데이터베이스 시스템에서 연관 규칙 탐사 기법을 이용한 비정상 행위 탐지

박 정 호† · 오 상 현† · 이 원 석††

요 약

컴퓨터와 통신 기술의 발달로 사용자에게 많은 정보가 편리하게 제공되는 반면, 컴퓨터 침입 및 범죄로 인한 피해가 증가하고 있다. 특히, 고객 개인 정보, 기업 기밀과 같은 주요 정보가 저장되어 있는 데이터베이스의 보안을 위해서 데이터베이스 관리 시스템의 기본적인 보안 기능 및 기존의 오용 탐지 모델이 사용되고 있다. 하지만, 다양한 시스템 침입 유형에 대한 분석 결과에 따르면 외부 침입자에 의한 시스템 파괴보다는 내부 사용자에 의한 기밀 정보 유출과 같은 권한 오용 행위에 의한 손실이 더 큰 문제가 되고 있다. 따라서, 효과적으로 데이터베이스 보안을 유지하기 위해서 사용자의 비정상 행위 판정 기술에 대한 연구가 필요하다. 본 논문에서는, 연관 규칙 마이닝 방법을 이용하여 데이터베이스 로그로부터 사용자 정상 행위 프로파일을 생성하는 방법을 제안한다. 이를 위해서 데이터베이스 로그를 의미적인 패턴 트리로 구조화하여 생성된 정상 행위 프로파일을 온라인에서 발생된 해당 사용자의 트랜잭션과 비교하여 온라인 데이터베이스 작업에 대한 비정상 행위 여부를 탐지할 수 있다. 다양한 실험을 통해 제시된 알고리즘의 효용성을 분석하고 결과를 제시하였다.

Anomaly Intrusion Detection based on Association Rule Mining in a Database System

Jeong Ho Park† · Sang Hyun Oh† · Won Suk Lee††

ABSTRACT

Due to the advance of computer and communication technology, intrusions or crimes using a computer have been increased rapidly while tremendous information has been provided to users conveniently. Specially, for the security of a database which stores important information such as the private information of a customer or the secret information of a company, several basic security methods of a database management system itself or conventional misuse detection methods have been used. However, a problem caused by abusing the authority of an internal user such as the drain of secret information is more serious than the breakdown of a system by an external intruder. Therefore, in order to maintain the security of a database effectively, an anomaly detection technique is necessary. This paper proposes a method that generates the normal behavior profile of a user from the database log of the user based on an association mining method. For this purpose, the information of a database log is structured by a semantically organized pattern tree. Consequently, an online transaction of a user is compared with the profile of the user, so that any anomaly can be effectively detected.

키워드 : 비정상 행위 판정(Anomaly Detection), 침입 탐지(Intrusion Detection), 데이터마이닝(Data Mining), 연관 규칙 마이닝(Association Rule Mining)

1. 서 론

컴퓨터와 통신 기술의 발달로 사용자에게 다양한 정보와 편리성이 제공된 반면, 컴퓨터 침입 및 범죄로 인한 피해가 증가하고 있다. 침입이란 권한이 없는 사용자가 발생시키는 문제 또는 합법적인 사용자가 권한을 남용하는 행위로 정의된다[1]. 이와 더불어 자원의 유용성, 기밀성, 및 무결성 등에 저해되는 행동 집합을 침입이라 정의하기도 한다[2]. 이러한 침입을 탐지하기 위한 방식은 크게 오용 탐지 모델[3, 4]과 비정상 행위의 탐지 모델[5, 6]로 분류된다. 오용 탐지 모델은 사전에 침입자의 공격 패턴을 모델링하여 이와 일치하는 침입자의 패턴을 탐지한다. 따라서 오용 탐지 모델은 기존에 알려진 공격 패턴만을 탐지할 수 있다는 단점을 갖는다. 하지만, 분석 결과에 따르면 외부 침입자에 의한 시스템 파괴보다는 기밀 정보 유출과 같은 내부 사용자의 권한 남용에 의한 손실이 더 큰 문제가 되고 있다[2]. 이러한 문제를 해결하기

† 준 회 원 : 연세대학교 대학원 컴퓨터과학과
†† 종신회원 : 연세대학교 컴퓨터과학과 부교수
논문접수 : 2002년 8월 12일, 심사완료 : 2002년 10월 22일

위해서, 사용자의 비정상 행위 탐지 모델이 연구되고 있다. 비정상 행위 탐지 모델은 사용자의 정상적인 행위를 모델링 하며 정상 행위 패턴에서 벗어나는 작업을 비정상 행위로 탐지하는 방식이다.

침입자들의 시스템 공격은 초기에는 침입 기법이 단순하였지만 정보 통신의 발전과 더불어 시스템 침입 기법도 고도화되고 전문적으로 변화해가고 있다. 따라서 이에 대응하는 침입 탐지 기법들도 그 복잡성을 더해가고 있으므로 과거와 같이 각 침입 방식에 대한 개별적인 대처 방안으로는 충분한 보안 유지를 기대할 수 없다. 이러한 문제를 해결하기 위해서 자동화된 판정 시스템 개발이 필요하게 되었고 방대한 양의 감사 자료를 필터링 등의 방법으로 자료의 저장 및 분석에 따른 오버헤드를 최소화시킬 수 있는 기술이 필요하게 되었다. 특히 비정상 행위 탐지 모델의 핵심이라 할 수 있는 비정상 행위 판정 기술과 관련하여 보안 관련 감사 자료의 수집, 저장, 분석 및 해석 기술에 대한 연구가 활발히 추진 중이다[6-10]. 최근에는 방대한 데이터 분석을 지능적이고 자동적으로 수행하기 위해서 데이터마이닝 기법을 이용하여 사용자의 정상 행위를 모델링하고 있다[5-7].

한편, 고객 정보, 기업 비밀과 같은 주요 정보가 저장되어 있는 데이터베이스 보안을 위해서 데이터베이스의 기본적인 보안 기능[7] 및 오용 탐지 모델[3, 4]이 사용되어왔다. 하지만 데이터베이스 시스템의 경우 알려진 공격 패턴이 많지 않고 운영 시스템의 권한을 획득한 경우 저장된 데이터베이스의 정보가 무방비 상태로 되며 사용자 암호 유출인 경우 데이터베이스에 대한 접근 권한의 정의 단위가 크며 세밀한 권한 검사가 어려워 외부 침입자 뿐만 아니라 사용자의 권한 오용으로 인한 정보 유출도 심각한 문제로 대두되므로 데이터베이스 보안을 위해서는 오용 탐지 모델보다 비정상 행위 탐지 모델의 적용이 적합하다. 본 논문에서는 사용자가 수행한 데이터베이스 명령어들에서 연관 규칙을 탐사하여 사용자 정상 행위 프로파일 생성하는 방법을 제안한다. 이를 위해서, 데이터베이스 로그를 의미적 계층 구조를 가지는 패턴 트리로 구조화하고 패턴 트리로부터 생성된 연관 규칙들로 구성된 정상 행위 프로파일을 온라인에서 발생된 사용자 트랜잭션의 명령어와 비교함으로써 효과적으로 비정상 행위를 탐지할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 관련 연구를 소개하며, 3장에서는 데이터베이스의 SQL 로그 파일을 분석하여 연관 규칙을 탐사하여 정상 행위 프로파일 추출 방법을 제시한다. 4장에서는 온라인상의 SQL 문장과 프로파일의 연관 규칙과의 유사도를 측정하는 방법을 제시하고 이를 이용한 비정상 행위 판정 기법을 설명한다. 5장에서는 제시한 알고리즘을 활용한 실험 환경 및 분석 결과를 보여주며 6장에서 결론을 맺는다.

