패킷캡쳐를 이용한 침입탐지 시스템의 구현

김영진* · 한승조**

Implementation of Intrusion Detection System Using Packet Capture

Young-jin Kim* · Seung-jo Han**

이 논문은 1999년도 조선대학교 학술연구비를 지원받았음

요 약

컴퓨터 확산 및 네트워크 이용의 급격한 증가에 따른 부작용으로 컴퓨터 보안 문제가 중요하게 대두되고 있다. 공격자들의 공격은 운영체제, 프로토콜, 응용프로그램에서 취약점을 이용하고 있으며 그 기술이 고도화, 전문화 되어가고 있다. 그러므로 정보통신망의 기반구조를 구성하는 구성요소들에 대한 구조, 관리에서의 문제점을 해결하기 위한 기반구조 보호기술이 필요하다. 본 논문에서는 효과적으로 침입자를 차단하여 중요 시스템에서 분리시키기 위한 침입탐지시스템을 개발하고, IDS 모델을 설계 및 구현한다.

ABSTRACT

Computer security is considered important due to the side effect generated from the expansion of computer network and rapid increase of use of computers. A attack of intruders using a vulnerability of operating system, protocol and application programs. And so, The attack methods is to be high technology and professional. Thus It must be necessity that we necessary a solution to structure, management for framework of information technology. This paper develope intrusion detecting system for separating intruders form critical system and design IDS model and implementation of it.

키워드

침입탐지, 침입자, 기반구조, 보호기술

Ⅰ. 서 론

최근 컴퓨터 통신망 및 인터넷의 급속한 확산 및 일반화에 따라 각종 온라인업무, 전자 상거래, 사이버 강의는 물론 화상회의 시스템에 이르기까지 다양한 분야에 응용을 가능하게 만들어 우리 사회를 편리하게 만들고 있다. 그러나 이와 더불어 정보 누설, 정보시스템의 비인가 사용, 전산망의 해킹 등의 역기능이 증대되어 사회, 경제적으로 심각한 문제를 초래하게 되었으며, 컴퓨터 보안 문제가 중요하게 대두되고 있다. 이러한 컴퓨터 보안 문제에 대처하기 위해 정보보호를 요하는 시스템에 대한 불법 침입을 분석하고 탐지하는 감사 기술의 형태인 침입탐지 시스템(Intrusion Detection System : IDS)에 관한 연구가 활발히 진행되고 있다.[1,2,3]

침입이란 컴퓨터가 사용하는 자원의 무결성(Integrity),

*조선대학교

접수일자 : 2002. 10. 21

비밀성(Confidentiality), 이용성(Availability)을 저해하는 일련의 불법적인 행위와 시스템의 보안정책을 파괴하는 행위를 말한다. 침입탐지 시스템은 침입을 목적으로 사용자의 시스템에 불법으로 권한을 획득하거나 서비스 거부 공격과 같은 행위를 하는 것을 탐지하여 관리자에게 보고하는 시스템을 일컫는다. 일반적인 침입탐지시스템의 중요 요구사항은 시스템 관리자 없이도 지속적인 관리가 수행되어야 하며, 컴퓨터 시스템에 트래픽에 의해 발생되는 오버헤드를 최소화해야 한다. 또 새로운 침입 유형의 변화에 대한 자체 학습 기능과, 어떤 침입탐지모듈에 결함이 발생되어도 전체 침입탐지시스템에 큰 영향을 주지 않는 결함 허용 관리 기능, 그리고 시스템에 정상상태를 침입이라고 탐지하는 긍정적 결함(false positive) 및 시스템의 침입상태를 정상상태로 판단하는 부정적 결함(false negative)과 같은 잘못된 침입 탐지를 방지해야 하고 실시간에 침입을 탐지해야 한다.[4,5,6,7] 본 논문에서는 이러한 문제를 해결하기 위한 방법으로 침입자의 공격으로부터 시스템과 네트워크 구조를 보호하기 위한 방법으로 네트워크와 호스트를 보호할 수 있는 제어기능과 시스템 내부의 피 공격자 시스템에 대한 가용성을 보장하는 보안모델을 제안하고자 구현하고자 한다.

본 논문의 구성은 Ⅱ장에서 침입탐지시스템에 대해서 분류하고 그 문제점 및 특징을 살펴보고 Ⅲ장에서는 Libpcap를 이용한 침입탐지 시스템을 설계하고 개선된 보안모델의 개념과 구성요소의 기능에 대하여 설명한다. Ⅳ장에서는 제안된 침입탐지시스템을 구현한다. 마지막으로 본 논문의 결론 및 향후 연구 방향에 대하여 기술한다.

