情 報 保 護 學 會 誌
第 15 卷 第 2 號, 2005. 4

리눅스 커널 보안 동향

강 정 민*, 장 인 숙*, 남 택 준*, 이 진 석*

요 약

리눅스가 전자정부 및 다양한 분야에 적용되면서 항상 거론되는 문제점은 보안이다. 더욱이 최근 들어 커널 취약점과 이를 악용한 공격 사례가 발표되면서 응용 수준에서의 보안 노력이 사상누각(砂上樓閣)임을 보여주고 있다. 본 논문에서는 리눅스 커널의 잠재적 혹은 알려진 취약점을 해소하기 위한 커널 수준의 리눅스 보안 요소 기술과 보안 솔루션들을 살펴본다. 또한 본 저자들이 제안하는 모바일 디바이스용 리눅스 보안 커널에 대해 소개한다.

Ⅰ. 서 론

최근 오픈 소스의 장점을 반영하면서 리눅스 운영체제는 전자정부 시스템을 비롯한 다양한 분야의 시스템에 적용, 운영되고 있다. 하지만 리눅스 운영체제가 지닌 보안 취약점으로 인해 해결해야 할 과제 또한 많이 존재한다. 본 논문에서는 리눅스 운영체제의 핵심인 리눅스 커널의 잠재적 혹은 알려진 취약점을 해소하기 위한 리눅스 커널 보안 기술과 리눅스 시스템의 보안을 향상시키기 위한 커널 수준에서의 보안 솔루션들을 살펴보고자 한다.

리눅스 커널의 취약점 현황을 살펴보고, Ⅲ장에서는 리눅스 커널의 취약점을 해소하기 위한 보안 기술들을 설명하며, Ⅳ장에서는 리눅스 시스템의 보안 강화를 위한 커널 수준의 보안 솔루션들을 소개하고, Ⅴ장에서는 결론을 맺는다.

Ⅱ. 리눅스 커널 취약점 분석

리눅스 커널(2.4.23 기준)은 약 4,890,000 라인으로 구성된 거대한 소프트웨어이다. 리눅스 커널이 제공하는 기능은 크게 프로세스 관리, 파일 시스템, 메모리 관리, 네트워크 관리 등으로 나누어 볼 수 있다. 사용자 프로그램은 이러한 기능들을 정해진 인터페이스를 이용하여 커널에 요구하게 되며, 커널은 이러한 요구들을 만족시켜 준다.[1] 리눅스 커널의 개발은 오픈 소스로 진행되고 있다. 모든 사람이 자유롭게 복사하고 사용할 수 있으며, 마음대로 소스를 수정하여 재배포할 수 있는 GPL(GNU Public License)을 따른다. 소스가 공개되어 있기 때문에 관심 있는 사람들은 잘못된 코드를 찾아서 수정하거나, 부족한 기능을 직접 구현할 수 있다. 자신이 가지고 있는 하드웨어를 리눅스가 지원하지 않으면 이를 만들어서 추가하고 고쳐진 코드는 리눅스 커널의 창시자이자 관리자인 Linus Tovalds에게 보내지고, 그는 여러 곳에서 보내온 코드를 취합하여 커널을 수정한 후 배포한다.[2]

리눅스 커널 취약점은 SecurityFocus, CVE(Common Vulnerability and Exposure) 및 Security Tracker 등의 사이트에 응용 소프트웨어의 취약점과 함께 공지되고 있다.[3-5] SecurityFocus에서 1999년부터 2004년까지의 리눅스 커널 코드에서 나타난 취약점의 발표 건수는 120여 가지로 표 1에 나타나 있다.

