□특집□

리눅스 보안 연구개발 동향

이 철 원[†] 김 홍 근[††] 박 태 규[†††]

◈ 목 차 ◈

1. 서 론

인터넷을 통해 전 세계의 컴퓨터들이 네트워크로 연결되면서 해킹이나 바이러스와 같은 침해사고들이 빈번하게 발생하여 많은 피해가 발생하고 있다. 이에 따라, 시스템의 안전에 대한 인식도 증가하여 컴퓨팅 환경에서 보안 기능을 추가하려는 노력도 증가하게 되었다. 최근 국내 한 연구기관에서 발표한 다양한 침해사례를 살펴보면, 컴퓨터 시스템에 대한 해킹 수법이 지능화되고 다양해지면서 특히 운영체제 등의 시스템의 취약점을 이용하는 수법이 해킹의 상당 부분을 차지하고 있음을 알 수 있다[1]. 또한, 일반 사용자에게 많이 알려진 UNIX와 유사하고, 다양한 하드웨어 플랫폼 지원, 강력한 네트워크 지원, 다양한 형식의 파일시스템을 지원하는 등 막강한 성능을 가지고 있는 리눅스(Linux)라는 걸출한 운영체제가 무료로 보급되면서 리눅스의 취약성을 이용한 해킹사례가 국내·외에서 많이 보고되고 있는 실정이다. 리눅스는 소스코드의 공개와 관련 문서의 풍부한 제공으로 쉽게 수정이 가능하며, 많은 사람의 검증을 거친 운영체제로 안정성이 뛰어나며, PC 기반에서 운영이 가능하고 설치비용이 적게 들어 최근 들어 급속도로 보급이 확산되고 있다. 최근 매년 25%의 성장률을 보이고 있는 리눅스의 보급추세로 미루어 짐작할 시 향후에는 리눅스가 주요 서버시장의 큰 부분을 잠식할 것이라는 것을 쉽게 예측할 수 있다. 따라서, 그 어느때보다도 리눅스 운영체제의 성능을 향상시키기 위한 노력도 필요하지만 리눅스 운영체제에 보안기능을 첨가하는 것도 시급한 일이라 할 수 있다. 또한 최근의 운영체제 개발동향이 커널의 최소화 방향으로 나아가고 있으므로 마이크로 커널 기반의 안전한 리눅스 개발은 필수적인 일이라 생각된다.

본 논문에서는 리눅스에서 발견되고 있는 주요 취약성을 고찰하고, 이러한 취약성을 방지하기 위한 리눅스 보안 연구개발동향에 대하여 고찰한다. 또한, 응용 프로그램 수준이 아닌 운영체제 수준에서 보안을 제공하기 위한 마이크로 커널 기반의 안전한 리눅스 접근방법도 분석하였다.

본 고의 구성은 다음과 같다. 리눅스 운영체제의 보안 취약성을 제 2장에 기술하고, 안전한 리눅스를 개발하기 위한 요구사항을 제 3장에 기술하며, 현재 개발되어 있는 리눅스 보안제품 및 마이크로 커널을 기반으로 하는 안전한 리눅스 접근방식을 제 4장에 기술하고 향후 연구방향에 대하여 제 5장에서 기술한다.

† 정회원 : 한국정보보호센터 선임연구원
†† 정회원 : 한국정보보호센터 시스템기술팀장
††† 정회원 : 한서대학교 컴퓨터과학과 부교수

2. 리눅스의 보안 취약성

리눅스는 개방성과 우수한 성능으로 인해 점차적으로 각광받고 있는 유닉스 계열의 운영체제로 다음과 같은 특징을 가지고 있다[2][3].

- 유닉스 운영체제와 유사하며 POSIX 표준을 준수하여 호환성이 높다.
- 인텔 호환 프로세서, 디지털 알파, Sun Sparc, PowerPC, M68K 칩 등 지원 환경이 다양하다.
- 완벽한 멀티 유저, 멀티 테스킹 시스템을 지원한다.
- 안정성이 뛰어나고 소스까지 완전 공개시스템으로 수정하기가 쉽다.
- TCP/IP부터 모뎀 PPP까지 다양한 네트워크 프로토콜을 지원한다.
- NTFS, FAT32, SysVFS, SunOS, FreeBSD 등 다양한 형식의 파일시스템 지원한다.
- 빠른 업그레이드로 운영체제의 버그에 대한 대응이 신속하다.
- 리눅스 기반의 풍부한 유틸리티가 무료로 제공된다.
- 사용자를 위한 여러 가지 공개 문서들이 제공되어 있다.
- Samba 등의 유틸리티를 이용해서 유닉스와 윈도즈와의 상호 이용성이 뛰어나다.

2.1 리눅스 보안 취약성

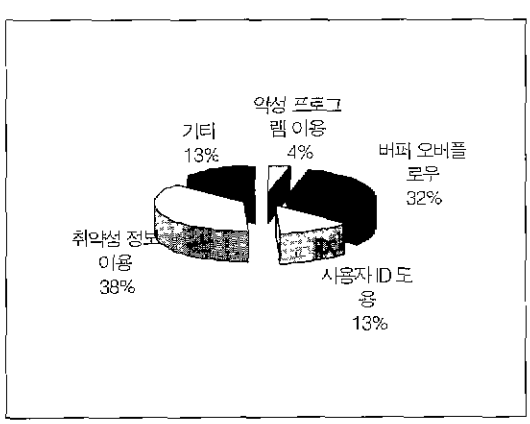
리눅스의 오픈 정책으로 인하여, 운영체제 커널에서부터 일반 사용자 프로그램들까지 거의 모든 것이 공개된 리눅스는 과거 유닉스가 소수만이 사용하던 운영체제라는 인식을 종식시키고 일반 PC에서도 강력한 유닉스 서버 프로그램을 돌릴 수 있게 해준다는 점에서 매우 매력적이다. 하지만 이러한 소스의 공개로 인하여 해커들이 쉽게 버그를 찾아내고 테스트할 수 있는 주요 공격