Ⅱ. 본 론

침입탐지 시스템은 외부의 침입 시도로부터 내부의 특정 호스트나 네트워크를 보호하기 위한 보안기술로 접근정책을 통한 내외부간 트래픽을 제어하는 기술이다. 그 기본 목표는 보호 대상이 되는 네트워크에 불법 사용자들의 접근해 컴퓨터 자원들을 사용하는 것을 방지하고, 자신의 정보들이 불법적으로 외부에 유출되는 것을 방지하는 것이다. 침입탐지시스템은 1987년 Denning에 의해서 최초로 Model이 제안되었다. 이 보안기술은 정보접근, 조작, 시스템무력화 등의 특정 시스템에 불법적으로 접속하여 시스템을 사용, 오용, 남용하는 침입자들을 감지하고 문제점에 대응하는 기술을 말한다.[11,12,13]

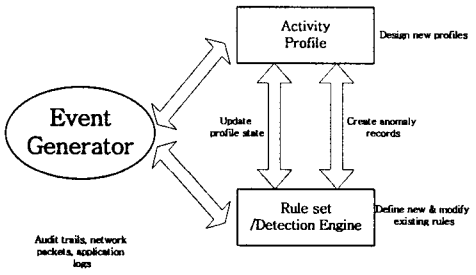


그림 1. 침입탐지시스템의 모델
fig 1. model of Intrusion Detection System

침입차단시스템은 전반적인 호스트 보안을 증진시키는데 많은 이점을 제공한다. 침입차단시스템은 네트워크 보안을 증가시키고 취약성을 내재하는 서비스를 보호하기 위한 필터링 기능을 제공함으로써 서브넷 상에 있는 호스트의 위험을 감소시키며, 호스트 시스템으로의 접근을 제한할 수 있다. 또한 인터넷 안팎으로 모든 엑세스가 침입차단시스템을 통과한다면 침입차단시스템은 엑세스를 기록할 수 있고 네트워크 사용에 관한 유용한 통계자료를 제공할 수 있다. 의심스러운 활동이 있을 때 통지기능을 가진 침입차단시스템과 네트워크가 침입 시도를 받고 있는지 또는 침입되었는지에 대한 자세한 정보를 제공해 줄 수 있다. 침입탐지시스템의 일반적인 구조는 그림 1과 같이 Event generator, Activity Profile, Rule set/Detection engine로 이루어져 있다. Event generator는 호스트나 네트워크로부터 발생된 데이터를 수집하며, 감사자료의 형식이 되는 Rule set과 비교되어 탐지되거나 행위 Profile의 형식에서 비정상 행위 레코드를 생성시켜 탐지된다. Activity profile은 패턴 생성기를 말하는데 패턴 생성기는 정상패턴이나 비정상패턴을 생성하며 생성된 패턴은 Rule set에 의해서 Detection engine이 탐지해낼 수 있다. 탐지된 결과는 해당관리자에게 보고된다. 침입탐지시스템의 중요 요구사항은 새로운 침입 유형의

변화에 대한 자체학습기능, 결함허용, 시스템 환경 변경시 유지 및 관리에 있다. 침입탐지시스템 연구들은 대규모의 구조를 지닌 네트워크에서의 정보수집과 분석이 각각의 전담 시스템에서 수행되는 경우가 많고, 이에 따른 트래픽 집중이 문제시된다. 또한 접근통제가 용이하지 않기 때문에 대규모 네트워크 상에서의 분산적이고 협력적인 효율적인 구조가 필요하다.

III. 침입탐지 시스템의 분류

침입탐지시스템은 탐지 후 행위, 감사대상, 탐지 빈도 수 그리고 탐지방법 등에 의해 분류할 수 있다. 탐지 후 행위에 따른 분류 방법은 소극적인 방법과 적극적인 방법이 존재한다. 소극적인 방법은 탐지에 대한 보고만을 행하는 것을 뜻하고, 적극적인 방법은 탐지된 항목에 대해 침입을 방지하기 위한 사후 처리를 능동적으로 행하는 것을 의미한다. 감사 대상에 따른 분류는 호스트의 로그 파일을 이용한 탐지와 네트워크의 패킷을 기반으로 한 탐지로 나누어진다. 전자를 이용한 시스템을 호스트 기반(Host based)의 침입 탐지 시스템이라 하고 후자를 이용한 시스템은 네트워크 기반(Network based)의 침입 탐지 시스템이라 한다. 호스트 기반 침입 탐지 시스템은 시스템 내부에 설치되어 하나의 시스템 내부 사용자들의 활동을 감시하고 해킹 시도를 탐지해내는 시스템으로써 해킹 시도에 대한 차단기능이 가능하지만, 서비스되고 있는 자체 서버에 Agent를 올려야 하기 때문에, 일부 성능이 낮은 시스템은 패킷 분석에 과부하가 걸려 서버가 다운되는 경우가 발생된다.[14,15]

네트워크 기반의 침입탐지 시스템은 네트워크의 패킷 캡쳐에 기반하여 네트워크를 지나다니는 패킷을 분석해서 침입을 탐지해낸다. 시스템 기반 침입탐지 시스템은 모니터링하려는 시스템마다 하나씩 설치가 되어야 하지만, 네트워크 기반 침입탐지 시스템은 네트워크 단위에 하나만 설치하면 된다. 네트워크 기반 침입탐지시스템은 네트워크 영역 전체를 탐지 대상으로 하며, 대부분 서비스를 제공하는 서버들이