〔표 1〕 연도별 커널 취약점 발표 현황

	2004. 1~11월	2003	2002	2001	2000	1999
발표 건수	57	12	16	12	5	18

그러나 SecurityFocus에서 공식으로 bugtraq id를 부여한 것 외에 커널 공식 사이트에서 특별한 공지 없이 다음 패치 버전에 조용히 반영되는 보안 취약점도 상당수 존재한다. 주목할만한 사항은 2004년도

* 국가보안기술연구소 ({jmkang, jis, tjnam, jinslee}@etri.re.kr)

에 커널 취약점의 발표 건수가 4배정도 늘었다는 점이다. 이것은 해커들과 보안 종사자들의 리눅스 커널 취약점에 대한 관심이 커졌음을 의미한다. 커널 취약점의 경우에는 그것이 실제 취약한지를 검토하기 위한 기술의 난이도가 높고, 기간이 오래 걸리며 패치 발표까지 수개월이 걸리는 경우가 많다. do_brk() 취약점의 경우, 데비안 개발자 웹사이트가 do_brk() 커널 함수의 취약점을 이용한 익스플로잇에 의하여 해킹된 이후에야 그 중요도를 인지하고 보안 권고문을 발표하는 사례도 있었다. 또한 2003년 12월에는 브라질의 해커 그룹이 PHP 웹 취약점을 이용하여 리모트 접근 권한을 획득한 뒤 do_brk() 커널 취약점에 의한 익스플로잇을 이용하여 로컬 관리자 권한을 취득하는 방법으로 미 항공 우주국(NASA)의 여러 웹서버를 해킹한 사례도 있다. 이와 같이 해커들에 의하여 커널 취약점이 악용되는 사례가 발생하고 있다.

2.1 리눅스 커널 취약원인 분석[6,7]

취약점의 발생 원인은 특정 기능에 대한 설계 오류에서부터 커널 소스 프로그래밍 오류에 이르기까지 다양하다. 커널 취약점은 발생 원인에 따라 표 2에서와 같이 7가지로 분류할 수 있다. 실제 발견된 커널 취약점의 경우 설계상의 오류가 가장 많고, 단순한 프로그래밍 상의 실수에 의한 예외 상황 처리 실패 오류와 경계값 검사를 제대로 수행하지 않음으로써 발생되는 경계조건 오류가 그 다음으로 많은 비중을 차지하고 있다. 설계상의 오류는 프로그램 구현상에서 데이터 타입을 잘못 설계하거나 시스템 자원의 최대값, 최소값을 잘못 설계함으로서 발생하는 경우를 모두 포함한다. 그 외 알려지지 않은 취약점도 상당수 존재한다. 취약점 발생 원인들을 프로그래밍 관점으로 재분류하면 표 3에서 보는 바와 같이 분류할 수 있다. 권한 오류와 입력 값 오류의 비율이 전체의 50%를 차지하고 있음을 알 수 있다. 커널 서브시스템별 취약한 현황은 네트워크 부분과 파일 시스템 부분이 다른 부분에 비해 많은 양을 차지하고 있다. 취약점에 대한 익스플로잇을 살펴보면 대부분의 공격 목적은 로컬에서의 권한상승과 서비스 거부 공격으로 나타나고 있다.

Ⅲ. 리눅스 커널 보안 요소 기술

본 장에서는 리눅스 커널의 취약점을 해소 또는 보안을 향상시키기 위한 요소 기술들을 소개한다.

〔표 2〕 커널 취약점 발생 원인과 현황

발생 원인	설 명	발생비율 (%)
환경 오류	특정 환경에서만 오류를 낼 경우	2.5
입력 정당성 오류	입력값의 유효성을 검사하지 않는 프로그래밍 실수로 인하여 발생하는 오류	2.5
예외 상황 처리 실패	시스템이 함수 모듈, 장치 또는 사용자 입력에 의해 생기는 예외적인 상황을 처리하지 못함으로써 오류가 발생하는 경우	17.5
경쟁 상태 오류	두개 연산 사이의 시간 윈도우 동안 오류가 악용되는 경우	5.9
설계 오류	커널의 기능을 구현하기 위한 설계 시의 오류에 의하여 취약점이 발생하는 경우	33.3
접근 정당성 오류	주체의 접근 도메인 외에 존재하는 파일이나 메모리 영역을 읽거나 쓰려고 할 때 시스템이 주체를 적절하게 인증하는데 실패함으로써 오류가 발생하는 경우	7.5
경계조건 오류	프로세스가 정당한 주소범위 외의 영역을 읽거나 쓰려고 할 경우 정적으로 정해진 크기의 테이터 구조의 오버플로우로 인하여 오류가 발생한 경우	13.3

