동질형 호스트들로 구성된 정보시스템에 적합한 침입탐지시스템의 설계

이 종 성*, 조 성 언**, 조 경 룡***

Design of Intrusion Detection System to be Suitable at the Information System Organized by Homogeneous Hosts

Jong-Sung Lee, Sung-Eon Cho, Kyung-Ryong Cho

요 약

컴퓨터 및 네트워크 기술이 발전하고 이에 대한 의존도가 증가함에 따라 컴퓨터의 결함은 인적 물적 손실뿐만 아니라 조직의 경쟁력을 약화시키는 결과를 초래하게 되어 정보사회의 역기능으로 컴퓨터 보안 문제가 중요하게 대두되고 있다. 침입탐지시스템(Intrusion Detection System : IDS)은 불법적인 침입에 의한 시스템 결함으로부터 컴퓨터를 보호하기 위해 침입을 탐지하고 이에 대한 적절한 조치를 취하는 역할을 수행한다. 최근까지 IDS에 대한 다양한 기법과 모델들이 개발되고 있으나 컴퓨터 통신망의 복잡성, 대상 시스템의 원초적 취약성, 정보 보호에 대한 이해 부족 및 새로운 불법 침입 기법의 개발 등으로 기존의 어떤 기법 또는 모델도 완전하지 못한 실정이다. 본 논문에서는 동질형 호스트들로 구성된 정보시스템에 적합한 침입탐지시스템을 제안하고, 이를 설계하고 프로토타입을 구현하여 그 타당성을 보인다. 제안한 침입탐지시스템은 여러 동질형 컴퓨터에 단위 센서 침입탐지시스템을 설치하고, 분산된 단위 센서 침입탐지시스템들 중 어느 하나가 프로세스에 의해 발생된 시스템 호출 순서 중 비정상적인 시스템 호출을 탐지한 경우 이를 다른 센서 침입탐지시스템들과 서로 동적으로 공유하여 전체 정보시스템에 대한 새로운 침입에 대하여 효율적으로 탐지할 수 있게 한다.

Abstract

With the development of computer&network technology and the growth of its dependance, computer

* 한국정보보호센터 개발부

** 순천대학교 공과대학 정보통신공학과

접수일자 : 1999년 12월 31일

failures not only lose human and material resources but also make organization's competition weak as a side-effect of information society. Therefore, people consider computer security as important factor. Intrusion Detection Systems (IDS) detect intrusions and take an appropriate action against them in order to protect a computer from system failure due to illegal intrusion. A variety of methods and models for IDS have been developed until now, but the existing methods or models aren't enough to detect intrusions because of the complexity of computer network, the vulnerability of the object system, insufficient understanding for information security and the appearance of new illegal intrusion method. We propose a new IDS model to be suitable at the information system organized by homogeneous hosts and design for the IDS model and implement the prototype of it for feasibility study. The IDS model consist of many distributed unit sensor IDSs at homogeneous hosts and if any of distributed unit sensor IDSs detect anomaly system call among system call sequences generated by a process, the anomaly system call can be dynamically shared with other unit sensor IDSs. This makes the IDS model can effectively detect new intruders about whole information system.

Ⅰ. 서 론

정보화의 급속한 진전에 따라 국가의 주요 기반 구조도 정보시스템 및 정보통신망에 대한 의존도가 나날이 높아가고 있다. 이러한 상황에서 컴퓨터의 결함은 인적 물적 손실뿐만 아니라 국가 안보 및 경쟁력을 약화시키는 결과를 초래하게 되어 정보사회의 역기능으로 컴퓨터 보안 문제가 중요하게 대두되고 있다. 일반적으로 침입자가 컴퓨터 시스템에 침입하는 과정은 크게 3단계로 구분하는데, 침입대상 컴퓨터가 연결된 지역 네트워크에 침입하는 네트워크 침입단계와 침입 대상 컴퓨터의 일반 사용자 권한을 획득하는 사용자 권한 접근단계, 그리고 일반 사용자 권한으로 시스템에 내재된 결함을 이용하여 루트 권한을 획득하여 침입대상시스템을 완전하게 침입하는 완전침입단계로 구분할 수 있다. 이때 첫 번째 단계의 보안 문제를 해결하기 위해 네트워크 기반 보안 기술이 요구되고 두 번째 세 번째 단계의 보안 문제를 해결하기 위해 호스트 기반 보안 기술이 요구된다. 이러한 침입 위협에 대처하기 위해 정보보호를 필요로 하는 문서나 시스템에 대한 불법 침입을 분석하고 탐지하는 감사 기술의 발전적 형태인 침입 탐지 시스템(Intrusion Detection System : IDS)에 관한 연구가 활발히 진행되고 있다[1]-[4].

침입 탐지 시스템은 불법적인 침입으로부터 컴퓨터를 보호하기 위해 침입을 탐지하고 이에 대한 적절한 조치를 취하는 역할을 수행한다. 침입 탐지 시스템은 크게 데이터의 소스(source)를 기반으로 하는 분류 방법과 침입의 모델을 기반으로 하는 분류 방법으로 나눌 수 있으며, 데이터 소스를 기반으로 하는 분류 방법은 단일 호스트로부터 생성되고 모아진 감사(audit) 데이터를 침입 탐지에 사용하는 단일호스트 기반(host based)과, 여러 호스트들로부터 생성되고 모아진 감사 데이터를 침입 탐지에 사용하는 다중호스트 기반(multihost based), 그리고 네트워크의 패킷 데이터를 모아 침입을 탐지하는데 사용하는 네트워크 기반(network based)으로 구분할 수 있다. 또한 침입 모델을 기반으로 하는 침입탐지시스템의 일반적인 분류 방법은 정상적인 시스템 사용에 관한 정상 행위 프로파일과 시스템 상태를 유지하고 있는 동안 이 프로파일에서 벗어나는 행위들을 탐지하는 비정상적인 행위 탐지(anomaly detection) 방법과, 시스템의 알려진 취약점들을 이용한 공격 행위들에 대한 공격 특징 정보를 통해 침입을 탐지는 오용 침입탐지(misuse detection) 방법, 그리고 이 두 방법을 결합하여 침입을 탐지하는 하이브리드 침입탐지 방법으로 분류할 수 있다[4],[5].

일반적인 침입탐지시스템의 중요 요구 사항은 시스템 관리자 없이도 지속적으로 수행되어야 하며, 컴퓨터 시스템에 최소한의 오버 헤드를 부과해

야 하고, 새로운 침입 유형의 변화에 대한 자체 학습 기능과, 어떤 침입탐지 모듈에 결함이 발생되어도 전체 침입탐지시스템에 큰 영향을 주지 않는 결함 허용 관리 기능, 그리고 시스템의 정상상태를 침입이라고 탐지하는 긍정적 결함(false positive) 및 시스템의 침입상태를 정상상태로 판단하는 부정적 결함(false negative)과 같은 잘못된 침입 탐지를 방지해야 한다[3],[4],[6].

이와 같은 침입 탐지 서비스의 요구에 따라 다양한 기법과 모델들[7-12]이 개발되고 있으나 컴퓨터 통신망의 복잡성, 대상 시스템의 원초적 취약성, 정보 보호에 대한 이해 부족 및 새로운 불법 침입 기법의 개발 등으로 기존의 어떤 기법 또는 모델도 완전하지 못한 실정이다. 특히, 탐지대상에 대한 정상 개념이 시간이 지나감에 따라 지속적으로 변화되므로 비정상적인 행위 탐지 방법에 따라 IDS를 구현하는 것이 오용탐지 방법으로 IDS를 구현하는 것 보다 많은 어려운 점이 존재하므로 현재 상용화된 IDS의 대부분은 오용탐지 방법에 따른 IDS이다. 따라서, 대부분의 IDS가 오용탐지 방법의 원초적인 문제점인 새로운 침입을 탐지하지 못하는 문제점을 안고 있다.

이에, 본 논문에서는 탐지 대상을 동질형 호스트들에서 공통적으로 수행되는 특권 프로세스로 하고, 특권 프로세스(privilege process)가 수행할 때 발생하는 시스템 호출 순서 중 비정상적인 시스템 호출을 탐지하여 이를 분산된 각각의 침입탐지 시스템들이 서로 동적으로 공유하여 침입자로부터 시스템의 침입에 대한 면역력을 향상시키는 동질형 호스트들로 구성된 정보시스템에 적합한 침입탐지시스템을 설계하고 프로토타입을 구현하여 그 타당성을 보인다.