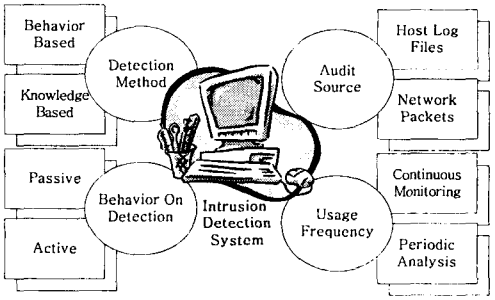


그림 2. 침입탐지시스템의 분류
fig 2. classification of Intrusion Detection System

상위에 설치된 스위치의 Uplink 포트에 Mirroring을 적용하여 미러링된 포트가 침입탐지시스템의 모니터링 포트가 되도록 하여 서버들과 통신하는 모든 패킷을 분석한다. 미러링 포트에 침입탐지시스템을 연결함으로써 장애가 발생하더라도, IDS 자체만 네트워크에서 고립될 뿐 상위 스위치에 미치는 영향은 전혀 없다. 탐지 빈도수에 따라 분류는 지속적인 탐지 방법과 구간 분석에 의한 탐지 방법으로 나눌 수 있다. 지속적인 탐지 방법은 보통 네트워크 기반의 침입 탐지 시스템에서 사용을 하고 구간 분석에 의한 탐지 방법은 호스트 기반의 침입 탐지 시스템에서 활용을 한다.

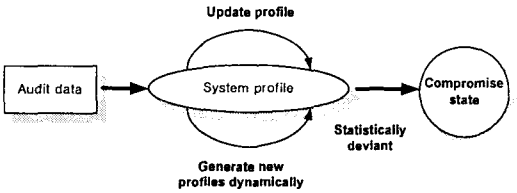


그림 3. 비정상 행위 탐지 시스템의 모델
fig 3. Model of Anomaly Detection System

탐지 방법에 따른 분류는 행위에 기초한 탐지 방법과 지식 기반의 탐지 방법으로 나누어진다. 행위에 기초한 탐지 방법은 실제 탐지한 행위를 가지고 침입 유무를 가리는 것이다. 탐지 행위가 기존에 존재하는 정상적인 패턴과 비교하여 어긋날 경우 침입이라 규정하는 탐지 기법이다. 이러한 방식을 비정상 행위 탐지기법이라고도 한다. 지식 기반의 탐지 방법은 기존에 알려진 침입에 대한 유형과 비교하여 탐

지를 행하는 것이다. 지식 기반 탐지 기법을 오용탐지 기법이라고도 한다.

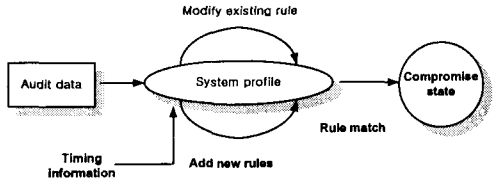


그림 4. 오용 탐지 시스템의 모델
fig 4. Model of Misuse Detection System

네트워크 기반 침입탐지시스템(NIDS)의 경우 침입을 탐지하기 위해서 Raw 데이터를 구성하는 패킷 정보에 의존한다. 패킷으로부터 얻을 수 있는 정보는 헤더의 각 필드의 값을 통해서 트래픽 또는 연결정보를 얻을 수 있다. 또한 헤더 이외에 데이터 Payload의 내용을 분석하는 Stateful 방식의 분석도 고려할 수 있다. 일반적인 침입탐지시스템은 수집된 Raw 데이터를 여러 가지 방식을 적용하여 정보를 만든 후 탐지에 적합한 형태의 패턴으로 만들어지는 정보를 기반으로 탐지를 수행하게된다. 그러나 NIDS 구현 원리 자체가 인터페이스 카드의 Promiscuous Mode를 이용해 같은 네트워크상의 모든 패킷을 볼 수 있음을 전제로 하기 때문에 암호화된 통신이나, 스위칭 허브를 통해 스위칭 된 환경 즉, 대역폭이 제한되어 있는 환경에서는 적용이 힘들며 속이기 쉽다는 단점이 있다. 또한 호스트기반 침입탐지시스템(Host based IDS)은 시스템 자체에서 감사자료를 기록할 수 있으므로 호스트에서 일어나는 모든 상황이 모니터링 가능하게 되고, 암호화된 통신이나 스위치 된 환경 등에서도 사용이 가능하다는 장점을 가진다. 그러나 가장 큰 문제는 시스템 자체가 해커에 의해 장악되었을 경우 탐지기능을 전혀 하지 못할 수 있으며, 시스템 자체에 종속되므로 그 개발이 OS 마다 달라져야 한다는 단점이 있다. 침입탐지에 요구되는 기술의 조사를 위해 기존에 개발된 네트워크 기반 침입탐지 시스템을 분석하여 전체적인 모델을 설계하면 그림5와 같다. 외부로부터 들어오는 패킷을 감사자료로 수집하여(Audit data)로 수집하여 필요한 정보만을 감사자료 축약과정을 통해 추출한다. 추출된 감사자료는 침입탐지 모델에 따른 침입판정 기술을 이용하여 침입행위를 분석하게되며, 그 결과를 Arbitrator에게 준다. 여기서는 다양한 침입 판정 기술이 적용될 수 있으며, 1개 이상의 기술이 적용 가능하다. Arbitrator는 침입판정에 대한 최종 결정을 내리고, 정해진 대응 정책을 수행토록 한다. 사용자 인터페이스는 이러한 정보들을 조합하여 관리자에게 해당 정보를 알려준다. 네트워크 기반 침입탐지 모델의 전체적인 설계를 통하여 침입탐지에 요구되는 기술을 모듈별로 구성할 수 있다. 감사자료를 수집과 축약하는 모듈, 축약된 감사자료를 통하여 침입을 탐지하는 모듈, 탐지된 결과를 통하여 관리자에게 결과 보고 및 대응하는 모듈로 나눌 수 있다.