〔표 3〕 프로그래밍 관점에서의 취약점, 공격 현황

취약 원인		설 명	취약점 발생 비율 (%)	권한상승 공격 비율 (%)	서비스 거부 공격 비율 (%)
정수 오류	정수 변수 오버플로우	정수 변수 오버플로우가 발생하는 경우	10.7	11.0	11.1
	정수 연산 오류	부호 있는 정수와 부호 없는 정수 간 연산으로 오류가 발생하는 경우			
한계 오류		최대값이 지정된 변수의 최대값을 초과하여 사용하는 경우	2.9	2.4	1.2
권한 오류		Capability 기능을 이용하여 권한상승을 할 수 있는 경우	25.6	24.3	27.3
초기화 오류		변수 및 시스템 설정이 잘못 초기화된 경우	12.9	12.2	11.1
반환 값 오류		반환 값이 있는 함수를 호출한 후, 반환 값을 정확히 확인하지 않는 경우	10.0	11.0	12.3
입력 값 오류		인자를 정확히 확인하지 않고 함수를 호출하는 경우	24.3	22.0	17.3
값 비교 오류		값 비교 후 잘못된 비교 결과를 사용하였을 경우	3.6	6.1	7.4
기타		분류 기준에 해당하지 않는 경우	10.0	11.0	12.3

3.1 취약점 패치 기술

리눅스 진영에서는 리눅스 커널 자체에 대한 문제점이나 취약점이 발견되면 공개 패치를 발표하여 이를 보완하여 왔다. 패치 파일은 커널의 일부분을 수정하고 diff 유틸리티를 이용하여 생성한다. 새로운 패치는 수정된 커널과 기존 커널과의 차이를 diff 유틸리티를 이용하여 생성하기 때문에, 이전 패치에 대한 정보가 포함되지 않는다. 그렇기 때문에 모든 패치가 순서대로 적용되어야만 정상적으로 패치작업이 수행된다.[8] 이러한 패치 방식은 패치파일을 빠르게 생성하여 배포하고, 쉽게 패치 작업을 수행할 수 있는 장점을 가지고 있는 반면, 버전 차이가 많이 날 경우에는 패치의 순서를 맞추어서 비슷한 작업을 반복적으로 수행해야 하는 번거로움을 가지고 있다. 그러므로 버전 차이가 많이 날 경우에는 가장 최근의 커널을 이용하여 최소한의 패치만을 적용하는 것이 편리하다. 또한 패치완료 후 새로운 커널을 적용하기 위해서는 시스템을 종료해야 되는데 상용서비스를 운용하는 기관에서는 이러한 과정이 부담이 되어 패치를 쉽게 적용시키지 못하는 경우도 있다.

3.2 접근 통제 기술

접근 통제의 목적은 컴퓨팅 자원, 통신 자원 및 정보 자원 등에 대하여 허가되지 않은 접근을 제한하는 것이다. 허가되지 않은 접근이란 불법적인 자원의 사용, 노출, 수정, 파괴와 불법적인 명령의 수행을 포함한다. 즉, 접근통제는 각 자원에 대한 기밀성, 무결성, 가용성 및 합법적인 이용과 같은 정보보호 서비스에 직접적으로 기여하게 되며 이러한 서비스들의 권한부여를 위한 수단이 된다. 현재 주류를 이루고 있는 접근 통제 정책은 다음과 같이 3가지로 대표할 수 있다.[9,10]

3.2.1 임의적 접근 통제

임의적 접근 통제(DAC: Discretionary Access Control)는 현재 대부분 유닉스 계열에서 채택하는 접근 통제 정책이며, 주체 또는 그들이 속해 있는 그룹들의 신분에 근거하여 객체에 대한 접근을 제한하는 방법이다. DAC 정책에서 내재적으로 지닌 결점은 DAC 속성상 접근 통제는 주체의 신분에 전적으로 근거를 두고 있으므로 만약 다른 사람의 신분을 사용하여 행위가 이루어진다면 DAC은 파괴될 수 있다.

