임베디드 리눅스에서 IPS의 설계 및 구현

채경철[0], 김상욱
경북대학교 컴퓨터과학과
{kccai,swkim}@woorisol.knu.ac.kr

A Design and Implementation of the IPS for Embedded Linux

Kyung-Chul Chae[0], Sang-Wook Kim
Dept. of Computer Science, Kyungpook National University

요 약

인터넷과 내장형 시스템의 발전으로 인하여 가정내의 모든 정보가전들이 홈 네트워크를 통하여 인터넷과 연결되어 있고 원격에서 제어가 가능하다. 또한 휴대폰, PDA와 같은 무선단말기를 이용한 정보검색, 메일 송수신, 증권거래, 은행거래와 같은 다양한 전자상거래가 활발히 진행되고 있다. 이와 반면에 이런 정보가전이나 무선단말기에 대한 해킹 바이러스 사례가 갈수록 많아지고 있다. 본 논문에서는 내장형 시스템인 임베디드 리눅스에서 IPS(Intrusion Prevention System)를 설계하고 구현하여 이런 정보가전이나 무선단말기에 대한 보안 방안을 제시한다.

1. 서 론

정보가전의 시대에 가정내의 모든 가전기기 및 PC 관련 제품들은 유무선 네트워크를 통하여 홈 네트워크를 구성하며 인터넷과 연결되어 원격에서 이런 정보가전에 대한 제어가 가능하다. 또한 휴대폰이나 PDA를 통하여 인터넷에 접속하여 정보 검색, 메일 송수신은 물론 인터넷 쇼핑, 금융결재, 증권거래와 같은 전자상거래가 활발히 진행되고 있다. 이런 정보가전이나 무선단말기는 인터넷에 연결되어 있기 때문에 PC와 마찬가지로 바이러스, 웜, 트로이목마 등 악성코드와 해킹의 위험에 노출되어 있다. 따라서 정보가전이 작동불능의 상태에 빠지거나 오동작을 할 수 있으며 개인 정보의 유출과 시스템에 대한 파괴행위를 초래할 수 있다.

2000년 8월에 Palm OS에서의 Liberty라고 불리는 트로이목마가 발견되었고 9월에 Phage라고 불리는 첫 번째 바이러스가 등장하였으며 앞으로 내장형 시스템에서의 해킹 바이러스 사례가 더 많아 질 것으로 예견된다. 이에 대응하여 McAfee, Symantec, Trend, F-Secure, 안철수연구소 등 회사에서 이미 무선단말기용 바이러스 백신 제품을 출시하고 있다. 하지만 이런 백신 프로그램은 일반적으로 PDA 플랫폼에 제한되어 있고 바이러스 정보를 계속 업데이트 해야 하며 새로운 바이러스에 대한 발견 및 치료가 불가능하다는 단점들이 있다.

2000년도 초반에 침입방지 기술이라는 새로운 개념으로 상용화된 IPS제품이 등장하였고 실시간으로 침입에 대한 대응, 알려지지 않은 공격으로부터의 방어 기능 등 장점으로 사람들의 각광을 받고 있다.

본 논문에서는 내장형 시스템의 운영체제인 임베디드 리눅스에서 IPS를 설계하고 구현하여 운영체제 수준에서 해킹 바이러스에 대한 대응 방안을 연구하고자 한다.

본 논문의 제 2절에서는 임베디드 리눅스의 취약점을 분석하고 대응 방안을 연구한다. 제3 절에서는 IPS의 설계와 구현에 대하여 기술한다. 4장에서는 IPS의 개발환경과 구현 예에 대하여 기술하고 5장에서는 결론 및 향후 과제에 대하여 기술한다.

2.관련 연구

2.1임베디드 리눅스의 취약점

임베디드 리눅스는 낮은 성능의 프로세서와 작은 크기의 메모리를 가진 내장형 시스템용으로 개발된 리눅스이다. 현재 임베디드 리눅스는 저렴한 가격, 공개된 소소코드, 우월한 안정성과 기능 등 장점으로 정보 가전, 통신기기, 제어기기, 무선단말기 등의 내장형 시스템 운영체제로 널리 사용되고 있다.