2. 관련 연구

데이터베이스 보안은 크게 접근 제어기법과 추론 제어기법으로 구분할 수 있다. 접근 제어기법은 세부적으로 임의 접근 제어기법(Discretionary Access Control)과 강제 접근 제어기법(Mandatory Access Control)으로 나누어진다. 임의 접근 제어기법은 특정 데이터에 접근할 수 있는 권한을 사용자에게 부여함으로써 보안을 유지하는 방법이고[8] 강제 접근 제어기법[9, 10]은 데이터와 사용자들을 다양한 보안 등급으로 분류함으로써 다단계 보안이 필요한 곳에서 사용된다. 추론 제어(Inference Control)기법[11]은 통계 데이터베이스 보안을 위해 사용된다. 통계 데이터베이스는 다양한 분야에서 통계를 산출하기 위해 사용되지만 통계를 위한 목적이 아닌 비밀정보를 얻기위해 접근하는 사용자로부터 통계의 추론을 통한 상세정보의 유출을 통제하여야 한다. 추론 제어기법에서는 개별적인 속성값 조건들에 기반한 질의를 금지해야 하고 통계적인 함수만을 포함한 질의만이 수행되도록 한다.

데이터베이스 관리 시스템은 기본적인 보안 기능을 위해 자체적으로 다양한 기능을 보유하고 있다. 그러나 이러한 기능들은 시스템 침입자가 운영 시스템의 권한을 획득하였거나 다른 사용자의 계정 암호(password)를 요용하여 침입한 경우에는 저장된 데이터베이스의 정보를 보호할 수 없다. 따라서 불법적인 데이터의 조작에 대해 의심되는 사용자를 감시하기 위해서는 상당한 양의 로그 데이터를 분석해야 하는 부담을 가지게 된다. 따라서 자동화된 로그 데이터의 분류 및 분석을 통한 보안 시스템의 필요성이 대두되고 있다. 최근에 침입 탐지를 위해서 로그 데이터를 지능적이고 자동적으로 분석하기 위해서 데이터마이닝 기법이 이용되고 있다. 데이터마이닝 기법 중에 연관 규칙 마이닝 방법[5-7]은 데이터 내에 항목들간의 연관성을 탐사한다. 연관 규칙 마이닝 방법 중에서 Apriori[5]는 사용자가 정의한 최소 지지도를 이용하여 동시에 자주 나타나는 항목(빈발 항목 집합)들을 정제하고 빈발 항목 집합(frequent itemset)에서 생성된 규칙들은 신뢰도를 이용하여 정제하는 방식이다. Apriori는 후보 항목 집합에 대한 지지도를 계산하여 사용자가 정의한 최소 지지도 보다 크면 빈발 항목 집합으로 생성한다. Partition[6] 알고리즘은 데이터 집합을 최대 두 번 검색함으로써 데이터 집합에 대한 탐색횟수를 감소시킨다. 이 알고리즘의 기본적인 접근 방법은 분석 대상 데이터 집합을 가용 메모리 공간에 적합한 크기의 블록으로 분할한다. 빈발 항목 집합을 구하기 위한 기본적인 알고리즘은 Apriori와 동일하다. DIC[7] 알고리즘은 서로 다른 길이를 갖는 항목 집합들의 출현 빈도수를 동시에 분석하며 잠재적으로 빈발인 항목들의 출현 빈도수만 래티스(Lattice)에서 관리된다. 데이터 집합은 몇 개의 블록으로 분할되고 전체 데이터 집합은 최대 두 번 탐색 된

다. 연관 규칙 마이닝을 이용한 대표적인 침입 탐지 시스템은 JAM[3-5]이다. JAM에서는 연관 규칙 마이닝과 더불어 frequent episode[20]를 이용하여 정상행위 패턴을 생성한다. JAM은 데이타마이닝 응용 프로그램을 평가하는 데에 있어 일반적 접근 방법인 메타 학습(Meta-Learning)을 채용하고 있으며 분산환경에서의 이식성과 확장성을 제공하는 에이전트 기반 데이타마이닝 시스템이다. 즉, 분산된 여러 사이트에서 데이타마이닝 결과를 상위 사이트에서 조합(메타 학습)하여 사용자의 부정행위를 탐지한다.

DEMIDS 시스템[12]은 데이터베이스 사용자의 로그 데이터에 대한 빈발 항목 집합을 정상 행위 패턴으로 추출하여 데이터베이스 보안에 적용하였다. 정상 행위 패턴을 추출하기 위한 DEMIDS 시스템의 접근 방식은 스키마 거리(Schema Distance)와 접근 연관성(Access Affinity)개념을 사용한다. 스키마 거리는 기본키(Primary Key)과 외부키(Foreign Key)로 연결된 데이터베이스 스키마를 사용하여 객체간의 거리를 표현한다. 접근 연관성이란 의미적 접근에 근거하여 속성들간의 연관성을 표현하는 개념을 말한다. 즉, 둘 이상의 속성을 동시에 접근하는 횟수가 많을 경우 그 속성들 간의 연관성을 수치로 표현함으로써 의미적인 연관성을 나타낸다. 하지만 DEMIDS에서는 SQL 문 단위로 빈발 항목 집합을 생성하기 때문에 의미적인 사용자의 행위 단위인 세션 단위의 모델링이 불가능하다. 따라서 내부 권한 오용자에 대한 효과적인 탐지가 어렵다. 본 논문에서는 DEMIS에서와 달리 사용자 행위의 단위인 세션을 트랜잭션으로 정의하고 트랜잭션에 기반한 연관 규칙 탐사 알고리즘을 제안한다. 이를 통해서 규칙적이고 반복적으로 나타나는 데이터베이스 명령어군을 정상 행위로 간주한다.

3. 정상 행위 프로파일의 추출

사용자의 정상 행위 프로파일을 추출하기 위해서는 해당 사용자의 로그 데이터를 수집하고 분석하는 과정이 필요하다. 이를 위해서 데이터베이스 로그로부터 각 사용자의 SQL 문을 추출하고 추출된 SQL 문에 대한 파싱 및 코드 변환을 수행하여 패턴 트리를 생성한다. DEMIS에서는 SQL 문 단위의 빈발 항목 집합을 생성하였다. 하지만 본 논문에서는 사용자 트랜잭션 단위의 연관 규칙 마이닝 방법을 적용하여 사용자의 정상 행위 프로파일을 생성한다. 데이터베이스 로그에는 사용자에 따라 다양한 형태의 SQL 문이 기록된다. 따라서 사용자의 SQL문의 구성 단위에 대해서 각각 식별자를 부여하여 서로 다른 SQL 문을 구별하고 동시에 공통된 부분을 나타낼 수 있다.