3.2.2 강제적 접근 통제

강제적 접근 통제(MAC: Mandatory Access Control)는 객체에 포함된 정보의 비밀성과 이러한 비밀성의 정보에 대하여 주체가 갖는 권한에 근거하여 객체에 대한 접근을 제한하는 방법이다. 주체가 객체에 대한 접근 시 다음의 규칙을 만족해야 한다. 아래 두 가지 특성은 정보의 기밀 분류 환경에서 상위 등급의 정보가 하위 등급으로의 흐름을 제한한다.

- ss-property: 주체의 보안등급이 객체의 보안등급을 지배하면 해당 객체를 read 할 수 있다.
- *-property: 객체의 보안등급이 주체의 보안등급을 지배하면 해당 객체에 write 할 수 있다.

3.2.3 역할기반 접근 통제

역할기반 접근 통제(RBAC: Role Based Access Control)는 현대의 상업용 환경에서 잘 적용되는 정책이며, 정보에 대한 사용자의 접근은 조직 내에서 개인의 직무(사용자가 한 조직에서 행하는 정해진 기능)에 따라서 결정된다. 여기서 직무는 주체 그룹에 해당하고(DAC), 직무 이외의 접근을 강제적으로 금지(MAC) 한다는 측면에서 RBAC은 DAC과 MAC의 특성이 동시에 존재한다.

3.3 침입 방지 기술

침입 방지 기술은 외부로부터의 불법적인 침입을 탐지 및 방어하는 Anti-Hacking 기술들을 의미한다. 커널 취약점을 악용하는 코드들은 대부분이 로컬 공격이므로 외부로부터의 불법적인 시스템 접근을 차단한다면 커널 취약점을 악용하여 공격하기 어렵게 되므로 침입방지 기술은 일차적 방어선이 될 수 있다. 침입 방지 기술에는 방화벽, 네트워 침입탐지시스템(IDS) 등 네트워크 수준 침입 방지 기술과, 로컬에서 수행중인 프로세스의 이상 행위를 탐지 및 차단하는 호스트 침입탐지시스템 등의 시스템 수준 침입 방지 기술로 구분할 수 있다.

네트워크 수준의 침입방지 기술은 초기에 응용 수준에서 별도의 프로그램으로 구현되었으나, 최근에는 커널 수준의 커널 모듈 형태로 개발되는 추세이다. 시스템 수준의 침입방지는 주로 버퍼 오버플로우 방지, 루트 쉘 공격 방지, 스푸핑 및 스니핑 방지 등의 기능들로 구성된다. 버퍼 오버플로우는 주로 응용 프로그램의 오류를 이용하여 스택을 오버플로우 시킴으로써 쉘 프로그램을 실행시키는 공격 방식으로 스택에서 코

드가 실행되지 않도록 함으로써 이를 방지하는 기술들이 널리 사용되고 있다. 루트 쉘 공격 방지는 루트 권한의 쉘을 실행 시킬 때 강력한 권한 검사를 수행하도록 하는 방식이며, 스푸핑 방지는 신뢰된 경로에 존재하는 프로그램만을 실행하도록 하는 기술이다. 스니핑 방지는 네트웍 카드가 promiscuous mode로 동작하는 것을 탐지하여 이를 종료시키는 기술이다. 또한, 커널 루트킷과 같은 악성 커널 모듈은 일단 시스템에 적재되면 탐지가 어려우므로, 커널 모듈들이 악의적으로 탑재되지 않도록 보호하는 커널 Sealing 기술도 침입방지 기술에 해당된다.

3.4 암호화 기술

접근통제, 침입방지 기술은 시스템 내의 데이터에 대한 불법적인 접근을 효율적으로 통제할 수 있는 방법을 제공할 수 있지만, 백업 매체나 디스크 자체의 도난에 대해서는 해당 매체 내에 저장되어 있는 데이터의 보호가 불가능하다. 또한 응용 수준에서의 암호 메커니즘은 악성 응용프로그램 및 악의적 사용자에 의해서 우회되거나 위장될 수 있다.[11] [11]은 응용 수준에서의 대칭키 방식의 암호 기술을 사용하는 Kerberos 인증 시스템의 취약성을 잘 설명해주고 있다. 응용 수준에서의 암호화 기술의 단점은 다음과 같이 정리될 수 있다.[12]

첫째, 비밀 파일에 대해 사용자의 실수 혹은 고의에 의해 암호화를 하지 못하는 경우가 생길 수 있다.