(표 1) 리눅스 보안 취약성

Linux 시스템명	보안취약점
Red Hat Linux 6.0	mars_nwe, XFree86, inews, amd, vixie-cron, wu-ftpd, in.telnetd, New boot images, linuxconf, gnome, enlightenment, rdist, rpm, libtermcap, pump, squid, samba, gnumeric, net-tools, KDE, dev, rxvt, screen, INN, xscreensaver, netkit-base, traceroute, talk, mod php3, utempter, apmd
Red Hat Linux 5.2	mars_nwe, inews, amd, vixie-cron, in.telnetd, libtermcap, squid, rpm, timetool, imap, mod_perl, rsync, procmail, lpr, pine, mutt, zgv, dump, perl, Xconfogulator, fvwm2, lsof, wu-ftpd, minicom, XFree86, kernel, pam, ftp client, nfs server, samba, sysklogd
Red Hat Linux 5.1	wu-ftpd, minicom, XFree86, NFS, pam, ftp client, samba, sysklog, bash, xscreensaver, linuxconf, apache, imapd, SysVinit, mutt, initscripts, glibc, libtermcap, tin, slang, bind, metamail, mailx, elm, dhcp, dhcpcd, bootp, xosview
Red Hat Linux 5.0	wu-ftpd, minicom, XFree86, NFS, pam, ftp client, samba, sysklogd, bash, xscreensaver, apache, SysVinit, mutt, ncurses, imap, glibc, libtermcap, bind, tin, slang, metamail, mailx, elm, findutils, dhcp, dhcpcd, bootp, lpr, procps, lynx, kbd, mh, ncftp, perl, textutils
Debian GNU/Linux	lsof, wget, Eterm, super, cfengine, Debian FTP packages, ftpwatch, netstd, sshd, fsp, zgv, samba, junkbuster, tcsh, bash, nslookup and dig, rpc.mountd, minicom, seyon, sail, apache, News, lprm, eperl, ncurses, mutt, cfingerd, faxsurvey, filerunner, cxhextrix, mailx, premail, kdebase, samba, gzip, shadow-su, procps, super, irc, bind, perl, netstd, lincity, gzip, gcc, textutils, dwww, sudo, smail, deliver

대상이 되고 있는 것도 사실이다. 물론 이러한 시도들로 인하여 리눅스는 더욱 안정화되고 보안이 강화되고 있다. 실제로 리눅스를 많이 사용하고 있는 국내 대학들은 외국의 해커에 의해 끊임없는 공격에 시달리고 있는 실정이다. 대표적이 예로 1999년 초 모 대학의 연구실에 설치된 리눅스 시스템이 미국의 모 ISP업체 사용자로부터 mscan과 imap 취약점 등을 이용하여 해킹 당한 뒤 다시 NASA, 미 국방부 등 미국의 주요 시스템을 공격하는 데 사용된 사고가 있었다[1]. 이와 같이 리눅스는 전세계 해커의 주요 침입대상 시스템이 되고 있으며, 현재까지 약 160여가지의 취약성이 보고되고 있다. 리눅스 시장의 가장 많은 부분을 차지하고 있는 RedHat에서 발표한 자료에 의하면 '99년 5월부터 9월 현재까지 Red Hat 리눅스 6.0에서 mars_nwe, XFree86, innews 등 34여가지의 취약성이 발견되었으며, 지금까지 발표된 각 버전별 리눅스 관련 보안 취약성은 다음과 같다.

자세한 정보는 http://www.redhat.com/corp/support/errata를 참조하기 바란다[4]. 상기와 같은 보안 취약성을 이용하여 '99년 1/4분기 국내에 발견된 리눅스 관련 해킹 사고는 총 53건이었으며, 이중에서 Root 권한을 획득하는 침입수법이 34건으로 전체의 63%를 차지하고 있으며, 이중 버퍼 오버



(그림 1) '99년 1/4분기 리눅스 관련 해킹사고

플로우를 이용하는 방법은 17건으로 Root 권한 획득 방법의 50%를 차지함을 알 수 있다[5]. 여기서, 흥미로운 사실은 Root 권한 획득 또는 시스템 침입을 위한 기본적인 방식으로 시스템의 취약성 정보를 수집하는 mscan 또는 sscan 등을 많이 활용한다는 점이다.

위에서 열거한 다양한 리눅스의 취약성중 최근에 발견된 공격에 가장 많이 사용되는 취약점은 다음과 같다[1].

2.1.1 IMAP 취약점

IMAP 서버는 리눅스 운영체제를 설치할 때 기본적으로 설치되는 것으로 사용자의 메일을 서버에서 관리하도록 해주는 프로그램이다. 보통 일반 사용자들은 POP서버를 사용하며 IMAP 서버는 관리자가 인지하지 못하는 사이에 데몬으로 동작하고 있는 경우가 많다. IMAP 서버가 사용자 확인을 위하여 아이디와 패스워드를 입력받을 때 그 길이에 대한 한계 값에 대한 검사를 충분히 하지 않아 조작된 값을 입력함으로써 공격자는 원격에서 루트 권한으로 임의의 명령을 수행시킬 수 있다. 다음은 imap 취약점 이용 공격사례로서 시스템의 관리자 권한의 쉘을 획득하는 공격이다.

2.1.2 POP 서버 취약점

POP 서버는 PC 등 클라이언트에서 메일서버에 접속하여 메일을 송·수신하도록 하는 서비스를 제공한다. 사용자가 유닉스 시스템의 POP서버에 접속할 때 사용자 아이디와 패스워드를 입력하게 되고 이러한 인증 절차를 거쳐서 사용자는 유닉스 시스템에 저장된 자신의 메일을 확인할 수 있게된다. 하지만 POP 서버가 사용자 확인을 위하여 아이디와 패스워드를 입력받을 때 그 길이에 대한 한계 값을 검사하지 않아 조작된 값을 입력함으로써 원격에서 루트 권한으로 임의의 명령을 수행시킬 수 있다.

