한국ITS학회논문지
제 7권, 제 4호 (2008년 8월)
pp. 97~106

패킷 인터셉터를 이용한 DB 쿼리 수집 및 분석기 구현

An Implementation of DB Query Collecting and Analyzer Using Packet Interceptor

임 재 덕* 임 성 한** 백 남 철***
(Jae-Deok Lim) (Sung-Han Lim) (Nam-Chul Baek)

요 약

본 시스템은 ITS 정보 감사를 지원하기 위한 내부 통제 시스템의 일부이다. 이 시스템은 중요 도로정보 시스템 서버에 접근하는 패킷을 가로챈 후 SQL 쿼리를 분석하여, 접속한 사용자 정보와 데이터베이스에 접근하려는 DB 정보, 데이터에 대한 조회, 변경, 삭제 정보를 로그로 기록한다. 이 로그 정보는 사용자의 내부 통제 및 데이터베이스 접근 증거로 사용될 수 있다.

Abstract

This proposed system is a part of internal control system that national highway need to support their ITS information audit. This paper explains the design and implementation of a packet interceptor and a DB query analyzer. The packet interceptor sniffs users' query packets, and then the DB query analyzer parses the SQL queries and stores the users' DB access information such as SQL queries, access data and changing data. The information may be used as the evidences on internal control of users and users' accesses.

Key words: ITS, log machine, packet interceptor, internal control system, Query analyzer, computer forensic

Ⅰ. 서 론

1990년대 이후 지속적으로 확대 구축된 지능형 교통 체계(Intelligent Transportation Systems, ITS)로 인해, 국내 교통 분야에서는 양질의 교통자료를 축적할 수 있게 되었다. 수집된 자료는 교통시설의 계획, 설계, 운영 등에 유용하게 활용되고 있으며, 자료 수집의 어려움으로 인해 불가능하게 여겨졌던 다양한 교통 분석이 가능하게 된 것은 학문적으로 큰 성과라 할 수 있다.

ITS에서 제공되는 교통정보는 차량검지기로부터 관측된 원자료(raw data)의 분석·처리를 통해 이루어지기 때문에, 신뢰성 있는 교통정보를 제공하기 위해서는 신뢰성 있는 관측 자료의 수집이 전제되

* 주저자 : 한국건설기술연구원 첨단도로교통연구실 연구원
** 공저자 : 한국건설기술연구원 첨단도로교통연구실 연구원
*** 공저자 : 한국건설기술연구원 첨단도로교통연구실 선임연구원
† 논문접수일 : 2008년 4월 16일
† 논문심사일 : 2008년 5월 20일
† 게재확정일 : 2008년 6월 17일

어야 한다.

정확한 데이터 수집을 위해 가장 필요로 하는 요소 기술은 컴퓨터 포렌식(Computer Forensic) 기술이다[1]. 디지털 시스템에서 컴퓨터 포렌식의 역할은 이 시스템을 가지고 이루어지는 모든 사용자(또는 프로그램) 행위에 대하여 정확한 자료를 수집하는 것을 말하기 때문에 정보 처리 및 수집에 대하여 "누가(who), 언제(when), 어디서(where), 무엇을(what), 어떻게(how)" 접근하였는가에 대한 로그 기록을 저장하는 것부터 출발해야 한다. 컴퓨터 포렌식 기술 적용의 대상은 ITS 시스템의 데이터베이스다. 따라서 데이터베이스에 접근하는 내부 사용자에 대한 관리 및 감독을 위한 시스템이 필요하다.

본 논문에서 제시하는 시스템은 ITS 시스템의 정확한 관리 및 감독을 위한 내부 통제 시스템의 일부로서, 중요 데이터베이스 서버에 접근하는 패킷(Packet)을 가로챈 후 DB 쿼리(Query) 패킷을 분석하여 접속한 사용자 정보와 데이터베이스에 접근하려는 데이터의 정보 및 데이터에 대한 변경, 삭제 정보를 로그로 기록하여 사후에 이를 사용자의 접근 증거로 사용하고자 함이다. 만약 접속한 사용자가 내부 정보를 불법적으로 유출하여 피해를 입히는 경우에 로그 기록을 감사 증거로 활용할 수 있으며, 악성 사용자에 대한 사전 차단 정보로 방화벽이나 보안 OS에서 사용할 수 있다[2].

Ⅱ. 이론 고찰

1. 컴퓨터포렌식

1980년대 중반부터 디지털 증거의 보존, 신원확인, 증거 확보 등에 관한 기술을 다루기 시작하였으며 법적인 문제와 연관된 학문으로 발전하기 시작하였다. 최근 컴퓨터가 일상생활에 밀접하게 사용되면서 컴퓨터에 저장되어 있는 자료가 법정에서 다루어지는 경우가 많이 발생하여 이와 관련된 분야를 컴퓨터 포렌식이라고 부른다. 컴퓨터 범죄 발생 시 책임을 규명하기 위해서 시스템의 접속, 프로그램의 실행 및 처리 결과, 시스템 자원의 접근 내역 등의 휘발성, 비휘발성 로그 기록을 확보, 분석함으로써 불법 사용자를 식별하여 이 증거를 법정에서 증거로 채택할 수 있도록 지원하는 기술이다.