둘째, 응용 프로그램에서 파일을 읽거나 변경하는 등의 경우에 불가피하게 파일 단위로 전체를 복호화한 후, 저장하여 처리하기 때문에 비밀 수준의 파일이 평문 형태로 보조기억장치에 저장되어 존재하는 경우가 많다.

셋째, 처리 속도의 측면에서 커널 수준에서의 암호화/복호화에 비하여 늦다.

넷째, 암호화/복호화에 사용되는 키 관리를 위해 별도의 키 관리 방법이 제공되어야 한다.

리눅스 커널 수준에서 암호화 기술은 군부대 같이 내부자에 의한 정보유출이 민감한 상황에서 저장 데이터에 대한 안전한 보호를 제공할 것이며, 네트워크 트래픽을 암호화 함으로서 안전한 통신환경을 제공할 수 있다.

3.5 감사기록 기술

커널 수준에서의 감사기록은 로그파일 삭제 및 위조·변조에 대한 안전성, 신뢰성을 제공할 수 있다. 또한 침입방지에 필요한 데이터를 제공하기도 한다. 리눅스 운영체제에 포함되어 운용되는 디폴트 감사 시스템(syslog, log)은 수많은 커널 루틴, 각종 데몬들, 또는 전자메일 프로그램등과 같은 시스템 유틸리티들에서 생성하는 로깅 관련 감사 자료들만을 처리하고 있다. 감사 시스템이 존재하지 않거나 충분치 못한 감사 자료를 제공하는 시스템에 대한 직접적인 응용 프로그램 수준에서의 제어도 가능하나 불안정한 감사 자료 생성 및 성능상의 심각한 결과를 초래한다.

Ⅳ. 리눅스 커널 수준 보안 솔루션

본 장에서는 Ⅲ장에서 소개된 기술을 이용하여 제작되는 솔루션들을 살펴본다. 아울러 본 저자들이 제안하는 솔루션을 소개한다.

4.1 커널 패치 솔루션

패치관리시스템(PMS : Patch Management System)은 금융기관, 대기업 등의 애플리케이션이나 안티바이러스 업데이트를 자동화하기 위해 많이 사용되고 있다. 애플리케이션 레벨에서는 이러한 패치관리시스템의 도입으로 패치 자동화가 정착되었다. 그러나 현재 적용되고 있는 패치관리시스템은 OS(윈도우, 리눅스 등)에 관계없이 커널에 대한 패치가 일어나면 해당 시스템을 종료해야 되는 한계를 여전히 가지고 있다.

최근에는 이러한 한계점을 극복하기 위한 새로운 시도가 이루어지고 있다. 커널에서 취약점이 발견되었을 때 이 취약점을 해결하는 패치코드를 커널에 적용하는 것이 아니라 커널의 보안모듈에 적용시키려는 시도이다. 커널 단으로 내려가는 모든 시스템 콜을 후킹하여 해당 시스템 콜에 커널이 가지는 취약점 패치 코드를 적용한다. 이렇게 하게 되면 커널 단에서 발견되는 취약점을 모듈 단에서 차단하게 되고, 그 후 패치를 적용하는 과정에서 시스템을 종료할 필요가 없이, 해당 모듈만을 다시 적재해주면 된다. 이처럼 모듈 단에서 보안 관리를 해주게 되면 패치과정에 소요되는 시간 자원을 줄일 수 있다. 또한, 커널 자체의 취약점뿐 아니라 응용 프로그램이 가지는 취약점에 대한 공격을 차단하는 기능도 쉽게 추가할 수 있다.

