LKM을 이용한 센서기반 침입 탐지 및 보호 시스템

장철연°, 조성제, 최종무
단국대 정보컴퓨터학부
aceoftop@daum.net°, sjcho@dankook.ac.kr, choijm@dankook.ac.kr

Sensor based Intrusion Detection and Prevention System using LKM

Chul-Yean Chang°, Seongje Cho, Jongmoo Choi
Division of Information and Computer Science, Dankook University

요 약

우리 생활은 컴퓨터와 인터넷의 발달로 많이 편리해 졌고 향상되었다. 그러나 예전엔 중요하게 생각하지 않았던 보안이라는 문제가 발생되었다. 그 해결책으로 많은 침입 탐지 시스템이 개발되었다. 본 논문에서는 주요 디렉토리 및 파일의 접근을 감시하며 중요한 정보가 외부로 유출되는 것을 막고 시스템을 보호하는 SIDPS를 제안한다. 이 시스템은 특정 패턴을 이용한 기존의 방식과는 다른 "센서 파일/데이터"를 이용한다. 또한 LKM방식을 이용해 실행하도록 함으로서 손쉬운 설치 및 성능향상이 가능하도록 하였다.

1. 서론

컴퓨터와 인터넷의 발전으로 생활의 편리함과 동시에 보안이라는 문제가 새롭게 나타났다. 인터넷의 급속한 확대로 인터넷에 연결된 호스트에 대해 항상 연결이 가능해 졌고, 인터넷에서 구할 수 있는 많은 공개된 공격 도구들에 의해 쉽게 시스템이 해킹되는 등 최근 몇 년 사이에 컴퓨터 관련 범죄가 급증하였다.

현재 많은 침입 탐지 시스템의 존재하고 있지만 공격에 대해 완벽할 수는 없다. 또한 기존의 방법인 특정 패턴(*signature*)을 비교하여 침입 여부 및 공격을 판단하는 방법은 새로운 침입 방법에 대한 탐지가 어렵다. 따라서 "센서 파일/데이터"를 사용하여 새로운 침입에 대한 유연한 대처가 가능한 새로운 시스템 SIDPS를 제안한다. 이 시스템은 LKM을 사용함으로써 손쉬운 설치 및 성능향상을 제공하며 새로운 침입을 발견할 수 있다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구를 기술하며, 3장에서는 SIDPS의 "센서 파일/데이터" 기반 시스템 보호 방법을 설명한다. 4장에서는 제안한 시스템을 구현하고 실행한 결과를 보이며, 5장에서 결론 및 향후 연구에 대해 기술한다.

2. 관련 연구

침입 시도 기술로는 네트워크를 통해 어떤 시스템을 접근할 수 있는지, 제공되는 서비스는 무엇인지를 알아내는 스캐닝(scanning)기법과, 시스템으로부터 노출되는 자원의 이름이나 유용한 계정을 추출하는 *enumerating* 기법이 있다[1,8]. 이런 정보로부터 침입을 하기 위해 많은 새로운 침입 방법 및 도구들이 만들어지고 있으며 그로 인해 시스템의 보안이 더욱 중요해 졌다.

2.1 침입 탐지 시스템

침입 탐지 시스템(Intrusion Detection System, IDS)은 컴퓨터 시스템 또는 네트워크에서 일어나는 사건 및 사용자 행위들을 감시하고 침입 여부를 파악하기 위해 그 사건들을 분석하여 침입에 대응하는 소프트웨어 또는 하드웨어를 말한다[2,8,9]. IDS는 크게 호스트 기반 IDS(*HIDS*)과 네트워크 기반 IDS(*NIDS*)로 나눌 수 있다.

*HIDS*는 각 호스트에 설치되어 운영체제 감사로그(audit logs)와 다른 지역 데이터를 조사하고 시스템 보안 상태 변화를 추적하여 실제 공격을 탐지 할 수 있다는 장점이 있지만 네트워크 및 다른 호스트에 대한 공격을 알지 못하고 대부분 설치 및 유지가 복잡하다는 단점이 있다.