SQL 문의 구성 단위를 단순히 문자열을 비교하여 별도의 식별자로 매핑할 경우 비 표준 SQL 문이나 SQL 문에서 부분적으로 생략된 내용의 차이로 인해 동일한 의미의 SQL 문이 수행되었다 하더라도 다른 식별자가 부여될 수 있다. 예를 들어, <표 1>에서 첫 번째 트랜잭션에서 수행된 SQL 문들의 구성 단위들은 모두 같지만 SQL 문의 해당 문자열이 다르기 때문에 다른 식별자가 부여된다. 또한, SQL 문자열을 그대로 식별자로 매핑하게 되면 부속절의 처리에 한계를 갖는다. 이와 같은 문제들을 해결하고 효율적인 정상 행위 프로파일을 추출하기 위해서 데이터베이스 로그에 포함된 SQL 문 집합에 대한 의미적 패턴 트리를 생성한다. 이를 위해서 각 SQL 문에서 생략된 문장의 보충 및 비 표준 SQL 문의 표준화 작업을 수행한다. 예를 들어, 테이블 Table1과 테이블 Table2의 소유자가 Owner라 했을 때 'INSERT INTO Table1 SELECT×FROM Table2'와 같은 SQL 문은 파싱을 통해서 Owner.Table1, Owner.Table2와 같이 생략된 문장을 보충한다. 또한 SQL 문 내부에 존재하는 부속절에 대한 분리 작업을 수행해야 하며 부속절을 포함하는 SQL 문은 해당 부속절의 포인터를 가짐으로써 데이터의 손실을 막을 수 있다. 이때, 부속절을 식별할 수 있는 플래그(Flag)를 이용하여 해당 SQL 문이 부속절로 사용되었는지 독립적으로 사용되었는지의 여부를 표시하게 된다. 따라서, 부속절 플래그와 포인터를 확장하면 부속절을 포함하는 전체 SQL 문을 표현할 수 있다. 예를 들어 어떤 부속절이 SELECT 문 내에만 사용되었다면 그 문장은 해당 SELECT 문장 이외의 부속절이나 독립절로 사용될 경우 비정상적인 행위로 간주될 수 있다. 부속절 플래그는 여러 단계로 확장할 수 있으므로 관리자의 설정에 따라 비정상 행위 판정을 보다 세밀하게 제어할 수 있는 장점을 갖는다.

〈표 1〉 사용자 로그 데이터

트랜잭션 ID	SQL 문
1	SELECT Column1 FROM Table1 SELECT Column1 FROM Owner.Table1 SELECT Table1.Column1 FROM Table1 SELECT Table1.Column1 FROM Owner.Table1
2	SELECT Column2 FROM Table1 SELECT Column3 FROM Table1
3	SELECT Column2 FROM Table1 SELECT Column3 FROM Table1

데이터베이스 로그에 포함되어 있는 SQL 문의 파싱이 완료된 후 SQL 문의 각 구성 요소를 고정된 길이의 문자열로 변환하는 코드 변환 작업을 수행한다. 코드 변환 과정을 수행함으로써 SQL 문중 필요한 정보만을 용이하게 추출할 수 있다. 즉, SQL 문에서 고정 길이의 코드 변환이 수행되면 SQL 문간의 비교 연산이 효율적으로 수행될 수 있다. 코드 변환을 위해서 SQL 문을 구성하는 의미적 구문을 <표 2>와

같이 6개의 의미 단위로 구분하여 각각의 구문에 대응되는 코드 정보 데이터 사전(Data Dictionary)을 사용한다. 데이터 사전에는 <표 2>와 같이 SQL 구문을 위한 정보와 사용자에 의해 정의되는 상수 및 객체 정보 등을 저장하게 된다. 데이터베이스 시스템에서 제공되는 함수, 연산자 등은 프로파일 생성 이전 단계에서 데이터 사전에 저장되므로 서로 다른 데이터베이스 시스템에서 정상 행위 프로파일을 추출할 경우 데이터 사전만을 변경함으로써 일관성을 유지할 수 있다. <표 2>를 이용하여 <표 1>의 사용자 로그에 대해서 코드 변환을 수행하면 <표 3>과 같다.

〈표 2〉 데이터 사전

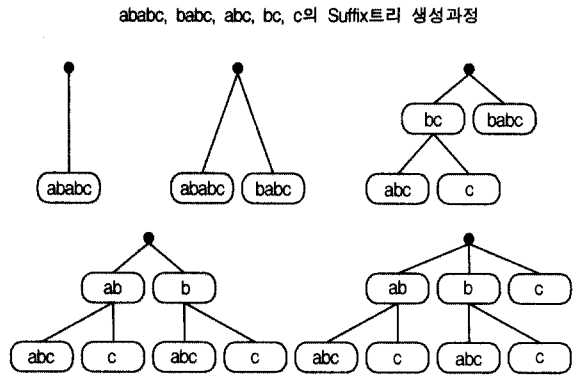
구 문	설 명	코 드	
Action	DDL 또는 DML 명령	SELECT INSERT	'A 0001' 'A 0002'
Function	데이터베이스에서 제공하는 함수	ABS SQRT	'F 0001' 'F 0002'
Operator	데이터베이스에서 제공하는 연산자	+ -	'P 0001' 'P 0002'
Object	사용자에 의해 정의되는 테이블 또는 뷰	Owner.Table 1	'J 0001'
Column	사용자에 의해 정의되는 컬럼	Owner.Table 1.Column 1 Owner.Table 1.Column 2 Owner.Table 1.Column 3	'L 0001' 'L 0002' 'L 0003'
Constant	사용자에 의해 정의되는 상수	'SEOUL'	'T 0001'

〈표 3〉 사용자 로그 데이터

트랜잭션 ID	변 환 된 코 드
1	'A 0001 # J 0001 # L 0001' 'A 0001 # J 0001 # L 0001' 'A 0001 # J 0001 # L 0001' 'A 0001 # J 0001 # L 0001'
2	'A 0001 # J 0001 # L 0002' 'A 0001 # J 0001 # L 0003'
3	'A 0001 # J 0001 # L 0002' 'A 0001 # J 0001 # L 0003'

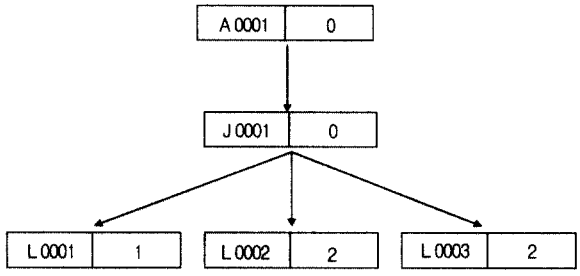
본 논문에서는 SQL 문을 효과적으로 구조화하기 위해서 Suffix 트리[13, 14]를 적용한 패턴 트리를 제안한다. Suffix 트리는 문자열 비교를 빠르게 수행할 수 있는 구조로 일반적인 텍스트 비교 알고리즘에서 가장 널리 사용되고 있다. Suffix 트리의 기본적인 생성 및 확장 과정은 (그림 2)와 같다.