Ⅱ. 제안한 침입탐지시스템 모델

제안한 침입탐지시스템은 네트워크를 통해 동질형의 여러 호스트에 분산된 단위 센서 침입탐지부를 포함하고, 각각의 호스트는 자신의 단위 센서 침입탐지부를 통해 호스트에서 발생하는 이벤트들을 모니터링 하면서 기설정된 이벤트 패턴 정보(self[1] 정보)에 따라 침입 여부를 판단한다. 이때, 각각의 호스트에서 감시하는 대상은 모든 호스트에 존재하는 동일한 프로세스이며, 각 호스트는 상기 프로세스에 대한 비정상 이벤트를 공유하면서 새로운 침입으로부터 전체 시스템의 침입탐지율을 향상시킨다.

2.1 탐지 대상

일반적으로 비정상행위 침입 탐지시스템에서 어떤 것을 탐지대상으로 정해야 시스템 침입을 탐지할 수 있는지가 명확하지 않으므로 비정상행위 침입 탐지시스템에서 탐지대상을 정하는 작업이 가장 중요하다. 본 논문에서 탐지 대상 객체는 특권 프로그램을 수행하는 특권 프로세스[2]와 시스템 서버[3]로 한다. 그 이유를 일반적인 운영체제 구성 특징에 의해 살펴보면 다음과 같다.

일반적인 운영체제는 (그림 2-1)에 도시된 바와 같이, OS 커널이 모든 시스템 자원(메모리, 디스크, 파일, CPU)을 관리하고, 모든 시스템 자원은 단지 시스템 호출을 통해 접근될 수 있다. 일반적으로 커널은 시스템 자원을 보호하기 위해 접근제어를 통한 보호 메커니즘을 제공한다. 따라서, 커널은 사용자 프로세스의 행위를 제한하여 사용자 프로세스에 의한 보안 위반을 방지할 수 있다. 그

1) self와 noself는 면역시스템에서 유래된 용어로서 본 논문에서 self는 합법적인 사용자, 허가된 행동 등을 의미하고, nonself는 침입자, 컴퓨터 바이러스, 트로이목마, 스프핑 등을 의미한다.

2) setuid 프로그램(예를 들어, ping, ufsrestore, rdist 등)을 수행시키는 프로세스와 같이 시스템 관리자 권한으로 수행하는 프로세스를 칭하며, 이후 본 논문에서는 프로세스와 혼용하여 사용한다. solaris 2.6(SunOS 5.6)에 75개의 setuid 프로그램 존재.

3) ftpd 프로그램 등을 수행시키는 프로세스를 칭하며 이 또한 시스템 관리자 권한으로 수행하는 프로세스이므로 특권 프로세스와 동일한 것으로 간주한다.

러나, 특권 프로세스들과 시스템 서버들은 고유의 작업을 수행하기 위해 커널의 보호 메커니즘을 우회하여 시스템자원을 접근할 수 있다[13]. 예를 들어, 사용자가 패스워드를 변경하기 위해서는 시스템 자원인 /etc/passwd 파일 변경을 요구하는데 이때 사용자에게 루트 권한이 부여되어야 한다.

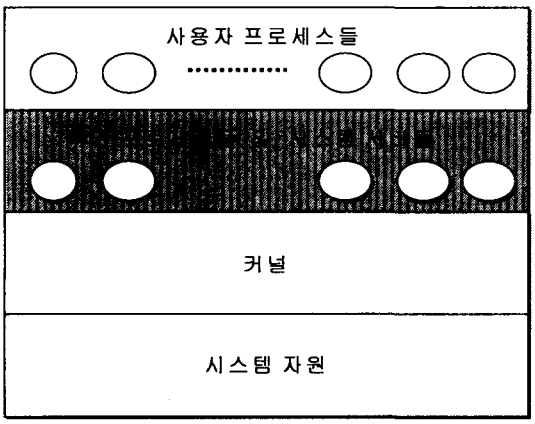

그림 2-1 탐지 대상 시스템의 특징
Fig. 2-1 Character of System Monitored

특권 프로세스들과 시스템 서버들은 커널의 일부분으로 구성할 수 있으나, 일반적으로 커널이 비대해지는 것을 방지하기 위해 그림 2-1과 같이 커널 밖에 구성한다. 이에 따라 사용자 프로세스의 경우 OS 커널의 보호 메커니즘에 의해 자원 접근에 제한을 받으나 특권 프로세스와 시스템 서버의 경우 관리자 권한을 획득하므로 인해 OS 커널에 의한 접근 제어에 제한을 받지 않고 시스템 자원을 사용할 수 있으므로 악의적으로 시스템 자원을 사용할 수 있다. 이와 같은 이유로 본 논문에서 특권 프로세스를 탐지 대상으로 한다.

따라서, 제안한 침입탐지시스템은 탐지 대상인 특권프로세스가 비정상적인 행위를 수행하여 시스템에 장애를 발생시키는 공격, 이를테면, setuid 프로그램을 수행시킨 특권프로세스가 수행도중 악성코드를 내포하여 시스템에 장애를 발생하는 공지된 공격인 버퍼오버플로우 공격과 같은 공격을 탐지할 수 있다. 다시 말해, 제시한 침입탐지시스템은 버퍼오버플로우 공격 외에 어떤 공격이 특권 프로세스 행위를 정상 행위와 다르게 하는 경우 이를 침입으로 탐지할 수 있는 비정상 행위 탐지 방법을 기반으로 하는 침입탐지 시스템이다.

2.2 모델 정의

【정의 1】 시스템 행위 집합(System Behavior Set : SBS)

시스템 S가 수행하는 동안 생성할 수 있는 모든 이벤트들의 집합을 의미한다.

$$SBS = \{ \cdots e_a, e_b, \cdots, e_k, \cdots \}$$

이때, 첨자는 이벤트의 종류를 의미한다.

【정의 2】 시스템 행위 궤적(System Behavior Trace : SBT)

시스템 S가 수행하는 동안 발생하는 이벤트 순서를 의미한다.

$$SBT = e_1, e_2, \cdots, e_n, e_{n+1}, \cdots$$

각각의 이벤트들은 발생된 시간 정보를 가지며 $T(e_1)$, $T(e_2)$, …, $T(e_n)$, $T(e_{n+1})$, …, 으로 나타내고, 시스템 행위들 간의 시간 순서는

$$T(e_n) < T(e_{n+1}), \text{ 단 } n \geq 1$$

【정의 3】 시스템 감사 행위 집합(Audit Event : AE)

시스템 S의 감사 서브시스템이 제공하는 이벤트 집합을 의미하며,

$$AE = \{ \cdots Ae_a, Ae_b, \cdots, Ae_k, \cdots \}$$

시스템 감사 행위 집합은 시스템 S에서 발생하는 시스템 행위 집합(SBS)의 부분집합이다.

즉, $SBS \supset AE$

예를 들어, Solaris 2.x의 경우 기본적으로 250개의 OS 커널 수준의 감사 이벤트를 제공하고 63개의 사용자 수준의 감사이벤트를 제공한다.

【정의 4】 시스템 감사 궤적(System Audit Tail : SAT)

시스템 S가 수행하는 동안 감사 서브시스템에 정의된 이벤트의 발생 순서를 의미한다.

$$SAT = Ae_1, Ae_2, \cdots, Ae_n, Ae_{n+1}, \cdots$$

각각의 이벤트들은 발생된 시간 정보를 가지며 $T(Ae_1)$, $T(Ae_2)$, …, $T(Ae_n)$, $T(Ae_{n+1})$, …으로 나타내고, 감사 이벤트들 간의 시간 순서는

$T(Ae_n) < T(Ae_{n+1})$, 단 $n \geq 1$

【정의 5】 프로세스 행위 궤적(Process Behavior Trace : PBT)

프로세스 i에 대한 프로세스 행위 궤적은 $PBT_i = SC_{i_1}, SC_{i_2}, \cdots, SC_{i_n}, SC_{i_{n+1}}, \cdots$, 단 $1 \leq i \leq M$[4]이며, 프로세스가 수행하는 동안 발생하는 시스템 호출 순서[5]를 의미한다.