2.1.3 bind 취약점

BIND 4.9.7 이전 버전과 BIND 8.1.2 이전 버전에서 inverse query 요청에 대한 응답시 적절한 한계값 검사를 하지 않아 버퍼 오버플로우 취약점이 존재한다. 공격자는 교묘히 조작한 패킷을 전송하여 루트 권한으로 임의의 명령을 실행시킬 수 있다.

2.1.4 mountd 취약점

NFS(Network File System)는 네트워크를 통하여 컴퓨터간에 파일 시스템을 공유하기 위한 클라이언트/서버 프로그램이다. NFS 클라이언트가 NFS 서버의 파일에 접근하기 위해서는 먼저 파일 시스템을 마운트한다는 요청을 하게되는데, 이 NFS 마운트 요청을 처리하는 소프트웨어(mountd 프로그램)에 버퍼 오버플로우 취약점이 존재한다. 특히 리눅스 시스템에서는 기본적으로 NFS 서버가 mountd를 구동하기 때문에 매우 위험하며, 공격자는 시스템 관리자 접근 권한을 획득할 수 있다. 모든 Red Hat 리눅스가 취약하며 새 nfs서버 패키지를 구해 설치하여야 한다.

3. Secure Linux를 위한 요구사항

현재 우리나라의 보안제품 연구 개발은 기존의 컴퓨터 시스템 상에 보안 서비스를 추가하는(add-on) 방식으로 진행되어 왔다. 그러나, 이러한 방식으로는 기존에 알려진 우회, 수정 등의 문제점 및 새로운 보안상의 문제점인 트로이 목마, 비밀채널(Covert Channel) 등의 신종 컴퓨터 범죄를 해결할 수 없으며, 문제점 발생 시마다 보안제품을 계속적으로 추가하는데 드는 비용과 번거로움이 뒤따름에 따라, 미국을 비롯한 선진 외국에서는 이미 1980년대부터 컴퓨터 시스템의 OS의 내부 커널에 보안기능을 포함시키는 보안 커널의 연구 개발을 추진하여, 제품을 생산하고 있는 추세이다. 미 국방부에서는 보안 커널을 "컴퓨터 보안에 있어서, 참조 모니터(Reference Monitor) 개념을 구현한 TCB의 하드웨어, 펌웨어, 소프트웨어 요소이며, 이 보안 커널은 모든 접근을 중재해야만 하며, 수정으로부터 보호되어야 하며, 정확성이 검증되어야 한다"로 정의하고 있다. 여기서, TCB란 컴퓨터 보안에 있어서, 하나의 컴퓨터 시스템(하드웨어, 펌웨어, 소프트웨어)내의 모든 보호 메커니즘을 의미한다[6].

본 절에서는 현재 리눅스가 안고 있는 취약성을 제거하여 안전한 리눅스 시스템을 만들기 위한 요구사항에 대하여 정의하였다.

3.1 안전한 운영체제의 설계요소

안전한 컴퓨터 시스템을 구현하기 위해서는 시스템이 유지하는 정보에 대한 기밀성(confidentiality), 무결성(integrity), 가용성(availability)을 보장하여야 한다[7]. 기밀성, 무결성, 가용성을 보장하기 위하여 안전한 운영체제를 설계 및 구현할 시에는 다음과 같은 요소를 고려하여야 한다.

3.1.1 특권의 최소화

부적절하거나 악의가 있는 공격으로부터 피해를 최소화하기 위해 각 사용자와 프로그램들은 가능한 최소한의 특권을 사용하여 운영되어야 한다.

3.1.2 메커니즘의 경제성

보호 시스템의 설계는 작고 단순하고 직접적이어야 한다. 이와 같이 작고 단순한 보호 시스템은 보안성에 대한 완전분석, 시험 및 검증될 수 있다.

3.1.3 완전한 중재

모든 접근 시도는 체크되어야만 한다. 직접적인 접근 시도뿐만 아니라 접근통제 메커니즘을 우회하려는 시도가 고려되어야만 한다.

3.1.4 허가 기반

기본적으로 접근은 거부되어야만 한다. 일반적인 설계자는 접근하지 못하는 항목을 정의하기보다는 접근가능한 항목을 정의한다.

3.1.5 특권의 분리

이상적으로, 객체에 대한 접근은 사용자 인증, 암호키 등과 같이 한가지 이상의 조건들을 사용하여야만 한다. 이 방식에 의하면 하나의 보호 메커니즘을 우회한 사람도 완전하게 보호하고자하는 대상에 접근할 수 없을 것이다.

3.1.6 최소한의 공통 메커니즘

공유된 객체는 정보흐름 가능성이 있는 채널을 제공한다. 물리적, 또는 논리적인 분리를 이용하는 시스템은 공유로부터의 위험을 제거한다.

3.1.7 사용의 용이

쉬운 사용방법은 보호 메커니즘을 우회할 가능성을 제거한다.

3.2 Secure Linux 보안 요구사항

전통적으로 군사적 목적 및 정부기관의 중요 문서를 보호하기 위하여 적용되었던 안전한 운영체제 설계 및 구현은 TCSEC(Trusted Computer System Evaluation Criteria)의 요구사항을 적용하였다. TCSEC은 C1(최하위 등급), C2, B1, B2, B3, A1(최상위 등급)등급의 6가지 등급을 가지며 각 등급별로 보안 요구사항을 정의하고 있다[8]. TCSEC의 요구사항에 맞게 구현된 시스템은 시스템 내부의 중요한 정보를 효율적으로 보호하였지만, 시스템의 성능을 저하시키는 하나의 원인이 되기도 하였다. 최근 들어 폭발적으로 증가하는 인터넷 사용추세를 감안할 시 안전한 운영체제에 대한 필요성은 급증할 것으로 예측되는바, 군사용이 아닌 일반 사용자를 위한 안전한 운영체제의 개발이 필요하리라 생각되며, 이러한 요구사항을 바탕으로 안전한 리눅스를 개발하기 위한 보안 요구사항으로는 다음과 같은 것이 있을 수 있다.

