情 報 保 護 學 會 誌
第 13 券 第 4 號, 2003. 8

보안 리눅스(Secure Linux) 연구개발 동향

박 태 규*

요 약

응용 수준에서 정보보호를 위한 침입차단시스템(Firewall)과 침입탐지시스템(IDS)은 조직 내의 컴퓨터 서버 보안 대책으로는 그 한계를 갖고 있다. 이에 따라 보안 운영체제(Secure OS)에 관한 필요성이 점차 사회적으로 공감대를 형성하고 있다. 본 고에서는 보안 리눅스 운영체제의 필요성, 기존 리눅스의 보안성, 보안 리눅스의 개발에 따른 요구사항과 개발 방법을 기술한다. 또한 최근 보안 리눅스 연구 동향으로 미국, 일본, 독일 등의 리눅스 보안 연구동향을 살펴보고, 국내 연구기관과 업체의 제품 출시 현황을 살펴본다. 특히 최근 리눅스 커널 2.5.29부터 표준기능으로 포함되고 있는 커널 보안 모듈 방식인 LSM(Linux Security Module)의 기본 구조를 살펴본다. 현재 국내에서 개발하여 보급되고 있는 보안 리눅스 운영체제는 기존 리눅스 커널에 시스템 호출 후킹을 통한 LKM 방식으로 추가적인 접근제어 외에 해킹 차단, 감사 추적, root의 권한 제한, 통합보안관리 등의 추가적 기능을 제공한다. 향후 Firewall, IDS의 한계를 보완하는 서버 보안 대책으로 활발한 보급이 예상된다.

Ⅰ. 서 론

1991년 최초의 리눅스 버전이 공개된 이래 리눅스 운영체제는 십여 년이라는 비교적 짧은 기간에도 불구하고 매우 빠른 속도로 발전해 왔다(1977년 최초의 상용 UNIX 발표, 1985년 Windows 1.0 발표). 리눅스는 소스 코드의 공개, 저렴한 구축비용, 벤더 독립적인 하드웨어 지원, 점차적으로 다양해지는 응용 프로그램의 보급에 힘을 입어, 현재 국내·외적으로 웹 서버, 응용 업무용 서버, 연구 개발용 서버 등의 서버 분야는 시장에서 비교적 성공적으로 그 인기를 증명하고 있다. 이러한 빠른 보급으로 십대라는 비교적 어린 나이인 리눅스는 사용자나 응용 프로그램 개발자들에게 강한 어필을 하고 있으나 이에 따른 보안 위협도 급증하고 있는 현실이다.

보안은 시스템이 증가되면 될 수록 공격에 대한 동기가 덩달아 발생되는 고질적이고 끊임없이 일어나는 문제이다. 리눅스는 이러한 위협에 아직 면역이 없으며, "눈(目)이 많으면 많은 버그를 보게 된다."는 말에 견디질 못하고 있다. 리눅스 시스템은 많은 소프트웨어 상의 취약성을 경험하고 있다.

이러한 소프트웨어 상의 취약성을 제거하는데 있어 가장 중요한 방법은 접근제어의 효과적인 사용을 통하는 것이다. 임의적 접근제어(DAC: Discretionary Access Control, root, user-IDs, mode bits)는 사용자 자신들의 비밀성(Privacy) 관리에는 적합하나 해킹 공격에 대하여 시스템을 방어하기에는 충분하지 못하다. 이러한 비-임의적 접근제어(Non-discretionary Access Control) 모델에 대한 집중적인 연구는 30년[12,14,15,26,30]이 넘게 진행되어 오고 있으나, 하나의 진정한 접근제어 모델에 대한 실질적 합의는 아직 없다. 이러한 합의 부재로 인하여 접근제어를 개선하려는 많은 리눅스 커널 패치(patch)가 나오고 있으나, 그 중 어떤 것도 리눅스 커널의 표준이 되질 못하고 있는 실정이다. 리눅스에서 제공하는 접근제어와 로그를 생성하는 감사 기능 등 보안기능은 미국의 컴퓨터 신뢰성 평가기준인 TCSEC 기준으로 C2급에 해당한다. 다소의 강력한 보안 시스템이 연구용 시제품 혹은 매우 전문적인 제품으로 구현되어왔지만 시스템 관리자 및 일반 사용자에겐 아직도 어려운 도전으로 남고 있다. 즉, 기존의 시스템에서 어떻게 새로운 보안 기능들을 활

* 한서대학교 교수(tkpark@hanseo.ac.kr)

용할 것인지를 쉽게 결정하지 못하고 있다. 특히 이러한 문제점을 해결하기 위해서 2001년 미국 NSA의 Peter Loscocco가 리눅스 커널 정상회의(summit)에서 2000년 NSA를 중심으로 NAI Lab, SCC (Secure Computing Co.), MITRE가 공동으로 수행한 SELinux 연구 프로젝트의 결과를 리눅스 커널에 반영할 것을 제안 발표함으로써 LSM 프로젝트[10]가 태동하게 된 씨앗을 뿌리게 되었다. 이후 리누스 토발즈가 이 LSM 제안을 받아들임으로써 Crispin Cowan과 WireX 사가 중심이 되어 LSM 프로젝트를 본격적으로 시작하게 되었다. 현재 이 프로젝트를 위한 메일링 리스트(lsm.immunix.org)가 활발히 운영되고 있다.

본고에서는 2장에서 운영체제 차원에서의 보안의 필요성, 3장에서는 기존 리눅스에서 발생되고 있는 보안 취약성 현황을 분석해 보고, 4장에서 이러한 보안 문제를 해결하기 위해서 보안 리눅스(Secure Linux)를 어떻게 개발할 것인가에 대한 요구사항과 5장에서 그 개발 방법에 대하여 설명한다. 6장에서는 국내·외에서 현재 보안 리눅스를 위해서 어떤 연구와 제품들을 내놓고 있는지를 알아보고, 7장에서는 이제부터 리눅스 커널 수준에서 보안성을 제고함에 중요한 역할을 담당하게 될 LSM 구조의 기본 원리를 분석하며 기존 커널에 미치는 성능 부하율과 보안성을 기술하고자 한다.

Ⅱ. 보안 리눅스의 필요성

인터넷 환경에서 최근의 보안 대책은 주로 네트워크 응용 수준의 보안 대책인 침입차단시스템(firewall), 침입탐지시스템(IDS, Intrusion Detection System)으로 이루어지고 있다. 그러나 이러한 응용 수준에서의 보안 대책은 근원적으로 중요한 정보를 담고 있는 인트라넷 내부 서버에 대한 보호를 위해서는 한계를 가지고 있으므로 충분한 보안 대책이 될 수 없다. 따라서 이러한 응용 수준에서의 보안 한계를 극복하기 위해서는 보안 운영 체제(Secure OS)의 활용이 서버 보안 측면에서 기본적인 전제가 되어야만 한다. 인터넷을 구성하는 기본 요소인 컴퓨터 시스템의 운영체제 자체가 기본적으로 보안상 취약성을 내포하고 있는 상태에서는 응용 수준의 어떠한 보안 대책도 사상누각을 쌓는 결과를 초래할 것이라는 지적들이 이미 제기 되었으며[22,24,25], 침입차단시스템과 침입탐지시스템, 보안 관제시스템 등을 갖추고도 불법적인 해킹을 당하여 중요한 정보들이 파괴, 유출되는 사건들이 빈번히 발생하고 있다[23,27,28].