임베디드 리눅스는 일반적인 리눅스 커널의 메커니즘을 따르며 시스템 구조와 환경도 리눅스와 유사하다. 웹 브라우징, 메일 등 인터넷 접속기능을 지원하며 telnet, ftp, 웹 서버 등 인터넷 서비스 기능도 지원한다. 따라서 임베디드 리눅스도 리눅스와 마찬가지로 바이러스, 트로이목마와 같은 악성코드나 버퍼 오버플로우, 백도어 등 공격에 노출되어 있다. 또 한 임베디드 리눅스에서 사용자는 루트 권한으로 작업을 수행하기 때문에 트로이목마나 바이러스는 루트 권한을 쉽게 얻을 수 있으며 시스템 자원에 치명적인 파괴를 초래할 수 있다.

2.2 대응 방안 연구

아래에 백도어, 버퍼 오버플로우 등 공격에 대한 분석을 통하여 이런 공격들이 이루어 질 때 호출되는 시스템콜과 그 특징을 살펴보고 대응방안을 연구한다.

백도어는 해커가 시스템을 해킹 한 이후의 접근을 용

이하게 하기 위하여 설치되는 일종 트로이목마로서 일반적인 루트킷과 커널 백도어 2가지 종류가 있다. 일반적인 루트킷은 login, ls, ps, netstat등 주요 시스템 실행파일을 트로이버전으로 바꾸어 특정한 목적을 수행한다. 리눅스에서 파일에 대한 접근은 주로 open 시스템 콜을 통하여 이루어지며 전달되는 인자는 파일이름과 파일에 대한 접근권한을 나타내는 flag이다. 따라서 프로세스가 open 시스템 콜을 호출할 때 접근하고자 하는 파일과 프로세스의 정보에 대한 분석을 통하여 인가되지 않은 프로그램의 시스템의 중요한 디렉토리와 파일에 대한 접근제어를 통하여 시스템 자원을 보호할 수 있다. 리눅스에는 현재 커널에서 실행되는 프로세스의 정보를 저장하는 전역 변수인 task_struct 자료구조가 있다. 이 자료구조를 이용하여 실행되는 프로그램 이름, 사용자 식별자, 유효한 사용자 식별자와 같은 정보를 얻을 수 있다. 시스템의 중요한 실행 파일, 설정 파일 및 디바이스 장치들은 /bin, /sbin, /etc, /var, /dev 등과 같은 디렉토리에 위치하고 있다.

커널 백도어는 시스템 파일을 변경하지 않고 악의적인 기능을 수행하는 모듈을 커널에 로드한다. 커널 모듈을 로드할 때 create_module, init_module 등 시스템 콜을 호출하며 init_modue 시스템 콜을 호출할 때 모듈 이름을 인자로 넘겨준다. 임베디드 리눅스는 커널 모듈은 일반적으로 /lib/module 디렉토리에 위치한다. 따라서 init_module 시스템 콜이 호출될 때 전달되는 모듈 이름을 검사하여 정상적인 모듈이나 인가된 모듈에 대한 로드는 허용하고 그 외의 모듈은 로드하지 못하게 하여 커널 백도어가 설치되는 것을 방지할 수 있다.

버퍼 오버플로우 공격은 C, C++ 등 프로그램 언어에서 메모리 영역에서 스택에 대한 경계 조건을 검사하지 않는 취약점을 이용하여 함수의 복귀 주소에 쉘 코드를 삽입하는 기법이다. 버퍼 오버플로우 공격의 대상은 주로 백그라운드 데몬과 SUID 프로세스이다. 따라서 이런 프로세스가 실행될 때 호출하는 execve 시스템 콜과 전달되는 인자에 대한 분석을 통하여 인가되지 않은 프로세스의 생성을 방지하여 버퍼 오버플로우 공격에 대응할 수 있다.