4.2 보안 커널 솔루션

보안 커널이란 하드웨어 펌웨어, 운영체제, 소프트웨어 어플리케이션과 같은 보안 컴퓨팅 베이스(TCB: Trusted Computing Base)의 모든 요소들로 참조

모니터 개념이 구현된 것을 의미한다. 참조 모니터(Reference Monitor)는 주체에 의한 객체의 모든 접근을 중재하는 일련의 소프트웨어를 말한다. 참조 모니터는 객체에 대한 접근 통제 기능을 수행하고, 감사, 식별 및 인증, 보안 매개 변수 설정 등과 같은 다른 보안 메커니즘과 데이터를 교환하면서 상호 작용을 한다.[9,13]

위와 같이 보안 커널은 Ⅲ장에서 언급했던 접근 통제 기술을 적용한다. 표 4의 제품들을 살펴보면 대부분 역할 기반 접근 통제와 강제적 접근 통제를 적용하고 있다. 이들 대부분은 시스템 콜을 후킹 하여 접근 통제를 수행 하는 동적으로 로드/언로드 가능한 모듈로서 개발되어 지고 있다.

〔표 4〕 보안 커널 제품과 접근 통제

	소속	제품 및 연구	접근 통제
국내	시큐브	SecuveTOS	RBAC
	티에스온넷	RedOwl	MAC
	시큐브레인	Hizard	RBAC
	레드캐이트	Redcastle	MAC
국외	NSA	SELinux	Type Enforcement
	Hamburg대학	RSBAC	MAC, RBAC
	FreeBSD	TrustedBSD	MAC
	Argus	PitBull	MAC

4.3 커널 수준의 침입 방지 솔루션

외부의 침입으로부터 시스템을 안전하게 보호하기 위한 침입 방지 솔루션을 커널 수준에서 구현한 프로젝트로서 LIDS[14], PaX[15] 등이 있다.

4.3.1 LIDS

LIDS(Linux Intrusion Detection System)는 호스트기반의 침입 탐지 시스템으로서 커널의 보안을 증진시키기 위한 커널 패치와 보안 관리 도구로 구성된다. LIDS는 ACL(Access Control List) 방식을 이용하여 파일에 대한 접근 통제를 수행하고, 수퍼유저 권한의 제한, 프로세스 보호, 커널 잠금 기능 등을 제공한다.

- 파일 보호 기능: 시스템의 파일들을 Read only 속성으로 설정하여 임의로 수정할 수 없도록 보호하고, 시스템 로그 파일들은 Append only로 구분하여 크기가 늘어나는 경우만 가능하도록 보호하는 기능이다.
- 수퍼유저 권한 제한: 최소권한(least privilege) 원칙에 입각하여 수퍼유저에 모든 권한이 집중되지 않도록, 수퍼유저라 할지라도 모든 프로세스와 파일을 핸들링 할 수 없도록 제한하는 기능이다.
- 프로세스 보호: ps 명령어로 프로세스를 볼 수 없도록 숨기는 기능과 kill 명령어로 프로세스를 종료시킬 수 없도록 함으로써 프로세스를 보호하는 기능이다.
- 커널 잠금: 시스템이 시작된 이후에는 동적으로 모듈을 로드 할 수 없도록 하는 기능으로 커널 루트킷과 같은 악의적 모듈이 시스템에 적재되는 것을 방지하는 기능이다.
- 네트워크 보안: 방화벽이나 라우팅 규칙 변경 권한을 제한하고, 현재 네트워크 카드가 promiscuous mode로 동작하지 않도록 제한하는 기능 및 포트 스캐닝 탐지 기능을 제공한다.

4.3.2 PaX

PaX는 최근 가장 빈번히 발생되고 있는 소프트웨어의 취약점을 이용한 버퍼 오버플로우 공격을 커널 수준에서 방지하려는 프로젝트이다. PaX의 목적은 메모리와 관련된 보안 취약점들을 보호하는 것이다. PaX에서 제공하는 보안 메커니즘은 크게 NOEXEC (Non-Executable)과 ASLR(Address Space Layout Randomization) 기법으로 구성된다.

