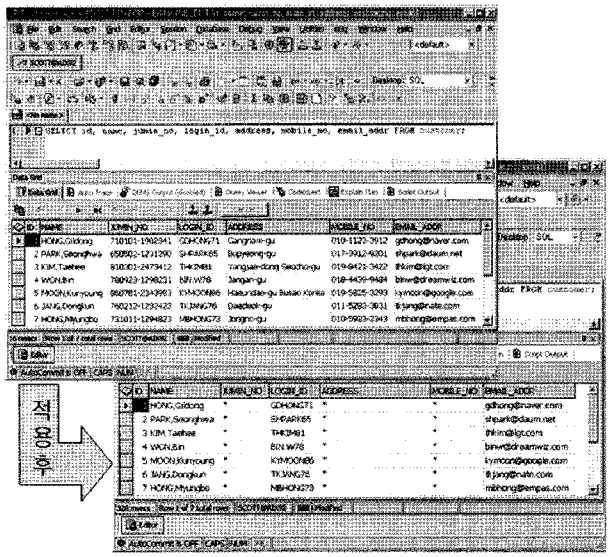

그림 11 TOAD 컬럼마스킹 적용 전, 후 수행결과

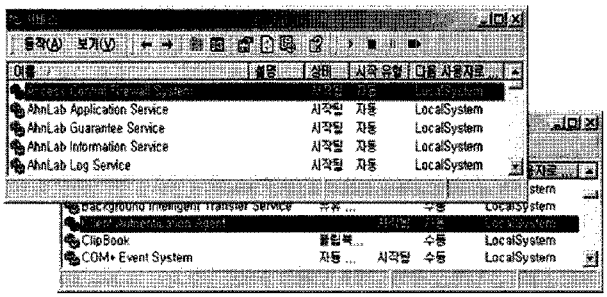

그림 12 CAA와 ACS 실행 화면

차후 데이터베이스의 관리를 위해서 제안한 데이터베이스 보안 시스템을 사용할 경우 위의 그림과 같이 CAA와 ACS를 서비스 형태로 올려놓고 사용할 수도 있도록 구현하였다.

6. 비교 평가

본 절에서는 기존에 연구된 데이터베이스 보안 시스템[7]과 제안하는 접근 통제 방식에 대해 비교한다. 기존의 다양한 크기의 데이터 그룹을 위한 데이터베이스 보안 시스템에는 관련 연구에서 언급한 바와 같은 문제점이 있었다. 반면에 제안하는 방법은 데이터에 대한 보안을 지원하는 동시에 데이터베이스의 통합 관리와 자세한 접근 통제가 가능한 장점이 있다. 이와 관련된 더욱 자세한 분석은 다음 표 9와 같다.

표 9에서 데이터 자체에 대한 보안 지원 항목은 데이터베이스 보안을 위해서 반드시 보장되어야 할 특징이다. 왜냐하면 이 항목이 보장되지 않을 경우 데이터베이스 보안 자체가 의미가 없기 때문이다. 기존의 방법에서는 데이터 각각에 대한 어떤 조치가 되어 있지 않기 때문에 데이터 전송 시 패킷이 도난될 경우 데이터의 내

표 9 기존 방법과 제안 방법의 비교

항목	다양한 크기의 데이터 그룹을 위한 기존 방법	제안 방법
데이터 자체에 대한 보안 지원	불가능	가능
정책 처리에 대한 시간 지연	있음	없음
이질적인 데이터베이스의 통합 관리	불가능	가능
결과의 특정 컬럼에 대한 통제	가능	가능
특정 시간/ 특정 SQL에 대한 통제	불가능	가능
데이터 인덱싱	가능	가능

용이 그대로 노출될 수 있지만, 제안하는 방법은 마스킹 처리를 하기 때문에 패킷이 도난당한다 하더라도 중요 데이터 내용을 노출하지 않을 수 있다.