운영체제의 보안성 평가의 잣대는 미 국방성 표준 규정(DoD 5200.28-STD)인 신뢰성 컴퓨터시스템 평가기준(TCSEC, Trusted Computer System Evaluation Criteria)과 ISO 국제공통평가기준인 CC(Common Criteria)의 Operating System 관련 Protection Profile[21]이 될 수 있다.

리눅스 운영체제의 보안성은 TCSEC 기준으로 볼 때 C2급(사용자 식별 인증, 임의적 접근제어(DAC), 객체 재사용 방지, 감사 추적 등의 기능)에 해당되는 보안 기능을 갖고 있다[11,15]. 즉, DAC의 내재된 결점으로 인한 트로이 목마와 해킹에 대하여 취약한 보안성의 문제를 안고 있다. 따라서 리눅스를 중요한 정보처리용 서버로 안전하게 사용하기 위해서는 C2급의 한계를 보완하여 상위 B1급 수준 이상의 운영체제로 개발하고자 하는 요구가 제기되어 왔고, 관련 연구와 제품들이 등장하고 있는 상황이다.

Ⅲ. 리눅스의 보안성

리눅스의 취약성을 분류해보면 사용자 인증 시의 패스워드 공격, 버퍼 오버플로우 및 경주 조건(Race Condition) 공격 등을 통한 root 권한탈취 공격, 파일 소유자의 임의적 판단에 의한 접근제어, 데몬 공격, 서비스 거부 공격(Denial of Service), 바이러스 및 웜 감염, IP 스푸핑, 패킷 스니퍼링 등으로 다양한 유형의 취약점을 지니고 있다[19].

리눅스와 타 운영체제의 보안성을 비교함에 있어서 일부 사람들은 공개 소스인 리눅스가 본질적으로 비공개 운영체제보다 더 안전하다고 주장하고 있고[1], 또 다른 일부는 그렇지 않다고 주장하고 있다[2]. 두 경우 모두 한 개의 동전의 양면과 같아서 절대적으로 한쪽의 주장이 옳다고 할 수는 없다. 공개 소스는 공격자와 방어자 모두에게 소프트웨어의 취약성에 대한 위협과 보안 대책을 마련할 기회를 제공함으로써 비공개 소스보다는 분명 더욱 많은 분석 기회를 갖는다. 공격자에게는 새로운 공격 수단(exploit)을 만들도록 하며 방어자에겐 보안의 수준을 더욱 높일 수 있도록 해준다. 만약 방어자가 보안에 대한 어떤 노력도 하지 않는다면, 비록 공개 소스라 할지라도 그러한 이점을 공격자에게 줘버리는 꼴이 된다. 그러나 또한 공개 소스는 비공개 소스 소프트웨

어라면 불가능한 보안 대책에 관한 연구를 가능케 하는 등의 많은 장점을 방어자에게 제공한다. 그리고 비공개 소스는 사용자들에게 소프트웨어 벤더가 제공하는 보안 설정 수준을 그대로 수용하도록 강요하지만, 공개 소스는 사용자로 하여금 그들이 원하는 만큼의 보안 설정 수준을 선택하도록 해준다[3]. 그러나 비공개 소스에 의도적인 백-도어, 트랩도어와 같은 악성 코드가 삽입되는 경우에는 노출될 기회가 없기 때문에 잠재적인 보안 취약성은 심각한 문제가 될 수 있다. 그 좋은 예가 백-도어가 내재되어 있는 것으로 의혹을 받고 있는 마이크로소프트의 Windows가 될 것이다. 2000년 7월 8일자 뉴욕타임즈는《중국, 마이크로소프트사의 힘을 꺾고 있다》라는 제목으로 다음과 같은 기사를 보도하였다. 『Chen Chong 중국 정보산업부 부부장은 "중국에서는 은행에서부터 개인의 전자우편 박스까지 모든 것을 통제하는 Windows 운영체제에 대한 의존도가 급속히 높아지고 있다. 그러나 중국 정부는 마이크로소프트가 소프트웨어 시장을 독점하길 원하지 않는다. 리눅스로서 중국은 보안을 통제할 수 있으며, 중국 자신의 운명을 통제할 수 있다. 어떤 사람들은 마이크로소프트의 컴퓨터 코드 내에 비밀 구멍들은 미국으로 하여금 중국의 네트워크에 접근할 수 있도록 하며, 전쟁 시에 중국의 네트워크를 다운 시킬 수도 있다고 경고한다. 이러한 관심은 미국에서 일하고 있는 캐나다 소프트웨어 회사의 한 암호 기술자가 1999년에 자신이 Windows 운영체제에서 NSAKey라는 백-도어를 발견했다고 말함으로써 고조되었다.』이와 관련하여 중국에서는 리눅스를 국가적 운영체제로서 개발하고 있다[5].

〔표 1〕은 한국정보보호진흥원 CERTCC-KR에서 분석한 2000년도부터 3년 반 동안의 운영체제별 해킹통계분석 자료이다. 이 보고서에 따르면 보고 된 해킹 건수가 리눅스의 경우 2000년 595건, 2001년 1,505건, 2002년 575건, 2003년 6월(반기)까지 272건으로 3년 반 동안 총 2,946건으로 Windows 95/98(6,426건)에 비하여 46%로 나타나고 있다.

〔표 1〕 국내 CERTCC-KR의 운영체제별 해킹 건수 현황

운영체제	2000년	2001년	2002년	2003년 (6월까지)	합계
Linux	595	1,505	574	272	2,946
Solaris	75	172	90	14	351
Windows 95/98	776	1,042	2,098	2,510	6,426
Windows NT	27	987	2,279	4,479	7,772
HP-UX	2	3	12	4	21
DEC	1	2	3	0	6
CISCO	2	4	2	0	8
합계	1,478	3,715	5,058	7,279	17,530

따라서 리눅스와 같은 공개 소스의 경우에 보안 취약성에 대한 적극적인 대책이 마련되고, 사용자의 적절한 보안 관리가 유지된다면 Windows 등과 같은 비공개 소프트웨어에 비하여 더욱 높은 보안성을 유지할 수 있을 것이다.

Ⅳ. 보안 리눅스 운영체제 요구사항

보안 리눅스를 개발함에 있어 중요하게 고려하는 2가지 요구사항은 설계를 위한 일반적 요구사항과 구현을 위한 기능적 요구사항이다. 설계를 위한 요구사항은 일반적인 보안 시스템 개발에서 요구하는 것과 유사하게 고려될 수 있으며, 구현시의 기능적 요구사항은 중요한 정보에 대하여 강제적 접근제어를 수행하는 기능과 리눅스 운영체제 자체에서 잠재적으로 가지는 해킹 취약성을 방어하는 해킹방지 기능이 필수적으로 요구된다.