NIDS는 하나의 탐지기로 여러 호스트에 대한 트래픽을 감시하여 분산공격을 인지할 수 있지만 암호화된 네트워크 트래픽을 취급할 수 없고 공격이 실제로 성공하였는지 판단 할 수 없다.

본 논문에서 제안하는 SIDPS는 HIDS 와 NIDS의 기능이 결합된 시스템으로 각 IDS의 문제를 해결하기 위해 "센서 파일/데이터"를 사용해 침입 탐지 및 보호를 하게된다.

2.2 LKM (Loadable Kernel Module)

커널의 구조는 크게 마이크로 커널(micro kernel)과 단일 커널(monolithic kernel)로 구분을 한다.

마이크로 커널은 특정 기능을 수행하는 부분으로 구성되고 각각의 부분들은 통신을 통해서 연결된 구조로 되어 있다. 커널의 크기가 작고 단순하며 필요한 기능들은 커널 모듈을 동적으로 로드와 언로드 할 수 있다. 장점으로는 모든 운영체제 계층이 잘 정의된 소프트웨어 인터페이스를 통해 다른 계층과 상호작용을 할 수 있게 만들어져 시스템 프로그래머들이 모듈화된 접근을 채택하도록 만든다. 또한 하드웨어 의존적인 구성 요

소를 마이크로커널 속에 넣었기 때문에 다른 아키텍처로 포팅이 쉽고 램을 단일 커널 보다 잘 활용하는 경향이 있다. 그러나 운영체제의 다른 계층 사이에서 메시지를 전달하는 비용이 들기 때문에 일반적으로 단일 커널보다 느리다는 단점이 있다. 대표적인 마이크로 커널 시스템으로는 Mach, L4, QNX, AIX, Windows NT등이 있다. [6,10]

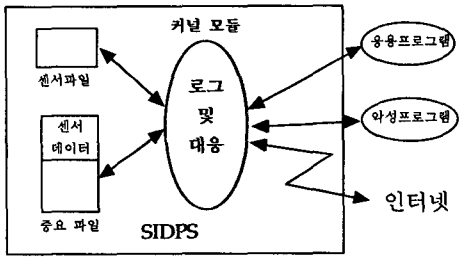
단일 커널은 커널에서 쓰이는 모든 함수와 자료구조를 하나의 프로그램으로 만드는 것이다. 엄격한 검증을 통해 새로운 기능이 추가되며, 추가 될 때마다 새로 커널을 컴파일하고 재부팅을 해야하는 단점이 있다. 단일 커널의 대표적인 시스템으로는 LINUX를 들 수 있다.[7]

커널 모듈은 단일 커널이 마이크로 커널의 여러 이론적인 장점을 활용할 수 있게 하는 기능으로 실행 중에 커널로 로드 및 언로드가 가능하며, 정적으로 링크된 다른 커널 함수와 마찬가지로 커널 모드에서 작동을 한다. 모듈화된 접근이 가능하며, 플랫폼에 독립적이고, 주 메모리를 적게 사용하며, 로드된 후 커널코드와 동등해지므로 성능이 떨어지지 않는다는 많은 장점을 가지고 있다[10].

SIDPS는 단일 커널 시스템인 LINUX에서 LKM로 구현함으로서 마이크로 커널의 장점을 활용할 수 있도록 하였다.

3. SIDPS

본 시스템의 논리적인 구성은 [그림1]과 같으며 이 시스템의 주요 목표는 "센서 파일/데이터"를 사용하여 자료의 외부 유출을 방지하는데 있다.