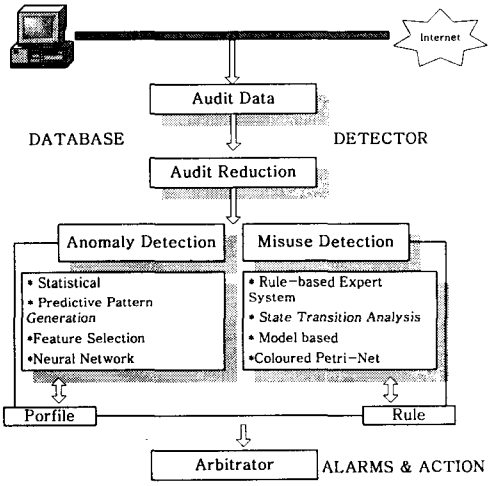


그림 5. 네트워크기반의 침입탐지시스템 블록도
fig 5. Block diagram of Network-based Intrusion Detection System

IV. 침입탐지 시스템의 설계

4.1 패킷캡처의 개요

패킷캡처란 네트워크상의 흘러 다니는 패킷들을 모니터링하는 것을 말한다. 일반적인 이더넷 환경에서 라우터는 내부 네트워크로 향하는 패킷들을 브로드캐스팅(Broadcasting)하게 되고 각 시스템은 자신의 주소가 목적지인 패킷만을 받아들여 운영체제(Operation System)가 처리하게 된다. 이때 이더넷

인터페이스의 수신 Mode가 Promiscuous로 설정이 되면 목적지가 어디든 상관없이 같은 세그먼트에 물려있는 네트워크상의 모든 패킷을 수신할 수 있게된다. 이러한 패킷 캡처는 네트워크의 사용에 대한 통제나 보안을 목적으로 하는 모니터링, 네트워크를 디버깅하기 위한 목적, 스니퍼링 등 다양한 형태로 응용이 가능하다. Libpcap는 커널 수준이 아닌 사용자 레벨에서 네트워크상의 패킷을 캡쳐해 주는 유용한 라이브러리로 다양한 네트워크 툴 개발에 응용되고 있다. 초창기의 Libpcap은 BSD 계열의 OS에서 동작하도록 설계되었다.

표 1. Libpcap를 이용한 애플리케이션
table 1. Application using of Libpcap Library

Application	comment
Ethereal	네트워크 프로토콜 분석툴 현재는 윈도우용으로 포팅
Ngrep	네트우크 패킷용 grep
Nmap	리모트 호스트 스캐닝/탐지 툴
Ntop	실시간 네트워크 트래픽 분석 툴
Snort	패킷 스티퍼/logger 및 간단한 NIDS 기능 수행
Tcpdump	Basic command line interface to libpcap functionality

이러한 BSD 계열의 OS들은 BPF(Berkeley Packet Filter)라고 알려진 커널 레벨의 인터페이스를 갖고 있다. BPF는 커널영역에 패킷 필터를 컴파일 시킬 수 있다. 이는 커널 영역과 사용자 영역을 오가는 횟수를 많이 줄였기 때문에 패킷캡처 애플리케이션 개발에서 많은 이점을 제공하였으며, 또한 CPU의 부하를 감소시켰다. 그러나 Raw socket이 모든 프로토콜을 다루지 못하며, 사용되는 OS에 따라 많은 영향을 받았다. 이러한 이유로 Libpcap가 등장하게 되었다. 현재 Libpcap는 많은 개발자들에게 유용하게 사용되고 있으며, 현재 인터넷에서 널리 사용되는 여러 가지 유용한 애플리케이션에서 사용되고 있다.

4.2 Libpcap의 구조 및 패킷 필터링

패킷 필터란 지나가는 패킷의 헤더를 보고 패킷의 네트워크로의 유입여부를 결정짓는 소프트웨어이다. 네트워크를 통하는 모든 데이터들은 패킷의 형태를 띠고, 패킷이 어디로 향하고 있는지 어디서 왔는지, 어떤 종류의 패킷인지 그리고 그 밖의 관리 상 필요한 세부 사항이 적혀 있다. 이 부분을 패킷의 헤더(header)라 부르고 나머지 부분에는 전송하고자 하는 실제 자료가 들어있으며 몸체(body)라 부른다. Libpcap를 이용한 패킷 추출 과정은 그림 6과 같으며, 파일을 다룰 때 사용되는 File Descriptor와 유사한 개념의 Packet Capture Descriptor를 사용한다. Packet Capture Descriptor는 다음과 같은 구조를 가진다.