각각의 시스템 호출은 호출된 시간 정보를 가지며 $T(SC_{i_1})$, $T(SC_{i_2})$, …, $T(SC_{i_n})$, $T(SC_{i_{n+1}})$, …으로 나타내고, 시스템 호출들 간의 시간 순서는

$T(SC_{i_n}) < T(SC_{i_{n+1}})$, 단 $n \geq 1$

【정의 6】 시스템 감사 행위 궤적과 프로세스 행위 궤적과의 관계

시스템 S에서 수행되는 모든 프로세스에 의해 생성된 프로세스 행위 궤적(PBT)은 시스템 감사 행위(SAT)의 부분집합이다.

$SAT \supset PBT_i$, 단 $1 \leq i \leq M$

【정의 7】 실존정상행위패턴KB (Live Normal behavior Pattern Know- ledge Base : NP)

일정시간 정상적으로 수행되는 프로세스에 의해 생성된 프로세스 행위 궤적(PBT)을 그림 2-2와 같이 일정한 크기 단위로 나누어 구성한 궤적들의 집합을 의미한다.

프로세스 P에 대한 정상 행위 패턴은

$P_{NP} = \{P_{NP1}, P_{NP2}, P_{NP3}, P_{NP4}, \cdots, P_{NPn}\}$

이때 n은 프로세스 P에 대한 정상 행위 패턴 수를 의미한다.

한편, 정상적으로 수행하는 프로세스 P의 프로세스 행위 궤적(PBT)을 P_{PBT}라고 하면 궤적의 단위 길이에 따라

$P_{PBT} \supseteq P_{NP}$

이다.

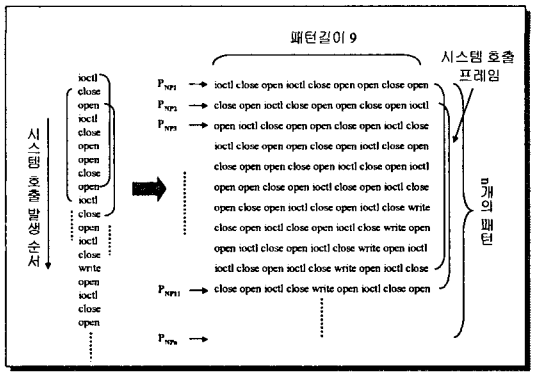

그림 2-2 시스템 호출 순서를 패턴 길이 9로 나눈 예
Fig. 2-2 Example of Dividing System Call Sequence by Pattern Length 9

【정의 8】 시스템 호출 프레임 크기(system call Frame Size : FS)

시스템 호출 프레임은(그림 2-2)에 도시된 바와 같이 프로세스에 의해 생성되는 시스템 호출에 대해 r-contiguous-bits 방식에 의해 생성되는 연속적인 패턴들을 의미하며 시스템 호출 프레임 크기(FS)는 시스템 호출 프레임에 포함되는 시스템 호출 패턴 개수를 의미한다. (그림 2-2)에서 시스템 호출 프레임 크기가 10임을 알 수 있다.

【정의 9】 합성정상행위패턴KB(Composition Normal behavior Pattern Knowledge Base : CNP)

침입탐지 대상 프로세스들 (P_1, P_2, P_3, …, P_{PN})이 동질형 호스트들 (H_1, H_2, H_3, …, H_{HN})에서

4) M은 시스템 S가 제공하는 최대 한도로 생성할 수 있는 프로세스 수로서 최대 프로세스 수는 커널에 존재하는 프로세스 테이블의 총 엔트리 수를 의미한다.

5) 본 논문에서는 시스템 호출 중 open, close, ioctl, write, read, exit 등과 같은 호출함수 이름만을 사용한다.

행위패턴KB	입력패턴 A와 차	입력패턴 B와 차	입력패턴 A의 HD	입력패턴 B의 HD
open write close close	3	*2*		
open close write ioctl	*1*	3	1	2
ioctl ioctl ioctl ioctl	4	3		

입력패턴 A : open close write write　　입력패턴 B : open write ioctl read

그림 2-3 입력패턴 A, B에 대한 Hamming Distance 계산 예

Fig. 2-3 Example of Calculating Hamming Distance about Input Pattern A, B

정상적으로 수행하면서 가질 수 있는 거의 모든 프로세스 행위 궤적(PBT)들의 집합을 의미한다.

$$CNP = \{ \{P_{1(PBT)H_1}, P_{1(PBT)H_2}, P_{1(PBT)H_3}, \cdots, P_{1(PBT)H_m}\}, \{P_{2(PBT)H_1}, P_{2(PBT)H_2}, P_{2(PBT)H_3}, \cdots, P_{2(PBT)H_m}\}, \{P_{3(PBT)H_1}, P_{3(PBT)H_2}, P_{3(PBT)H_3}, \cdots, P_{3(PBT)H_m}\}, \cdots\cdots \{P_{PN(PBT)H_1}, P_{PN(PBT)H_2}, P_{PN(PBT)H_3}, \cdots, P_{PN(PBT)H_m}\}\}$$

【정의 10】 탐지자KB(Detector Pattern Knowledge Base : DP)

비정상적으로 수행되는 프로세스에 의해 생성된 프로세스 행위 궤적(PBT)을 일정한 크기 단위로 나누어 구성한 궤적들의 집합(ASP) 중 합성정상 행위 패턴(CNP)에 존재하지 않는 비정상 행위 패턴(AP)들 중 hamming distance가 일정 수(α) 이상인 패턴들의 집합을 의미한다.

즉, 프로세스 P에 대한 비정상 행위 패턴(AP)은 $P_{AP} = P_{ASP} - P_{CNP}$ 이고, 탐지자KB는

$$P_{DP} = \{x \in P_{AP} \mid HD(x) \geq \alpha\}$$

【정의 11】 어떤 행위패턴 i의 Hamming Distance

행위패턴 KB의 모든 패턴들 중 행위패턴 i와 가장 유사도가 가까운 패턴과의 차이 값을 의미한다. 즉, $HD(i) = \min$ {행위패턴KB의 모든 행위 패턴들과 행위패턴 I와의 차}

그림 2-3은 open, close, write, write 순으로 입력되는 입력패턴 A와 open, write, ioctl, read 순으로 입력되는 입력패턴 B가 행위패턴KB에 저장된 패턴들과 hamming distance를 구하는 것을 예시하고 있다.

【정의 11】의 의미를 그림 2-3을 참조하여 입력패턴 A에 대한 hamming distance를 구하는 과정을 통해 살펴보면, 먼저, 입력패턴 A에 대해 현재 행위패턴KB에 저장된 3종류의 패턴(즉, open write close close, open close write ioctl, 그리고 ioctl ioctl ioctl ioctl)과 각각 hamming distance를 계산하면, 행위패턴KB에 저장된 각각의 패턴에 대한 입력패턴 A의 hamming distance는 3, 1, 4가 된다. 따라서 입력패턴 A에 가장 유사한 open close write ioctl 패턴에 대한 hamming distance 1을 입력패턴 A에 대한 hamming distance로 정한다.

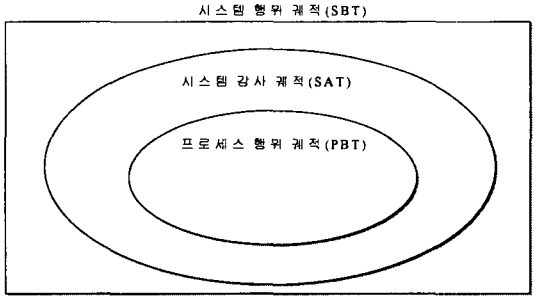

그림 2-4 궤적 관계

Fig. 2-4 Trace Relation

본 논문에서 제안한 침입탐지 시스템의 탐지 범위를 살피기 위해 전술한 시스템 행위 궤적(SBT)과 시스템 감사 궤적(SAT), 그리고 프로세스 행위 궤적(PBT)간의 관계를 도시하면 (그림 2-4)과 같다. 제안한 침입탐지 시스템은 탐지 대상을 특권 프로세스로 하므로 프로세스 행위 궤적 정보를 통해 침입을 탐지한다. 따라서, 제안한 침입탐지 시스템은 프로세스 행위 궤적에 침입 증후가 표현되지 않으면 침입을 탐지하지 못한다. 그러나, 대부분의 침입은 전술한 이유에 따라 특권 프로세스를 악용하여 이

루어지므로 본 논문에서는 모든 시스템 행위 궤적을 감사 데이터로 사용하지 않고 그중 프로세스 행위 궤적만을 감사 데이터로 사용한다.