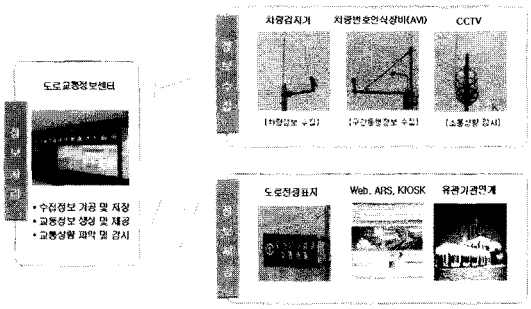
결국 컴퓨터 포렌식은 '정보처리 기기를 통하여 이루어지는 각종 행위에 대한 사실 관계를 확정하거나 증명하기 위해 행하는 각종 절차와 방법'이라고 정의하고 있다. 그러므로 컴퓨터 포렌식은 단순히 과학적인 컴퓨터 수사 방법 및 절차뿐만 아니라 법률, 제도 및 각종 기술 등을 포함하는 종합적인 분야라고 할 수 있다. 또한 컴퓨터 포렌식 기술은 '컴퓨터를 매개로 이루어지는 행위에 대한 법적인 증거 자료를 확보하기 위하여 컴퓨터 저장 매체 등의 컴퓨터 시스템과 네트워크로부터 자료를 수집, 분석 및 보존 절차를 통하여 법적 증거물로서 제출 할 수 있도록 하는 일련의 행위'로 정의하고 있다[3].

2. 데이터베이스 보안

최근 내부자에 의한 기업 기밀정보 누출 사건이나 고객 개인정보 누출 사건 등이 연이어 발생함에 따라, 내부 데이터 관리의 허술함이 그대로 드러나고 있다. 이에 대부분의 기업에서 내부 데이터 누출 방지를 위한 기술적 대책 마련에 고심하고 있으나, 대부분 운영체제와 네트워크 계층에서의 침입탐지 측면에 국한되어 있는 것이 현실이다. 그러나 최근 일부 기업을 중심으로 내부자에 의한 데이터 누출을 막기 위해 데이터베이스 자체에 대한 보안 대책을 수립하는데 노력하고 있다[4].

내부 정보 자산 보호 방안으로는 기존의 법체계에서의 저작권 보호를 들 수 있으나 이 경우 법적인 문제는 사후의 구제 수단과 실제적인 방지책으로는 한계가 있다. 이러한 문제를 해결하기 위해서 지적 재산권의 컴퓨터 사용에 대한 법적인 증거 개념인 컴퓨터 포렌식 기술이 필요하다[5].

여러 가지 증거 수집 방법 중에서 본 시스템은 네트워크 패킷을 수집하여 데이터베이스 쿼리를 분석하는 방법으로 이 방법은 "누가, 언제, 어디서, 무엇을, 어떻게" 접근하였는지 모니터링 및 추적을 통하여 내부 정보 관리 · 감독에 대한 시스템을 구현하는데 필요하다.

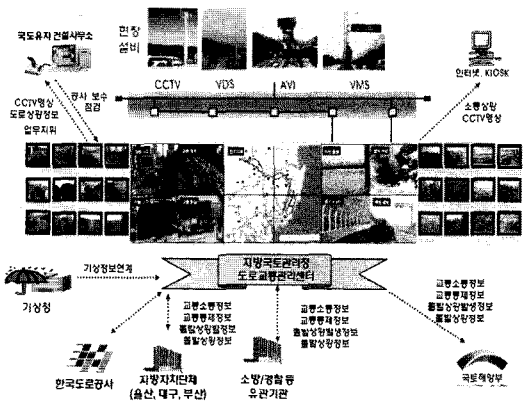


<그림 1> 국도 ITS 정보 처리 체계
<Fig. 1> Information system of national highway ITS

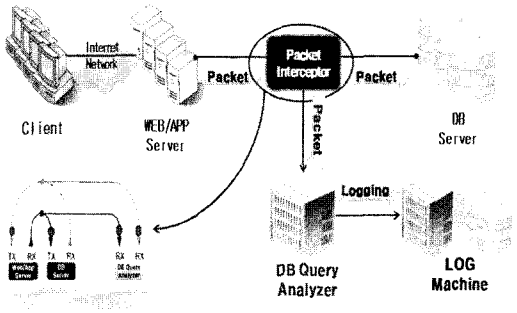
Ⅲ. 국도 ITS 시스템

국도 ITS에서 수집된 도로교통정보는 다양한 정보 제공 매체를 통해 <그림 1>과 같이 제공되고 있다.