3.2.1 식별 및 인증(Identification and Authentication)

시스템 내부의 주요 자원에 대한 접근통제의 근간이 되는 것이 시스템을 사용하려는 개인의 식별 및 인증이다. 따라서, 사용자에 대한 식별 및 인증은 정확하여야 한다. 특별히, MLS(Multi Level Security)를 위하여 사용자 식별 및 인증과정에 사용자의 신원허가(Clearance)를 추가로 입력하여야 한다. Secure Linux는 사용자의 안전한 식별을 제공하여야 하고 각 사용자는 유일하게 신분이 확인되어야 한다.

3.2.2 접근통제

안전한 운영체제의 핵심은 시스템내의 자원에 대한 접근을 통제하는 접근통제 메커니즘이다. HRU 모델, BLP 모델, RBAC 모델 등 지금까지 많은 접근통제 모델이 개발되어져 왔다. 전통적으로는, 강제적 접근통제(MAC, Mandatory Access Control)를 위하여 BLP 모델이 가장 많이 활용되었으나, 최근 들어 의료, 금융 등에서 역할기반접근통제를 많이 활용하고 있다.

- BLP 모델
  - 읽기(read), 실행(execute) 정책
    한 주체가 어느 한 객체에 대하여 읽기 또는 실행 접근권한를 얻기 위해서는 그 주체의 보안레이블은 객체의 보안레이블을 지배(dominate)하여야 한다.
    여기서, 주체라 함은 사용자 혹은 사용자를 대신하는 프로세스를 의미하며, 객체라 함은 파일과 같은 프로세싱 능력이 없는 엔터티를 의미한다.
  - 쓰기(write) 정책
    한 주체가 어느 한 객체에 대하여 쓰기 접근

권한을 얻기 위해서 그 객체의 보안레이블은 주체의 보안레이블을 지배하여야 한다.

여기서, 지배한다는 의미는 보안레이블 S1의 비밀등급이 S2의 비밀 등급보다 크거나 같고, S1의 카테고리 집합은 S2의 카테고리 집합을 포함하는 것을 의미한다.

- RBAC 모델

역할기반 접근통제 모델은 사용자(U, User), 역할(R, Role), 허가(P, Permission)의 세가지 엔티티들로 정의된다. 사용자는 컴퓨터의 네트워크, PC, 로봇과 같은 도구들도 포함할 수 있으나 일반적으로 사람이라고 정의한다. 역할은 역할의 구성원에게 수여된 책임과 권한에 관련된 의미를 가진 조직내의 직무 기능이나 직무 이름이다. 허가는 시스템내에 하나 이상의 객체에 대한 접근 승인이다. 허가에서 객체는 컴퓨터 시스템내의 자원뿐만 아니라 데이터도 포함된다.

사용자는 여러 역할들의 멤버가 될 수 있고 역할은 여러 사용자들을 가질 수 있다. 유사하게 역할은 많은 허가를 가질 수 있고 허가는 여러 역할들에 할당될 수 있다. 이러한 두 관계들을 나타내는 것이 사용자 할당(UA)과 허가 할당(PA)이며 다음과 같이 정의된다.

PA $\subseteq$ P $\times$ R, a many-to-many permission to role assignment relation

UA $\subseteq$ U $\times$ R, a many-to-many user to role assignment relation

사용자는 워크스테이션에서 여러 윈도우를 여는 것처럼 동시에 여러 세션을 열 수 있으며 다음과 같이 정의된다.

user : S $\rightarrow$ U, a function mapping each session s_i to the single user $user(s_i)$ (constant for the session's lifetime)

roles : S $\rightarrow$ 2R, a function mapping each session s_i to a set of roles

$roles(s_i) \subseteq \{r \mid (user(s_i), r) \in UA\}$(which can change with time) and session s_i has the permissions $U_{r \in roles(s_i)}\{p \mid (p, r) \in PA\}$

여기서 세션(S, Session)은 한 사용자와 역할의 가능한 매핑을 나타낸다. 이 역할기반 접근통제 모델은 최소 권한(least privileges)의 원칙을 지원한다. 여러 역할의 구성원인 사용자는 그 세션에 수행되는 작업을 위해 이 역할들의 일부를 실시할 수 있다. 그러므로 역할의 구성원인 사용자는 정상상태에서 이 역할을 활성화하지 않다가 필요할때 활성화할 수 있다. 세션의 개념은 전통적인 접근통제 개념에서 주체(subject)와 같다. 주체는 접근통제의 단위이고 사용자는 동시에 다른 허가를 활성화시키는 여러 주체들을 가질 수 있다[9].

3.2.3 완전한 중재(Complete Mediation)

접근통제를 효과적으로 수행하기 위해서는 시스템 내부 모든 자원에 대한 모든 접근을 중재하여야 한다. 메모리 또는 외부 포트, 네트워크, 비밀(Covert) 채널과 같은 것을 통해서 공격이 가능하므로 단지 파일에 대해서만 접근을 통제하는 것으로는 불충분하다. 안전한 운영체제의 설계와 구현의 어려움은 접근을 위한 모든 통로들을 모두 통제해야 하기 때문이다.