위에서 분석한 바와 같이 실행되는 프로세스가 호출하는 시스템 콜에 대한 실시간 모니터링과 전달되는 인자에 대한 분석을 통하여 프로세스의 악의적인 행위에 대한 탐지가 가능하며 이에 대한 적절한 조치를 취하여 악성코드와 공격에 대한 방지가 가능하다. 표 1은 공격에 가장 많이 이용되는 시스템 콜이다.[1]

system calls	dangerous parameter
chmod, fchmod	a system file or a directory
chown, fchown, lchown	a system file or a directory
execve	an excutable file
mount	on a system directory
rename, open	a system file
link, symlink, unlink	a system file
setuid, setreuid, setfsuid, setreuid	UID set to zero
setgroups, setgid, setfsgid, setresgid, setregid	GID set to zero
create_module	modules not in /lib/modules

표 1 보안 관련 시스템 콜

3. IPS의 설계 및 구현

본 논문에서 구현하는 IPS는 커널 모듈로 구성되며 커널에 로드된 후 커널에 있는 시스템 콜[2] 테이블을 변경하는 기법으로 시스템 콜을 모니터링 한다. 따라서 프로세스가 특정 시스템 콜을 호출할 때 IPS에서 프로세스의 정보와 전달되는 파라미터에 대하여 분석하고 적절한 보안정책을 적용하여 시스템 자원을 보호한다.

3.1 IPS의 구성

IPS는 Rules Loading Component, System Call Intercept Compont, System Call Analyzing Component, Action Engine 4개의 세부 모듈과 Rule List, Original System Call Table 등 필요한 자료구조로 구성된다. 그림 1은 IPS의 전체적인 구조와 각 세부 모듈사이의 관계를 나타낸다.

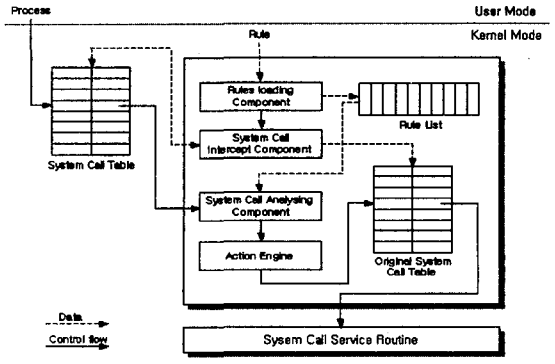

그림 1 IPS의 구성

3.2 IPS의 구현

본 논문에서 IPS는 커널 모듈로 구현된다. 리눅스 커널 모듈은 동적으로 커널에 로드하거나 제거할 수 있는 커널의 구성요소이다. 따라서 IPS는 동적으로 커널에 링크되어 커널 공간의 자원(함수, 자료 구조)에 접근할 수 있으며 시스템 콜 테이블에 대한 변경이 가능하다.[3]

Rules Loading Component는 사전에 정의된 룰(보안

정책)을 커널에 로드하여 룰 리스트를 생성한다.

System Call Intercept Component는 원래의 시스템 콜 테이블을 복제하여 *Original* 시스템 콜 테이블에 저장한다. 그리고 모니터링 하고자 하는 시스템 콜의 시스템 콜 테이블에서의 해당 엔트리를 System Call Analyzing Component에서 정의한 대응 처리함수의 주소로 변경한다. 따라서 시스템 콜을 호출되면 직접 시스템 콜 서비스 루틴을 호출하지 않고 System Call Analyzing Component의 대응 처리함수를 실행하게 한다.

System Call Analyzing Component에서는 모니터링 하고자 하는 시스템 콜의 대응 처리함수를 정의한다. 이 처리 함수는 전달되는 시스템 콜의 인자와 룰 리스트에 저장된 룰이 매칭되는가 비교하고 그 결과를 인자로 Action Engine에 전달한다.