[그림1] SIDPS 의 구성도

3.1 센서 파일

"센서 파일"은 시스템 및 사용자가 접근할 필요가 전혀 없는 숨겨진 파일로 중요 디렉토리에 설치를 하게 되며, 만일 이 파일을 접근하면 프로세스 및 사용자는 수상한 행동을 하는 것으로 가정한다. 트로이 목마와 같은 악성 프로그램이 디렉토리내의 모든 파일을 외부로 유출하려고 할 때는 "센서 파일"을 접근하게 되고, 이 때 해당 프로세스에 대한 관련 정보를 기록하고 종료 시킴으로서 정보를 보호하게 된다.

3.2 센서 데이터

"센서 데이터"는 중요 파일 내부에 설치가 되며, 로컬 시스템에서는 접근이 가능하나 외부로 유출될 수 없으며, 특정 파일 하나만을 네트워크를 통해 외부로 유출하는 악성 프로그램 및 사용자를 차단함으로서 파일을 보호하게 된다.

3.3 보호

이 시스템에서 가장 중요한 부분은 "센세 파일"과 "센서 데이터"를 관리하는데 있다. "센서 파일"은 /boot, /etc, /dev, /lib, /proc, /root 등과 같은 시스템 디렉토리에, "센서 데이터"는 /etc/{passwd, group, services, lilo.conf}등과 같은 시스템 파일 및 사용자가 중요하다고 생각하는 파일에 설치하여 디렉토리 및 특정 파일에 대한 접근을 감시하고 정보를 보호하게 된다. "센서 파일/데이터"의 이름이 고정되어 있다면 악성 프로그램 및 사용자가 회피할 수 있으므로 주기적으로 바뀌게 하여 시스템의 보안을 높였다. "센서 파일/데이터"를 접근이 확인된 순간부터 파일의 삭제 기능을 중단 시켜 자료의 파괴를 막았으며, 외부에서 접속을 한 경우 해당 사용자ID로는 더 이상 접근이 불가능하며 관리자가 확인 후 재접속 여부를 결정할 수 있도록 하였다.

4. 구현

현재 SIDPS는 IBM PC 펜티엄 III, 와우리눅스 커널버전 2.4.20상에서 시스템 콜 가로채기(hooking)방법을 사용하여 구현되었으며 모듈 프로그램의 기본적인 형태는 [그림2]와 같다.

```
int init_module(void)
{
    orignal_sys_open = sys_call_table[__NR_open];
    ...
    sys_call_table[__NR_open] = SIDPS_sys_open;
    ...
}
void cleanup_module(void)
{
     sys_call_table[__NR_open] = orignal_sys_open;
     ...
}
```

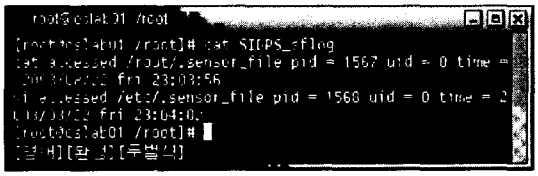
[그림2] 모듈 프로그램의 기본 형태

insmod 명령으로 SIDPS의 모듈을 실행 할 때 init_module()함수가 실행되며 본래 시스템 콜 함수의 주소를 바꾸게 된다. rmmod 명령으로 SIDPS의 작동을 중지시킬 때는 cleanup_module()함수가 실행되어 원래의 시스템 상태로 돌아가게 된다[3,4,11].

기존 시스템 콜 함수 sys_read(), sys_write(), sys_socketcall(), sys_fork(), sys_unlink()등을 수정하였으며 "센서 파일/데이터"의 설치 및 제거를 위해 시스템 콜 함수 SIDPS_sys_install()과 SIDPS_sys_delete()를 추가하였다. 네트워크관련 시스템 콜은 sys_socketcall()함수를 통해 실행되며, 현재는 UDP 네트워크 함수인 sys_sendto()이 호출될 때 데이터의 내용을 검사하여 정보의 유출을 감시하게 된다. 빠른 "센서 파일/데이터" 검색

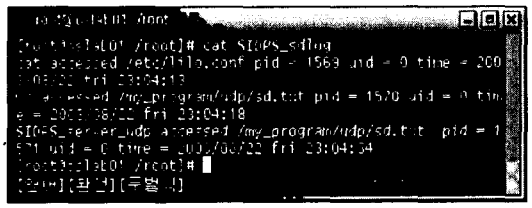
을 위해 boyer_Moore 알고리즘[5]을 사용하였다.