3.2.4 신뢰 경로(Trusted Path)

공격자는 안전한 운영체제내의 보안을 시행하는 부분(예를 들어, TCB)에 직접적으로 침투하여 시스템의 보안을 붕괴하고자 한다. 따라서, 패스워드를 변경하거나 접근 권한을 바꾸는 것과 같은 중요한 보안관리 기능을 수행할 때에는 시스템의 보안을 시행하는 부분 즉, TCB와의 신뢰성

있는 경로가 설정되어야 한다. 일부 시스템은 특정 키를 누르게 되면 신뢰 경로가 설정되며 또 다른 시스템은 보안 관련 변경은 단지 시스템 시작시에만 변경할 수 있도록 구현되어 있다.

3.2.5 감사(Audit)

보안과 관련한 사건은 파일과 같은 객체에 대한 개인적인 접근처럼 단순할 수도 있고 모든 접근 관련 정보가 저장되어 있는 데이터베이스의 접근과 같이 중요할 수도 있다. 감사는 일반적으로 발생한 보안 관련 사건의 로그를 유지하는 것을 수반해야 한다. 이러한 감사 로그는 외부자로부터 보호되어야만 하며 모든 보안 관련 사건은 기록되어야만 한다.

3.2.6 감사로그 축약(Reduction)

감사에 대한 가장 큰 문제점은 그 양과 분석이다. 감사 기록이 실행되는 모든 명령과 파일에 대하여 감사기록이 생성되면 로그의 양은 엄청날 것이다. 이는 시스템의 성능저하와도 연관이 된다. 감사기록이 방대해 지면 감사기록에서 침입의 흔적을 발견하는 것이 어렵게 되므로 감사 데이터의 양을 줄이는 별도의 도구를 사용하여 감사 축약을 수행한다.

이밖에도 암호화, 안전한 네트워크 서비스를 위한 다양한 보안 요구사항이 존재한다.

4. 리눅스 보안기술

리눅스는 인터넷 웹서버, 전자우편서버, 데이터베이스서버, 프록시 서버, 파일 서버, DNS 서버, FTP 서버, 침입차단시스템 등에 이르기까지 인터넷의 여러 분야에서 다양하게 활용되고 있는 반면에 앞에서 기술한 바와 같이 많은 취약성을 가지고 있는 것도 사실이다. 그러나, 리눅스는 많은 사람들의 테스트와 계속적인 버그 리포트로 안정성을 가지고 있으며, 공개된 무료 보안도구의 지원을 받고 있다. 본 장에서는 리눅스 보안과 관련한 제반사항 즉, 관리적 보안대책, 리눅스와 관련한 무료 보안 도구를 알아보고 안전한 리눅스 개발을 위한 바람직한 개발 방향에 대하여 고찰하였다.

4.1 리눅스 보안설정 방법

시스템 보안의 기본 사항은 자신의 시스템이 어떠한 서비스를 수행하는지, 그리고 어떠한 중요 파일이 있는지를 파악하고, /var/log/messages 등의 로그를 주기적으로 감시하는 것이다. 또한 항상 최신 버전의 소프트웨어를 설치하도록 하여 최소한의 보안을 유지하는 것이 중요하다. 이와 더불어 다음의 보안 관련 여러 설정 파일들과 몇 몇 유용한 명령들을 이용하여 리눅스 시스템의 보안을 강화할 수 있다[1].

4.1.1 시스템 관련 보안 설정

/etc/security	루트가 접속할 수 있는 터미널을 표시. 레드헷 리눅스에는 지역 가상 콘솔(local virtual consoles: vtys)만을 기본 값으로 가지며 다른 원격 콘솔에서는 root로 접속하지 못하도록 한다
/etc/fstab	SUID/SGID 파일을 제한한다. 일반적으로 /var 파티션을 포함 사용자의 홈 파티션에 "nosuid" 옵션을 설정한다
/etc/exports	와일드카드를 사용하지 않도록 하며, 가능한 읽기 전용으로만 마운트하도록 한다.
/etc/profile	사용자의 파일생성 umask를 지정하며, 가능한 제한된 값으로 조정한다. 자주 쓰이는 값은 022, 033이고, 가장 제한적인 값은 077이다
/etc/pam.d/limits.conf	파일시스템 제한을 설정할 수 있다. 기본값인 "무제한"이 아닌 다른 값으로 설정하여 파일시스템의 사용을 제한한다.
/etc/issue	telnet 등을 이용하여 시스템 버전 정보 유출을 방지한다.
/etc/pam.d/login	rhosts 파일을 사용하지 못하도록 한다 [10]
chattr(1)	파일에 추가만 가능하도록 하고 지우거나 수정할 수 없도록 한다. 이는 /etc/passwd나 /etc/shadow와 같은 중요한 파일을 지우거나 덮어쓰는 것을 방지하도록 하여 많은 공격을 방지할 수 있다.

4.1.2 네트워크 관련 보안 설정

네트워크 서비스는 주로 /etc/inetd.conf, /etc/services, /etc/rc.d/rcN.d(N은 시스템 실행수준을 표시), 그리고 TCP-Wrapper의 설정 파일인 /etc/hosts.allow, /etc/hosts.deny 파일로 제어가 가능하다.

이밖에 최근 많이 발생하고 있는 리눅스 관련 버퍼 오버플로우를 방지할 수 있는 방법은 참고문헌 [11][12][13] 등을 참조하기 바란다.

4.2 리눅스 보안도구

리눅스 보안을 위한 보안 도구로는 무료로 공개된 것과 상용제품 두 가지가 있다.