또한 기존 방법은 두 단계에 거쳐 세 가지의 정책을 만족시켜야 하므로 데이터 처리에 시간이 걸리는 반면에 제안한 모델은 데이터를 암복호화 하는 대신 마스킹 하는 방법을 적용하므로 지연 시간이 거의 없으며, 그 근거는 아래의 그림 13의 실험 결과에서 볼 수 있다.

이질적인 데이터베이스의 통합 관리에 대해서는 기존 방법에서는 권한 부여가 모두 각각의 데이터베이스에서 일어나기 때문에 권한 부여 구조를 확장할 경우 서로 다른 데이터베이스들 간의 권한 충돌 등의 문제가 발생하고 통합적으로 정책을 반영하지 못한다. 그러나 제안한 방법에서는 적용 대상이 되는 데이터베이스 혹은 데이터베이스 팜의 바로 앞단에 처리 시스템을 두어 데이터베이스에 독립적으로 작업을 수행할 수 있다.

뿐만 아니라 기존의 방법은 세 가지의 정책(MAC, RBAC, S-DAC)을 사용함으로써 위의 표에서 볼 수 있듯이 사용자 쿼리 결과의 특정 컬럼에 대해서는 통제가 가능하도록 하였지만, 특정 시간 혹은 특정 SQL 쿼리에 대한 자세한 통제는 할 수 없다. 그러나 제안한 시스템에서는 패킷 모니터링을 이용하여 분석된 결과를 바탕으로 특정 컬럼 뿐만 아니라 특정 시간, 특정 SQL, 특정 사용자 등에 대한 통제를 적용할 수 있다. 이것은 앞서 서론에서 언급한 문제점 중의 하나인 다양한 접근 요구에 대한 해결방안이 될 수 있다.

마지막으로 데이터 인덱싱은 데이터 암호화 방법을 쓸 경우 데이터의 비밀성은 보장을 할 수 있지만, 데이터 암복호화에 시간이 지연될 수 있고, 데이터 인덱싱이 어려운 단점이 있었다. 이를 해결하기 위해 기존의 연구에서는 데이터의 비밀성은 보장하지 않는 상태에서 여러 가지 정책들을 적용함으로써 특정한 데이터 컬럼에 대한 접근 통제는 가능하도록 하는 것이므로 데이터를 암호화 하지는 않는다. 따라서 인덱싱이 가능하다. 제안한 방법에서는 데이터를 암호화하지는 않지만 네트워크로 데이터 패킷이 전송되기 직전에 마스킹을 작업을 수행하므로 데이터의 비밀성을 보장하는 동시에 데이터베이스에서 인덱싱이 가능하다.

마스킹 시 발생할 수 있는 시간의 지연 정도를 객관적으로 살펴보기 위하여 본 논문에서는 다음과 같은 실험을 수행하였다. 실험은 총 8개의 컬럼으로 이루어진 테이블의 데이터를 검색하는 형식으로 진행하였으며, 1,000건씩 1,000단위로 10,000까지 10번을 수행하였다. 마스킹을 하는 데이터는 총 4개의 경우로 분리하여 실행하였으며, 그 종류는 마스킹한 데이터가 없을 때, 하나의 컬럼을 마스킹할 때, 두 개의 컬럼을 마스킹할 때, 세 개의 컬럼을 마스킹할 때와 같이 마스킹 컬럼을 증가하면서 진행하였다. 이 때, 여러 가지 타입의 데이터를 실험하기 위해서 컬럼 A는 숫자 형식의 데이터로, 컬럼 B는 이미 암호화된 형식을 가지는(예를 들어 패스워드) 데이터로, 컬럼 C는 텍스트 형식을 가진 데이터로 구성하였으며, 각각의 데이터 길이의 평균은 14byte, 32byte, 19.875byte로 구할 수 있었다.

다음 표 10은 마스킹 데이터의 각각의 평균 길이를 보여주고 있다.

표 10 마스킹 데이터의 평균 길이

마스킹 데이터	A	A+B	A+B+C
마스킹 데이터 평균길이 합산	14	46	65.875

(길이단위 : byte)

위의 표 10의 데이터를 가지고 실험한 경과는 다음 그림 13과 같다.