SIDPS가 작동하는 순간부터 시스템에서 접근하는 모든 파일에 대해 "센서 파일/데이터"에 대한 접근인지 감시하게 된다. [그림3]은 "센서 파일"을 접근했을 때 실행된 명령, 절대 경로, PID, 사용자ID, 접근 시간이 기록된 것을 보여준다. "센서 파일"은 어떤 프로그램에서도 접근될 필요가 없는 파일이므로 루트권한의 사용자라도 접근을 했다면 로그를 기록하고 해당 프로세스를 종료시키게 된다.



[그림 3] SIDPS_sflog에 기록된 내용.

"센서 데이터"가 포함된 파일은 로컬 시스템에서만 접근 및 활용가능하며 외부로 유출되어서는 안 되는 중요한 파일이므로 접근한 모든 프로세스에 대한 정보를 기록하며, [그림4]는 기록된 로그 정보를 보여준다.



[그림 4] SIDPS_sdlog의 내용.

UDP를 사용하여 파일을 전송하는 SIDPS_server_udp 프로그램을 작성하여 실험해 보았다. "센서 파일"에 대한 외부유출 여부는 SIDPS가 기록한 로그 파일 [그림4] 과 [그림5]로부터 SIDPS_server_upd 라는 프로그램이 "센서 데이터"가 포함된 sd.txt파일을 외부로 유출하는 것을 알 수 있다.

[그림 5] sids_nlog의 내용.

5.결론 및 향후 연구

본 논문에서는 "센서 파일/데이터"를 이용하여 새로운 침입에 대응하고 주요 디렉토리 및 파일의 접근을 감시하여 중요한 데이터가 외부로 유출되는 것을 막는 SIDPS를 제안하였다.

SIDPS는 LKM 방식으로 구현함으로서 커널을 직접 컴파일 해야 하는 어려움과 시스템을 재부팅 할 필요 없이 간단히 설치가 가능하도록 하였다.

향후 연구로는 "센서 파일/데이터"가 접근되었을 때 커널 수준에서의 대응뿐만 아니라 시스템 프로그램과도 연계해서 전체적으로 시스템이 작동 방식이 변화되는 방법 및 입력 방법의 변화에 대해서도 연구할 계획이다.

참고 문헌

[1] Anonymous, "Maximum Linux Security", SAMS, pp 538~549, 1999

[2] R. Bace and P.Mell, "Intrusion Detection System", Spectial Publication SP800-31, National Institute of Standards and Technology, Gaithersburg, MD, Released by NIST in August 2001

[3] pragmatic/THC, version 1.0 "Complete Linux Loadable Kernel Modules" (http://packetstormsecurity.org)

[4] Peter Jay Salzman, "The Linux Kernel Module Programming Guide" (http://www.linuxdoc.org)

[5]C.Charras and T.Lecroq, "Handbook of Exact String-Matching Algorithms" (http://www-igm.univ-mlv.fr/~lecroq/biblio_en.html)

[6] http://os.korea.ac.kr/kernel/link/m_kernel.html

[7] http://www.wikipedia.org/wiki/Monolitic_kernel

[8] 유일선, "네트워크 취약점 검색공격에 대한 개선된 탐지 시스템", 단국대 박사학위 논문 2001

[9] 유정각 "침입탐지 시스템의 개요" (http://doit.ajou.ac.kr)

[10] 이호, 심마로, "리눅스 커널의 이해", 한빛미디어

[11] 이호, "Linux Kernel Programing" (http://linuxkernel.net)