(그림 2)에서 ababc와 babc를 Suffix 트리에 삽입할 경우 두 문자열은 일치되는 접두 부분이 없기 때문에 각각 하나의 노드를 생성한다. 여기에 abc가 입력될 경우 abc는 ababc와 ab접두 부분이 일치하기 때문에 ababc는 abc와 c로 분할되고 각각은 ab의 하위 노드로 생성된다. 이런 과정을 반복함으로써 입력된 문자열에 대해서 Suffix 트리를 확장해 나가



(그림 2) Suffix 트리의 생성 과정

게 된다. 패턴 트리는 SQL 문의 각 구문마다 Suffix 트리를 적용함으로써 빠른 문자열 비교 작업을 수행할 수 있고 원하는 구문을 쉽게 추출할 수 있도록 지원한다. <표 3>을 이용하여 Suffix 트리를 적용한 패턴 트리 생성 과정은 (그림 3)과 같이 이루어진다.



(그림 3) 패턴 트리

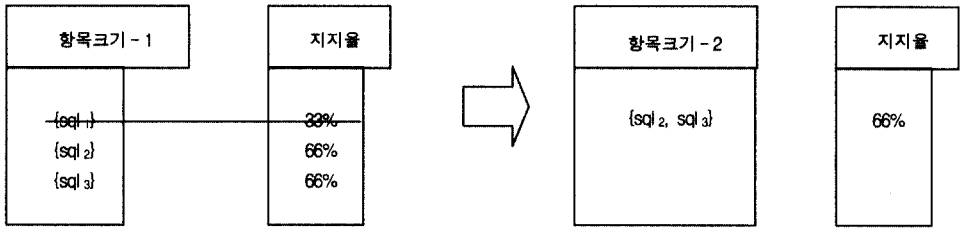
(그림 3)에서 패턴 트리의 각 노드는 SQL 문장의 각 구성 단위에 대해 변환된 코드를 갖고 이와 더불어 해당 노드까지의 경로에 존재하는 노드들의 구성 단위들로 표현된 SQL 문이 사용자 로그에서 발생된 트랜잭션의 횟수 정보를 갖는다. 이러한 패턴 트리 구조는 사용자 SQL 문들에 대한 정상 행위 모델링의 상세 범위를 필요에 따라 유연하게 적용할 수 있다. 예를 들어, 사용자 SQL 문중 객체까지만을 고려한 정상 행위 프로파일을 생성하고자 할 경우 패턴 트리의 컬럼 구문 이하의 노드들을 분석 대상에서 제외시킬 수 있다. SQL 문의 식별자는 SQL 문이 발생된 트랜잭션의 수가 0이 아닌 노드까지의 경로에 대해서 생성된다. 따라서, <표 3>의 각 SQL 문에 식별자를 부여하면 <표 4>와 같다.

〈표 4〉 사용자 로그 데이터

변 환 된 코 드	식 별 자
'A 0001 # J 0001 # L0001'	sql_1
'A 0001 # J 0001 # L0002'	sql_2
'A 0001 # J 0001 # L0003'	sql_3

{querytype = select, Table 1.Column 1 = 1, Table 1.Column 2 = 0, Table 1.Column 3 = 0} [50%]
~~{querytype = select, Table 1.Column 1 = 0, Table 1.Column 2 = 1, Table 1.Column 3 = 0} [25%]~~
~~{querytype = select, Table 1.Column 1 = 0, Table 1.Column 2 = 0, Table 1.Column 3 = 1} [25%]~~

(a) DEMIDS 시스템의 빈발 항목 집합



(b) 연관 규칙을 적용한 본 시스템의 정상 행위 프로파일

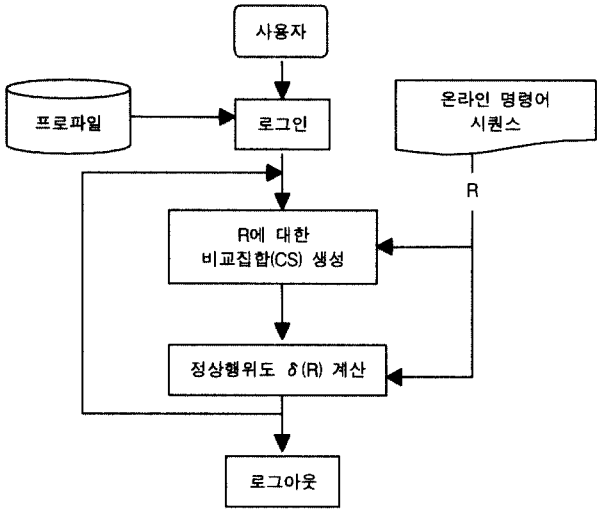
(그림 4) DEMIDS 시스템과 본 시스템의 정상 행위 프로파일(최소 지지도 = 50%)

사용자의 로그 데이터가 <표 1>과 같을 경우 DEMIDS 시스템의 정상 행위 프로파일과 본 논문의 프로파일의 추출 결과는 (그림 4)와 같다. 테이블 Table 1의 Column 1, Column 2 및 Column 3이 포함된 select문은 DEMIDS에서는 {querytype = 'select', Table 1.Column 1, Table 1.Column 2, Table 1.Column 3}로 표현된다. 여기에서 테이블 Table 1의 Column 1을 접근할 경우 Table 1.Column 1은 1로 설정되며 접근하지 않을 경우 0으로 설정된다.

SELECT Column 1 FROM Table 1문장은 전체 사용자 로그 데이터 중 4번 수행되었기 때문에 빈발 항목을 기준으로 정상 행위를 모델링하는 DEMIDS 시스템에서는 {querytype = 'select', Table 1.Column 1 = 1, Table 1.Column 2 = 0, Table 1. Column 3 = 0}에 해당되는 항목 집합의 지지도는$(4/8)\times100 = 50(\%)$이 된다. 결과적으로 DEMIDS 시스템에서는 SELECT Column 1 FROM Table 1 문만이 프로파일로 생성된다. 반면 (그림 4)(b)에서와 같이 본 논문에서 제시한 방법에서는 연관 규칙 마이닝 적용시 항목을 SQL에 해당하는 식별자를 이용한다. 따라서 항목 집합 $\{sql_2\}$, $\{sql_3\}$ 및 $\{sql_2, sql_3\}$이 빈발 항목 집합으로 구할 수 있다. 또한 최소 신뢰도가 90% 일 때 항목 집합 $\{sql_2, sql_3\}$에 대한 연관 규칙이 각각 $\{sql_2\} \rightarrow \{sql_3\}$, $\{sql_3\} \rightarrow \{sql_2\}$와 같고 신뢰도가 각각 100%이므로 두 개의 연관 규칙이 생성될 수 있고 해당하는 SQL 문들의 규칙이 정상 행위 프로파일로 생성된다. (그림 4)의 결과에서 DEMIS의 경우 사용자 트랜잭션의 개념을 적용하지 않기 때문에 사용자의 행위를 정상적인 작업 단위로 모델링할 수 없으며 소수의 특정 트랜잭션에 많은 작업이 집중될 경우 이들 작업이 정상 행위로 파악될 수 있다. 반면, 본 논문에서 제시하는 방법은 사용자 SQL 문 단위의 연관 규칙 마이닝을 수행하므로 트랜잭션 빈도수를 고려할 수 있기 때문에 사용자의 비정상 행위를 보다 정확히 탐지할 수 있다.