1. 보안 운영체제의 설계 요구사항

보안 운영체제를 구현하기 위해서 설계 시에 OS가 유지하는 정보에 대한 기밀성, 무결성, 가용성을 보장하도록 하여야 한다[19,26]. 따라서 다음과 같은 요소를 고려하여야 한다. ① 특권의 최소화(Least of Privilege) : 부적절하거나 악의가 있는 공격으로부터 피해를 최소화하기 위해 root를 포함한 사용자와 프로그램은 가능한 최소한의 권한을 사용하여 운영되어야 한다. ② 메커니즘의 경제성 : 보호 시스템의 설계는 작고 검증이 가능하고 처리가 빨라야 한다. 그 이유는 보안성에 대한 완전한 분석, 시험, 검증이 가능하며, 부가적인 처리로 인한 성능상의 오버헤드가 최소가 되어야 하기 때문이다. ③ 완전한 중재 : 모든 접근 시도는 통제되어야 하고, 우회하려는 어떠한 시도도 접근 통제가 가능해야 한다. ④ 허가기반 : 기본적으로 접근은 거부되어야만 한다. 일반적으로 설계 시 접근불가 항목을 정의하기보다는 접근가능하게 설계하는 오류로부터 벗어나야 한다. ⑤ 특권의 분리 : 이상적으로, 객체(Object)에 대한 접근은 사용자 인증과 보안등급, 또는 암호

키 등과 같이 한 가지 이상 조건들을 사용하여야 한다. 이러한 방식으로 보호 메커니즘을 우회한 프로세스도 보호하고자 하는 대상에 접근하지 못하게 한다. ⑥ 사용의 용이성 : 사용자 및 보안 관리자가 쉽게 사용하고 쉽게 보안 설정 사항 등을 통제하며 로그 정보를 이용함으로써 보호 메커니즘을 우회 할 가능성을 제거한다.

2. 보안 운영체제의 기능적 요구사항

기존의 리눅스는 TCSEC C2 수준으로 보안성의 한계를 가지므로 이를 기반으로 한 단계 이상의 보안 기능적 요구를 만족하는 연구 개발이 대두되어 왔다. 또한 전통적으로 군사적 목적 및 비밀 수준의 중요 정보를 보호하기 위하여 적용되었던 보안 운영체제 개발은 B1급 이상의 요구사항을 적용하고 있다. 다시 말해서 CC의 EAL4 등급이상의 MLS(Multi-Level Security) OS Protection Profile이 적용되고 있다. 부가적으로 TCSEC의 요구사항에 맞게 구현된 시스템은 시스템 내부의 중요한 정보를 효율적으로 보호하였지만, 시스템의 성능을 저하시키는 하나의 요인이 되기도 한다. 따라서 이러한 성능 부하율을 최소화 하도록 설계되고 구현되어야 한다. 한편 최근 들어 폭발적으로 증가하는 인터넷 사용 추세를 감안할 때 보안 운영체제에 대한 필요성은 급증할 것으로 예측되는바, 군사용, 금융기관 등의 중요 정보처리용과 일반사용자를 위한 공통적인 보안 운영체제의 개발이 필요하며, 이러한 요구사항을 바탕으로 보안 리눅스를 개발하기 위한 기능적 요구사항은 사용자 식별 및 인증, 강제적 접근제어, 레이블 보안, 감사 추적, 보안 관리, 해킹 방지 등으로 크게 정의하고 특수한 기능들은 사용자가 선택적으로(optional) 적용할 수 있도록 하는 추세이다.

V. 보안 리눅스 운영체제 구현 방법

리눅스는 사용자 수의 증가 못지않게 커널 코드의 크기도 급격히 증가해 왔다. 1996년에 발표된 커널 버전 2.0은 커널 소스 코드가 C 언어 코드 50만 라인과 수 천라인의 어셈블리어 코드 분량이었고, 커널 버전 2.2에서는 전체 커널 소스 코드의 크기가 약 150만 라인 가량 되었다. 이러한 방대한 코드들이 여러 개발자에 의해 다소 원칙 없이 작성되는 과정에서 코딩의 일관성이 결여되고 일부 개발에 관련된 문서가 충분히 제공되지 않는 등 소스 코드를 이해하고 수정하는데 어려움이 따르는 것도 사실이다. 보안 리눅스 커널을 개발하는 방법은 크게 두 가지로 나눌 수 있다. 하나는 커널 수정 방식이고 다른 하나는 보안 모듈을 커널 속에 필요에 따라 적재/제거하는 방식인 동적 LKM(Dynamically Loadable Kernel Module) 방식이다. 리눅스와 같은 통합 커널(monolithic kernel 또는 integrated kernel)은 간단히 커널을 수정하는 경우나 일부 시스템의 구성을 바꾸는 경우에도 매번 전체 커널을 다시 컴파일 해야 하는 단점을 가지고 있다. 따라서 이러한 번거로움을 피하기 위해서 리눅스는 모듈이라는 개념을 제공하고 있다. 모듈은 실행 중에 커널에 동적으로 연결하거나 제거할 수 있는 오브젝트 코드를 말한다. 필요한 보안 정책에 따라 그 기능을 기존의 리눅스의 보안 기능에 추가적으로 수행(집행)하는 모듈을 구성해 필요한 경우에만 커널에 포함시켜 사용함으로써 선택적으로 보안 커널의 역할을 수행하도록 할 수 있다. 또한 새로운 보안 정책에 따른 기능을 개발하여 테스트를 원하는 경우에도 모듈로 구성함으로써 테스트 과정에서 여러 번 재 부팅 해야 하는 번거로움을 피할 수 있다. 시스템 운용 중에도 동적으로 보안 모듈을 사용하기 위해서는 사용자가 "insmod/rmmod"와 같은 명령어를 사용하여 필요할 때마다 명시적으로 추가, 제거할 수도 있고, 커널 데몬(kerneld)에 의해 필요시마다 자동적으로 추가와 삭제도 할 수는 장점이 있다[4]. 예를 들어 NSA의 SELinux[29]와 NPS의 Policy-Enhanced Linux[27]는 TCSEC B1 기능을 갖도록 한 커널 소스 수정방식으로서 리눅스 커널의 Version-up, Patch 등이 발생할 경우에는 구현된 보안 기능의 재개발이 필요하며, 기존 응용 프로그램의 호환성이 떨어질 뿐만 아니라 보안 운영체제 설치 시에는 서버를 정지시켜 서 커널의 재 컴파일을 행한 후, 재부팅이 불가피하다는 문제점이 발생되게 된다. 특히 리눅스 커널의 변경(Version-up, Patch)이 다른 운영체제에 비하여 빈번한 점을 고려하면 보안 기능의 재개발에 따른 비용은 무시할 수 없을 만큼 크다 할 수 있다. 한편 LKM 방식은 다시 두 가지로 나누어지는데 하나는 시스템 호출 후킹 LKM 방식으로 리눅스 시스템 호출을 가로채기(Hooking 혹은 Interception)하여 원래의 서비스 루틴을 실행하기 전에 삽입되는 보안 커널 모듈에서 보안 체크를 한 후 원래의 서비스 루틴을 수행하는 방식이다. 다른 하나는

〔표 2〕 보안 리눅스 구현방식 비교

방식	구현 방법	장점	단점
커널 수정 방식	커널 소스 코드확보가 전제됨/커널 소스 분석 후 커널 수정/커널 재 컴파일 필요	고 수준 보안 OS 개발 가능(B2/B3등)/개발기간 단축 가능/개발이 비교적 용이함	OS 변경에 따른 빈번한 재개발(버전 갱신, 패치)/개발 비용 증대/호환성 유지 어려움
LKM 방식	시스템 호출 후킹/정의된 후킹 인터페이스/모듈 적재, 제거 가능	OS 변경에 비교적 독립/관리의 용이성(설치/유지보수 용이/호환성 유지 용이)	고도의 구현기술/고 수준 보안 OS 개발 어려움/한정된 인터페이스 사용

7장에서 소개할 LSM(Linux Security Module) 방식으로 리눅스 커널의 특정 포인트에서 여러 개의 인터페이스를 만든 후 이를 통하여 보안 모듈을 삽입하거나 삭제하도록 하는 방식이다. 〔표 2〕는 커널 수정 방식과 LKM 방식[13]에 대한 장단점 비교를 보여준다.