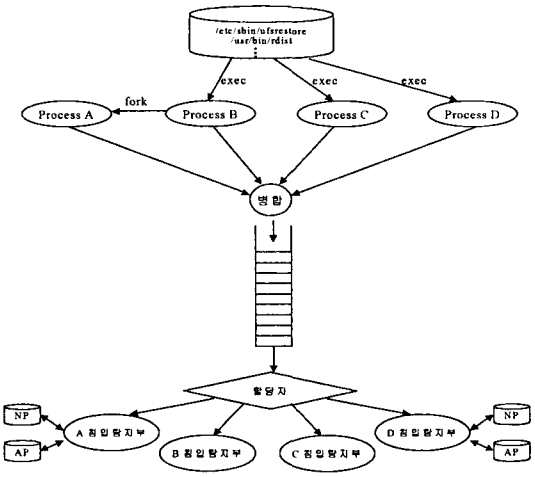

그림 2-5 특권프로세스 모니터링 모델
Fig. 2-5 Model of Monitoring Privilege Process

2.3 특권프로세스 모니터링 모델

특권프로세스의 행위를 모니터링하여 이를 통해 침입유무를 판단하는 특권프로세스 모니터링 모델을 그림 2-5를 참조해서 살펴보면 다음과 같다. 사용자 프로세스 B, C, D는 파일 시스템에 존재하는 특권 프로그램을 exec류[6] 시스템 호출을 통해 각각의 코드 세그먼트에 로딩하여 특권 프로그램을 수행하는 특권 프로세스로 그 성격이 바뀐 후 특권 프로그램을 시스템 호출을 발생하면서 수행한다. 한편, 특권 프로그램을 수행하는 특권 프로세스 B는 fork 시스템 호출을 통해 특권 프로세스 A를 생성하고, 특권 프로세스 A는 시스템 호출을 발생하면서 고유의 작업을 수행한다. 각각의 특권 프로세스에 의해 발생하는 시스템 호출은 발생 순서에 따라 병합되어 할당자에게 전달되고, 할당자는 전달된 시스템 호출 순서들 중 특권 프로세스에 의해 발생된 시스템 호출들을 추출하여 해당하는 침입탐지부를 기동시킨 후 전달한다. 기동된 침입탐지부는 전달된 시스템 호출을 실존정상행위패턴KB(NP)과 비정상행위패턴KB(AP)과 비교하여 특권 프로세스의 행위를 정상 또는 비정상으로 판단한다.

Ⅲ. 제안한 침입탐지시스템의 설계

본 장에서는 분산된 각 호스트에 설치된 침입탐지시스템이 SunOS BSM의 서브감사시스템을 통해 특권프로세스에 의해 생성되는 시스템 호출 순서 정보를 추출하여 기설정된 비정상 시스템 호출 패턴과 정상 시스템 호출 패턴과 각각 비교하여 비정상적인 시스템 호출을 탐지하고 이를 분산된 각각의 단위 센서 침입탐지 시스템들이 서로 동적으로 공유하여 모든 호스트에 설치된 침입탐지시스템들이 새로운 침입에 대한 지식을 증가시켜 이와 같은 침입으로부터 면역력을 향상시키는 침입탐지 시스템을 설계한다.

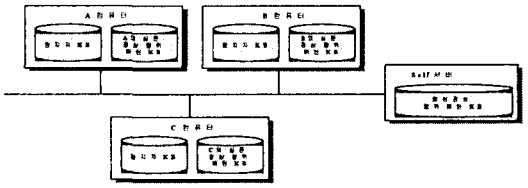

그림 3-1 제안한 침입탐지 시스템 구조
Fig. 3-1 Structure of Proposed Intrusion Detection System

제안한 시스템은 그림 3-1과 같이 구성되며, 각 컴퓨터의 침입탐지시스템은 공지된 침입에 대한 비정상행위 시스템 호출 패턴으로 구성된 탐지자 지식베이스(이하, '탐지자KB'라 칭함)와, 특권 프로세스에 의해 일정시간동안 발생한 시스템 호출 패턴을 수집하여 구성한 실존정상행위패턴지식베이스(이하, '실존정상행위패턴KB'라 칭함)를 포함하고, self 서버는 동질형 시스템들로부터 침입탐지

6) exec류 함수는 execl(2), execv(2), execle(2), execve(2), execlp(2), 그리고 execvp(2)로 모두 6종이 있는데 이들 모두 하는 기능은 동일하다[14].

대상 프로세스가 정상적으로 수행하면서 생성한 시스템 호출 패턴을 수집하는 방법으로 구성한 합성정상행위패턴지식베이스(이하, '합성정상행위패턴KB'라 칭함)을 포함한다.

3.1 합성정상행위패턴KB 구성

합성정상행위패턴KB를 구성하는 방법은 전술한 바와 같으며, 두 개의 호스트 시스템을 이용하여 합성정상행위패턴을 수집하는 것을 수행환경차이에 민감하고 복잡한 프로그램 중 하나인 sendmail 프로그램을 예로 하여 그림 3-2를 통해 살펴보면 다음과 같다. 그림 3-2는 호스트 A에서 case1의 경우에 생성한 sendmail에 대한 정상 행위 패턴을 정상행위패턴*KB*로 하여 *case2*, case3인 경우에 호스트 A에서 생성된 sendmail에 대한 시스템 호출 패턴들과 case1, 2, 3인 경우에 호스트 B에서 생성된 sendmail에 대한 시스템 호출 패턴들에 대한 hamming distance를 구한 결과를 나타낸다. 그림 3-2의 수행패턴 수는 각 경우에 sendmail의 시스템 호출 순서를 패턴 길이 9로 나누어 생성한 패턴순번을 의미하며, *case1*~case3은 sendmail 프로그램을 다르게 수행한 경우를 나타낸다.

그림 3-2에서 SE부분은 두 개의 호스트(A, B)의 환경차이로 인해 발생된 패턴 차이 부분이고, UA 부분은 사용자에 의해 *sendmail*이 다르게 수행될 때 발생되는 패턴에 의한 패턴차이 부분을 나타낸다. UA 부분은 호스트 A, B에서 동일하게 발생하며 이는 시스템 환경차에 의해 발생하는 것이 아니고 사용자의 *sendmail* 사용 패턴에 의해 결정된다. 따라서, 호스트 A에서 UA부분의 패턴을 취득하여 이를 호스트 B에 적용할 수 있으며, 환경차이로 인해 발생된 패턴 불일치 부분은 hamming *distance* 차이가 적으므로 무시할 수 있음을 알 수 있다.

따라서, 제안한 침입탐지 시스템에서는 여러 호스트로부터 탐지대상 프로세스의 정상행위를 수집하는 방법으로 하나의 합성정상행위KB를 구성한다.

각 KB의 크기는 탐지 대상프로세스에 의존하는데 ufsrestore 프로세스를 이용한 버퍼오버플로우공격[15], ps 프로세스를 이용한 race-condition 공격

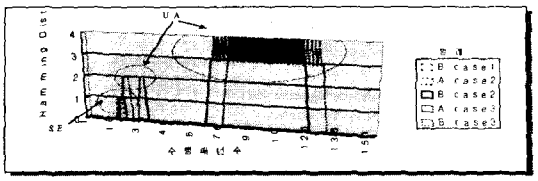

그림 3-2 sendmail 프로그램을 두 개의 호스트 시스템에서 수행한 경우

Fig. 3-2 Case of Executing Sendmail Program at Two Host Systems

3.2 패턴KB 구성

제안한 침입탐지시스템은 특권 프로세스에 대한 정상 및 침입 시스템 호출 패턴정보를 관리하기 위해 self 서버에 존재하는 합성정상행위패턴KB, 그리고 각 컴퓨터에 존재하는 실존 정상행위패턴KB와 침입패턴을 저장하는 탐지자KB를 포함한다.