정보 제공 매체로는 VMS(가변정보표지판), 건교부 및 지방국토관리청 도로교통정보센터의 인터넷 홈페이지, 자동응답전화 서비스(1333), 방송국 등이 활용되고 있다. 현재 민간업자, 방송사 및 공공기관으로부터 국도 교통정보 연계 및 제공 요청이 증가하고 있는 상황이다. 국토해양부는 1995년 9월 한국도로공사와 정보연계를 시작으로 타 기관과 도로교통정보를 연계하고 있다. 현재 한국도로공사, 교통방송, 경찰청, 지자체 등과 정보를 <그림 2>와 같이 연계하고 있다.



<그림 2> 국도 ITS 시스템 연계
<Fig. 2> Interworking of national highway ITS system



<그림 3> 쿼리 수집 시스템의 동작 원리
<Fig. 3> Operating principles of Query collection system

Ⅳ. 시스템 설계 및 구현

본 논문에서 구현한 시스템은 ITS 시스템의 관리 및 감독을 위한 로그 머신을 설계하는 것으로 설명할 수 있다. 이를 위해서는 네트워크에서 수집된 패킷 정보를 근거로 하여 DB 접근에 대한 쿼리 정보를 수집하여야 한다. 이를 바탕으로 각 서버 내에 접근 통제와 함께 상세 정보의 로깅을 실행하여, 로그 자료를 바탕으로 정확한 관리·감독을 수행할 수가 있다. 본 논문에서 구현한 시스템은 패킷 인터셉터와 DB 쿼리 분석기로 나눌 수 있고 그 동작 원리는 <그림 3>과 같다.

패킷 인터셉터에서 웹 서버와 DB 서버 간의 패킷을 가로채어 가져오면 DB 쿼리 분석기에서 이를 조립하여 어떤 클라이언트(사용자)가 언제 어떠한 데이터에 접근하고 어떤 데이터를 변경하거나 새롭게 추가하였는지 확인하고 로그 머신에 이러한 정보를 기록함으로써 컴퓨터 포렌식 시스템을 구성할 수 있게 된다.

1. 패킷 인터셉터(Packet Interceptor)

본 시스템에서는 패킷을 수집하기 위해 네트워크 탭을 사용하였다[6]. 네트워크 탭은 특별한 장비를 구입하지 않고도 구현 가능하다. UTP 케이블의 RX 선과 TX 선에 오가는 패킷을 데이터 흐름의 중단 없이, 네트워크에 전혀 영향을 주지 않으면서 트

EIA/TIA 568A

핀 번호	케이블의 색	기 능
1	흰색 녹색	Tx+
2	녹색	Tx-
3	흰색 주황	Rx+
4	파랑	사용하지 않음
5	흰색 파랑	사용하지 않음
6	주황	Rx-
7	흰색 갈색	사용하지 않음
8	갈색	사용하지 않음

EIA/TIA 568B

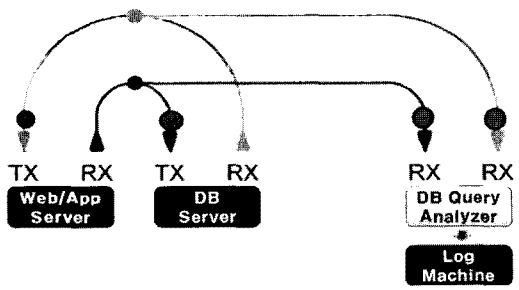
핀 번호	케이블의 색	기 능
1	흰색 주황	Tx+
2	주황	Tx-
3	흰색 녹	Rx+
4	파랑	사용하지 않음
5	흰색 파랑	사용하지 않음
6	녹색	Rx-
7	흰색 갈색	사용하지 않음
8	갈색	사용하지 않음

<그림 4> UTP 케이블
<Fig. 4> UTP cable

래픽을 모니터링 할 수 있는 네트워크 모니터링 방법이며, 한번 설치하면 언제나 네트워크 장애 없이 사용이 가능하다.

UTP 케이블에 대한 표준은 EIA/TIA에서 규정한 것으로서 <그림 4>와 같이 두 가지 표준 방식이 있으며 어느 규격을 사용해도 문제는 없다. 미국의 경우 EIA/TIA 568A를, 국내의 경우 EIA/TIA 568B를 많이 사용한다. 그렇지만 네트워크와 전화를 같이 사용하는 경우는 EIA/TIA 568A를 사용하여야 하는데, 그 이유는 표준 방식과 호환되지 않을 수도 있기 때문이다. UTP 케이블의 끝단에는 RJ-45라는 커넥터가 연결되어 있어 다른 장치와 접속을 용이하게 해준다[7]. 네트워크 탭 장비의 내부 구성도는 <그림 5>와 같다.