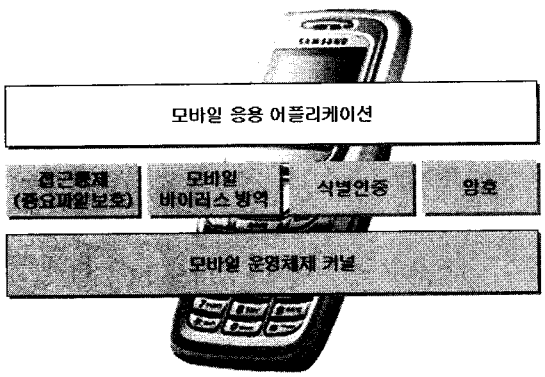
- NOEXEC : Non-Executable 보호 메커니즘은 메모리에 대한 접근 통제에 가장 많이 사용되는 기법이다. Non-Executable 기법은 메모리 상의 읽기 가능 페이지와 쓰기 가능 페이지를 분리하고, 스택 영역은 실행 불가능하도록 해야 하며, 실행 가능한 메모리와 실행 불가능한 메모리간의 변환을 방지하도록 제한한다.
- ASLR : ASLR은 특정 태스크에 의해 사용되는 주소를 임의적으로 할당하는 방법이다. 대부분의 익스플로잇 기법들은 공격되는 태스크의 현재 스택 포인터 또는 라이브러리 주소 등의 특정 주소에 관한 선행 지식을 필요로 한다. 만약 공격 대상 태스크의 랜덤화된 주소 공간 레이아웃에 대한 정보가 노출되지 않는다면 익스플로잇 기법은 주소 랜덤화로 인한 수많은 가능성 때문에 성공하기 힘들게 된다.

4.4 제안: 모바일 디바이스용 리눅스 보안 커널

리눅스는 많은 모바일 디바이스에 탑재되고 있지만, 몇 해 전부터 카비르(Cabir) 등 모바일(휴대폰) 바이러스들이 출현하기 시작했다.[16] 모바일 바이러스에 감염되면 휴대폰 안의 주소록을 지우거나 블루투스를 이용해 다른 단말기에 광고 메시지를 마구 보내는 등의 특징을 보이고 있다. 이는 휴대폰이 바이러스와 결합해 생길 미래의 위험을 예고한다고 할 수 있다. 모바일 디바이스들은 결국 미니 컴퓨터로 진화해 운영체제를 기반으로 작동될 것이기 때문에 바이러스의 감염은 치명적일 수 있다고 보안 전문가들은 경고하고 있다.

본 저자들이 제안하는 모바일 디바이스용 보안 커널의 기능을 살펴보면 다음과 같다.

- 식별인증: 제 3자에 의한 개인 단말의 불법사용 방지와, 모바일 디바이스 사용자간의 커뮤니티 및 그룹 형성 시 상호 인증 등을 위해 필요한 기능이다.
- 접근통제: 모바일 디바이스에 다운로드 되는 수많은 모바일 코드들에 의한 파일 및 리소스 접근이 불법적이지 않음을 보장한다.
- 암호: 주소록 등 중요 파일의 분실 및 도난 시 이들의 보호와, 네트워크 전송 시 중요 데이터에 대한 암호화 기능이다.
- 모바일 바이러스 방역: 수많은 모바일 코드들을 받아들이는 환경에서 악성 모바일 코드에 대한 탐지 및 방역 기능이다.



(그림 1) 모바일 디바이스용 리눅스 보안 커널

각 보안기능들을 구현함에 있어서 고려해야 할 사항들은 다음과 같다.

- 성능 및 속도: 모바일 디바이스들은 대체로 CPU 및 메모리 같은 시스템 자원이 풍족하지 않다. 암호기능에 있어서 가볍고, 빠른 속도의 암호 알고리즘이 요구된다. 또한 식별인증, 접근통제, 바이러스 방역을 위한 시그니쳐 및 정책 파일에 대한 접근을 빠르게 하기 위해서는 파일 접근이 아닌 메모리 및 캐쉬 접근에 의한 메커니즘을 고려해야 할 것이다.
- 유연성, 확장성: 모바일 디바이스들은 항상 켜져 있어서 네트웍에 연결되어 있는 상태이다. 보안 기능들을 코어 커널 자체에 구현한다면 보안 기능의 적용을 위해 시스템이 재부팅 되어야 한다. 또한 모든 보안 기능들이 모든 디바이스에 필요한 것은 아닐 것이다. 이런 상황에서는 보안 기능들을 유연하게 선택 적용할 수 있으면서, 재부팅이 필요없이 언제든지 적용, 해제 할 수 있는 커널 모듈 형태의 구현이 타당할 수 있다.