4. 비정상 행위 판정 시스템

비정상행위 판정은 사용자의 데이터베이스 시스템에 로그인을 수행하면 그 사용자의 프로파일을 미리 가져와서 메모리에 상주시킨다. 이후에 사용자가 SQL 문들을 수행할 때 프로파일과 비교하여 정상행위도가 계산된다. (그림 5)는 비정상행위 판정의 개요를 나타낸다.



(그림 5) 비정상행위 판정 시스템의 개요

예를 들어, 사용자가 온라인 상에서 수행 SQL 문의 식별자 집합이 $\{sql_1, sql_2\}$일 때 sql_1에 해당하는 SQL 문이 수행된 후 sql_2가 수행되었다고 가정하자. sql_1이 수행되었을 경우에는 오로지 하나의 SQL문이 수행되었기 때문에 sql_1의 지지도만을 고려함으로써 비정상 행위를 판정할 수 있으나 sql_2가 수행된 시점에서는 Supp($\{sql_2\}$)와 Supp({sql1, sql2}) 가 모두 고려된다. 또한 sql_1이 수행된 후 sql_2가 수행되었으므로 이에 대한 신뢰도 Conf($\{sql_1\}\{sql_2\}$)도 함께 고려된다. 이때, 프로파일과 온라인에서 수행된 SQL 문들과의 비교를 위해서 [정의 1]에 의해서 비교 집합 CS를 구할 수 있다.

[정의 1] 온라인에서 현재 수행되는 SQL 문을 R이라 하고 R이전에 수행된 SQL 문의 집합 SS = $\{sql_1, sql_2, \cdots, sql_n\}$이라 하자. 이때 R에 대한 정상 행위도

를 계산하기 위해 프로파일과 비교되는 집합 CS는 SS에 대한 멱 집합(power set)의 모든 원소(element)에 R을 추가한 집합이다. 즉, SS와 R에 대해서 비교 집합 CS = {{sql_1, R}, {sql_2, R}, ⋯, {sql_1, sql_2, ⋯, sql_n, R}}으로 정의한다. □

예를 들어 온라인 SQL 문이 sql_1, sql_2, sql_3의 순서대로 수행되었다면 sql_3에 대한 SQL 문이 수행될 때 비교 집합 CS = {{sql_1, sql_3}, {sql_2, sql_3}, {sql_1, sql_2, sql_3}}와 같다. 따라서, 사용자의 온라인 SQL 문장 R에 대한 정상 행위도 $\delta(R)$은 다음과 같이 계산된다. 여기에서 Diff(R)은 온라인 SQL문장 R에 대해서 과거 트랜잭션내에서의 평균 발생 빈도수 P(R)과 온라인 트랜잭션에서의 발생 빈도수 O(R)과의 차이를 나타내고 r은 사용자 정의 명령어 차이 반영 비율을 나타낸다. 명령어 차이 반영 비율은 특정 SQL 문의 과거와 현재의 빈도수 차를 정상행위도 계산에 어느 정도 적용시킬지를 설정하는 변수이다. 따라서 명령어 차이 반영 비율이 커지면 특정 SQL 문의 빈도수 차가 정상행위도에 많은 영향을 미치게 된다.

$$Diff(R) = 1 - |P(R) - O(R)| \times r$$

$$\delta(R) = \frac{Supp(R) \times Diff_R + \sum_{x \in CS} Supp(x) \times Conf((x - R) \rightarrow R}{1 + \sum_{x \in CS} Supp(x)}$$

예를 들어, <표 1>의 사용자 로그에 대한 프로파일은 <표 5>와 같다.

〈표 5〉 사용자 정상 행위 프로파일

P(R)	지 지 율(%)	신 뢰 도(%)
$P(sql_2)$ = 1 $P(sql_3)$ = 1	Supp({sql_2}) = 80 Supp({sql_3}) = 90 Supp({sql_2, sql_3}) = 70	Conf({sql_2} {sql_3}) = 100 Conf({sql_3} {sql_2}) = 100

온라인에서 sql_3과 sql_2순으로 SQL 문이 수행되었을 때 sql_3과 sql_2에 대한 정상 행위도는 다음과 같이 계산된다. 이때 명령어 차이 비율은 r = 0.05와 같이 설정한다. sql_3이 수행되었을 경우 $P(sql_3)$ = 1이고 $O(sql_3)$ = 1이므로 $Diff(sql_3)$ = 1 - |1 - 1|× 0.05 = 1로 계산된다. 따라서 $\delta(sql_3)$ = (Supp ({sql_3}) ×Diff (sql3))/1 = 0.7×1/1 = 0.7과 같다. 즉, sql_3은 70%의 정상 행위를 가지게 된다. sql_2가 수행될 경우 이전 SQL 문장의 집합은 {sql_3}이며 현재의 SQL 문장이 sql_2가 되므로 CS = {{sql_2, sql_3}}과 같다. sql_2의 정상 행위도는 다음과 같이 계산된다.

$$\delta(sql_1) = \frac{Supp(\{sql_2\}) \times Diff_2}{1 + Supp(\{sql_2, sql_3\})} + \frac{Supp(\{sql_2, sql_3\}) \times Conf(\{sql_3\} \rightarrow \{sql_2\})}{1 + Supp(\{sql_2, sql_3\})}$$

$Diff(sql_2)$ = 1 - |1 - 1|×0.05 = 1이 되므로 (sql_2) = ((0.8×1) +(0.7×1)) / (1 + 0.7) = 0.88(88%)와 같다.

5. 실험 결과 및 분석

실험을 위해 유닉스 기반의 데이터베이스 시스템인 Oracle 8.05(Solaris)을 사용하여 로그 데이터를 수집하였다. 또한 사용자 SQL 문을 추출하기 위해서 Oracle Trace Utility를 사용하였으며 부속절의 분리를 위한 SQL 문 파싱을 위해서 Java를 사용하였다. 적합한 환경 변수를 추출하기 위해 본 실험에서는 <표 4>와 같은 환경 변수를 설정하여 실험을 수행하였다.