쿼리 패킷을 웹 서버와 DB 서버 중간에서 네트워크 탭을 사용하여 가로채고 DB 쿼리 분석기에서 쿼리 패킷을 조립, 분석하여 클라이언트가 어떠한



<그림 5> 네트워크 탭의 구성도
<Fig. 5> Network tap

<표 1> 패킷에 대한 검사 항목
<Table 1> Test items for packet

Header	Field	사용 목적
IP	Protocol	TCP 통신인지의 여부 확인
	Length	패킷의 전체적인 길이 파악
TCP	Push	패킷 조립 작업
	Window	
	Length	TCP 헤더의 끝 또는 DATA 영역의 시작

정보를 조작했는지 확인하고 로그 머신에 기록함으로써 DB 정보 관리 · 감독 시스템을 구성하게 된다. 웹 서버와 DB 서버 사이에서 모니터링을 하기 위해서는 서버 사이의 송 · 수신 데이터를 모니터링해야 하므로 단일포트 랜카드 2장을 설치하거나, 이중포트 랜카드를 설치해야 하며, 반드시 탭의 모니터링 포트에서 나온 케이블이 랜카드의 RX 포트에 연결되어야 한다.

<표 1>은 패킷에 담겨있는 쿼리문을 가져오기 위한 검사 항목으로서 패킷으로부터 추출하여 정리한 것이다. 들어오는 패킷을 검사하여 해당 데이터베이스 고유의 프로토콜이나 형식에 맞는지 검사한다. 정상적인 쿼리문 여부를 판단하고, 패킷을 조립한다. 이러한 작업을 하는 이유는 긴 쿼리문의 경우 패킷의 최대 전송 길이(MTU:Maximum Transper Unit)를 초과하여 단편화가 수행되는 경우에 대응하기 위해서이다. 그렇기 때문에 반드시 잘려온 패킷을 조립하여 하나의 세션으로 만들고 그 안에 존재하는 쿼리를 읽어내야 한다.

패킷 조립은 TCP 헤더 중 Push Flag를 먼저 검사하게 되는데, 이 값이 '0'인 경우 패킷은 조각이기 때문에 조립을 하여야 하며, '1'인 경우에는 마지막 패킷의 조각이거나 단일 패킷인 경우이므로 Push Flag가 '0'인 패킷과 조립하여 하나의 세션으로 만들고 단일 패킷인 경우에는 패킷 자체가 하나의 쿼리 정보를 가진다. 이와 같이 단일 패킷 또는 나누어진 패킷을 조립하여 데이터베이스 쿼리를 추출할 수가 있다.

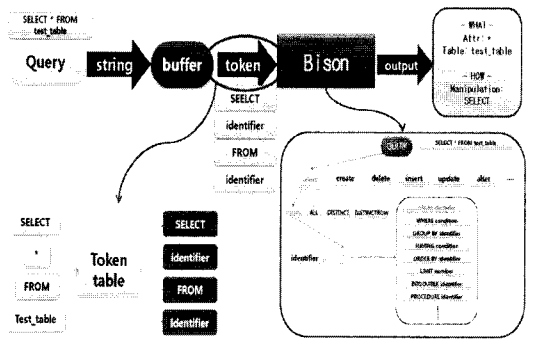
<그림 6> 패킷에서의 쿼리 추출 화면
<Fig. 6> Query extract process for packet

<그림 6>은 패킷을 조립하고 저장한 내용을 편집기로 불러온 화면이다. 쿼리문이 포함되어 있는 것을 확인할 수 있으며, 이를 추출하여 DB 쿼리 분석기를 거쳐 로그머신에 저장하게 된다.

2. DB 쿼리 분석기(DB Query Analyzer)

DB 쿼리 분석기는 Lex & Yacc[8]를 응용하였다. 패킷 인터셉터에서 가로챈 쿼리 패킷을 DB 쿼리 분석기가 분석하여 클라이언트가 접근하려는 데이터에 대한 정보 및 접근 후 데이터의 변경 정보를 추출하기 위해서 구성해야 하는 단계는 <그림 7>과 같다.

