2001년 한국정보처리학회 춘계 학술발표논문집 제8권 제1호

리눅스 보안 모듈을 이용한 응용 프로그램 로깅 시스템 설계 및 구현

박남열, 송춘환, 김정일, 노봉남
전남대학교 전산학과
e-mail:pulmaru@athena.chonnam.ac.kr

Design and Implementation of An Application Program Logging System with Linux Security Module

Namyoul Park, Choonhwan H. Song, Chongil Kim, Bongnam Noh
Dept of Computer Science, Chonnam National University

요약

리눅스 시스템의 급격한 사용증가에 비해 리눅스 보안에 대한 인식은 취약한 상태이며, 응용수준에서 제공되는 로깅시스템은 위.변조의 가능성이 높다. 특히 서버 데몬에서의 감시나 추적은 전적으로 응용수준에 의존하고 있는 상태이므로 현재의 특정한 자료나 연결 요청이 공격인지 아닌지를 판단하기에는 어려움이 많다. 본 논문에서는 리눅스의 시스템 호출 로깅 모듈인 리눅스 보안 모듈(LSM:Linux Security Module)을 이용하여 서버 데몬이나 유틸리티 등에 대해 선택적으로 응용프로그램에 대한 동작상태를 감사 및 추적하여 침입여부를 판단할 수 있도록 지원하는 응용 프로그램 로깅 시스템(ALOGS:Application Program Logging System)을 설계 및 구현하고자 한다.

1. 서론

로깅시스템은 시스템 내에서 수행된 각종 응용 프로세스에 대한 내역과 외부로부터 통신망을 통해 접근하여 수행된 작업 내역 등의 정보를 기록한다. 수집된 정보는 추후에 시스템 무결성을 확인하거나, 제반 운영 정책의 효과적인 적용 여부를 확인하기 위한 분석자료로 사용된다. 또한, 불법적인 시스템 자원의 사용이나 통신망을 통한 불법 접근 등의 사건이 발생하였을 때, 사용 및 접근경로를 추적하는 감사추적 자료로 사용될 수 있다.[1].

이러한 로깅시스템을 이용하여 아무리 간단한 프로그램을 수행할 때라도 프로그램을 실행할 때 발생하는 오류 및 오류발생 시간과 오류발생 이유, 프로그램의 UID(ruid, euid)와 PID, 특정 프로그램의 사용자와 사용시간, 프로그램의 정상작동 여부와 같은 실질적인 정보를 가진다면 유용하게 사용될 수 있을 것이다. 그러나 현재 대부분의 LINUX 시스템에서 기본적으로 응용수준에서 제공되어지는 Logging Package인 wtmp, utmp, syslogd는 로그 클리너에 의해 위.변조의 가능성이 높다[5].

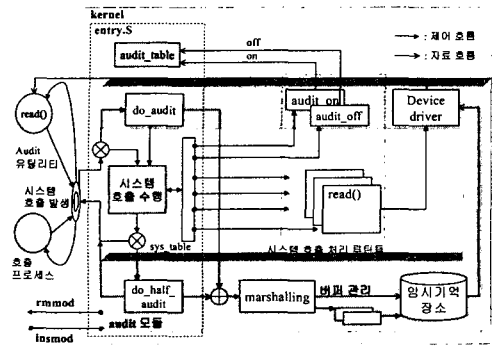
또, 각 서버 데몬들에 의해 제공되어지는 기본적인 로그 파일의 내용을 보면 세부적인 동작에 대한 명세는 존재하지 않고, 요청의 성공, 실패 여부에 따른 간단한 명세만이 존재함을 알 수 있다[8].

따라서 본 논문에서는 리눅스의 시스템 호출 로깅 모듈인 리눅스 보안 모듈(LSM:LINUX Security Module)을 이용하여 선택적으로 응용프로그램(서버 데몬, 유틸리티 등)에 대한 동작상태와 추이를 감사 및 추적하여 침입여부를 판단할 수 있도록 지원하는 응용 프로그램 로깅 시스템(ALOGS : Application Program Logging System)을 제안하였다.

2. LSM 구성 및 기능소개

LSM은 시스템 호출전 정보를 기록하는 do_audit

* 본 연구는 정보통신부 대학정보통신 연구센터 지원사업의 지원 및 한국소프트웨어진흥원의 관리로 수행하였습니다.



<그림 1> LSM 커널 모듈의 구성 요소

처리 루틴, 시스템 호출후 반환되는 정보를 기록하는do_half_audit 처리 루틴, 생성된 정보를 저장하기 위한 임시 기억장소를 유지하고 관리하는 버퍼 관리 부분, 커널내의 로그모듈의 임시 기억장소에 접근하기 위한 디바이스 드라이버, 로그파일의 크기가 급속적으로 커지는 것을 방지하기 위한 audit_filter 처리 루틴, 그리고 로그파일을 토대로 정보를 분석할 수 있는 모니터링 도구로 구성된다[7](그림 1).

2.1 do_audit 처리 루틴

audit 모듈의 기본 기능은 모든 시스템 호출에 대한 관련 정보를 생성하는 것이다. 이를 위해서 이 모듈은 커널 내에 위치하여 시스템 호출이 처리되는 바로 전후의 지점에서 시스템 호출 관련 정보를 생성하게 된다. 이러한 기능을 제공하는 부분은 audit 모듈 내에서 do_audit() 루틴(routine)과 do_half_audit() 루틴이다. 두 개의 루틴(routine)으로 분리된 이유는 시스템 호출 전 또는 시스템 호출 후에 생성할 수 있는 정보가 제한이 있기 때문이다. do_audit()루틴과 do_half_audit() 루틴을 통해서만 비로소 하나의 시스템 호출에 대해 완전한 정보를 구성할 수 있다.

do_audit() 함수는 시스템 호출이 일어나기 바로 전의 지점에서 시스템 호출 관련 정보를 생성한다. 여기서 생성되는 정보는 시스템 호출의 생성 시점, 시스템 호출 자체에 관련한 정보, 시스템 호출을 요청한 주체(subject)에 대한 정보 등을 포함한다.