4.2.1 공개도구

- Tcp Wrappers : 접속에 대한 필터링 기능으로 TCP Wappers는 사전에 등록해 놓은 설정 파일(/etc/hosts.allow, /etc/hosts.deny)을 보고 접속에 대한 권한이 있는지를 검사한다.
- Trinux(Linux Security Toolkit) : 리눅스 부트 디스크 보안 패키지로 네트워크 보안 도구를 가지고 있어 TCP/IP 네트워크를 관리하고 모니터링한다. 현재 v0.45로서 계속 개발중이며 궁극적으로는 침입탐지시스템, 모니터링, 스캐너(Scanner), 침입차단시스템 등 모든 네트워크 보안 기능을 포함하도록 개발할 예정이다.
- Crack, John The Ripper : 패스워드를 깨는 프로그램으로 사전에 있는 모든 단어와 그 변화형을 패스워드로 시도한다. 최근에는 사전뿐만 아니라 패스워드 파일에 있는 정보까지 유추하는 형태로 발전하였다. John The Ripper는 기능은 Crack과 유사하나, Crack보다 빠른 프로그램이다.
- Tripwire : 해커가 시스템에 침입했을 경우 파일의 손상여부 및 트로이 목마 프로그램 등 시스템을 스캔해 파일의 무결성을 점검해 준다.
- COPS(Computer Oracle and Passwork System) : 알려진 취약점에 대한 시스템의 보안상 문제점을 검사해주는 도구이다. 시스템 관리자에게 취약성을 알려주고 일부는 스스로 수정하기도 한다. Tiger도 이와 유사한 기능을 한다.
- SATAN(Security Administrators Tool for Analyzing Networks) : 네트워크 스캐너 프로그램으로 웹 인터페이스를 가지고 있다. 원격지 컴퓨터의 가능한 모든 포트에 연결하려고 시도하며, 어떤 서비스가 그 곳에서 수행되고 있는지를 찾을 수 있다. ISS(Internet Security Scanner)는 또 다른 스캐너로서 SATAN보다 빠르지만 제공하는 정보는 적다.
- Secure Shell(SSH) : rlogin, rsh, rcp, rdist 등에 대한 패킷 스니퍼링의 대책으로 사용될 수 있으며, 강력한 암호 및 인증기능을 제공한다. 이것은 man-in-the-middle attack과 DNS 스푸핑도 방지하며, 원격 호스트에 로그인하거나 호스트끼리 데이터를 복사하는 경우에 사용될 수 있다.
- PAM(Pluggable Authentication Modules) : 인증기능을 제공하며, 서비스 부인공격 방지에 사용된다. 레드햇에서 제공되며, 패스워드가 DES외의 암호화를 가능하게 하고, 사용자들이 쓸 수 있는 자원을 제한할 수 있다.
- 커버러스(Kerberos) : MIT 대학에서 개발한 커버로스에 대한 GNU 버전이 존재하며, 비밀키 방식의 인증기능을 제공하여 스푸핑을 방지하도록 하여 준다.
- qmail : 메일은 많은 취약점을 가지고 있다. 일반적으로 sendmail을 많이 사용하는데 이것은 많은 취약점이 알려져 있기 때문에 항상 최신의 sendmail 데몬을 유지해야만 하다. qmail은 이에 대한 해결책으로 기존 Sendmail의 보안 취약성을 제거한 mail 데몬이다.

- Fefe's finger daemon : 기존 finger 서비스의 취약성을 제거한 finger 데몬이다. 이 데몬은 루트로서 수행되지 않으며 사용자에 대한 많은 정보를 노출시키지 않는다. 또한, 보여주는 사용자 정보도 조정할 수 있다.
- lsof : 리눅스 시스템의 모든 open 파일의 리스트를 보여주는 것으로 서비스 거부 공격 및 로그 파일 수정에 대한 감지가 가능하다.
- pidentd : identd는 inetd에서 수행되는 작은 프로그램으로 어느 사용자가 어떤 tcp 서비스를 수행하는지 추적하고, 요구하는 누구에게든 추적 결과를 보고한다. pidentd는 일반적인 identd보다 설정하기 쉽고 보안기능이 향상된 ident 데몬이다.
- rhosts.dodgy : .rhosts 파일에서 주의해야 할 점을 검사해주는 도구이다.
- VPNd : 리눅스에서 TCP/IP VPN 서비스를 제공하는 패키지로써, 인터넷과 같이 안전하지 않은 라인을 통해 두 네트워크를 연결할 때 두 네트워크로 이루어진 하나의 논리적인 네트워크를 만들 수 있도록 해준다. 두 네트워크 사이의 데이터는 Blowfish 알고리즘을 가지고 암호화된다.
- Psionic software : 호스트 기반의 로그인 부정 행위 탐지 및 대응을 위한 도구인 HostSentry, 시스템 데몬이나 보안 도구들의 로그 파일의 처리를 도와주는 도구인 Logcheck 및 포트 스캔을 탐지하는 PortSentry 등의 무료 패키지를 제공한다.

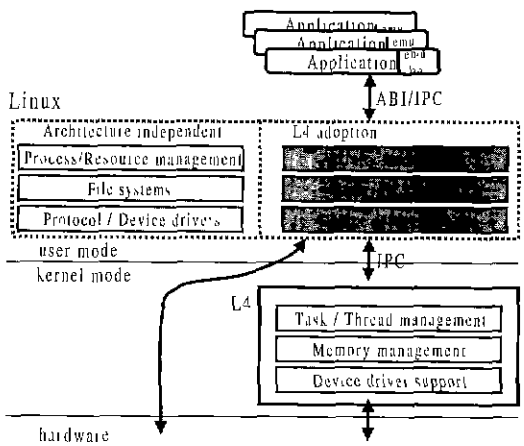
이밖에도 암호화된 파일시스템을 지원하여주는 CFS(Cryptographic File System), SSL, Shadow Password, Log Scanner(TSI), TIS Internet Firewall Toolkit(FWTK) 및 IP Port Scanner인 IPPS 등이 있다. 또한, 상용제품으로 개발되어 판매되는 것으로는 다음과 같은 것이 존재한다.