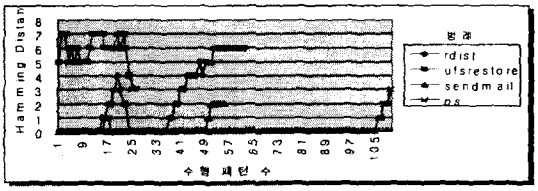

그림 3-3 공격에 사용된 특권 프로세스의 시스템 호출과 정상패턴과의 비교 (패턴길이가 7인 경우)

Fig. 3-3 Compare Normal Pattern with System Call Sequence of Privilege Process Used by Attack (Case of Pattern Length 7)

[16], rdist 프로세스를 이용한 버퍼오버플로우공격[17], 그리고 sendmail 프로세스를 이용한 버퍼오버플로공격[18]에 따라 발생되는 시스템 호출 순서를 시스템 호출 패턴 길이 7로 나누어 얻은 패턴들과 각각의 프로세스들에 대한 합성 정상 행위 패턴들과의 hamming distance 차이를 이용하여 탐지자KB를 구성하는 것을 그림 3-3을 통해 설명한다. 이때, 테스트를 통해 구한 각 프로세스에 대한 합성 정상 행위 패턴 개수는 각각 50, 71, 450, 512이고, 실존 정상 행위 패턴은 상기 정상행위패턴보다 적은 13, 16, 130, 125 이다. 도시된 바와 같

이 ufsrestore의 경우 hamming distance가 7이상인 패턴들만으로 탐지자KB를 구성할 경우 7개의 패턴으로 구성되고, rdist의 경우 hamming distance가 6이상인 패턴들만으로 탐지자KB를 구성할 경우 10개의 패턴으로 구성되며, sendmail의 경우 hamming distance가 2이상인 패턴들만으로 탐지자KB를 구성할 경우 5개의 패턴으로 구성되고, ps의 경우 hamming distance가 1이상인 패턴들만으로 탐지자KB를 구성할 경우 3개의 패턴으로 구성된다. 이때, 각 프로세스에 적용되는 hamming distance 값은 보안 강도에 의해 결정된다.

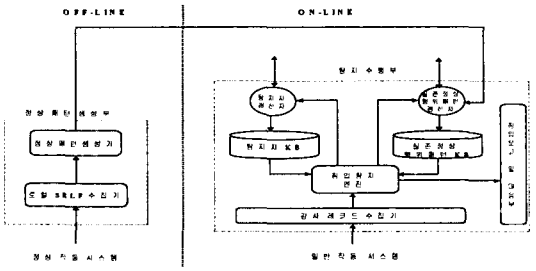

그림 3-4 각각의 컴퓨터에 대한 제안한 침입탐지 시스템

Fig. 3-4 Proposed Intrusion Detection System about Each Computer

3.3 시스템 구성

제안한 시스템에서 분산된 각각의 컴퓨터에 설치되는 침입탐지 시스템의 구성 요소를 그림 3-4를 참조하여 살펴보면 다음과 같다. 각 컴퓨터의 침입탐지 시스템은 크게 오프라인으로 수행되는 정상패턴생성부와 온라인으로 수행되는 탐지수행부로 대별된다.

3.3.1 정상패턴생성부

정상패턴생성부는 각 컴퓨터가 정상 상태, 즉 정상 사용자가 정상적인 수행을 할 때 발생하는 프로세스의 시스템 호출 순서를 로컬 self수집기를 통해 수집한 후, 패턴생성기를 통해 구성한다. 프로세스에 대한 시스템 호출 순서는 Solaris 2.6 BSM(Basic Security Module)의 감사서브시스템(audit subsystem)을 통해 구한다[19].

한편, 패턴생성기는 입력된 프로세스의 시스템 호출 순서를 r-contiguous-bits 방식[20]에 따라 r 크기 단위로 분리하여 시스템 호출 순서를 트리로 표현한다. 프로세스 A에 의해서 생성되는 시스템 호출에 대해 r을 3으로 하여 정상 트리를 구성하는 것을 그림 3-5을 참조하여 살펴보면 다음과 같다.

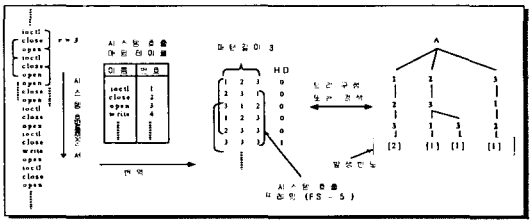

그림 3-5 프로세스 A에 대한 시스템 호출 패턴 생성 과정 또는 검색 과정

Fig. 3-5 Example of System Call Pattern Generation Step or Search Step about Process A

먼저, 프로세스 A가 수행하면서 연속해서 시스템 호출을 생성하면, 생성되는 순서에 따라 시스템 호출 매핑 테이블에 시스템 호출 이름을 등록하고 번호를 부여하여 추후 해당하는 시스템 호출 이름을 정수 값으로 번역한 후 자주 발생하는 패턴에 대해 추후 검색을 빠르게 하기 위해 패턴 발생 빈도에 따라 트리를 구성한다.

3.3.2 탐지수행부

탐지수행부는 공지된 침입 패턴 정보를 저장한 탐지자 KB와 이를 관리하는 탐지자갱신자와, 정상패턴생성부로부터 전달된 특권 프로세스의 정상행위 패턴을 저장한 정상 패턴 KB와 이를 관리하는 패턴갱신자와, 감사서브시스템에서 제공하는 감사레코드를 수집하는 감사레코드수집기와, 수집된 감사레코드로부터 시스템 호출을 분리하여 해당되는 침입탐지부를 기동시켜 침입을 탐지하는 침입탐지엔진과, 그리고 침입 발생을 알리고 해당 프로세스를 강제로 종료시키는 침입보고 및 대응부로 구성된다. 한편, 탐지자 KB와 정상 패턴 KB에 저장된 시스템 호출 패턴은 그림 3-5와 같은 트리구조로 각각 저장된다.

침입탐지엔진을 보다 상세히 살펴보면, 감사레코드수집기를 통해 감사서브시스템에서 제공하는 감사 레코드를 입력받은 후, 할당자가 수집된 감사 레코드에 특권 프로그램을 수행시키기 위한 execve 시스템 호출이 있는지 조사하여, 존재하는 경우 프로그램 프로세스 매핑 테이블에 등록시키고 해당되는 침입탐지부를 기동시킨다. 한편, 수집된 감사레코드 중 fork 시스템 호출이 있는 경우, 부모 프로세스의 프로그램과 생성된 프로세스의 PID를 매핑시켜 해당 프로그램의 침입탐지부를 기동시킨다.

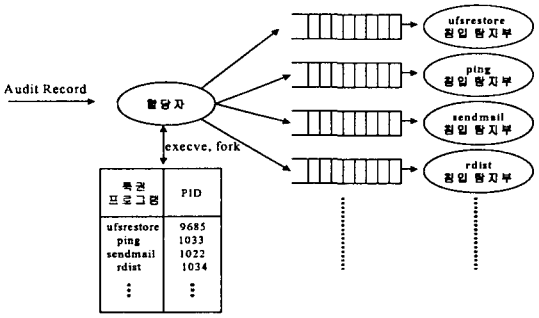

그림 3-6 단위 센서 침입탐지 엔진

Fig. 3-6 Unit Sensor Intrusion Detection Engine

프로그램 프로세스 매핑 테이블을 사용하는 이유는 BSM에서 제공하는 감사 레코드에는 현재 수행하는 프로그램에 대한 정보가 존재하지 않고, 프로그램을 수행하는 프로세스 번호만 존재하기 때문에 프로그램과 프로세스를 매핑시키는 수단이 필요하다.

이후, 할당자는 입력되는 감사 레코드 중 그림 3-6과 같이 프로그램 프로세스 매핑 테이블에 등록된 PID에 해당하는 감사 레코드의 시스템 호출 부분(이를테면, open, close, write 등을 의미)을 분리하여 침입탐지부에 전달하여 탐지를 수행한다. 침입탐지부는 전달된 시스템호출 순서를 패턴 크기 단위로 분리하여 패턴KB와 패턴 매칭을 하여 침입유무를 판단한다.