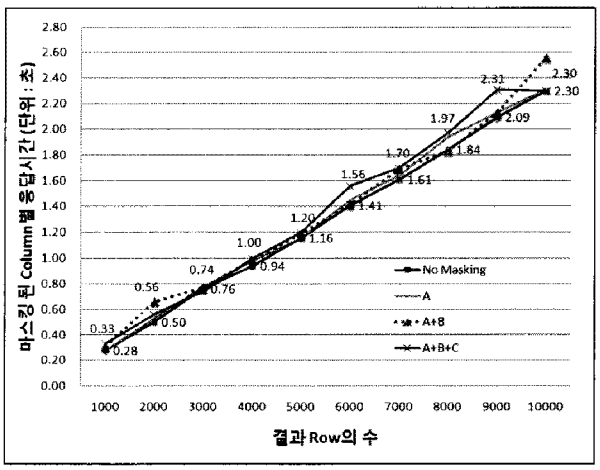

그림 13 마스킹된 컬럼별 응답 시간의 그래프

위의 그림 13에서 위쪽에 숫자로 표시된 데이터는 'A+B+C'에 대한 마스킹 응답 시간의 결과이고, 아래쪽에 숫자로 표시된 데이터는 마스킹을 하지 않고 데이터를 처리한 결과이다. 여기에서 볼 수 있듯이, 전반적으로 마스킹을 하지 않은 데이터가 응답 시간이 적게 걸리긴 하지만 마스킹된 컬럼이 많을수록 처리속도가 항상 증가되지는 않음을 알 수 있다. 심지어 어떤 경우에는 데이터의 마스킹 처리결과가 마스킹 하지 않는 처리결과보다 빠르기도 하다. 이것은 마스킹이 시간 지연에 영향을 미치지 않고 있다는 것을 의미한다.

또한 위에서 언급한 개선점과 더불어 적용 대상이 되는 데이터베이스 혹은 데이터베이스 팜의 바로 앞단에 위치하여 접근 시간, 특정 SQL 제한, 결과 row에 대한 권한 부여 등의 접근 통제가 가능하고, 데이터베이스와는 독립적으로 정책 부여와 관리가 이루어지기 때문에 권한 구조를 확장하거나 통합적인 관리를 하는데 높은 수행 능력을 보일 수 있다.

다음은 기존의 데이터베이스 보안 시스템과 제안한 시스템 각각에서 데이터베이스의 권한에 따른 데이터베이스 접근 통제 여부를 비교하였다. 먼저, 임의의 DB계정과, SQL을 생성하여 보안조건과 권한구분 항목을 두었다. 그런 다음 접근 통제의 여부를 확인해 보았다. 접근 통제 여부를 비교 분석한 결과는 표 11과 같다. 표에서 Yes는 SQL 작업 발생 별로 접근이 모두 통제가 됨을 의미하며, No는 통제가 되지 않음을 의미한다.

표 11에서의 비교 결과를 보면, 기존 시스템의 경우 인가된 사용자와 비인가된 사용자의 구분 없이 보안조건에 대해서 통제가 되지 않았다. 그러나 제안된 접근 통제 시스템의 경우, 보안조건에 대해서 100% 정확한 제어를 수행하였다. 특히, 시간대별 접근 보안조건을 보면, 각 사용자 별로 할당된 데이터 접근 시간 이외의 접근에 대해서 접근 통제를 정확히 수행한 것을 볼 수 있다. 또한 컬럼별 통제가 불가능한 일반 질의의 접근인 경우에도, 제안한 시스템에서는 데이터 마스킹이 수행된 결과를 확인할 수 있다.

7. 결론 및 향후 연구

현재의 데이터베이스 권한부여 방식은 SQL 문장 실행에 대한 권리로 데이터베이스 및 데이터베이스 오브젝트에 접근할 수 있는 권한을 사용자에게 부여하는 방식이다. 특히 가장 최근에 연구된 다양한 크기의 데이터 그룹을 위한 데이터베이스 보안 시스템[7]에서는 데이터 크기와 관련된 몇 가지 요구사항에 초점을 맞추고 있기는 하지만 데이터 자체의 보안은 보장하지 못하고 있었다. 또한 데이터 그룹을 정의할 때 오버헤드가 발생할 수 있으며 정책의 추가 시 시스템의 성능을 저하시키거나 정책의 중복 현상이 나타날 우려가 있었다. 뿐만 아니라, 여러 가지 종류의 데이터베이스를 통합적으로 관리할 수 없었다.