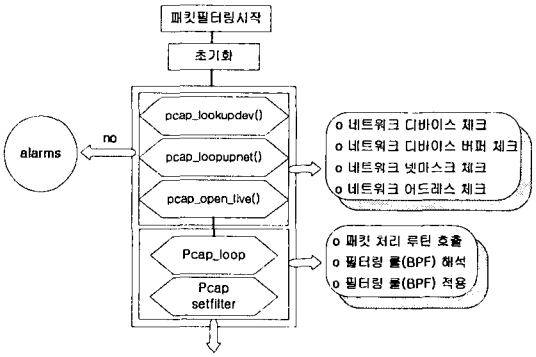


그림 6. Libpcap를 이용한 패킷 추출 과정
fig 6. Packet capture using Libpcap Library

o pcap_lookupdev() ; 패킷을 캡쳐하기 위한 디바이스를 검색해 앨리어스를 반환한다. 즉 device는 "eth0", "eth1"을 가르키는 스트링형 포인터가 되고, 시스템마다 인터페이스를 칭하는 앨리어스가 다르므로 어떤 운영체제를 쓰느냐에 따라 다른 값을 가지게 된다.

```
device = pcap_lookupdev(errbuf)
```

o pcap_loopupnet() ; 검색된 해당 디바이스의 네트워크, 마스크 주소 정보를 가져온다.

```
pcap_lookupnet
(device, &NETWORK, &MASK, errbuf)
```

o pcap_open_live() ; 파일 기술자와 유사한 역할을 하는 패킷 기술자를 반환한다. 파일 기술자는 인터페이

스 카드가 두 개 이상일 경우 각각을 구별해 주는 역할을 하며, 두 번째 인수는 캡쳐할 사이즈를 나타내고 세 번째 인수는 네트워크 상태를 변경해주는데 이때 모든 패킷을 잡기 위해서는 PROMISCUOUS모드로 전환해줄 필요가 있다.

```
packet_descriptor = pcap_open_live(device, 1500,
        PROMISC, 200, errbuf)
```

○ pcap_loop() ; 실질적으로 패킷을 읽어 들여오는 함수로 패킷 처리 루틴을 호출한다. 두 번째 인수는 캡쳐할 패킷의 개수를 나타내고, 패킷 처리 루틴 packet_process는 세 번째 인수로 함수 포인터 pcap_handler에 의해 호출된다.

```
pcap_loop(packet_descriptor,100, packet_process, 0)
```

○ pcap_handler()

패킷 처리 함수의 포인터로 사용자가 개발 목적에 적합한 형태로 코딩할 수 있다.

```
void packet_process()
//단순히 캡쳐한 패킷을 헥스 코드로 출력한다.'
for(length = 0; length < packet_header->len;
length++);
printf("%02x", *(packet++));
}
```

○ pcap_compile()

스트링 형태의 필터링 룰(BPF 방식)을 해석한 후 구조체 bpf_program에 넣는다.

```
pcap_compile(packet_descriptor,&BPF_CODE,
FILTER_RULE, 0, mask)
```

○ pcap_setfilter()

pcap_compile에 의해 해석된 필터링 룰을 적용한다.

```
pcap_setfilter(packet_descriptor, &BPF_CODE)
```

○ pcap_close()

열린 패킷 디스크립터를 닫는다.

```
pcap_close(packet_descriptor)
```

IV. 침입 탐지 시스템의 구현

본 논문에서 구현한 침입탐지시스템은 그림 7과 같이 Manager, Database, Agent, Sensor의 4부분으로 구성되어 있다.

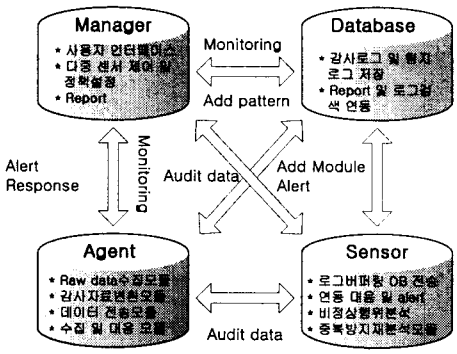


그림 7. 제안한 침입탐지시스템의 블록 구성도
fig 7. Block diagram of proposed Intrusion Detection System

5.1 침입탐지시스템의 동작

구현한 침입탐지시스템은 다음과 같은 순서로 동작한다.

○ 필터링 파일로부터 필터링 룰을 불러들인다.

특정 서버만을 보호하기 위한 침입탐지 시스템이라면 모든 패킷을 감사할 필요가 없으므로 필터링을 통해 해당 프로토콜을 가진 패킷만 신속히 처리한다.

```
void packet_cap::Open()
{
        fopen("FILTER_RULE", "r");
        fgets(FILTER_RULE,
STDBUF, FP);
}
```

○. 룰 파일을 읽어들여 메모리에 로드시킨다.

룰 구조체에 룰을 파싱하여 공격 코드와 대응 메시지 등을 집어넣는다.

```
void Detect::Rule_Reader(void)
  {
     fopen("./attack_rule", "r");
     for(rule_count = 0;
     rule_count        <        RULE_NUM;
rule_count++)
     {
        AttackRule[rule_count]              =
ReadRule();
        ParsingRule();
     }
  }
```

○ PCAPLIB를 통해 네트워크로부터 패킷을 수집한다. 라이브러리는 아래와 같은 순서로 사용하게 된다.

```
pcap_lookupdev()->pcap_lookupnet()
->pcap_open_live()->pcap_loop()
```

○ 캡쳐된 패킷을 분석 체크한다.