인터넷 또는 인트라넷 환경에서 일반적으로 24시간 서비스를 제공해야 하는 서버의 특성으로 보아 설치 및 유지보수를 위하여 서버의 동작을 중단시킨다는 점도 큰 문제가 아닐 수 없다.

LKM 방식의 이점은 첫째, 운영체제의 커널에 보안 모듈이 삽입되어 시스템 호출을 가로채기 때문에 어떠한 악성 코드라도 보안 커널을 우회하거나 피해갈 수 없다. 둘째, 기존 운영체제의 커널을 수정할 필요가 없다. 보안 커널 모듈은 운영체제의 실행 중에도 적재(loadable)와 삭제(removable)가 가능하기 때문에 운영체제의 소스 코드를 수정할 필요가 없다. 셋째, 다양한 보안 정책의 적용에 유연하다. 여러 가지의 서로 다른 보안 정책을 구현하는 보안 커널을 모듈화 구현하여 보안 정책에 따라 보안 모듈을 변경하여 사용할 수 있다. 넷째, 플랫폼에 비교적 독립적이다. 시스템 호출 자료구조에 접근하는 커널 모듈을 지원하는 어떠한 운영체제 플랫폼에서도 사용될 수 있다. 이 LKM 방식은 UNIX 계열의 운영체제와 Windows NT 등에도 적용이 가능하다. 다섯째, 어떠한 상용 소프트웨어(COTS, Commercial Off-The-Shelf)에 대해서도 그 소프트웨어의 수정 없이 투명하게(transparent) 동작의 감시가 가능하다는 점 등의 많은 장점들이 존재한다.

Ⅵ. 국내·외 연구 개발 동향

1. 국외 연구개발 현황

1.1 미국 NSA의 SELinux

SELinux 시제품은 1999년 9월부터 2000년 8월까지 NSA와 NAI Lab, SCC, MITRE와 공동의 노력으로 개발되었다. NSA 연구진들이 앞서 수행했던 MACH 마이크로 커널 기반의 Flask 보안 연구 결과를 리눅스 커널에 적용하기 위해서 주요 서브시스템 내에 보안 구조를 구현하였고, 타 공동 연구 기관들에 의하여 개량되었다. NAI Lab은 또한 응용 보안 정책 개발 시 NSA를 도왔으며, 시스템을 위한 각종 유틸리티를 개량하였다. 2001년 1월 2일 미국 NSA는 보도 자료를 통해서 그 연구의 취지를 발표하였다[6]. NSA는 리눅스 공동체와 협력하에 궁극적으로 리눅스에 포함시킬 목적으로 이 시스템을 개선하기 위해서 GNU GPL 측면에서 SELinux 시제품을 공개하고 있다. 이 시스템은 리눅스를 위한 완전한 보안 솔루션으로 개발되지는 않았으며 리눅스에 현존하는 어떤 허점을 수정하고자 함도 아니었다. 다만 NSA는 정보 보증(Information Assurance)의 한 부분으로서 운영체제 보안을 포함하여 다양한 컴퓨터 보안 관련 주제에 투자를 통하여 오랜 동안 컴퓨터 보안 연구 협의체와 관계를 지속하고 있다. 보안성을 좀 더 높은 수준으로 끌어올리기 위해서는 운영체제 보안 메커니즘이 중요하다는 인식하에, NSA 정보 보증 그룹 내의 연구진들은 다양한 컴퓨터 환경에서 보안 요구를 만족시킬 수 있는 방법으로 필요한 보안 기능을 제공할 수 있는 구조에 전념하였다. 운영체제 보안 메커니즘은 비밀성과 무결성 요구에 바탕을 둔 정보를 강제적으로 격리하는 기능을 시스템 보안에 제공하도록 해야만 했다. 운영체제 메커니즘만이 이러한 정보의 격리를 보장할 수 있는 기반이 된다고 생각하고 있으나, 불행하게도 기존 주류를 이루고 있는 운영체제들은 정보를 격리함에 필요한 중요한 보안 기능인 강제적 접근제어(MAC; Mandatory Access Control)가 없다. 결과적으로 응용 프로그램의 보안 메커니즘은 침입, 우회에 취약하며 악성 혹은 보안상 허점이 존재하는 응용 프로그램은 쉽게 시스템 보안에서 실패하게 된다는 것이다. 이러한 분야에서 종전의 몇몇 연구 프로젝트의 결과가 SELInux 시스템에

적용되었다. 이 리눅스 버전은 강하며 커널의 중요한 서브시스템 내에 포함된 유연한 강제적 접근제어 구조를 갖는다. 이 시스템은 비밀성과 무결성 요구사항에 바탕을 두고 정보의 격리를 집행하는 메커니즘을 제공하며, 이 방법은 침입의 위협과 응용 프로그램의 보안 메커니즘의 우회를 막을 수 있으며, 악성 혹은 보안상 허점이 존재하는 응용 프로그램에 의해서 발생될 수 있는 위험을 방지할 수 있다고 믿고, 이러한 작업을 위해서 리눅스가 플랫폼으로 선정되었다. 그 이유는 리눅스의 성장이 성공적이며 이러한 기능들이 널리 활용되는 시스템의 보안에 기여하고 동시에 다른 주요 운영체제에도 성공할 수 있다는 점을 시사할 수 있는 기회가 될 수 있기 때문이었다. 아울러 이러한 보안 연구를 종합함으로써 결과적으로 리눅스가 추가적인 운영체제 보안 연구를 자극할 수 있기 때문이었다. 이 연구는 super-user를 포함한 모든 프로세스의 행위를 제한할 수 있는 강제적 접근제어가 어떻게 리눅스에 추가될 수 있는지를 예로서 보여주었으며, 이 연구의 초점이 비록 보안 시스템에서 중요하긴 하지만 보안 감사와 같은 기타 보안 기능 혹은 시스템 보증에 맞춰지는 것은 아니었다.

이 시스템에서 구현된 보안 메커니즘은 다양한 보안 정책을 지원할 수 있도록 유연함을 제공하여 다양한 보안 요구에 맞추어 시스템을 구성할 수 있도록 하고자 하였다. 여러 가지의 보안 목표를 만족시킬 수 있게 설계된 범용 보안 정책 구성을 포함하고 있으며, 이 시스템의 유연성은 또한 정책을 수정할 수 있도록 해주며 보안 정책을 주어진 설치 환경에 필요한 보안 정책에 맞게 확장될 수도 있다. NSA는 SELinux를 완전한 보안 솔루션으로 만들기 위해서는 자원의 제약으로 말미암아 보안 메커니즘을 평가하고 성능을 최적화하지 못하고 있다고 한다. 현재는 x86 구조를 지원가능하며 RedHat 배포판에서 시험이 가능하며, SELinux는 원래의 소스 코드를 공개하고 있으며 수정 보완을 위한 참여를 유도하기 위해서 Mailing 리스트를 운영하고 있다.