〈표 4〉 실험을 위한 환경 변수

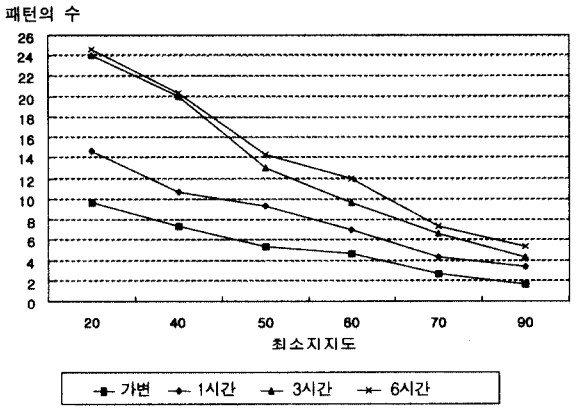
환 경 변 수	내 용	단 위
지지도	20, 40, 50, 60, 70, 90	퍼센트(%)
트랜잭션 구분 단위	고정 길이 트랜잭션(1, 3, 6) 가변 길이 트랜잭션	시간(Hour)

<표 4>에서 고정 길이 트랜잭션은 데이터베이스 로그 데이터에 대해서 시간간격을 1, 3 및 6시간으로 나누어서 각각을 별도의 트랜잭션으로 식별한다. 반면 가변 길이 트랜잭션에서는 한 사용자에 대해서 데이터베이스로의 로그인에서 로그아웃까지 동안에 생성된 SQL 문의 집합으로 식별된다. 실험에 사용된 데이터의 특징은 <표 5>와 같다.

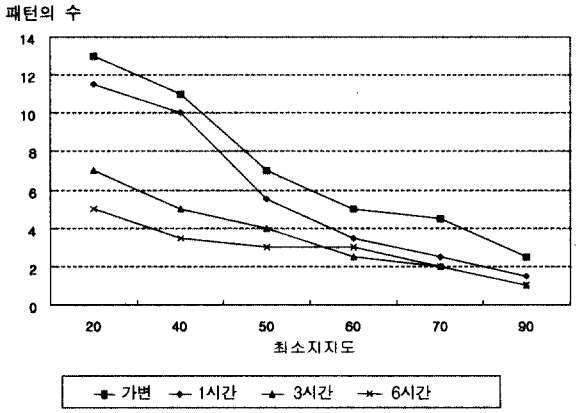
〈표 5〉 실험 대상자 정보

사 용 자	로그 데이터의 정보
프로그램 사용자	기간 : 3개월 전체 트랜잭션의 수 : 747 전체 SQL 문의 수 : 16217
대화 환경 사용자	기간 : 4개월 전체 트랜잭션의 수 : 67 전체 SQL 문의 수 : 605

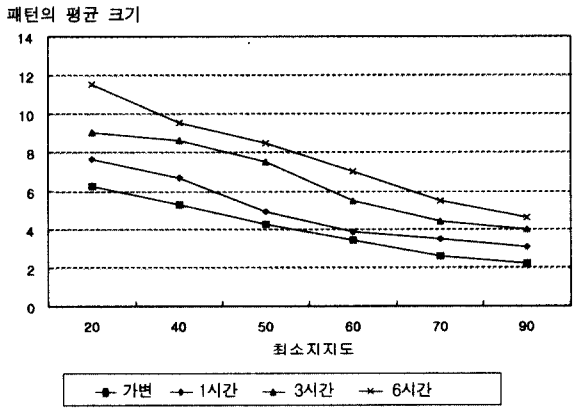
프로그램 환경에 있어서 정상 행위 프로파일의 추출에 영향을 미치는 것은 트랜잭션 단위 시간의 설정이다. (그림 6)은 트랜잭션의 단위 시간을 다양하게 설정하였을 때의 정상 행위 패턴 추출 결과를 보여준다. (그림 6)(a)는 프로그램 사용자의 최소지지도 변화에 대한 다양한 트랜잭션 시간 단위 설정에 따른 정상 행위 패턴의 수이며 (그림 6)(b)는 대화 환경의 사용자에 대한 정상 행위 패턴의 수이다. (그림 6)(c)와 (그림 6)(d)는 각각 프로그램과 대화 환경 사용자에 의해 생성된 패턴들의 평균 크기를 나타낸다. 프로그램 사용자의 경우 6시간을 트랜잭션으로 지정했을 때 정상 행위 패턴의 수가 가장 많이 생성된 반면 대화 환경 사용



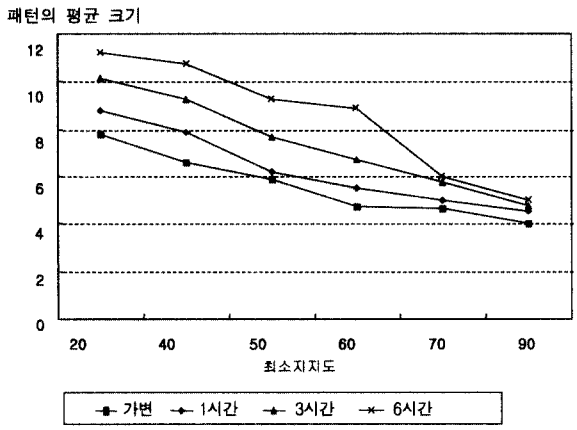
(a) 프로그램 사용자



(b) 대화환경 사용자



(c) 프로그램 사용자



(d) 대화 환경 사용자

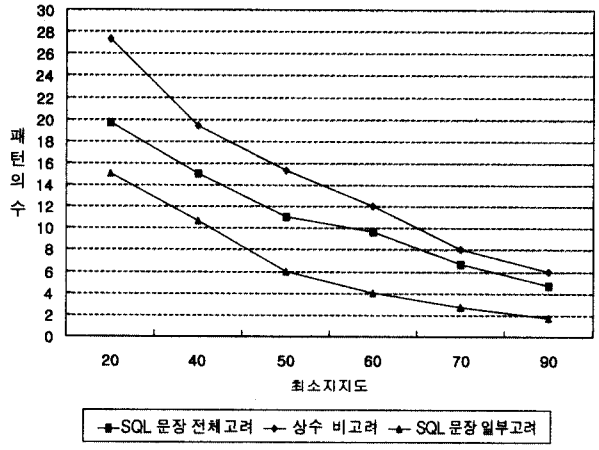
(그림 6) 트랜잭션 시간 단위별 정상 행위 패턴의 수

용자의 경우에는 가변 길이 트랜잭션의 경우 정상 행위 패턴의 수가 가장 많았다. 프로그램의 경우 패턴의 수와 패턴의 평균 크기는 비례하였지만 대화 환경의 경우는 패턴의 수가 가장 많은 가변의 경우가 크기가 가장 작았다. 이것은 대화 환경의 경우 사용자가 데이터베이스에 접근할 경우 사용자 인증, 언어 설정 등의 설정을 위한 시스템 질의가 반복되기 때문에 트랜잭션 시간을 크게 잡을수록 패턴의 크기가 커지게 됨을 알 수 있다.