- ISS에서는 System Scanner 및 Internet Scanner를 판매하고 있다.
- Network Associates에서는 시스템, 네트워크 스캐너인 CyberCop Scanner, 사용자가 간단하게 보안 취약성을 테스트할 수 있는 스크립트 언어를 만들 수 있는 도구인 CyberCop CASL을 판매하고 있다.
- Aventail에서는 IP 네트워크를 통해 응용 프로그램과 데이터를 공유하기 위하여, 사용하기 쉽고 안전한 보안 솔루션인 Aventail ExtraNet Center를 판매하고 있으며, 이 제품은 강력한 암호, 사용자 기반의 인증, 접근통제, 강력한 관리자 도구 등이 통합되어 있다.
- Progressive Systems에서는 리눅스용 Phoenix Firewall을 판매하고 있다.
- 노르웨이 Guardian Networks Secure Linux사에서는 IEEE POSIX.6에 따라 secure Linux를 개발하고 있으며, root 권한을 획득 방지를 위한 Capability를 개발하였으며, 향후 감사추적, 보안레이블, 강제적 접근통제, 접근통제리스트 등을 구현할 예정으로 있다.

4.3 마이크로 커널 기반 안전한 리눅스 개발 방법

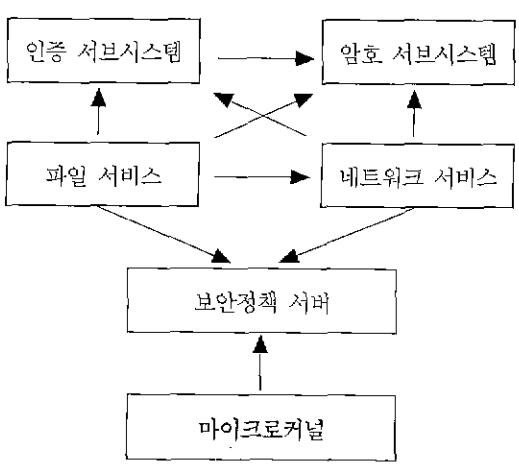
운영체제 보안에 대한 기술 개발을 활발하게 진행중인 미국의 안전한 운영체제 개발 현황을 보면, DTMach 보안 커널, Synergy 보안 커널, Flask 보안 커널 연구 등 주로 마이크로 커널을 기반으로 하는 안전한 운영체제 개발에 초점이 맞추어지고 있음을 알 수 있다[13]. 마이크로 커널은 강력한 신뢰성, 유연성 등을 보장해 주며 설계 및 구현시에도 통합 커널에 비해 용이하므로 향후 마이크로 커널을 이용한 보안 커널 연구 개발이 활발해질 것으로 예측된다. 본 절에서는 리눅스의 마이크로 커널인 L4에 기반한 안전한 운영체제 개발 방법에 대하여 소개하도록 하겠다[14].

L4 마이크로 커널은 독일의 드레스덴 대학에서 개발한 것으로, 원래 486과 Pentium 구조를 위하여 개발되었으나, 현재 Digital의 Alpha 21164와 MIPS R4600의 실험적인 구현도 존재한다[7]. L4 마이크로 커널이 탑재된 기본 시스템 구성도는 (그림 2)와 같다.



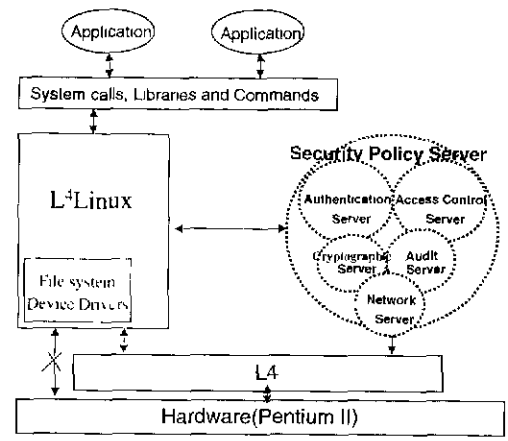
(그림 2) L4 마이크로 커널 기반의 리눅스

상기와 같은 환경하의 마이크로 커널 위에 3장에서 언급한 운영체제의 안전성 특징이 추가된 보안서버를 구현하여 탑재시킨다(그림 3).



(그림 3) 보안서버의 개념적 구성

보안 서버는 인증 서버, 접근통제 서버, 감사추적 서버, 암호화 서버, 안전한 네트워크 서버 등이며 접근통제 서버는 MAC 및 RBAC으로 별도로 구성할 수 있다(그림 4). 이 환경에서 파일 시스템은 L4Linux에만 존재하며, 보안 관련 서버는 필요 시 L4Linux의 파일 시스템을 사용토록 한다. 보안서버의 관리를 위한 보안 응용 프로그램이 별도로 존재할 수 있다.



(그림 4) L4 마이크로 커널에 기반한 보안서버 구조

일반적으로, L4 마이크로 커널에서는 ipc, ip_nearest, fpage_unmap, thread_switch, lthread_ex_regs, thread_schedule, task_new 등 7가지의 시스템 호출을 제공하며[12], 마이크로 커널과 리눅스 서버 또는 새로이 탑재될 보안서버간의 통신은 모두 ipc로 가능하므로, 이 ipc를 이용하여 보안서버를 통한 사용자 인증, 접근통제, 감사추적, 암호화 및 안전한 네트워크 서비스 제공이 가능하다. 예를 들어, 사용자의 시스템 자원에 대한 접근은 다음과 같은 시나리오에 의하여 통제될 수 있다.

- 사용자 응용 프로그램에서 시스템 자원, 파일, 데이터와 같은 객체에 접근을 요청한다.
- L4Linux 서버에서 보안정책 서버에 시스템

에 설정된 보안정책에 적합한지를 묻는 토큰을 ipc를 통해 보안정책 서버에 보낸다. 보안서버에의 접근 허가를 거치지 않고 직접적으로 마이크로 커널이나 하드웨어와 통신할 수 없도록 ipc 메커니즘을 수정한다.