<그림 7> DB 쿼리 분석기 구성도
<Fig. 7> Composition of DB Query analyzer

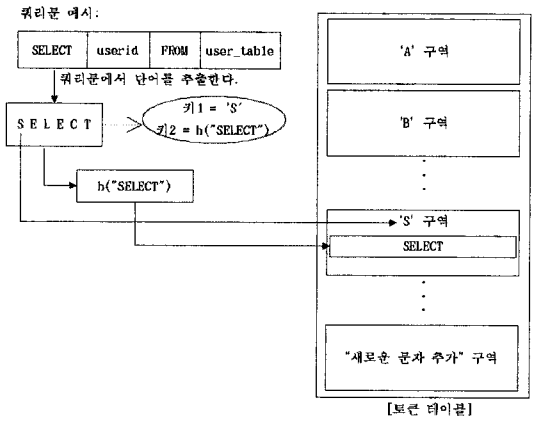


<그림 8> DB 쿼리 분석기 동작원리
<Fig. 8> Operating principles of DB Query analyzer>

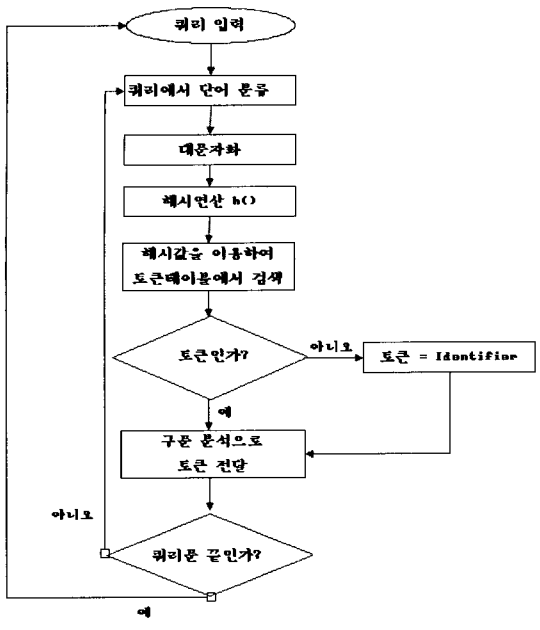
패킷을 조립하여 DB 쿼리가 들어오면 Lexer에서 어휘를 분석하고 Parser(Bison)[9]에서 구문을 분석하여 출력하는 절차이며 <그림 8>과 같이 설계하였다.

<그림 8>과 같이 쿼리 분석기에서는 "무엇을, 어떻게"에 대한 정보를 확인할 수 있다. 위와 같이 "SELECT * FROM test_table"라는 쿼리문이 들어오면 이를 토큰으로 나누고 검색하여 Bison에서 파싱 트리(Pasing Tree)를 따라서 구문을 분석하면 출력과 같이 "Attribute: *, Table: test_table, Manipulation: SELECT"인 것을 확인할 수 있다.

<그림 9>는 토큰 검색을 위한 처리 과정이며, 어휘 분석을 위해서 토큰을 나누고 인덱싱과 구현을 위하여 응용한 해시 알고리즘[10]을 이용하여 검색함으로써 메모리의 공간을 효율적으로 활용하여 속도를 빠르게 설계하였다. 예를 들어, "SELECT"를 찾고자 할 때 2개의 Key 값인 첫 번째 문자 'S'와 해시 함수를 이용한 h("SELECT")를 생성하여 쿼리문에서 추출한 각 단어와 Token Table에서의 해당 항목을 비교하여 토큰 검색을 수행한다. Token Table은 몇 개의 Group으로 나누어 구현한다. 이처럼 설계함으로써 추후에 새롭게 추가되는 예약된 단어(Reserved word)를 쉽게 삽입할 수 있는 장점이 있다. 이렇게 토큰들을 하나씩 찾아나간 후 Bison에서는 파싱 트리를 따라 분석한다.



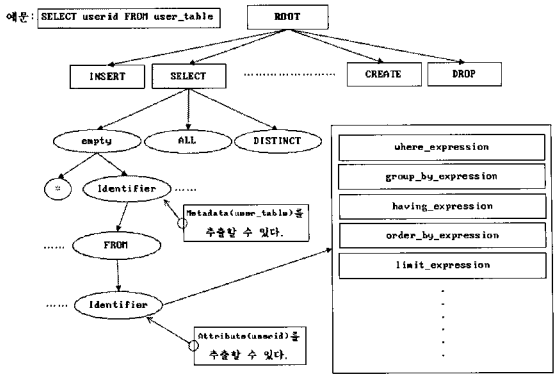
<그림 9> 토큰 검색을 위한 처리 과정
<Fig. 9> Token process



<그림 10> 어휘 검색 순서도
<Fig. 10> Vocabulary analysis flow chart

<그림 10>은 어휘 검색을 위한 순서도를 보여주고 있다. 어휘 분석은 쿼리문 내의 각 단어들이 예약된 단어인지 식별자인지를 확인한 후 토큰으로 나눈다. <그림 10>과 같이 쿼리가 입력되면 단어를 분류한다. 단어 분류는 문자열 내의 빈 공간을 기본으로 수행되며, 각 연산자에 한해서는 따로 플래그를 두어 검사 및 분류를 행한다. 분류된 단어는 해싱 연산을 위해 모두 대문자로 변환하고, 그 다음 해싱을 통해 32비트 정수 값을 생성한다. 본 시스템에서 사용된 해시 함수는 문자열(단어)의 각 바이트 논리합, 이동, 그리고 배타적 논리합 비트 연산을 이용한다. 그리하여, 토큰 테이블 내에 저장되는 각 토큰들의 해시 값은 서로 고유한 값을 가질 수 있게 구현하였다.