1.2 일본 전자정부 프로젝트의 보안 리눅스

일본 정부 역시 보안 문제를 내세워 Windows 대신 전자정부 컴퓨터의 운영체제로 오픈소스 소프트웨어를 채택하기로 하고, 총무성이 전문가 패널을 구성해 다른 나라의 사례를 조사한 후 플랫폼을 리눅스로 채택한 것으로 알려지고 있다[7]. 리눅스 운영체제를 가지고 독자적인 보안대책 수립이 가능하고 장애가 발생해도 대처가 용이하다는 게 장점으로 부각되었다. 한편 2003. 4. 21에 발간된 〈日經컴퓨터〉, "전자"안전보장을 정비하라" 특집기사[8]에서 다음과 같이 보도하고 있다. 『오픈 소스로 추진하여 안전을 확보: 2003년 3월 4일~6일, 일본, 한국, 중국, 대만 등 아시아를 중심으로, 16개국/지역의 오픈소스 관계자 약 100명이 타이에 모였다. 이 자리에서 아시아 각국/지역의 커뮤니티의 연락조직의 설립이 합의되어, 오픈 소스의 보급에 협력해 가기로 선언했다.이 심포지엄을 시작한 것은 일본 경제산업성이었다. 정보처리진흥과가 기획하고, 외곽단체인 국제정보화협력센터가 주최했다. 이 회의에 마이크로소프트 일본법인이 참가하고 싶어했으나 경제산업성 측은 "오픈 소스 관계자만의 회의로 하고 싶다." 고 거절했다. 작년 가을쯤부터 정부는 빠르게 오픈소스에 주목해 왔다. 경제산업성은 2003년도 오픈소스관련으로 10억 엔의 예산을 획득했다. 총무성도 "시큐어 OS에 관한 연구회"라는 명목으로 2,400만 엔의 예산을 획득했다. 실제는 "전자정부에서 이용하는 관점에서, Linux의 안전성을 검증하는" 연구회이다. 이러한 움직임의 배경에는 안전보장이라는 관점이 있다. 총무성의 예산을 뒤에서 밀어 준 자민당의 河野 의원은 "적어도 일본정부가 조달하는 소프트웨어는 내용을 알 수 있어야 한다. 내용을 알 수 없는 것을 어떻게 안전하다고 말할 수 있겠는가. 리스크 분산이라는 의미로도 현재의 과점상태는 좋지 않다." 라고 주장한다. 정부는 지금 오픈소스 OS에 관한 기술평가를 하고 있는 단계이지만, 독자 OS도 생각하고 있는 것 같다. 경제산업성 정보처리진흥과의 久米孝 과장보좌에 의하면 "금년 예산을 확보한 프로젝트와는 별도로, 시큐어 OS의 개발연구에 착수할 가능성이 있다." 그 때에는 Linux가 베이스가 될 것으로 보여 진다. 방위청은 올 여름을 목표로 오픈소스OS 연구모임을 실시할 모양이다. 관계자에 의하면 "최종적인 목표는 국방에 관계되는 시스템에 사용할 독자OS의 개발에 있다."고 말한다. 한편, 지금까지 각 省廳에 맡겨두고 있는 암호의 조달에 대하여, 정부에서 통일적으로 기술평가를 하여 안전성을 확인하려는 시도도 시작했다. 총무성과 경제산업성의 암호평가 공동프로젝트 "CRYPTREC"이다. 그 결과를 받아 정부는 금년 2월 28일, 행정정보시스템 관계 과장 연락회에 추장암호 알고리즘을 통달했다』.

1.3 독일 연방군 등 MS 사용금지

독일의 경우 군은 리눅스를 이용해 필요한 소프트웨어를 직접 개발해 사용할 것을 권하고 있고, 남미 일부 나라는 오픈소스 소프트웨어 사용을 강제하는 법 제정까지 추진 중인 것으로 전해지고 있다. 오픈소스 소프트웨어는 미국 국방성과 군에서도 채택하고 있다. 2002년 12월 23일 국내의 한겨레신문[9]은 독일 주간 〈슈피겔〉 최근호의 보도를 인용하여 다음과 같이 보도하고 있다. 『독일군, MS 사용금지, 미 국가안전국 침투 가능성 : 독일 연방군이 최근 보안문제를 이유로 미국 마이크로소프트의 소프트웨어의 사용을 금지했다. 독일 보안당국은 마이크로소프트 제품이 사용된 전산망에 미국 국가안전국(NSA)이 침투할 수 있는 백-도어(소프트웨어나 시스템에서 보안이 제거된 비밀통로)를 가지고 있을 가능성을 제기했으며, 이에 따라 독일 연방군은 민감한 정보를 다루는 업무영역에서 마이크로소프트의 모든 소프트웨어를 제거했다고 이 잡지는 전했다. 독일 연방군은 이를 다른 모든 업무영역으로 확대해갈 계획이며, 독일 보안당국은 다른 미국 소프트웨어 업체 제품에 대해서도 조사를 벌이고 있다. 미국 국가안전국은 1952년에 설립된 국방부 특별 활동국 소속 정보수집기관으로 암호의 작성 및 관리, 적성국의 암호 분석 및 해독을 주요 임무로 한다. 독일 외교부도 재외공관과 위성을 이용한 영상회의 시스템을 구축하려던 계획을 정보유출 가능성 때문에 전면 보류했다. 외교부가 추진 중인 이 시스템은 기술적인 이유로 중계에 사용될 위성서비스가 미국 콜로라도 주 덴버를 경유하도록 돼 있다. 도이체텔레콤, 지멘스 등 독일 기업들은 독일 정부의 요청으로 현재 사용 중인 미국소프트웨어와 시스템을 대체할 새로운 제품을 자체 개발 중인 것으로 알려졌다. 영국 역시 정부가 사용하는 기본 소프트웨어를 오픈소스 프로그램으로 사실상 제한하는 조처를 시행한 바 있다. 오픈소스 소프트웨어에 대한 정보통신업체들의 관심도 커지고 있다. 아이비엠, 휴렛패커드, 인텔, 후지쯔, 히타치 등 세계적인 컴퓨터 회사들은 최근 레드햇과 터보리눅스 등 리눅스 업체들과 함께 '오픈소스개발연구소'를 설립했다』.

2. 국내 연구개발 현황

현재 국내에서는 [표 3]에서 보는바와 같이 연구기관으로서 ETRI 정보보호본부 보안운영체제팀에서 리눅스와 FreeBSD 플랫폼에서 커널을 직접 수정하는 방식으로 TCSEC B2급 수준의 "SecuROS"를 2002과 2003년에 시제품으로 연구 개발하여 기업에 기술전수를 시행하고 있다. 업체로서는 시큐브(주)가 2000년에 PKI 방식의 사용자 인증 방식을

〈표 3〉 국내 Secure OS 연구 개발 업체 현황

기관	제품명	개발 플랫폼	개발방식	주요 특징	비고
ETRI	SecuROS	Linux FreeBSD	커널 수정	RBAC MLS 커널모드 암호화	TCSEC B2 수준
시큐브	TOS	Linux Solaris HP-UX Compaq IBM-AIX Windows	후킹 LKM	PKI 기반 인증 ACL 기반 해킹방지 통합보안관리	-
티에스온넷	RedOwl SecuOS	Linux Solaris HP-UX Compaq IBM-AIX Windows	후킹 LKM	MLS 기반 Labeled Security 해킹방지 통합보안관리	TCSEC B1 수준
시큐브레인	Hizard	Linux Solaris HP-UX IBM-AIX Windows	후킹 LKM	RBAC/ACL기반 해킹방지 통합보안관리	-

적용하고 ACL(Access Control List) 방식의 접근제어와 불법적인 root 권한 해킹을 커널에서 차단하는 기능을 갖는 TOS(Trusted OS)를, 티에스온넷(주)는 주체 객체에 보안 등급을 적용하는 다중 등급 보안(Multi-Level Security) 기반의 강제적 접근제어와 커널 수준의 3단계 해킹 방지 기능을 갖는 RedOwl SecuOS를, 시큐브레인(주)는 직무기반 접근제어(RBAC; Role Based Access Control)와 커널 수준의 해킹방지 기능을 갖는 Hizard를 출시하여 서버 보안 시장에서 서서히 활용되고 있다. 보안 운영체제 전문 3사 모두 장점이 많은 시스템 호출 후킹 LKM 방식으로 개발하고 있으며, 3사 모두 국가정보원에서 시행하는 보안성 검토를 완료하고 민수분야는 물론 정부 및 공공시장에서 시장 확대를 예상하고 있다.