전술한 각각의 침입탐지부는 그림 3-7의 탐지자 루틴에 도시된 바와 같이 탐지자KB를 이용한 반대측 탐지(negative detect)와 정상패턴KB를 이용한 긍정적 탐지(positive detect)를 수행하며, 임계치에 의해 경보를 발생하는 경우 탐지자갱신자를 통해 self 서버와 통신하여 현재 탐지된 패턴이 침입패턴인 경우 분산된 모든 탐지자KB를 갱신시켜 침입으로부터 분산된 모든 침입탐지 시스템의 면역력을 증가시킨다.

이를 상세히 살펴보면, 외부로부터 전달된 침입패턴 정보 즉 탐지자를 이용하여 반대측 선택 방식(negative selection)에 따라 현재 프로세스가 발생하는 시스템 호출들을 감시하여 탐지자KB에 존재하는 시스템호출 패턴(즉, 탐지자)과 일치하는 시스템호출이 존재하면 이를 침입으로 간주하여 침입보고 및 대응부를 통해 시스템관리자에게 침입을 알리고 현재 침입에 사용된 프로세스를 종료시킨다. 이때, 만일 침입탐지부가 오판하여 긍정적 결함이 발생한 경우, 현재 수행중인 프로세스의 시스템 호출과 일치된 탐지자(들)를 분산된 모든 침입탐지시스템의 탐지자KB로부터 제거하고, 이를 self서버의 합성정상행위패턴KB에 추가한다.

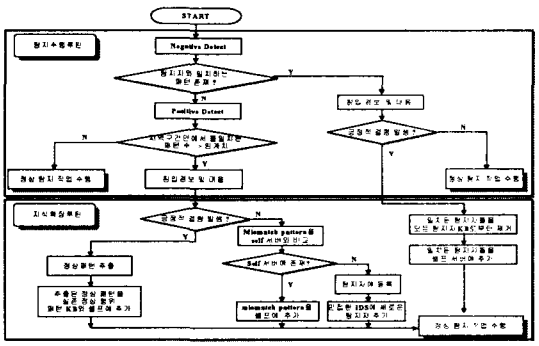

그림 3-7 제안한 단위 센서 침입탐지부의 동작 알고리즘

Fig. 3-7 Operation Algorithm of Proposed Unit Sensor Intrusion Detection Module

침입탐지엔진은 반대측 탐지를 통해 프로세스가 발생하는 시스템 호출을 감시한 후, 상기 프로세스에 의해 발생된 시스템 호출과 정상행위패턴KB의 내용과 비교하여 상기 프로세스에 의해 발생된 시스템 호출 중 정상행위패턴KB에 존재하지 않는 패턴(hamming distance가 1이상)들이 시스템 호출 프레임 크기(FS)안에서 임계치보다 많이 존재하면

이를 침입으로 간주하여 침입보고 및 대응부를 통해 시스템관리자에게 알리고, 현재 침입에 사용된 프로세스를 종료시킨다. 이때, HD 값과 FS 크기값 선정에 따라 탐지 시간과 탐지 정확도에 차이가 있으며 이에 대해서는 실험을 통해 후술한다. 임계치는 보안정책에 따라 조절할 수 있어, 임계치를 낮추면 보안강도가 높아지는 반면 긍정적 결함 발생 확률이 증가하며, 임계치를 높이면 부정적 결함 발생확률이 증가한다.

한편, 침입탐지엔진은 침입 경보가 발생하고, 발생된 경보가 긍정적 결함이 아닌 경우 self 서버와 통신하면서 그림 3-7의 지식확장루틴을 수행한다. 이를 상세히 살펴보면, 침입탐지엔진은 FS 안에 HD값이 일정크기인 패턴들을 self 서버에 전송하여 합성정상행위 KB에 일치하는 패턴(들)의 존재 유무를 판단하여 패턴(들)이 존재하는 경우(즉, hamming distance가 0인 경우) 상기 패턴(들)을 정상패턴 KB에 추가하여, 추후에 이와 동일한 패턴에 의해 긍정적 결함이 발생하는 것을 방지한다. 만일 합성정상행위 KB에 일치하는 패턴(들)이 존재하지 않는 경우, hamming distance가 일정치 이상인 패턴(들)을 분산된 모든 컴퓨터의 탐지자 KB에 추가하여 추후 이와 같은 시스템 호출 패턴을 발생하는 프로세스의 수행을 반대측 탐지 단계에서 빠르게 탐지할 수 있게 한다. 만일, 발생된 경보가 긍정적 결함인 경우 현재 수행 중인 프로세스의 로그 데이터를 분석하여 정상행위 패턴을 추출하여 추출된 정상행위 패턴을 실존정상행위KB와 셀프서버의 합성정상행위KB에 추가한다.

단위 센서 침입탐지부에서의 비정상 판단 방법

그림 3-7의 제안한 침입탐지 시스템의 동작 알고리즘 중 시스템 호출 프레임 크기(FS)안에서 불일치된 패턴 수와 임계치를 비교하는 것을 상세히 살펴보면 시스템 호출 프레임 크기(FS) 만큼 입력된 각각의 시스템 호출 패턴의 hamming distance가 C보다 큰 패턴들의 개수가 임계치를 넘는지를 판단한다. 즉, 다음의 경우 침입으로 판단한다.

$$\sum_{i=1}^{FS}\{HD_{\min}(i) \geq C\} > \text{임계치},$$

시스템 호출 프레임 크기(FS)를 사용하는 논리적 근거는 실험 결과 비정상 패턴이 전체 시스템 호출 중 일부분에 집중해서 발생하기 때문이며, 침입탐지부는 시스템 호출 순서의 비정상 패턴의 지역성을 고려하여 침입을 탐지하고 이때, 정상 행위 패턴과 미세하게 틀리는 시스템 호출은 고려하지 않는다.

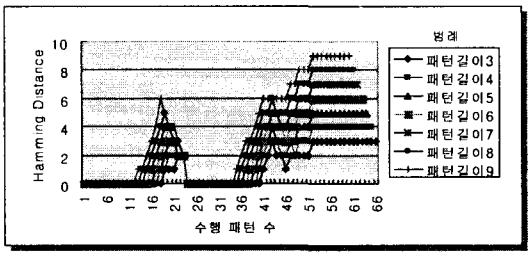

그림 3-8 rdist를 사용한 공격 분석

Fig. 3-8 Analysis of the Attack Using rdist

그림 3-8은 패턴 길이에 따라 rdist 프로그램을 이용하여 버퍼오버플로우 공격을 할 때 발생되는 시스템 호출을 분석한 것으로서 정상패턴과 불일치되는 부분이 지역적으로 편중되는 것을 알 수 있다. 이에, 제안한 시스템에서는 전술한 바와 같이 프레임 크기(FS)와 hamming distance를 이용하여 침입여부를 탐지한다.

Ⅳ. 프로토타입 구현

4.1 구현 환경

제안한 모델의 타당성을 입증하기 위해 전술한 모델에 대한 프로토타입을 단일 시스템에서 구현하였다. 프로토타입은 Solaris 2.6환경에서 C++ 언어를 사용하여 구현하였고, SunOS BSM의 서브감사 시스템을 사용하였으며, 탐지대상 프로그램으로는 다중 호스트 상에서 복사 파일들을 동일하게 유지하기 위한 유닉스 유틸리티인 rdist [21](Remote File Distribute Program)로 하였다. rdist프로그램을 탐지대상으로 사용한 이유는 setuid 프로그램이고 시스템 관리자와 일반사용자에 의해 많이 사용되고 있으며, rdist를 이용한 버퍼오버플로우공격 예[17]가 존재하기 때문이다.