하지만, 240여개의 토큰들을 저장하는 토큰 테이블에 사용하기에는 너무 큰 범위를 가지고 있다는 문제가 있다. 이를 해결하기 위하여 인덱싱 기법과 혼용하는 방법으로, 해싱 값을 인덱싱에 오프셋 값으로 사용하였다. 해싱과 인덱싱 기법을 사용하여 토큰 테이블 내의 토큰 주소를 찾는다. 해당 주소의 토큰과 사상이 이루어지면 해당 토큰을 전달하고,



<그림 11> 파싱 트리상에서 정보를 추출하는 과정
<Fig. 11> Information extract process for parsing tree

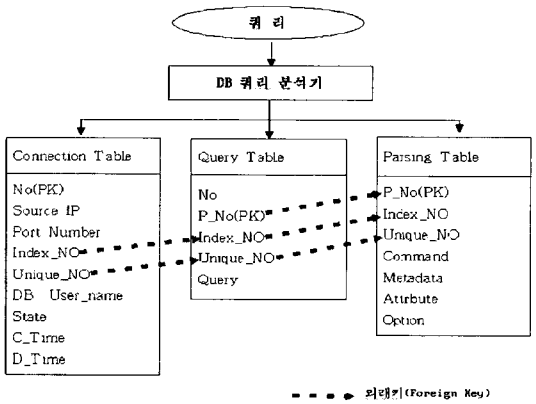
그렇지 않으면 식별자를 구문 분석으로 전달한다. 이와 같은 과정이 끝나면 쿼리의 끝을 확인하고, 새로운 쿼리에 대한 분석을 수행할지 결정을 내린다.

쿼리 분석기에서는 파싱 트리를 구현하기 위해서 Bison을 사용하였다. Bison은 문맥 규칙을 만들기 쉽고 C 언어와 호환되는 장점이 있다. 각 토큰을 파싱 트리를 따라서 분석함으로써 데이터베이스에 대한 어떤 조작 및 테이블에 접근하였는지 등에 관한 정보를 추출 할 수 있다.

<그림 11>과 같이 어휘 분석에서 토큰을 나누어 분류한 후 토큰을 검색한 후, 파싱 트리를 따라 구문을 분석한다. 파싱 트리에서 구현된 규칙에 따라서 토큰들을 순차적으로 사상시킨다. 처음 사상되는 "SELECT" 토큰으로 인하여 이 쿼리가 데이터베이스 내부 정보를 검색하는 쿼리라는 것을 알 수 있다. "SELECT" 이후의 토큰들은 select_manipulation 부트리(Sub-tree)의 규칙을 따라 사상된다. "SELECT" 토큰과 "FROM" 토큰 사이에 사상되는 첫 번째 "Identifier" 토큰은 애트리뷰트(attribute)가 포함된 테이블(user_table)을 가리킴을 알 수 있다. 구문 분석에서 두 번째 "Identifier" 이후에 나오는 "WHERE, GROUP 또는 HAVING" 등과 같은 토큰이 사상되는 경우에는 이 토큰 이후의 문장이 테이블 내의 데이터를 검색하기 위한 조건으로 사용되는 것을 유추할 수 있다. 이러한 과정을 통하여 추출한 정보는 <표 2>와 같다.

<표 2> 쿼리 예문으로부터 추출한 정보
<Table 2> Information extracted for Query example

구 분	내 용
Manipulation	SELECT
Metadata	user_table
Attribute	userid
Options	None



<그림 12> 로그 테이블 구조
<Fig. 12> Logging table structures

파싱이 끝나면 로그 머신에 저장하는데, 관리와 검색 등을 위해서 <그림 12>와 같이 구현하여야 한다.

Connection Table에서는 접속한 클라이언트의 Source IP, Port Number, DB Username 등과 같은 정보와 생성 시간 정보를 담고 있는 Index Number, 그리고 세션마다 고유의 Key Number인 Unique Number를 저장한다. Index Number와 Unique Number는 다른 테이블에 외래키(Foreign Key) 값으로 들어가 상호 연관이 될 수 있게 하는 역할을 한다.

Query Table에서는 쿼리문과 Parsing Table에 저장되는 값과 연동을 위한 P_No가 저장된다. 마지막으로 Parsing Table에서는 쿼리문을 분석한 값과 Command를 구분하여 저장하고 Metadata 부분은 대상이 되는 Table 또는 Database name, Attribute에는 테이블 내의 Attribute name, Option에는 조건 값이 저장된다.

<표 3>, <표 4>는 Connection_Table, Query_Table, Parsing_Table의 구조 및 내용을 정리한 것이다. 생성된 테이블은 시스템의 상황과 목적에 맞게 구조 변경을 할 수 있으며, 기본적으로 저장 프로시져, 시퀀스, 트리거 등을 사용하여 최대한 부하가 적도록 설계하였다. 또한, <표 3, 4>와 같은 테이블 형태로 저장된 로그는 원하는 날짜, 시간 등을 검색하거나 해당 사용자, 세션에서의 작업한 쿼리를 구분하

<표 3> 파싱 테이블 구조 및 내용
<Table 3> Parsing table

구 분	내 용
P_No(PK)	Parsing Number
Index_No(FK)	세션 생성 일시
Unique_No(FK)	CPU real clock time
MANIPULATION	DDL, DML, DCL
Attribute	Table Attribute
Metadata	DB Metadata
Option	Option, Condition

<표 4> Connection Table과 Query Table의 구조 및 내용
<Table 4> Connection table and Query table