Ⅲ. 리눅스 보안 모듈(Linux Security Module)

1. LSM 개요

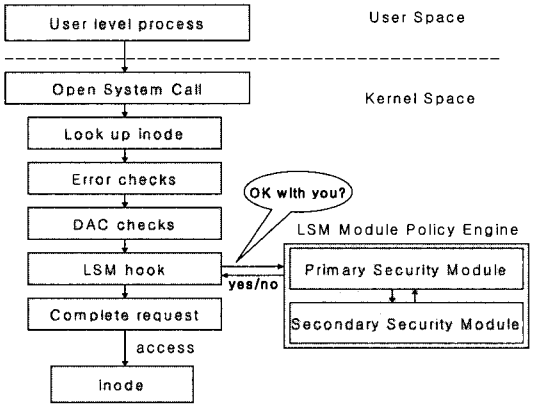
리눅스 보안 모듈(LSM; Linux Security Module) 프로젝트[10]는 2001년 3월 28-29일 San Jose에 있는 Hyatt Hotel에서 열린 리눅스 커널 2.5 정상회담(Linux Kernel 2.5 Summit)에서 NSA의 Peter Loscocco가 "Security Enhanced Linux"라는 주제 발표를 통하여 리눅스 커널에 유연한 접근제어 구조 구현인 SELinux[29] 프로젝트 결과를 발표하였다. 이 자리에서 토발즈 리누스는 리눅스 커널에 일반적인 접근제어 기본 구조가 필요하다는 인식은 받아들이게 되었으나, 리눅스 커널 보안에 대한 많은 프로젝트가 존재하고 리누스 자신이 세부적인 보안 정책에 대한 전문가적 식견이 부족하다는 이유로, 그는 탑재 가능 커널 모듈(Loadable Kernel Module)로 보안 모델이 구현될 수 있는 방법을 선호하게 되었다. 실제적으로 이러한 리누스의 응답으로 LSM 설계를 위한 씨앗이 잉태되게 되었으며, 리누스가 원하는 기본구조(Framework)를 개발하기 위해서 LSM 프로젝트가 WireX에 의해서 시작되었다. 2002년 6월에 토발즈 리누스는 Linux 2.5에 LSM을 받아들이기로 합의했으며, Kernel 2.5.29에 처음으로 LSM 기본 구조가 포함되게 된다. 2003년 7월 14일에 발표된 Kernel 2.6.0 테스트 버전에도 LSM 기본 구조가 포함되었다. 이 구조는 리눅스 표준 커널에 범용의 접근제어 방법을 제공함으로써 이러한 문제를 해결하려는 시도 중 하나이다. LSM은 커널 모듈로서 향상된 보안 정책을 선택적으로 적용할 수 있도록 해준다. 정책 집행 모듈(Policy Enforcement Module)을 위한 표준 API를 제공하는 리눅스를 제공함으로써 LSM 프로젝트는 보안이 강화된 리눅스 시스템을 널리 채용할 수 있도록 하고 있다.

LSM 프로젝트는 접근 제어를 위한 가볍고, 범용의 기본 구조를 제공한다. 현재 컴퓨팅 환경은 끊임없이 악의적이다. 커널에 향상된 접근제어를 추가함으로써 호스트의 보안성을 높이며, 서버로 하여금 악의적인 공격에 대응할 수 있도록 할 수 있다. 보안 연구는 각기 다른 환경에 다양한 향상된 접근제어를 제공해 왔다. LSM 기본 구조는 다양한 접근제어 모델을 LKM(Loadable Kernel Module)으로서 구현될 수 있도록 해준다. LSM 기본 구조는 GPL 라이센스를 가진다. LSM 프로젝트는 여러 종류의 보안 정책 모듈을 위한 범용의 기본 구조(framework)를 제공함으로써 이러한 바벨탑을 해결하고자 하는 시도이다. 이 LSM은 리눅스 커널과 독립적으로 다양한 보안 정책 엔진 개발이 가능하도록 하여, 각기 다른 여러 종류의 접근제어 모델을 탑재가능 커널 모듈(Loadable Kernel Module)로서 구현될 수 있도록 한다. 기존의 POSIX.1e capability, SELinux, DTE(Domain and Type Enforcement), LIDS(Linux Intrusion Detection System) 등과 같은 많은 접근제어 구현들이 LSM 기본구조를 사용할 수 있도록 이미 적용되고 있다[16].

2. LSM 설계 및 구현 개념

2.1 LSM 기본구조

LSM 기본구조[20]는 다음의 3가지 목표를 만족하도록 했다. 첫째, 다른 보안 모델을 사용하는 경우에는 단지 다른 커널 모듈을 적재하면 되도록 일반적일 것. 둘째, 개념적으로 단순하고 최소한으로 영향을 미치며 효율적일 것. 셋째, 선택 가능한 보안 모듈로서 기존의 POSIX.1e capability을 지원할 수 있을 것 등이다. 여러 가지 접근제어가 가능하면서 이러한 3가지 목표를 만족시키기 위해서 LSM은 [그림 1]에서 보는바와 같이 커널의 내부 객체(task, inode, open file 등)에 대한 접근이 가능하도록 하는 방법을 채택하게 되었다. 사용자 프로세스가 시스템 호출을 수행할 때 우선 자원을 찾아



〔그림 1〕 LSM Hook 구조

할당하기 위해서 리눅스 커널의 기존 로직을 거치며, 오류 체킹을 수행한 후, 기존의 UNIX DAC에 넘기게 된다. 커널이 내부 객체에 접근을 시도하기 직전에 LSM 훅(hook)이 "이 접근이 당신에게는 올바른가?"라는 질문을 부여하기 위해서 당신이 원하는 모듈로 out-call을 만들게 된다. 이 모듈이 이러한 정책적인 질문을 처리하고 "yes" 또는 "no" 중 하나를 되 돌려준다.

LSM 모듈의 조합은 다른 문제이다. 한편, 어떤 보안 정책들은 명시적으로 상호 충돌이 발생될 수 있기 때문에 여러 가지 보안 정책이 하나의 총체적 경우로 구성될 수 없는 경우가 있다. 다른 한편, 여러 가지의 다양한 보안 정책을 조합하여 구성하는 것이 분명 바람직하다. 여기서 LSM은 효과적으로 모듈 작성자에게 모듈을 "stack"할 수 있도록 방법을 제공한다. 즉, 첫 번째 적재된 모듈은 LSM 인터페이스를 다음에 적재될 모듈에게 전달할 수 있게 된다. 첫 번째 모듈은 두 번째 모듈에서 특정한 일을 수행 후 되돌아와 접근 결정을 종합적으로 조합할 책임을 지게 된다.