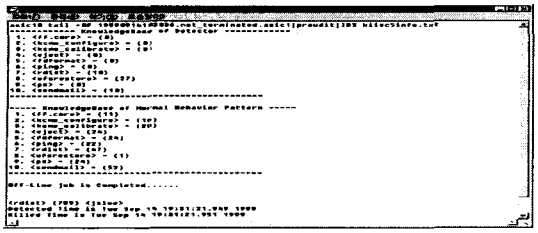

그림 4-1 프로토타입에서 rdist 프로그램을 이용한 공격 탐지 예

Fig. 4-1 Example of Detecting the Attack Using rdist Program at Prototype

그림 4-1를 통해 SunOS 5.6 Ultra-5_10에 설치된 제안된 모델에 대한 일부분의 프로토타입에 의해 rdist 프로그램을 이용하여 버퍼오버플로우 공격[17]이 발생한 경우 이를 탐지 및 대응하는 것을 살펴보면 다음과 같다. 도시된 바와 같이, 제안한 모델의 프로토타입은 탐지대상 프로그램들에 대한 각각의 탐지자KB와 실존정상행위KB를 구성한 후, 탐지를 수행하던 중 rdist 프로그램을 수행시키는 프로세스의 행위가 비정상인 경우, 이를 침입이라 판정하고 침입에 사용된 프로그램 이름과 이 프로그램을 기동시킨 프로세스 번호 및 프로세스 소유자 ID, 그리고 탐지 시간에 관한 정보를 출력한 후, 현재 침입에 사용된 프로세스를 강제로 종료시켜 침입에 대응한다. (그림 4-1)를 통해 프로토타입은 10개의 탐지대상 프로그램을 감시하고, self 서버로부터 rdist, ufsrestore, ps, sendmail에 대한 새로운 침입 탐지자패턴을 전달받았음을 알 수 있다.

4.2 성능 평가

제안한 시스템의 성능을 평가하기 위해 표 4-1의 조건으로 rdist 프로그램을 이용한 버퍼오버플로우 공격을 예로 침입탐지시간과 탐지정확도에 대하여 살펴본다.

성능평가에 사용되는 프로토타입은 rdist 프로그램을 포함한 67개의 setuid 프로그램의 행위를 감시한다. 표 4-1의 조건에서 동시수행프로그램은 표준 출력을 반복 수행하여 무한정으로 시스템 호출을 발생하는 프로그램으로 이를 통해 컴퓨터의 부하에 따라 제안한 침입탐지시스템의 탐지 능력을 평가하려고 한다. 또한, rdist 프로그램을 무한정 기동시켜 제안한 모델에서 탐지 대상 프로그램이 무한정 발생할 때 제안한 알고리즘의 처리 부하에 따른 탐지 능력을 평가한다.

표 4-1 탐지대상 시스템 조건

Table 4-1 Condition of System Monitored

조건 / 경우	패턴길이	동시수행 프로그램 수	rdist 무한정 수행
a	9	5	×
b	5	5	×
c	9	10	×
d	5	10	×
e	9	0	○

4.2.1 탐지시간 관점

그림 4-2 (a)~(d)를 통해 표 4-1의 a~d 경우에 따른 탐지시간을 살펴보면, 시스템 호출 프레임 크기(FS)가 20인 경우 FS가 10인 경우와 탐지자를 사용하는 경우보다 탐지 판단시간이 많이 소비됨을 알 수 있으며, 그림 4-2 (a)와 그림 4-2 (b)를 통해서 컴퓨터의 부하가 적게 걸려 있을 경우에는 패턴 길이가 침입을 탐지하는데 큰 영향을 미치지 않음을 알 수 있고, 동시수행프로그램 수에 따라 탐지시간도 그림 4-2 (a),(c)와 그림 4-2 (b),(d)를 통해서 큰 영향이 없음을 알 수 있으며, 패턴 길이가 길고 동시 수행 프로그램 수가 많을 경우 그림

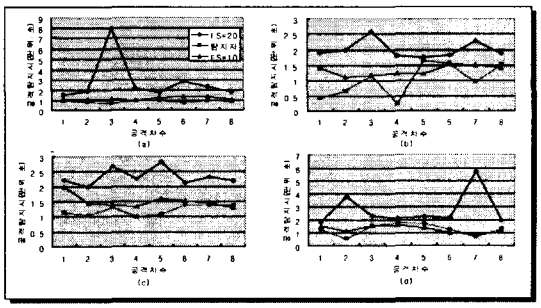

그림 4-2 4가지 경우에 따른 침입탐지시간 측정 결과

Fig. 4-2 Measure Result of Intrusion Detection Time According to Four Cases

4-2 (c)를 통해서 알 수 있듯이 탐지자를 이용하여 빠르게 침입을 탐지할 수 있다.

그림 4-3을 통해 표 4-1의 e 경우처럼 rdist 프로그램을 무한정 기동시키는 한 개의 프로그램을 수행하는 동안 rdist 버퍼오버플로우 공격을 수행할 때 탐지시간을 살펴보면, 이 경우 탐지자를 사용하는 것이 FS가 10인 경우보다 탐지시간이 빠른 것을 알 수 있다.

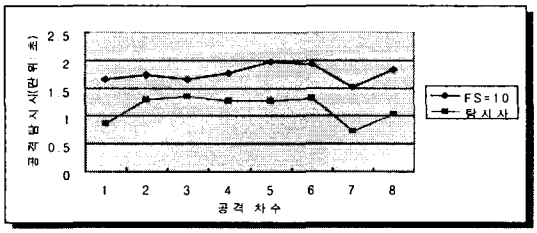

그림 4-3 (표 4-1)의 e 경우에 따른 침입탐지시간 측정 결과

Fig. 4-3 Measure Result of Intrusion Detection Time According to Case e of (Table 4-1)

결과적으로, 전체적인 탐지시간은 시스템 환경과 운영체제의 메모리 관리 방법 등 여러 가지 요인에 의해 공격 시도 차수에 따라 다르므로 큰 신뢰성을 갖기 힘드나, 자주 사용하는 탐지자 프로그램을 탐지할 경우와 패턴 길이가 길고 동시 수행 프로그램 수가 많을 경우 제안한 모델이 탐지 시간 측면에서는 빠름을 알 수 있다.

4.2.2 탐지정확도 관점

침입탐지시스템의 성능에 있어 중요한 요소 중 하나인 탐지 정확도를 측정하기 위해 rdist 프로그램에 의해 생성된 정상패턴수가 59개이고, 패턴 길이가 9인 경우에 정상적으로 각각의 rdist 명령을 수행할 때 긍정적 결함(false positive) 발생 유무를 실험해 보았다. 표 4-2를 통해 실험결과를 살펴보면, FS가 짧을수록 오판할 확률이 높으며, 제안한 시스템에서 FS를 30으로 하고 탐지자를 사용할 경우 긍정적 결함(false positive)이 발생하지 않음을 알 수 있다.

한편, 탐지자를 사용하지 않고 패턴 길이를 9로 하고, FS를 30으로 하고 hamming distance 값이 8 이상인 패턴의 개수를 20으로 하여 침입판단 기준을 높인 경우 긍정적 결함(false positive)은 발생하지 않으나 침입 발생 시 탐지를 못하는 부정적 결함(false negative)이 발생한다. 이때, 만일 상기 침입에 대한 시스템 호출 패턴을 다른 침입탐지시스템으로부터 전달받아 탐지자KB를 구축하여 표 4-2와 같이 침입 탐지자를 사용할 경우 상기 침입을 정확하게 탐지할 수 있다. 즉, 다른 침입탐지시스템으로부터 침입 패턴 정보를 수신하여 침입에 대한 면역력을 향상시켜 이와 동일한 침입을 받으면 이를 정확하게 탐지할 수 있다.

표 4-2 rdist 명령 옵션에 따라 false positive 발생 유무 점검

Table 4-2 Existence Check of False Positive According to rdist Instruction Option

탐지방법 / rdist 명령종류	FS = 10	FS = 20	탐지자 사용 FS = 30
rdist -b	○	○	×
rdist -D	×	×	×
rdist -h	○	×	×
rdist -i	○	×	×
rdist -n	×	×	×
rdist -q	○	×	×
rdist -R	○	×	×
rdist -v	○	×	×
rdist -w	○	×	×
rdist -y	○	×	×
rdist -f	○	×	×
rdist -m	×	×	×

※ ○ : false positive 발생, × : false positive 발생 무

4.2.3 다양성

제안한 모델에서 분산된 각각의 침입탐지시스템들은 동일한 특권 프로그램에 대해 각기 다른 침입판단 기준을 갖고 있으므로 어떤 침입탐지시스템은 침입을 탐지하지 못하였지만, 다른 침입탐지시스템은 동일한 침입공격에 대해 침입을 탐지할 수

있다. 예를 들어, 어떤 하나의 IDS에서 긍정적 결함(false positive)을 줄이기 위해 침입판단 기준을 높인 경우 침입을 탐지하지 못할 수 있으나, 반면 다른 IDS에서 엄격한 침입탐지를 위해 긍정적 결함(false positive)을 감수하면서 침입판단 기준을 낮춘 경우 침입을 탐지할 수 있다. 따라서, 제안한 모델은 A 침입탐지시스템이 설치된 컴퓨터에 침입이 성공한 경우, 동일한 방법을 적용하여 B 컴퓨터를 침입할 수 있다고 볼 수 없는 면역시스템의 특징인 다양성 성질을 갖고 있다.