- 보안정책 서버는 설정된 보안정책에 따라 접근여부에 대한 권한을 확인하고 허가/거부 토큰을 다시 ipc를 통해 L4Linux로 보낸다.
- L4Linux는 L4 마이크로 커널과 접근 요구된 객체를 접근하기 위하여 ipc를 이용하여 L4와 통신하고 해당 객체에 대한 접근을 시작한다.
- 위와 같은 기록은 감사추적 서버에 의하여 로깅된다.

5. 결 론

리눅스 운영체제는 저렴한 비용과 소스의 공개, 다양한 유틸리티의 지원 등의 장점으로 인해 최근 급속도로 사용이 늘고 있으나 이에 비례하여 리눅스에서의 해킹 사고 또한 증가하는 추세에 있다. 현재 제공되는 리눅스의 해킹사고 방지 대책은 취약성 또는 해킹사례가 보고될 때마다 관련부분을 수정한 패치버전을 사용할 뿐 보다 근본적인 방지책 마련에는 소홀함이 있다. 기존의 리눅스에 구현되어 있는 보안 기능이나 응용 프로그램을 통한 안전성 구현은 보안기능을 우회한 해킹의 우려가 있으며, 보안기능 자체가 Tamperproof하지 못하다는 한계가 있으므로 운영체제 자체에 보안 기능 추가는 필수적이라 할 수 있다.

이미, 미국, 유럽 등에서는 운영체제의 안전성을 보장하기 위한 기술개발을 1970년대부터 시작하여 많은 상용의 안전한 운영체제를 개발하여 판매하고 있으며, 리눅스 커널 자체를 안전하게 만드는 안전한 리눅스 개발도 이미 시작하고 있는 상황이다. 이를 이용하여 안전한 전자상거래, 안전한 웹환경 구축, 안전한 전자우편 서비스 제공뿐 아니라 전통적으로 보안이 강조되는 정부 및 군에서도 활용할 목적으로 막대한 예산을 투자하고 있는 실정이다. 특이할 만한 사항은 강력한 신뢰성, 유연성 등을 보장해 주는 마이크로 커널을 기반으로 하는 연구에 많은 관심을 기울이고 있다는 사실이다.

본 고에서는 리눅스의 보안 취약성을 고찰하고, 취약성 방지를 위하여 현재 활용가능한 보안 제품을 소개하고 근본적인 보안 취약성을 방지하기 위한 방법인 안전한 운영체제 개발방향에 대하여 고찰하였다.

리눅스의 성장 속도를 감안할 시 2000년대초에는 중·대형 서버시장의 큰 부분을 차지할 뿐 아니라 전자상거래 등 비즈니스 분야에서도 일정 부분을 차지할 것으로 예측되고 있다. 따라서, 리눅스의 안전성 강화는 필수 불가결할 것이며, 응용 프로그램 차원의 보안이 아닌 운영체제 수준에서의 보안이 반드시 필요하리라 사료되며, 이와 병행하여 보안 커널이 전체 시스템에 미치는 부하분석 및 보안 커널의 성능향상에 대한 연구도 지속적으로 추진하여야 한다. 무엇보다도 중요한 것은 리눅스란 운영체제를 대상으로 안전한 운영체제의 개발 타당성에 대한 학계, 정부, 관련 연구기관 종사자의 인식전환이 필요하고 전폭적인 지원이 필요하리라 사료된다.

참고문헌

[1] 임채호, "최근 해킹기법 분석과 대응책", 제4회 정보보호심포지움, 1999. 4.

[2] http:/www.linux.org/info/

[3] R. Magnus, U. Kunitz et al, Linux Kernel Internals, 2nd Ed., Addison-Wesley, 1999.

[4] http://www.redhat.com/corp/support/errata

[5] 천우봉외 3인, "효과적인 사고대응을 위한 해킹공격방법 분류", 제11회 정보보호와 암호에 관한 학술회의, 1999. 9.

[6] Dennis Longley, Data & Computer Security : Dictionary of Standards concepts and terms, 1989.

[7] Chales P. Pfleeger, Security in Computing, 2nd Ed., Prentice Hall, 1997.

[8] DoD, Trusted Computer System Evaluation Criteria", 1985.

[9] 이철원, 이병각외 4인, "역할속성을 이용한 역할기반 접근통제 메커니즘", 통신정보보호학회 논문지 제8권 제4호, 1998.

[10] http://www.kernel.org/pub/linux/libs/pam/index.html

[11] ftp://ftp.cac.washington.edu/mail/imap.tar.Z

[12] ftp://ftp.qua;comm.com/Eudora/servers/unix/popper

[13] http://www.isc.org/pub/new-bind.html

[14] 이철원, 이병각, 김홍근, "L4기반 Secure Linux", 제11회 정보보호와 암호에 관한 학술회의, 1999. 9.

이 철 원

1987년 충남대학교 수학과(이학사)
1989년 중앙대학교 전자계산학과 (이학석사)
1989년-1996년 한국전자통신연구원 선임연구원
1996년-현재 한국정보보호센터 시스템기술팀 선임연구원
관심분야 : 컴퓨터 및 네트워크 보안, 주요기반구조 보호, 정보보호시스템 평가기준

김 홍 근

1985년 서울대학교 컴퓨터공학과 (공학사)
1987년 서울대학교 컴퓨터공학과 (공학석사)
1994년 서울대학교 컴퓨터공학과 (공학박사)
1994년-1996년 한국전산원 선임연구원
1996년-현재 한국정보보호센터 시스템기술팀장
관심분야 : 컴퓨터 및 네트워크 보안, 병렬처리

박 태 규

1980년 경북대학교 전자전산기 공학과(공학사)
1989년 충남대학교 전산학과 (이학석사)
1996년 성균관대학교 정보공학과 (공학박사)
1981년-1982년 한국국방연구원 연구원
1982년-1992년 한국전자통신연구원 선임연구원
1996년-1997년 University of Western Sydney(Post-doc)
1992년-현재 한서대학교 컴퓨터과학과 부교수
관심분야 : 컴퓨터보안, 운영체제 보안