2.2 LSM 인터페이스

LSM 인테페이스는 150여개의 큰 함수 테이블이다. 이 함수들은 기본적으로 전통적인 super-user DAC 정책을 구현하는 호출로 알려져 있다. 제공하고 있는 인터페이스는 정책 등록(Policy Registration) 방법을 제공하며, 〔표 4〕에서 보는 여러 가지의 훅(Hook)을 위한 인터페이스를 제공하고 있다. 즉, 태스크 훅(Task Hook), 프로그램 적재 훅(Program Loading Hook), 파일 시스템 훅(File System Hook)(Super Block Hook, Inode Hook, File

〔표 4〕 LSM 자료 구조별 Hook 대상

구분	구조	객체별 Hooks
1	task_struct	Task(Process)
2	linux_binprm	Program
3	super_block	Filesystem
4	inode	Pipe, File 또는 Socket
5	file	Open File
6	sk_buff	Network Buffer(Packet)
7	net_device	Network Device
8	kern_ipc_perm	Semaphore/Shared Memory Segment 또는 Message Queue
9	msg_msg	Individual Message

Hook), 프로세스간 통신 훅(IPC Hook)(Common IPC, Object Specific IPC Hook), 모듈 훅(Module Hook), 네트워크 훅(Network Hook)(Socket & Application Layer, Packet, Transport Layer-IPv4, Network Layer-IPv4, Network Device, Netlink), 기타 시스템(System Hook) 등이다.

모듈 작성자가 자신들이 필요로 하는 함수를 구현할 책임이 있다.

정책 등록 방법에 있어 LSM 인터페이스는 callback method(security_ops) 구조로 구현된다. 보안 모듈은 집행하는 보안 정책에 따라 callback을 구현할 책임이 있다. 부팅할 때 security_ops 구조가 디폴트 callback으로 초기화되며 이 callback은 전통적인 superuser 시맨틱을 구현하게 된다. 보안 모듈은 동적으로 적재가능 모듈(dynamically loadable module) 또는 정적으로 커널에 링크되는 모듈로 만들어 질 수 있다. 이 모듈은 동적 적재 모듈 경우에는 모듈 적재 시에 초기화 되고, 정적 링크 모듈의 경우에는 do_initcalls()을 수행하는 동안 초기화된다. 초기화하는 동안 보안 모듈은 register_security()를 호출함으로써 LSM 기본구조에 이 callback을 등록해야만 한다. security_ops 구조를 디폴트 superuser 정책에 되돌려 주기 위해서 이 모듈을 제거(unload)할 때는 unregister_security()를 호출해야 한다.

LSM 기본구조는 한 순간에 오직 하나의 주가 되는 보안 정책만을 알고 있다. 하나의 보안 정책이 LSM 기본구조에 등록되게 되면, 새로운 보안 정책의 등록은 불가하다. 어떤 경우에는 여러 가지 보안 정책을 조합하는 것도 가능하다. 그러나 이 경우 LSM은 새로운 모듈을 다른 모듈과 함께 "stack"할 수

있도록 하며, 기본구조는 오직 하나의 security_ops 만을 알고 있을 뿐이다. 추가적으로 보안 정책을 등록하기 위해서는 후속 모듈은 mod_reg_security()를 이용해서 주가 되는 모듈에 등록해야 한다. 이러한 방법이 LSM 기본구조를 단순하게 해주며 주가 되는 보안 모듈에 복합적 보안 정책을 추가하게 된다.

3. LSM 시험 및 성능

진정한 LSM의 영향은 리눅스 커널의 표준으로 받아들여지고 있는 상황이므로 수용 가능한 것으로 평가되며, 일반 사용자들은 리눅스의 응용프로그램에 적용하듯이 보안 모듈을 적용할 수 있다. 리눅스에 수용되기 위해서 LSM은 높은 비용 대 효과가 있어야만 했다.

3.1 성능상의 영향

LSM 기본구조에 있어 성능 상의 영향은 매우 중요하다. 성능상의 영향은 LSM을 탄생시킨 Linux 2.5 개발자 정상회의에서 주요한 토의 주제가 되었다. 마이크로 벤치마킹의 경우, LMBech[20]를 사용하여 시스템 호출에 대하여 최악의 경우, stat()에서 6.2%, open/close에서 6.6%, file delete에서 7.2%의 부하율(overhead)이 측정되었다. 이러한 결과는 LSM에서 부가적인 처리에 의한 것으로 예측했던 대로 이며, 일반적인 경우에는 0%~2%로 부하율이 나타났다. 마크로 벤치마킹(커널 컴파일 수행)의 경우, 1 CPU(700MHz~1GB RAM-Ultra wide SCSI Disk) 환경에서 Linux 2.5.15(341sec) 대비 2.5.15-lsm(342sec) 0.3%, 4 CPU 환경에서 0%로 꽤 좋은 결과를 보였다.

3.2 보안상의 영향

LSM 기본구조를 받아들이기 위한 또 다른 중요 요인은 진정한 보안성을 제공하는가 하는 점이다. 이는 2가지 방법으로 볼 수 있다. 첫째는 LSM은 새로운 보안 구멍을 만들지 말아야 하며 총체적이며 일관성을 유지해야한다. 둘째는 LSM 기본구조는 다양한 접근 제어를 지원함에 있어 일반적이며 충분해야만 한다. LSM 기본구조의 정확성 증명은 LSM 프로젝트에 의해서 직접적으로 다루어지지는 않았으나, IBM 프로젝트에서 정적 및 동적 분석을 위한 도구를 개발하여 분석하였다. LSM의 이름에서 제시하는바와 같이 보안 모듈이 없는 상태에서 보안상의 영향이 없었다.

4. LSM 지원 모듈

LSM은 현재 다음과 같이 5개의 보안 모듈을 지원하고 있다.

4.1 SELinux

SELinux는 Flask[18]의 유연한 접근 제어 구조를 리눅스에 구현한 것으로, Type Enforcement, Role-Based Access Control, 옵션으로서 Multi-level Security을 지원하는 보안 서버의 한 예이다. SELinux는 원래 커널 패치로서 구현되었으며, 그 후에 LSM을 사용하여 보안 모듈 형태로 구현되었다. SELinux는 프로세스들로 하여금 최소 권한(Least Privilege)을 갖도록 한정하며, 프로세스와 데이터의 무결성과 비밀성을 보호하며, 응용 프로그램의 보안 요구 사항을 지원한다. SELinux의 일반성과 통합성은 LSM을 위한 요구사항을 추출함에 있어 큰 도움이 되었다.

4.2 DTE Linux

Linux를 위해서 Domain과 Type Enforcement이 구현되었다. SELinux와 같이 DTE Linux는 원래 커널 패치로서 개발되었으며 그 후에 LSM으로 바뀌었다. 이 모듈을 적재하여 Type이 객체에 부여되고, 프로세스들에게는 도메인이 부여된다. DTE 정책은 도메인간의 접근을 제한하며, 도메인에서 Type으로의 접근이 제한된다. DTE Linux 프로젝트는 또한 LSM의 설계와 구현에 유용한 도움을 주었다.

4.3 LSM port of Openwall kernel patch

Openwall 커널 패치는 일반적인 공격(예로 buffer overflow, temp file race)으로부터 시스템을 보호하기 위한 보안 기능을 모은 것으로, 이 LSM 모듈은 Openwall 패치의 일부를 지원하는 형태로 개발 중에 있다. 예로서 이 모듈을 적재하게 되면 대상 프로그램으로 하여금 악의적인 symlink를 따르지 못하게 한다.