IP Header, TCP Header, UDP Header, ICMP Header등 각 프로토콜별로 정상적인 필드 값을 가지고 있는지 체크를 한다. 비정상적인 패킷일 경우 경고를 한다.

```
//정상적인 패킷 체크
if(PACKET->pkth->caplen
    < ETHER_HEADER_LENGTH)
{  cout << "Capture Data Length < Ethernet
   Header Length";
   return; }
//필드 체크
if(!PACKET->TCP_HEADER->ack)
   PACKET->ack = 0x01;
//프로토콜별 디코드 함수를 호출한다.
switch(PACKET->IP_HEADER->protocol)
{
   IPPROTOTCP:
   DecodeIP(PACKET *P);
   break;
   IPPROTOUDP:
   DecodeUDP(PACKET *P);
   break;
   IPPROTOICMP:
   DecodeICMP(PACKET *P);
   break;
}
```

○ 침입 탐지를 시작한다.

미리 읽어들인 룰 파일과 패턴 매칭을 실시한다. 매칭 될 경우 침입으로 판정하여 로그를 남기고 이에 대응하며 매칭되지 않을 경우 다음 패킷을 읽어들인다.

```
for(i = 0; I < RULE_NUM; I++)
{
//검사하는 패킷을 매칭 코드와 검사해 일치할 경우 체크
섬을 증가시킨다.
   for(j = 0; j < rule_size; j++)
   if(*(buf + j) == Attack_Rule[i].Code[j]
     checksum++;
//침입으로 판정되었을 경우
   if(checksum == rule_size)
  {
   cout << "Found Attack!!\n";
//공격지의 주소와 공격 목표점의 주소를 보여준다.
   cout<<SrcAddress<<"--->"<<
DstAddress;
//어떤 공격이 들어왔는지를 보여준다.
   cout << Attack_Rule[i].msg;
//공격임이 판명 되었으므로 로그를 남긴다.
   LogToFile();
   }
}
```

5.3 침입탐지시스템의 구현 결과

침입탐지시스템은 C++환경에서 구현을 하였으며, g++ 컴파일러를 이용해 링크하였다. 컴파일시에는 반드시 libpcap library가 설치되어 있어야 하며 실행할 때 반드시 root의 권한으로 실행해야 한다. 데몬 모드로 동작하게 하려면 "-d" 옵션을 주면 된다.

본 논문에서 구현된 침입탐지시스템이 가상으로 구현된 네트워크에 외부의 공격을 적용한 실험결과이다. 다음의 그림은 Source IP 203.237.111.177에 대한 외부에서 공격이 들어왔을 때의 화면이다. 공격이 탐지되면 탐지시간과 탐지 내용, 출발지 주소, 도착지 주소 등의 정보가 남게된다. ps는 서버에서 어떤 프로세스들이 동작하는지 알 수 있게 해주는 명령어이다.

ps명령어를 실행하여 프로세스들을 확인하고 감시 변조 가능하기 때문에 외부에서 접근시 탐지할 필요가 있다. 그림은 유닉스 명령어 ps 커맨드를 웹상에

```
#make
  g++ -c main.cpp
  g++ -c packet_cap.cpp
  g++ -c analyzer.cpp
  g++ -c detect.cpp
  g++    main.o    packet_cap.o    analyzer.o
detect.c           log.o -o hl-ids -lpcap
 #./hl-ids
  Initializing Interface Device....
  Device       : eth0
  Network      : 211.213.45.0
  Mask         : 255.255.255.0
  Now Detecting....
```

서 요청했을 시 탐지된 것이다. 그림 8은 유닉스 명령어 ps 커맨드를 웹상에서 요청했을 시 탐지한 것이다.

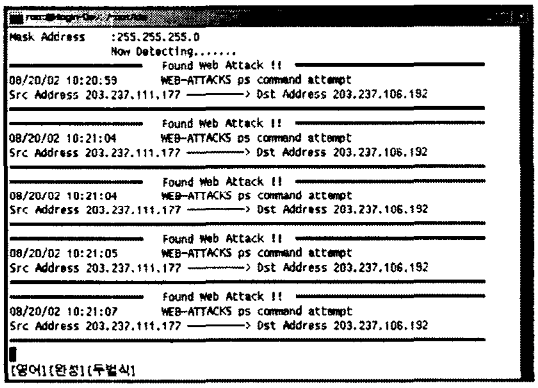


그림 8. 유닉스 명령어 ps 커맨드를 요청했을 시 탐지
fig 8. Detection of Unix Command ps

nc는 네트워크 유틸리티이지만 스크립트나 포트 연결 등을 통해서 해킹에도 이용된다. 이런 nc 명령어 또한 침입에 이용되기 때문에 탐지할 필요가 있다. 그림 9는 유닉스 명령어 nc를 웹으로 요청했을 시 탐지된 것이다.

/bin/sh명령은 쉘을 실행하므로 서 시스템에 명령을 내릴 수 있게 한다. 일반적으로 bufferoverflow나 formatstring공격의 마지막 단계는 이 쉘을 띄우는 것이다. 공격 프로그램이 /bin/sh에 해당하는 코드를 보냄으로서 공격을 수행한다. 그림 10은 리눅스 루트 쉘을 얻기 위한 공격을 했을 시 탐지된 것이다.