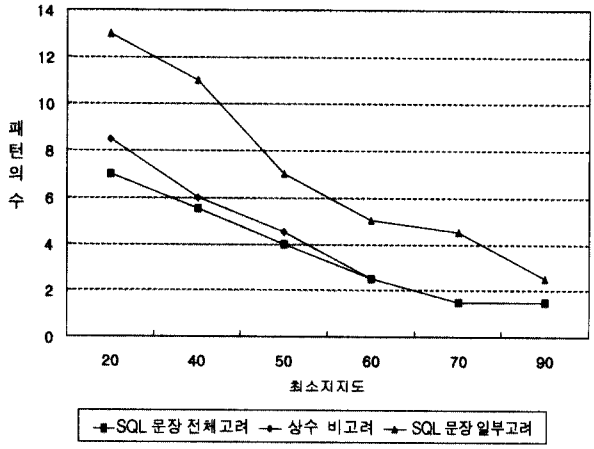
대화식 환경에서는 사용자가 자유롭게 SQL 문을 수행할 수 있으며 반복적이고 규칙적인 작업을 수행하는 확률이 적어진다. 반면에 특정 사용자는 그 사용자의 접근 권한에 따라 접근하는 객체가 한정됨으로써 사용된 SQL 문 전체를 고려하여 정상 행위 프로파일을 추출하는 경우보다 테이블과 컬럼 등과 같은 SQL 문의 일부만을 대상으로 하여 사용자가 사용한 SQL 문의 공통 사항을 파악하여 정상 행위 프로파일이 생성될 수 있음을 알려준다. 상수를 고려하지 않는다는 것은 SQL 문 전체를 고려하되 숫자/문자열 상수는 제외함을 뜻한다. 이에 대한 프로그램 및 대화 환경에서의 정상 행위 프로 파일 추출 결과는 (그림 7)과 같다.

(그림 7)(a)는 프로그램 사용자의 평균 패턴수를 나타내며 SQL 문 전체를 고려했을 경우보다 SQL 문의 상수를 고려하지 않았을 경우 더 많은 패턴이 생성되었음을 보여준다. (그림 7)(b)의 대화 환경에서는 SQL 문의 일부만을 고려했을 경우 지지도가 높은 패턴이 많이 생성되었음을 알 수 있다. 대화 환경에서는 SQL 문 전체를 고려할 경우나 상수만을 고려하지 않을 경우보다 테이블과 컬럼 등의 일부 정보만을 고려함으로써 보다 더 효율적인 정상 행위 프로파일을 생성할 수 있음을 알 수 있다. 따라서 사용자에 의해 다양한 로그 데이터가 존재할 경우 비효율적인 측면을 보이는 일반적인 비정상 행위 탐지 모델의 단점을 어느 정도 보완할 수 있다.

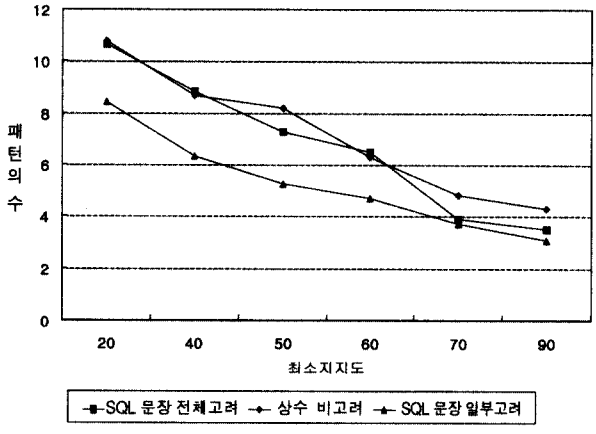
(그림 8)은 프로그램 환경과 대화 환경에서 각 트랜잭션에 대한 정상 행위도의 평균을 나타낸다. 최소 지지도를 높여가면서 실험을 한 결과 (그림 8)과 같이 정상 행위도는 감소하는 경향을 보였다. 또한, 대화 환경 사용자의 경우에 사용자의 행위가 불규칙적으로 발생이 되므로 프로그램 사용자보다



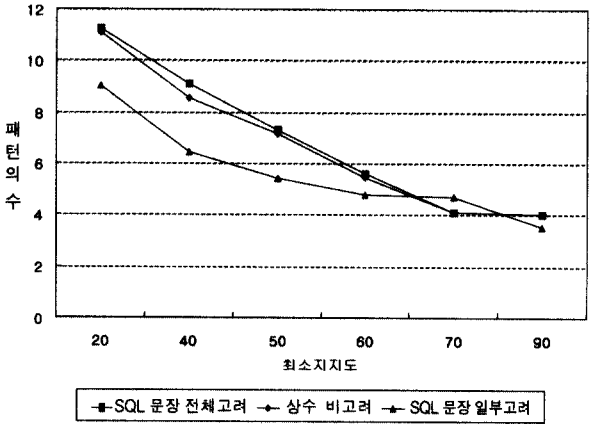
(a) 프로그램 사용자



(b) 대화 환경 사용자

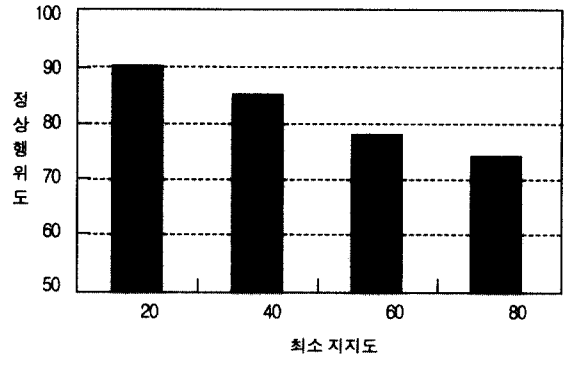


(c) 프로그램 사용자

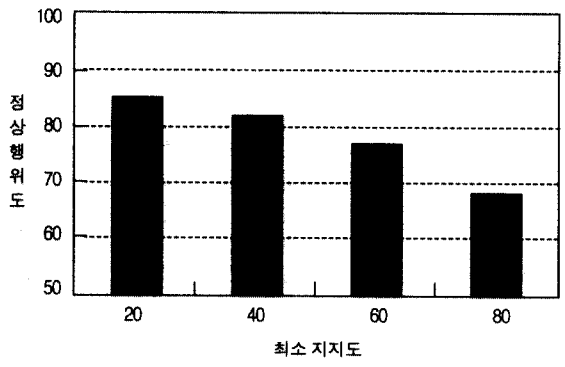


(d) 대화 환경 사용자

(그림 7) SQL 문의 고려대상에 따른 정상 행위 패턴



(a) 프로그램 사용자



(b) 대화 환경 사용자

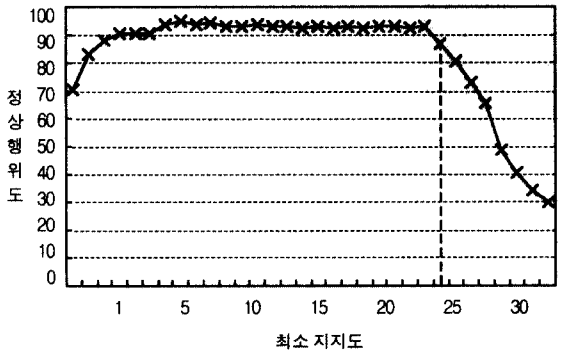
(그림 8) 최소 지지도에 따른 정상 행위도

낮은 정상행위도를 나타내고 있다.