4.4 POSIX.1e capability

POSIX.1e capability[26] 로직은 이미 리눅스 커널에 존재하였다. 그러나 LSM 커널 패치를 하게

되면 이 capability 로직을 보안 모듈과 별도로 격리 시켜서, 이 capability 기능을 필요로 하지 않는 사용자들에게는 자신의 커널에서 그 기능들을 빼버릴 수 있도록 해준다. 다시 말해서 capability 로직이 메인 커널 밖에서 독립적으로 개발될 수 있도록 해 준다.

4.5 LIDS(Linux Intrusion Detection System)

LIDS는 침입탐지시스템(IDS : Intrusion Detection System)으로 출발하였다. 그 후 SubDomain[17] (어떤 프로그램이 어떤 파일에 접근할 수 있는지를 기술하여 접근을 관리하는 방법)과 유사한 접근 제어 시스템의 형태로 침입방지시스템(IPS : Intrusion Prevention System)으로 발전되었다.

Ⅷ. 결 론

이제까지 리눅스 운영체제는 날로 증가하고 있는 다양한 해킹기법 및 불법적 사용 시도에 매우 취약한 면을 보이고 있다. 이러한 상황에서 기존의 방화벽이나 IDS 같은 보안제품이나 응용 프로그램으로 보안성을 확보하기에는 한계가 있다.

각국에서는 보안측면에서 자주적인 이유로 비공개 운영체제인 Windows 보다는 리눅스와 같은 공개 소스를 이용해서 독자적인 보안 운영체제를 선호하고 있다.

리눅스는 공유하는 놀이터로서 그 위에서 노는 대부분의 사용자들을 원칙적으로 행복하게 만들 필요가 있다. 그런 점에서 LSM은 2가지 기준을 만족시킬 필요가 있다. 즉 LSM을 원하지 않는 자들에게는 상대적으로 폐해가 없어야하며, LSM을 원하는 자들에게는 유용하고 효과적이어야 한다. 결과적으로 현재의 보안 리눅스 제품들은 LKM 방식으로, 표준 커널은 LSM 기본 구조로서 이러한 기준을 만족하고 있다. LSM의 경우 패치는 비교적 작으며 성능상의 수치는 LSM Patch가 거의 제로에 가까운 부하율을 가짐을 보여주고 있다. LSM을 구현하고 있는 세계의 보안 제품들은 리눅스 보안성 향상에 LSM API가 유용하며 효과적임을 보여주고 있다. LSM은 현재 Kernel 2.4와 2.5를 위해 커널 패치가 가능하며, 패치는 lsm.immunix.org에서 얻을 수 있다. Kernel 2.5.29 이후부터에서는 기본적으로 LSM를 위한 인터페이스가 포함되었으며 2003년 7월에 발표된 Kernel 2.6.0 베타 버전에도 LSM 인터페이스가 포함되어 있다.

보안 리눅스 운영체제는 향후 계속적인 연구를 통하여 다양한 환경에서의 기능 구현, 성능평가, 취약성 분석 등을 통하여, CC 기준이나 TCSEC 기준에 근거하여 등급의 상향화 연구개발이 계속될 것으로 보인다. 또한, DoS(Denial of Service) 공격 등 다양한 해킹방지에 대한 계속적인 연구가 지속적으로 필요할 것이다. 리눅스의 보안 취약성에 대한 적극적인 대책이 마련되고, 사용자의 적절한 보안 관리가 유지된다면 Windows 등과 같은 비공개 소프트웨어에 비하여 더욱 높은 보안성을 유지할 수 있을 것이다.

참 고 문 헌

[1] E. S. Raymond, The Cathedral and Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary, O'Reilly & Assoc., 1999.

[2] K. Brown, *Opening the Open Source* Debate, Alexis de Tocqueville Inst., 2002.

[3] Crispin Cowan, Software Security for Open-Source Systems, IEEE Security & Privacy, 2003.

[4] 손성훈, 리눅스 커널의 특징, 정보과학회지 VOL. 18 NO. 02 pp.0013~0017, 2000. 02.

[5] 한겨레신문, [리눅스]중, 윈도2000 규제로 리눅스 점유율 '쑥쑥' - 2000.2.18.

[6] http://www.nsa.gov/releases/selinux_01022001.html

[7] 한겨레신문, 일본, 전자정부사업에서 '윈도' 빨 듯, 2002. 11.17.

[8] 日經컴퓨터, 전자안전보장을 정비하라, 특집기사, 2003. 4. 21.

[9] 유럽·중국·남미서 바람 미국·독일 국방부도 채택(한겨레신문, 2002/12/23)

[10] http://lsm.immunix.org/lsm_doc.html

[11] DoD, Trusted Computer System Evaluation Criteria, DoD 5200.28. STD, 1985.

[12] Bell. D. and Lapadula, "Secure Computer System: Mathematical Foundations and Model," MITRE Report MTR 2547, v2 Nov 1973.

[13] R. Magnus et al, LINUX KERNEL

INTERNALS, 1999.
[14] Charles W. Flink II et al., "System V/MLS Labeling and Mandatory Policy Alternatives," Proc. of USENIX-Winter'89, pp. 413-427, 1989.
[15] D. D. Downs et al., "Issues in Discretionary Access Control," Proc. of IEEE Symposium on Security and Privacy, pp.208~218, 1985.
[16] Chris Wright, Crispin Cowan & Stephen Smalley, Linux Security Module Framework, Ottawa Linux Symposium, 2002.
[17] Crispin Cowan et al, SubDomain: Parsimonius Server Security, In USENIX 14th Systems Administration Conference (LISA), New Orleans, LA, Dec. 2000.
[18] http://www.cs.utah.edu/flux/fluke/
[19] Charles P. Pfleeger, Security in Computing, PTR, 1997.
[20] Larry W. McVoy and Carl Staelin, lmbench: Portable Tools for Performance Analysis, USENIX Annual Technical Conference, 1996.
[21] ISO/IEC 15408 Common Criteria, http://www.common-criteria.org 1999. 8.
[22] Peter A. Loscocco et al., The Inevitability of Failure: The Flawed Assumption of Security in Modern Computing Environments, 21st NISSC, 1998.
[23] Sue Hildreth, ASP Security: Why Firewall Are Not Enough, http:// www.ebizQ.net, 2001.2.
[24] Dixie B. Baker, Fortresses Built Upon Sand, ACM Proc. of the New Security Paradigms Workshop, 1996.
[25] Thomas H. Ptacek et al., Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection, NAI Lab., 1998.1.
[26] IEEE Std 1003.2c-Draft standard fot Information Technology Portable Operating System Interface(POSIX) Part 2: Shell and Utilities : Protection and Control Interfaces.
[27] Paul C. Clark, Policy-Enhanced Linux, 23rd NISSC, 2000.
[28] Http://www.police.go.kr 경찰청 사이버테러 대응센터 보도자료
[29] Security Enhanced Linux, http://www.nsa.gov/selinux/
[30] Immunix, http://immunix.org/

〈著 者 紹 介〉

박 태 규 (Tae-Kyou Park)
종신회원

1980년 10월 : 경북대학교 전자계산기공학과 졸업
1989년 8월 : 충남대학교 전산학과 석사
1996년 2월 : 성균관대학교 정보공학과 박사
1981년 2월~1982년 12월 : 한국국방연구원 연구원
1982년 12월~1992년 2월 : 한국전자통신연구원 선임연구원(부호5실장)
1997년 1월~1998년 1월 : Univ of Western Sydney, Post-doc.
1992년 3월~현재 : 한서대학교 교수
관심분야 : 보안 운영체제, 컴퓨터 보안