Action Engine은 System Call Analyzing Component의 분석 결과에 근거하여 시스템 콜의 호출에 대한 허용 여부를 결정하고 적절한 조치를 취한다.

3.3 보안 정책

보안 정책의 설정은 IPS에서 가장 핵심적인 부분이다. 리눅스는 기본적으로 사용자 식별자, 사용자 그룹 식별자에 근거하여 파일이나 디렉토리에 대한 접근 제어를 수행한다. 하지만 임베디드 리눅스에서 사용자는 루트의 권한으로 작업을 수행하기 때문에 바이러스, 트로이목마, 웜 등 악성코드가 실행되면 루트 권한으로 시스템의 모든 자원을 액세스할 수 있기에 시스템에서 제공하는 파일 Permission 메커니즘으로는 시스템 자원을 적절하게 보호할 수 없다. 따라서 임베디드 리눅스에서 시스템 자원과 사용자의 정보를 보호하기 위한 새로운 보안 메커니즘이 필요하다. 본 논문에서는 사용자가 아닌 프로세스 수준에서 파일 시스템과 시스템 콜에 대한 접근 제어를 통하여 임베디드 리눅스 환경에 적절한 보안 정책을 적용한다.

4. 평가

본 논문에서 구현하는 IPS는 임베디드 리눅스를 탑재한 Zaurus SL-5500에서 실행한다. Zaurus SL-5500의 기본사양은 아래와 같다.

-206Mhz StrongARM 프로세서
-64MB RAM(SDRAM)
-16MB ROM(FlashROM)

임베디드 리눅스 커널은 kernel2.4.6-rmk1-np2-embedix 이다. IPS를 개발하기 위하여 PC에 다음과 같은 개발 도구가 필요하다.

-크로스 컴파일러(cross-arm4l-gcc-2.95.2-10mz)
-라이브러리(cross-arm4l-glibc-2.2.1s-3mz)
-binutils-2.10-3mz
-cross-armv4l-kernel-headers-2.4.5_rmk6_np1-1mz

일반적으로 임베디드 리눅스에서 사용자는 루트 계정으로 로그인하기에 시스템의 모든 자원을 접근할 수 있다. 본 논문에서 구현하는 IPS는 관리자가 설정한 보안정책에 의해 시스템과 사용자의 중요한 파일이나 디렉토리를 적절하게 보호한다. 그림 2는 IPS의 구현의 예이다.

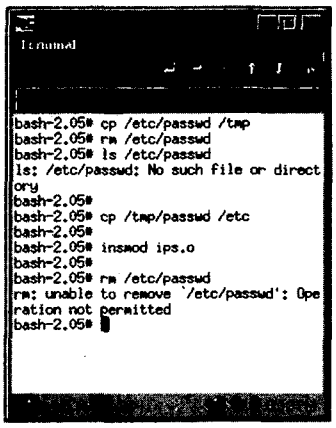

그림 2 구현 예

5. 결론 및 향후과제

본 논문에서는 임베디드 리눅스에서 IPS를 설계하고 구현하여 악성 코드와 해킹에 대한 대응 방안을 제시한다. 시스템 콜 테이블을 변경하는 기법으로 사용자 프로세스가 호출하는 시스템 콜을 분석하고 이에 보안정책을 적용하여 시스템과 사용자의 자원을 보호한다. 향후 연구과제로 해킹 바이러스 사례분석과 보안정책에 대한 연구를 통하여 IPS에 적절하게 적용될 수 있는 보안정책을 수립하는 것이다.

6.참고문헌

[1]Massimo Bernaschi, Emanuele Gabrielli, Luigi V. Mancini, "Operating System Enhancements to Prevent the Misuse of System Calls", 2000.

[2]Daneil p. Bovet and Marco Cesati, "Understanding the Linux Kernel", pp234~236, 2001.

[3]이호, "Linux Kernel Programming v1.01", http://linuxkernel.net/module/doc/lkp.pdf, pp.6.