V. 결 론

최근 리눅스는 다양한 시스템에 도입되어 널리 사용되어지고 있다. 오픈소스의 장점 또는 단점으로 논의되고 있는 보안 문제는 계속해서 해결해야 할 과제로 남아있다. 또한 응용 수준에서의 보안 노력을 무의미하게 만드는 커널의 취약점과 이를 이용한 공격 사례가 발표되면서 리눅스 커널 자체의 보안 노력이 절실히 요구되고 있다.

본 논문에서는 최근까지 발표된 리눅스 커널의 취약점을 살펴보고, 이를 해소하기 위한 리눅스 커널 수준의 보안 요소 기술인 취약점 패치 기술, 접근 통제 기술, 침입 방지 기술, 암호화 기술, 감사기록 기술들을 설명했고, 요소 기술들을 이용한 커널 수준의 보안 솔루션들을 살펴보았다. 마지막으로 모바일 디바이스들에 탑재되어 운영되는 리눅스의 보안을 향상시키기 위해 모바일 디바이스용 리눅스 보안 커널을 제안했다.

참 고 문 헌

[1] 이호, 심마로 역, "리눅스 커널의 이해", *한빛미디어*, 2001.
[2] http://www.osdl.org
[3] http://www.securityfocus.com
[4] http://www.cve.mitre.org
[5] http://www.securitytracker.com

[6] 장인숙, 남택준, 강정민, 이진석, "공개된 소스 레벨 운영체제 취약점 현황 분석", *한국정보처리학회 추계학술발표대회논문집*, Vol. 11, No 2, 2004.
[7] 김재광 외, "취약 원인에 따른 리눅스 커널 취약성 분류법", *한국인터넷정보학회 추계학술발표대회논문집*, Vol. 5, No. 2, pp. 119-122, 2004.
[8] http://www.kernel.org
[9] Edward G. Amoroso, "Fundamentals of Computer Security Technology", *Prentice Hall PTR*, 1994.
[10] Charles P. Pfleeger, Shari Lawrence Pfleeger, "Security in Computing", *Prentice Hall PTR*, 2003.
[11] Peter A. Loscocco 외, "The Inevitability of Failure: The Flawed Assumption of Security in Modern Computing Environments", *Technical report, NSA*, 1989.
[12] 박태규, 임연호, "리눅스 커널 기반의 안전한 OS 개발", *제5회 정보인프라 워크샵 논문집*, 2001.
[13] Dieter Gollman, "Computer Security", *Jone Wiley and Sons*, 1999.
[14] http://www.lids.org
[15] http://pax.grsecurity.net
[16] http://www.sophs.com

〈著 者 紹 介〉

강 정 민 (Jung Min, Kang)
정회원

2002년 2월: 광주과학기술원 정보통신공학과 석사 졸업
2002년 3월~2003년 5월: 삼성 SDS 근무
2003년 6월~현재: 국가보안기술연구소 근무

장 인 숙 (In Sook, Jang)
2001년 2월: 경북대학교 컴퓨터과학과 석사 졸업
2001년 3월~현재: 국가보안기술연구소 근무

남 택 준 (Taek Jun, Nam)
2003년 2월: 한국외국어대학교 컴퓨터공학과 석사 졸업
2003년 11월~현재: 국가보안기술연구소 근무

이 진 석 (In Sook, Jang)
1990년 2월: 한남대학교 수학과 석사 졸업
2002년 8월: 한남대학교 컴퓨터공학과 박사 졸업
1986년 1월~1999년 12월: 한국전자통신연구원 근무
2000년 1월~현재: 국가보안기술연구소 근무