(그림 9)에서는 정상 행위 프로파일을 생성한 사용자의 온라인 SQL 문에 다른 사용자의 SQL 문을 혼합함으로써 정상 행위의 변화 과정을 보이도록 한다. (그림 9)에서 프로그램 사용자의 정상 행위 프로파일에 대해서 24개의 사용자 자신의 SQL 문 이후에 8개의 다른 사용자의 SQL 문을 첨가한 트랜잭션에 대한 정상 행위의 변화 과정을 보여주고 있다.

그림에서 보는 바와 같이 자신의 SQL 문이 수행되었을 경우에는 대체적으로 80% 이상의 정상 행위도을 보였으나 다른 사용자의 SQL 문을 수행하였을 경우 급격하게 정상 행위도가 감소하는 것을 볼 수 있다. 따라서 타인의 계정을 도용하거나 자신이 조작할 수 없는 데이터베이스 정보에 접근

할 경우 비정상 행위로 판정함으로써 보안 관리자로 하여금 계정을 정지시키거나 유보할 수 있도록 할 수 있다.



(그림 9) 정상 행위도의 변화 과정

6. 결론 및 향후 연구 과제

본 논문에서는 데이터베이스 사용자의 정상 행위 프로파일을 추출함으로서 불법 침입자뿐 아니라 내부 권한 오용자의 비정상 행위를 판정하는 방안을 제안하였다. 이를 위해서, 패턴 트리를 이용하여 SQL 명령의 각 구문별로 정상 행위 프로파일을 생성할 수 있었으며 비정상 행위 판정시 온라인 명령과의 유사도 계산을 위해 빠른 탐색 방법을 제공할 수 있었다. 또한 프로파일 생성 이전에 정상 행위에 영향을 미치지 못하는 데이터를 제거함으로써 많은 시간을 소요하는 프로파일 생성 시간을 단축시킬 수 있었으며 정상 행위 프로파일은 프로그램 환경과 대화식 환경의 차이점을 감안하여 생성하였다. 향후 연구과제로는 최적의 정상 행위 프로파일 추출을 위한 다양한 환경에서의 실험과 이를 기반으로 실험 매개 변수의 최적의 임계치 값의 설정이 요구된다. 또한, 시스템의 목적과 용도에 따라 데이터베이스의 사용 환경이 달라지게 되며 사용자 또한 다양하기 때문에 각각의 시스템에 대한 효율적인 정상 행위 프로파일의 추출 실험과 효율적인 환경 변수 설정 알고리즘이 요구된다. 네트워크를 통한 프로그램의 경우에는 장기간 연결이 이루어지는 경우가 많기 때문에 정상 행위 프로파일 추출을 위해서 적합한 트랜잭션의 구분을 위한 효율적인 알고리즘의 연구가 필요하다.

참 고 문 헌

[1] William Stallings, Network And Internetwork Security Principles and Practice, Prentice Hall, Inc. 1995.

[2] Carter and Katz, Computer crime: an emerging challenge for law enforcement, FBI Law Enforcement Bulletin, pp.1-8, December, 1996.

[3] W. Lee and S. Stolfo, "Data Mining Approaches for Intrusion Detection," In Proc. Of the 7th USENIX Security Symposium, San Antonio, Texas, January, 1998.

[4] W. Lee, S. Stolfo and P. K. Chan, "Learning Patterns from Unix Process Execution Traces for Intrusion Detection," Proc. AAAI-97 Work. On AI Methods in Fraud and Risk Management, 1997.

[5] S. Stolfo, A. L. Prodromidis, S. Tselepis, W. Lee, D. Fan, P. K. Chan, "JAM : Java Agents for Meta-Learning over Distributed Databases," Proc. KDD-97 and AAAI97 Work. On AI Methods in Fraud and Risk Management, 1997.

[6] Sandeep Kumar, Classification and Detection of Computer Intrusions. Ph. D. Disertation, August, 1995.

[7] T. D. Garvey and Teresa F. Lunt, "Model Based Intrusion Detection," In Proc. Of the 14th National Computer Security Conference, pp.372-385, October, 1991.

[8] D. E. Denning, Cryptography and Data Security, Addision-Wesley, 1982.

[9] D. E. Bell, L. J. La Padula, Secure Computer Systems : Mathematical Foundations and Model, Technical Report M74-244, MITRE Corp, 1974.

[10] K. J. Biba, Integrity Considerations for Secure Computer Systems, Technical Report 76-372, MITRE Corp., 1977.

[11] F. Chin, "Security in Statistical Databases for Queries with Small Counts," TODS, 3 : 1, March, 1978.

[12] C. Chung, M. Gertz, K. Levitt, "DEMIDS : A Misuse Detection System for Database Systems," IFIP WG 11.5 1999.

[13] Yon-Wu Huang, Philip S. Yu, "Adaptive Query Processing for Time-Series Data," KDD-99, ACM August, 1999.

[14] G. M. Landau and U. Vishkin, Fast parallel and serial approximate string matching. Journal of Algorithms, (2), pp.157-169, 1989.

[15] Ming-Syan Chen, Jiawei Han, Philip S. Yu, "Data Mining: An Overview from Database Perspective."

[16] 윤정혁, 오상현, 이원석, "사용자 명령어 분석을 통한 비정상 행위 판정에 관한 연구", 한국정보보호학회논문지, 제10권 제4호, 2000.

[17] Rakesh Agrawal, Ramakrishnan Srikant, "Fast Algorithms for Mining Association Rules," In Proc. Of the 20th VLDB Conference, 1994.

[18] Computational Mathematics, Online Lecture, http://www.maths.bris.ac.uk/~macpc/lect5.

[19] Ian H. Witten and Eibe Frand, Data Mining practical machine learning tools and techniques with JAVA implementations, Morgan Kaufmann Publishers, pp.119-156, 1999.

[20] H. Mannila, H. Toivonen and I. Verkamo, "Discovery of frequent episodes in event sequences," Data Mining and Knowledge Discovery, 1, 3, pp.259-289, 1997.

박 정 호

e-mail : pjho@kr.ibm.com

1998년 대전 산업대 전자계산학과(학사)

2000년 연세대학교 컴퓨터학과(석사)

2000년~현재 한국 IBM

관심분야 : 침입 탐지, 데이터마이닝, 자바 프로그래밍

오 상 현

e-mail : osh@amadeus.yonsei.ac.kr

1996년 제주대학교 정보공학과(학사)

1998년 연세대학교 컴퓨터과학과(석사)

1998년~현재 연세대학교 컴퓨터과학과 박사과정

관심분야 : 침입 탐지, 데이터 마이닝, 에이전트 시스템

이 원 석

e-mail : leewo@amades.yonsei.ac.kr

1985년 미국 보스턴대학교 컴퓨터과학과 (학사)

1987년 미국 퍼듀대학교 컴퓨터공학과 (석사)

1990년 미국 퍼듀대학교 컴퓨터공학과 (박사)

1990년~1992년 삼성전자 선임연구원

1993년~1999년 연세대학교 컴퓨터과학과 조교수

1999년~현재 연세대학교 컴퓨터과학과 부교수

관심분야 : 분산 데이터베이스, 멀티미디어 데이터베이스, 객체 지향 시스템, 데이터마이닝