이와 같이, 탐지시간 및 정확도 관점에 따라 제안한 침입탐지시스템의 성능을 평가해본 결과, 제안한 시스템을 사용할 경우 인접한 침입탐지시스템간에 탐지자 정보를 공유하므로 각 컴퓨터의 침입에 대해 빠르고 정확하게 침입을 탐지할 수 있으므로 전체 컴퓨터 시스템들의 면역력을 향상시킴을 알 수 있다. 또한, 제안한 모델은 면역시스템의 특징인 다양성을 제공한다.

V. 결 론

본 논문에서는 여러 동질형 컴퓨터에 단위 센서 침입탐지시스템을 설치하고, 분산된 단위 센서 침입탐지시스템들 중 어느 하나가 프로세스에 의해 발생된 시스템 호출 순서 중 비정상적인 시스템 호출을 탐지한 경우 이를 다른 센서 침입탐지시스템들과 서로 동적으로 공유하여 새로운 침입에 대하여 효율적으로 탐지하는 침입탐지시스템을 제안하고, 이를 설계하고 프로토타입을 구현하여 그 타당성을 보였다.

본 논문에서는 제안한 침입탐지시스템의 타당성을 입증하기 위해 모델에 대한 프로토타입을 단일 시스템에서 구현하였고, 이를 통해 탐지시간 관점과, 탐지정확도 관점, 그리고 면역 시스템의 특징인 다양성에 관한 관점에서 제안한 모델의 성능을 평가하였다. 성능 평가 결과, 제안한 시스템은 인접한 침입탐지시스템간에 탐지자 정보를 공유하므로 각 컴퓨터의 침입에 대해 빠르고 정확하게 침입을 탐지하여, 전체 컴퓨터 시스템들의 면역력을 향상시킴을 알 수 있으며, 또한 제안한 모델은 면역시스템의 특징인 다양성을 제공함을 알 수 있었다.

따라서 제안한 침입탐지시스템은 수행하는 동안 인접 단위 센서 침입탐지시스템이 공격을 받으면 받을수록 전체 침입탐지 시스템의 면역력이 향상하여 새로운 침입을 효과적으로 방지할 수 있으므로 동질형 노드들로 구성된 정보시스템에 대한 침입을 효과적으로 탐지하고 이에 대한 빠른 대응을 가능하게 한다.

향후 연구과제는 현재 동질형 호스트간에 제안한 모델을 적용하였으나 이기종간의 감사 데이터 표준화를 통해 제안한 모델을 이기종 환경에 확장시키는 연구가 필요하다.

참고문헌

[1] James Cannady, Jay Harrell, "A comparative analysis of current intrusion detection technologies," http://iw.gtri.gatech.edu/Papers/ids_rev.html, Feb. 1998.

[2] Mansour Esmaili, Rei Safavi-Naini, "Case-based reasoning for intrusion detection," *Computer Security Applications Conference* pp.214 ~222 1996.

[3] Jai Sundar B. Spafford E, "Software agents for intrusion detection," Technical Report, Purdue University, Department of Computer Science, 1997.

[4] 이종성, 채수환, "분산 침입 탐지 에이전트를 기반으로 한 지능형 침입탐지시스템 설계," *한국정보처리학회 논문지*, 제6권 제5호, 1999.5

[5] 은유진, 박정호, "침입탐지 기술 분류 및 기술적 구성요소," *정보보호센터 정보보호 뉴스* 1998.7 통권 13호.

[6] Crosbie M, Spafford E, "Applying genetic programming to intrusion detection," Technical Report, Purdue University, Department of Computer Science, 1996.

[7] Paul Helman and Gunar Liepins, "Statistical foundations of audit trail analysis for the detection of computer misuse," *IEEE Transactions on Software Engineering*, 19(9):886-901, Sep. 1993.

[8] H.S. Vaccaro and G.E. Liepins, "Detection of anomalous computer session activity," *In Proceedings of the 1989 IEEE Symposium on Research in Security and Privacy*, pages 280-289, 1989.

[9] Cheri Dowell and Paul Ramstedt, "The computer watch data reduction tool," *In Proceedings of the 13th National Computer Security Conference*, pages 99-108, Washington, DC, Oct.1990.

[10] Paul Spirakis et al, "SECURENET:A network-oriented intelligent intrusion prevention and detection system," *Network Security Journal*, 1(1), Nov.1994.

[11] S. A. Hofmeyr, A. Somayaji, and S. Forrest. "Lightweight intrusion detection for networked operating systems," *Journal of Computer Security*, Vol. 6 pp. 151-180, 1998.

[12] A. Somayaji, S. Hofmeyr, and S. Forrest, "Principles of a computer immune system," *New Security Paradigms Workshop*, Sep.1997

[13] Calvin Cheuk Wang Ko, *Execution Monitoring of security-critical programs in a distributed system : A specification-based approach.* PhD thesis, Department of Computer Science, University of California DAVIS, 1996.

[14] 정진욱, 안성진, *UNIX 프로그램밍 기술 -SVR4 시스템 프로그래밍의 이론과 실제-*, 컴퓨터출판, 1996.

[15] Sun Security Bulletin #00169, 1998/4/28 http://www.certcc.or.kr/advisory/ka98/ka98-65.txt

[16] http://www.rootshell.com/archive-j457nxiqi3gq59dv/199707/psrace.c.html

[17] http://161.53.42.3/~crv/security/bugs/SunOS/rdist6.html

[18] http://www.rootshell.com/archive-j457nxiqi3gq59dv/199807/solaris-sendmail-8.8.4.sh.html

[19] SunSoft, Mountain View, Californina, *SunSHIELD Basic Security Module Guide*, 1995

[20] Kosoresow AP, S. Hofmeyr, "Intrusion detection via system call traces," *IEEE Software*, V.14 N.5, pp.35-pp42, Sep.1997

[21] Sun Microsystem, *Man Pages: Rdist-remote file distribution program*, Nov.1993.

이 종 성(Jong Sung Lee)

1994.2 한국항공대학교 전자계산학과 졸업 (이학사)

1996.2 한국항공대학교 전자계산학과 대학원 졸업 (이학석사)

2000.2 한국항공대학교 컴퓨터공학과 대학원 졸업 (공학박사)

1995.12 - 1999.4 한가람 국제특허 법률사무소 정보통신기술팀(대리)

1996.9 - 1998.8 한국항공대학교 컴퓨터공학과 시간강사

1998.3 - 2000.2 (국립) 순천대학교 정보통신공학과 시간강사

1999.3 - 1999.11 현대전자연수원 강사

1999.12 - 2000.4 현재한국정보보호센터 개발부 선임연구원

2000.5 - 현재 (주) SAFA 미디어 연구실장

조 성 언(Sung-Eon Cho)

1989. 2 한국항공대학교 항공통신정보공학 졸업 (공학사)

1991. 2 한국항공대학교 대학원 항공통신정보공학 졸업 (공학석사)

1997. 2 한국항공대학교 대학원 항공전자공학과 졸업 (공학박사)

1991. 3 ~ 1992. 2 한국항공대학교 항공통신정보공학과 조교

1997. 3 ~ 1999. 3 (국립) 순천대학교 공과대학 정보통신공학과 전임강사

1999. 4 ~ 현 재 (국립) 순천대학교 공과대학 정보통신공학과 조교수

*관심분야 : 전파공학, 전자파환경공학, Wireless

communication

조 경 룡(Kyung-Ryong Cho)
1987. 2 숭실대학교 전자공학과 졸업 (공학사)
1989. 2 숭실대학교 대학원 전자공학과 졸업 (공학석사)
1995. 2 숭실대학교 대학원 전자공학과 졸업 (공학박사)
1990.12 ~ 1996. 2 한국이동통신(주) 중앙연구원 선임연구원
1996. 3 ~ 현 재 (국립) 순천대학교 공과대학 정보통신공학과 조교수
*관심분야 : 이동통신, 채널코딩, 통신방식