구 분	내 용	구 분	내 용
No(PK)	일련번호	No	일련번호
Addr	접속자의 IP 주소	P_No(PK)	Parsing Number
Port	Port Number	Index_No (FK)	세션 생성 일시
Index_No	세션 생성 일시	Unique_No (FK)	CPU real clock time
Unique_No	CPU real clock time	Query	쿼리 전문 (全文)
Username	DB Username		
State	현재의 접속여부		
C_time	접속시간		
D_time	연결해제 시간		

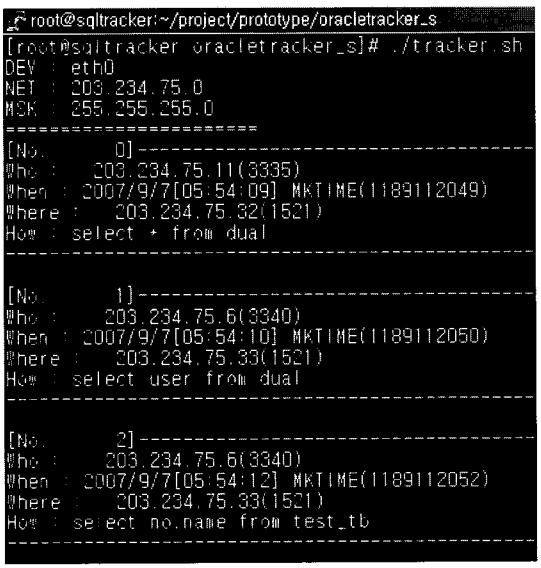
<표 5> "누가, 언제, 어디서, 무엇을, 어떻게" 정보 모니터링
<Table 5> "Who, When, Where, What, How" information for monitoring

구 분	내 용
누가(WHO)	Source IP Address, Source Port, Username
언제(WHEN)	MKTIME(Time stamp)
어디서(WHERE)	Destination IP Address, Destination Port
무엇을(WHAT)	Manipulation, Table, Attribute
어떻게(HOW)	DB Command

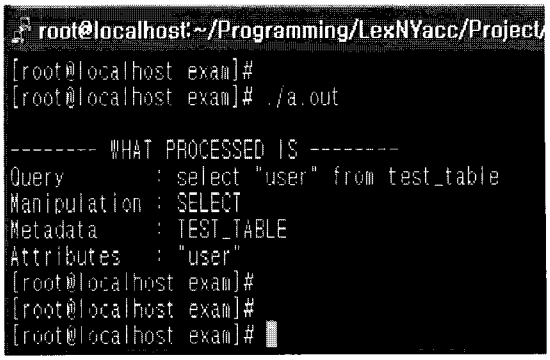
는 것도 가능하며, 파싱 결과에 따라서 더욱 자세한 결과를 얻을 수 있다.

3. 시스템 구현 결과

본 시스템에서는 <표 5>와 같이 "누가, 언제, 어디서, 무엇을, 어떻게" 접근하는지를 모니터링하고, 이를 분석하여 관리 및 감독할 수 있는 충분한 정보를 기록하여야만 한다.



<그림 13> "누가, 언제, 어디서, 어떻게" 정보 출력화면
<Fig. 13> Screen output of "Who, When, Where, How" information

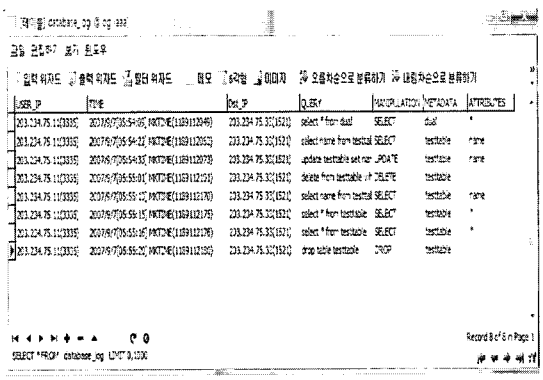


<그림 14> "무엇을" 정보 출력화면
<Fig. 14> Screen output of "What" information

따라서 <그림 13>과 <그림 14>에서는 본 연구에서 설계한 시스템으로 로그에 필요한 정보를 출력한 화면이다.

<그림 13>은 "누가, 언제, 어디서, 어떻게" DB에 접근하는지에 대한 구현 결과이며, <그림 14>는 "무엇에" 해당하는 정보를 추출하여 로그 머신에 기록을 하고, 내부 정보 유출 사건 등이 발생했을 경우 이 로그 기록을 DBMS를 활용하여 분석함으로써 불법 사용자를 식별할 수 있도록 하였다.

또한, <그림 15>는 "누가(USER_IP), 언제(TIME), 어디서(Dst_IP), 무엇을(QUERY), 어떻게(MANIPULATION, METADATA, ATTRIBUTES)"를 간단하게 확인할 수 있도록 로그를 저장하고 있다. 이와 같이



<그림 15> "누가, 언제, 어디서, 무엇을, 어떻게" 로그기록 출력화면
<Fig. 15> Screen output of "Who, When, Where, What, How" log data

로그를 저장함으로써 컴퓨터 포렌식에서 요구하는 디지털 자료 수집을 할 수 있다.