본 논문에서는 이를 해결하기 위해서 클라이언트에서 데이터베이스 서버로 요청되는 네트워크상의 패킷 분석을 통한 데이터베이스의 접근 통제 시스템을 제안하였다. 제안된 보안 시스템에서는, 데이터베이스에 접근하는 사용자 정보의 위변조를 방지하기 위해서 공개키 인증방식을 사용하였고, 클라이언트로부터 요청된 SQL의 무결성을 확보하기 위해서 클라이언트 인증 에이전트(CAA)와 접근 통제 서버(ACS) 사이에, 암호화된 SQL의 인증코드를 교환하였다. 또한, 강력한 컬럼별 접근 통제정책에 따른 SQL 작성의 불편함을 최소화 시키고, 권한별 데이터 접근을 수행하기 위해서 데이터 마스킹 기법을 구현하였다. 뿐만 아니라 이런 접근 통제의 결과로 데이터베이스 접근에 대한 모든 로그를 남김으로써, 사후추적에 대한 정보를 제공하도록 하였다.

결론적으로 본 논문에서 제안한 데이터베이스 보안 시스템은 마스킹 기법을 사용함으로써 데이터 자체에 대한 보안과 특정 컬럼, 특정 시간, 특정 SQL에 대한

표 11 데이터베이스 접근 통제 비교

보안조건	권한구분	기존 시스템		제안 시스템	
		일반 컬럼	통제 대상 컬럼	일반 컬럼	통제 대상 컬럼
일반 접근	AU	No	No	No	No
	NAU	Yes	No	Yes	Yes
시간대별 접근	AU	No	No	Yes	Yes
	NAU	No	No	Yes	Yes
DB계정도용 접근	AU	No	No	Yes	Yes
	NAU	No	No	Yes	Yes
IP주소 도용접근	AU	No	No	Yes	Yes
	NAU	No	No	Yes	Yes
통제 컬럼 접근 (Data Masking)	AU	No	No	No	Yes
	NAU	No	No	No	Yes

AU: 인가된 사용자(Authorized User) NAU: 비인가된 사용자(Non Authorized User)

상세 접근 통제가 가능할 뿐만 아니라 동시에 인덱싱이 가능하게 되었다. 또한 패킷 분석에 기반한 접근을 통해서 접근 통제에 대한 효율적인 작업 수행이 가능하고, 데이터베이스 보안 시스템을 데이터베이스 팜 앞에 위치하도록 함으로써 이질적인 데이터베이스 정책들의 통합적 관리가 가능하도록 하였다.

향후 연구로, 데이터베이스의 종류에 관계없이 모든 데이터베이스에 적용할 수 있는 표준적인 접근 통제정책 수립과, 데이터베이스에 대한 강력한 접근 통제를 완성시키기 위한 완벽한 프로토콜 분석이 필요하다. 또한, 감시 혹은 관리 데이터베이스의 수가 많을 때 효율적인 접근 통제 정책 관리에 대한 개선 연구가 필요하다.

참 고 문 헌

[1] J. J. Borking, B. M. A. Van Eck, P. Siepel, "Intelligent Software Agents: Turning a Privacy Threat into a Privacy Protector," *Information and Privacy Commissioner of Ontario*, 1999.

[2] H. G. Lee, S. M. Lee, T. Y. Nam, "Database Encryption Technology and Current Product Trend," *Electronics and Telecommunications Trend Analysis*, vol.22, no.1, pp.105-113, 2007. (In Korean)

[3] G. I. Davida, D. L. Wells and J. B. Kam, "A database encryption system with subkeys," *ACM Transactions on Database systems*, vol.6, no.2, pp.312-328, 1981.