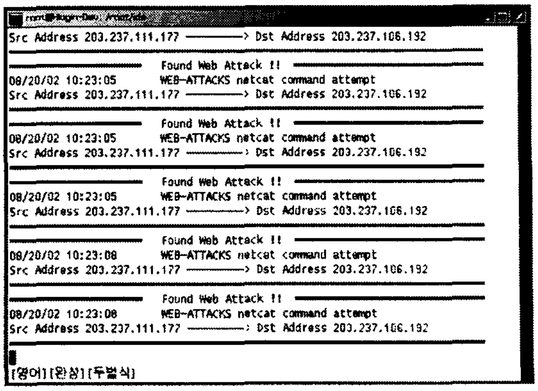


그림 9. 유닉스 명령어 nc를 요청했을 시 탐지
fig 9. Detection of Unix Command nc

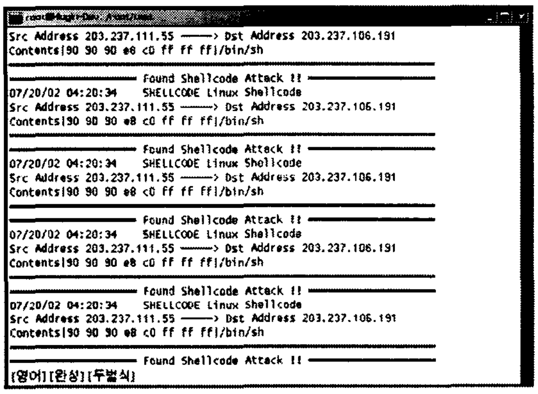


그림 10. 리눅스 루트 쉘을 얻기 위한 공격시 탐지
fig 10. Detection of obtain Linux Root shell

IGMP는 LAN상에서 호스트와 라우터 사이에서 그룹에 대한 정보를 교환할 때 사용되는 프로토콜로서 IGMP는 그룹에게 전달되는 메시지 중 목적지 IP 및 정보를 조작해서 한 호스트에게 집중시켜서 서비스 거부 상태에 이르게 된다. 그림 1a은 Denial Of Service 공격중의 하나인 IGMP 누크를 탐지한 것이다. pop3나 메일 관련 공격은 25번 포트를 통해서 이루어지며 buffer overflow는 쉘을 얻기 위한 공격이다. 메일데몬의 취약점을 이용해서 공격하게 되는데 25번 포트를 사용하기 때문에 공격 또한 25번 포트로 들어올 것이고 /bin/sh코드를 보내기 때문에 25번 포트와 /bin/sh 코드를 감지해서 침입을 알 수 있다. 그림 11은 리눅스 POP3 서버 버퍼 오버플로우 exploit 탐지한 것이다.

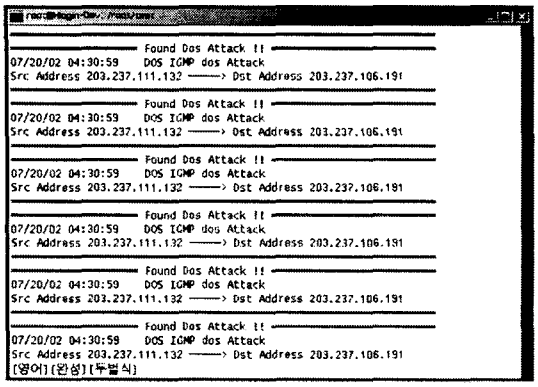


그림 11. Denial Of Service IGMP 누크 탐지
fig 11. Detection of Denial of Service IGMP nuk

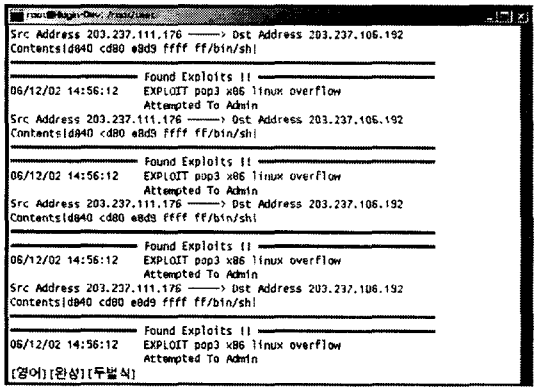


그림 12. 리눅스 POP3 서버 버퍼 오퍼플로우 exploit 탐지
fig 12. Detection of Linux POP3 Server buffer overflow exploit

VI. 결 론

본 논문에서는 패킷캡쳐를 바탕으로 한 IDS 모델을 제안하고, 이를 설계 및 구현하여 그 타당성을 보였다. 제안한 IDS 모델은 특권프로세스로 하여 이것들의 수행을 모니터링하여 침입여부를 판단한다. 제안한 IDS 모델은 특권 프로세스 행위를 정상 행위와 다르게 하는 경우 이를 침입으로 탐지할 수 있는 비정상 행위 침입탐지시스템이다. 본 논문에서는 침입탐지시스템의 타당성을 입증하기 위한 프로토타입을 단일 시스템에서 구현하여 탐지시간, 탐지정확도의 관점에서 제안한 모델의 성능을 평가하였다. 가상환경 하에서의 침입에 대한 평가를 해본 결과 기존의 HIDS나 NIDS의 문제점을 보완하기에 충분하였다. 본 논문에서 구현된 침입탐지시스템은 통신망 운영 및 이용기관의 인프라에서 사전 취약점 점검을 통해 전산망 침해사고를 예방 및 피해를 최소화하고, 효과적으로 침해사고에 대응할 수 있을 것으로 생각된다.