V. 결론 및 향후 연구 방향

본 논문은 국도 ITS 정보의 내부 통제 시스템을 지원하기 위한 시스템에 대해 다루고 있다. 컴퓨터 포렌식 기술을 적용하여, ITS 시스템의 관리 및 감독이 가능하도록 시스템을 구현하였다.

기존의 시스템[11]의 경우 DB 쿼리 수집은 가능하였지만, 분석을 통한 수집이 아니기 때문에 대량의 DB에서 필요한 부분의 쿼리 및 자료에 대하여 검색에 많은 시간을 소비하거나 수집된 쿼리가 "어떻게" 사용되었는지 확인이 어려웠다.

이 시스템에서는 웹 서버와 데이터베이스 서버 간의 오고가는 네트워크 패킷을 가로채어 데이터베이스 쿼리 분석기에서 패킷을 조립한 후에 쿼리를 추출하고, 그 쿼리를 분석함으로써 내부나 외부로부터 "누가, 언제, 어디서, 무엇을, 어떻게" 접근하는 가에 대한 정보를 수집할 수 있다. 또한, 수집된 정보를 로그 머신에 저장하여 관리 및 감독을 위한 자료로 사용하는 것이 본 시스템의 목적이다.

이는 데이터베이스 내의 중요한 정보 유출 사건이 발생했을 경우나 조작 사건이 발생했을 경우에 이를 추적할 수 있는 기반 지식 정보를 의미한다. 또한, 네트워크 탭을 이용하여 패킷을 가져오기 때문에 외부에서 본 시스템을 접근 및 조작할 수 없으며, 전송 속도에도 지장을 주지 않고 웹 서버와 데이터베이스 서버 간에 네트워크 간섭이 없어 기존 시스템에 영향을 주지 않으며 바로 연결하여 이용할 수 있다는 장점이 있다.

그러나, 현재 구현된 시스템에서는 UTP 케이블의 RX 선을 이용하여 수신만 가능하기 때문에 서버를 컨트롤 하거나 직접 방어 기능은 없다.

또한, 효율적인 로그 기록의 검색 및 관리 기능이 아직까지 부족하다. 따라서 상세한 많은 로그 정보들을 축소하여 상호 연관성이 있도록 저장하는 방법의 시스템 개발이 필요하며, 필요한 정보를 빠르게 검색하고, 연관 탐색이 가능한 데이터마이닝(Data-Mining) 등의 기법 도입이 추가적으로 필요할 것이다.

참고문헌

[1] W. G. Kruse II and J. G. Heiser, *Computer Forensics : Incident Response Essentials*, Addison-Wesley, 2001.

[2] 박태규, 임연호, "커널 기반의 보안 리눅스 운영체제 구현," *한국정보보호학회논문지*, 제11권, 제4호, pp. 33-43, 2001. 08.

[3] 이형우, 이상진, 임종인, "컴퓨터 포렌식스 기술," *한국정보보호학회지*, 제12권, 제5호, pp. 8-16, 2002. 10.

[4] 한국정보보호진흥원, *데이터베이스 보안 위협 및 국내외 제품 동향*, 연구보고서, 2007. 05.

[5] 황철, 황대준, "디지털 컨텐츠 보호를 위한 에이전트기반 포렌식 컴퓨팅 관리," *한국정보과학회 봄 학술발표논문집*, 제28권, 제1호(A), pp. 856-858, 2001. 04.

[6] 비오테크더블테크놀러지(주), *네트워크 패킷 모니터링의 필수품 NetworkTap*, http://www.bow.co.kr/, 2006.

[7] 원정석, *시스코 네트워킹 튼튼하게 배우기 (IP 네트워크 실무자를 위한)*, 성안당, 2003. 09.

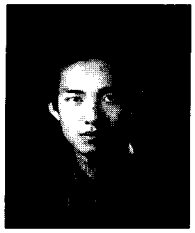
[8] J. R. Levine, T. Mason, and D. Brown, *Lex & Yacc*, O'REILLY, 1992. 10.

[9] 윤상배, "*Flex and Bison Howto*," http://www.joinc.co.kr/modules/moniwiki/wiki.php/article/lex_yacc_howto, 2004.

[10] B. Jenkins, "*Bob Jenkins' Web Site*," http://burtle burtle.net/bob/index.html

[11] Guardium, *DB보안 및 Data거버넌스 솔루션 Guardium version 6.0 제품*, 2007. 05.

저자소개



임 재 덕 (Lim, Jae-Deok)

2008년 현재 : 한국건설기술연구원 첨단도로교통연구실 연구원
1999년 3월 ~ 2005년 2월 : 한서대학교 컴퓨터정보학과 이학사
2006년 3월 ~ 2008년 2월 : 한서대학교 컴퓨터정보학과 이학석사

임 성 한 (Lim, Sung-Han)

2008년 현재 : 한국건설기술연구원 첨단도로교통연구실 연구원

백 남 철 (Baek, Nam-chul)

2008년 현재 : 한국건설기술연구원 첨단도로교통연구실 선임연구원