[4] D. F. Ferraiolo, *Role-Based Access Control*, Artech House, Computer Security, 2003.

[5] M. Piattini and E. Fernandez-Medina, "Secure databases: state of the art," Security Technology, *Proc. of the IEEE 34th Annual 2000 International Carnahan Conference*, pp.228-237, 2000.

[6] Y. Elovici, R. Waisenberg, E. Shmueli, and E. Gudes, "A Structure Preserving Database Encryption Scheme," *Proc. of the Secure Data Management in a Connected World 2004 (SDM 2004), LNCS 3178*, pp.28-40, 2004.

[7] M. A. Jeong, J. J. Kim, Y. Won, "A Flexible Database Security System Using Multiple Access Control Policies," *LNCS 2736*, pp.876-885, 2003.

[8] J. Gennick, *Oracle SQL*Plus: The Definitive Guide*, O'Reilly, 2005.

[9] B. Scalzo, D. Hotka, *Toad Handbook*, Sams Publishing, 2003.

[10] D. Steiner, V. Moore, *Oracle9i Net Services Administrator's Guide, Release 2 (9.2), Part no. A96580-02*, Oracle Corporation, 2002.

[11] D. Keesling, J. Womack, *Oracle9i Database Administration Fundamentals II, Production 2.0, D37492*, Oracle Corporation, 2002.

[12] S. Y. Kim, G. W. Nam, S. C. Kim, "Filtering Unauthorized SQL Query By uniting DB Application Firewall with Web Application Firewall," *Proc. of the Korea Institutes of Information Security and Cryptology Conference*, pp.686-690, 2003. (In Korean)

[13] G. O. Jang, H. O. Koo, C. S. Oh, "Detection of Internal Illegal Query Using Packet Analysis," *Proc. of the Korean Society Of Computer And Information*, vol.10, no.3, pp.259-265, 2005. (In Korean)

[14] G. Booch, J. Rumbaugh, I. Jacobson, *The Unified Modeling Language User Guide: second edition*, Addison-Wesley, 2005.

[15] K. Molnár, M. Kyselák, "Application Programming Interface offering classification services to end-user applications," *Proc. of the International Conference on Systems and Networks Communication 2006 (ICSNC'06)*, p.49(1page), 2006.

[16] Windows Packet Filter Kit, http://www.ntkernel.com

[17] SQL Parser, http://www.sqlparser.com

[18] Kozierok, M. Charles, *TCP/IP Guide*, O'Reilly & Associates Inc, 2005.

[19] Ethereal, Inc., http://www.ethereal.com.

조 은 애

정보과학회논문지 : 데이타베이스
제 36 권 제 2 호 참조

문 창 주

1997년 고려대학교 컴퓨터학과 학사. 1999년 고려대학교 컴퓨터학과 석사. 2004년 고려대학교 컴퓨터학과 박사. 2005년 고려대학교 정보보호대학원 연구교수. 2005년 건국대학교 컴퓨터응용과학부 컴퓨터시스템전공 조교수. 2006년~현재 건국대학교 공과대학 항공우주정보시스템공학과 조교수. 관심분야는 접근제어, 권한부여, RBAC, 프라이버시, 유비쿼터스 보안, 임베디드 시스템

박 대 하

1992년 고려대학교 컴퓨터학과 학사. 1994년 고려대학교 컴퓨터학과 석사. 2004년 고려대학교 컴퓨터학과 박사. 1999년~2003년 (주)시큐리티테크놀로지스연구소장. 2003년~현재 한국디지털대학교 디지털정보학과 교수. 관심분야는 XML 보안, 보안 프로토콜, 이동코드보안, 임베디드 시스템보안

홍 성 진

1999년 한밭대학교 전자계산학과 학사 2008년 고려대학교 컴퓨터정보통신대학원 석사. 2003년~현재 주)웨어밸리 기술연구소 책임연구원. 관심분야는 데이터베이스, 보안, 소프트웨어 공학

백 두 권

정보과학회논문지 : 데이타베이스
제 36 권 제 2 호 참조