향후 연구과제는 룰 파일을 실시간으로 업데이트 할 수 있도록 DB 서버와의 연동 및 새로운 룰이 침입탐지시스템이 설치된 서버로 자동으로 갱신 될 수 있도록 해야 할 것이며, 윈도우의 GUI환경을 이용해서 관리자가 쉽게 상황을 보고 대처할 수 있는 매니지먼트 프로그램의 개발이 필요할 것이다.

감사의 글

본 연구는 1999년도 조선대학교 학술 지원본부의 지원에 의하여 이루어진 연구로서, 관계부처에 감사 드립니다.

참고문헌

[1] James Cannady, Jay Harrell, "A Comparative Analysis of Current Intrusion Detection Technologies," 1998. 2.

[2] Mansour Esmaili, Rei Safavi-Naini, "Case-Based Reasoning for Intrusion Detection, "Computer Security Applications Conference PP.214-222. 1996.

[3] 이종성, 채수환, "분산 침입 탐지 에이전트를 기반으로 한 지능형 침입탐지시스템 설계," 한국정보처리학회 논문지 제6권 제5호, 1999년 5월.

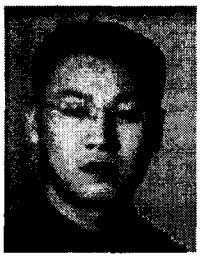
[4] Herve Debar, Marc Dacier and Andres Wespi, "Towards a Taxonomy of Intrusion-Detection Systems", Research Report of IBM Research Division, Zurich Research Laboratory, Jen, 1998.

[5] Taimur Aslam, Invan Krsul and Eugene Spafford, "Use of A Taxonomy of Security Faults". In 19th National Information System Security Conference Proceedings, Baltimore, MD, Oct. 1996

[6] Denning, Dorithy, "An Intrusion-Detection

Model", IEEE Transaction on Software Enginneering, Vol. SE-13, No.2, Feb.1987
[7] Mansour Esmaili, Rei Safavi-Naini, "Case-Based Reasoning of Intrusion Detection, "Computer Security Applications Conference PP.214-222, 1996.
[8] Teresa F. Lunt. "Detecting intruders in computer systems," 1993 Conference on Auditing and Computer Technology, 1993.
[9] Karl Levitt, Calvin Ko, and George Fink. "Automated detection of vulnerabilities in privileged programs by execution monitoring," 1994 Computer Security Application Conference, 1994.
[10] Debra Anderson, Teresa F. Lunt, Harold Javitz, Ann Tamaru, and Alfonso Valdes. "Detecting unusual program behavior using the statistical component of the Next-Generation Intrusion Detection Expert System (NIDES)".
[11] M. Bishop, "A STANDARD AUDIT TRAIL FORMAT", In Proceedings of the 18th National Information Systems Security Conference, Baltimore, Pages 136-145, 1995.
[12] Herve Debar, Marc Dacier and Andreas Wespi, "Research Report Towards a Taxonomy of Intrusion Detection Systems", IBM Research Division, Zurich Research Laboratory, June. 1998.
[13] Phillip A. Porras and Peter G. Neumann, EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances, SRI International, December 18, 1996.
[14] S. Staniford-Chen, S. Cheung, R.Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, D. Zerkle, "GrIDS-A GRAPH BASED INTRUSION DETECTION SYSTEM FOR LARGE NETWORKS", Department of Computer Science, UC Davis, CA 95616, January 26, 1999.
[15] Deborah Frincke, Don Tobin, Jesse McConnell, Jamie Marconi, Dean Polla, "A Framework for Cooperative Intrusion Detection, Center for Secure and Dependable Software", Department of Computer Science, University of Idaho, Moscow, ID 83844-1010
[16] Jai Sunder Balasubramaniyan, Jose Omar Garcia-Fernandez, David Isacoff, Eugene Spafford, Diego Zamboni, "An Architecture for Intrusion Detection using Autonomous Agents", COAST Laboratory, Purdue University, West Lafayette, IN 47907-1398, June 11, 1998
[17] Clifford Kahn, Phillip A. Porras, Stuart Staniford-Chen, Brian Tung, "A Common Intrusion Detection Framework", The Open Group, SRI, UC Davis, ISI, July,1998
[18] National Computer Security Center, A Guide to Understanding Audit in Trusted Systems, NCSC-TG-001 VERSION-2 Library No. S-228, 470, July, 1987.

저자소개



김영진(Young-Jin Kim)
1995년 2월 광주대학교 전자공학과 학사
1998년 2월 조선대학교 전자공학과 석사
2002년~현재 데이콤
2003년 2월 조선대학교 전자공학과 박사졸업 예정
※관심분야: 정보보호 및 정보이론, 암호학, VLSI설계

한승조(Seung-Jo Han)
1980년 2월 조선대학교 전자공학과 학사
1992년 2월 조선대학교 대학원 전자공학과 석사
1994년 충북대학교 대학원 전자계산학과 박사
1997년 조선대학교 전자정보통신공학부 교수
1986년~1987년 Univ. of New Oleans 객원교수
1995년~1996년 Univ. of Texas 객원교수
2000년 12월~2002년 2월 Univ. of Berkeley 객원학자
※관심분야: 통신보안 시스템 설계, 네트워크보안, ASIC설계, 음성합성