시스템 호출의 생성 시점은 시스템 호출을 처리하기 직전의 시각을 의미하며 단위는 백만 분의 1초(micorsecond)이다. 시스템 호출 자체에 관련한 정보는 처리될 시스템 호출의 고유 번호, 그 시스템 호출의 매개 변수의 값들을 의미한다. 시스템 호출에 따라서 매개 변수의 수나 값의 의미가 다르기 때문에 유효한 개수의 값들을 생성한다.

시스템 호출을 요청한 주체에 대한 정보는 그 시스템 호출을 요청한 프로세스의 식별자(process id), 부모 프로세스 식별자(parent process id), 사용자 식별자(real user id), 그룹 식별자(real group id), 권한 사용자 식별자(effective user id), 권한 그룹 식별자(effective group id) 등을 의미한다[2,3].

2.2 do_half_audit() 처리 루틴

do_half_audit() 루틴은 시스템 호출이 종료한 후에 시스템 호출 관련 정보를 생성한다. 여기서 생성되는 정보는 시스템 호출 전에는 생성할 수 없었던 것들인데 시스템 호출의 반환값, 시스템 호출의 종료 시각 등이다. 또한 이러한 부분적인 정보와 do_audit() 루틴에서 생성된 부분적인 정보를 하나의 온전한 레코드 단위로 구성하여 LSM 모듈 내의 임시 기억 장소에 저장하는 기능을 수행한다.

2.3 버퍼 관리 부분

LSM 모듈은 시스템 호출 관련 정보의 생성 기능 외에 몇 가지 주요한 기능을 제공한다. 즉, 생성된 정보를 저장하기 위한 임시 기억장소를 유지하고 관리한다. 보통 간단한 명령어에 대해서도 수십 개의 시스템 호출이 발생하기 때문에 대용량의 기억장소를 확보하여야 한다. 이를 위해 LSM 모듈은 커널 함수 중에서 get_free_page() 함수를 사용하여 필요시에 충분한 만큼의 페이지(page)를 할당받는다[4].

그러나 기억장소가 가득 차게 될 가능성을 배제할 수 없는데 이 경우에는 가장 오래 전에 기록된 정보들을 버리고 새로운 정보를 기록하게 된다.

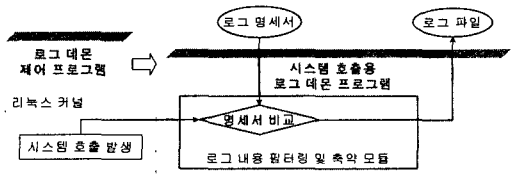
2.4 디바이스 드라이버(device driver)

LSM 모듈은 생성된 정보들을 유틸리티들이 사용할 수 있도록 하기 위해 소통점을 제공한다. 관련 유틸리티들은 직접 커널 내에 위치한 LSM 모듈의 임시 기억 장소에 접근할 수 없다[2]. 따라서 LSM 모듈은 외부 프로세스와의 소통점으로 디바이스를 생성하고 해당 디바이스 드라이버를 제공하여, 외부 프로세스가 이 디바이스에 read 명령(operation)을 적용하여 시스템 관련 정보를 읽어낼 수 있도록 한다.

2.5 audit_filter() 처리 루틴

보안 관리자는 로그 파일의 크기가 급속적으로 커지는 것을 방지하기 위하여 로깅하고자 하는 목록을

명세서에 서술할 수 있다. 그 목록은 추후에 다른 보안 프로그램에서 이용할 수 있는 것으로 선정한다. 이 로그 명세서에는 사용자 ID, 시스템 호출 종류, 기록되는 레코드의 내용을 명시할 수 있다. 리눅스 커널에서 시스템 호출이 발생하면 LSM 데몬은 로그 명세서와 비교하여 선택적으로 기록될 내용



<그림 2> LSM의 구성

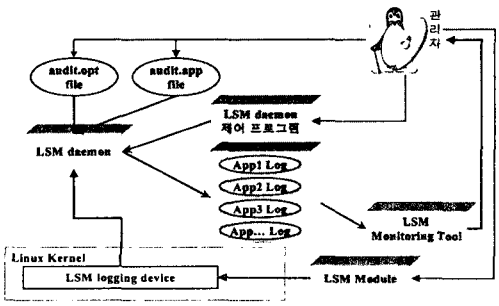
을 필터링하게 된다.

2.6 모니터링 도구

모니터링 도구에서는 시스템 호출, 사용자 식별자, 프로세스, 세션에 따라 선택적으로 결과를 보여줌으로써 관리자가 쉽게 시스템 보안을 관리할 수 있도록 해준다. LSM에서의 모니터링 도구는 기존의 Unix, 리눅스 등에 제공되어지는 로그 파일의 분석에 사용되어져 왔던 텍스트 기반의 출력의 한계를 벗어나 관리자에게 쉽게 작업하고 분석할 수 있는 GUI환경을 제공한다. 또한, 단순히 화면상의 출력만을 제공하여주는 것이 아니라 출력되어진 자료를 토대로 또 다른 정보를 출력하여 분석도구로서의 기능을 제공하고 있다.

3.LSM을 이용한 ALOGS의 구현

LSM을 이용한 ALOGS의 구현은 LSM이 동작되는 커널 버전 2.2.18이 장착된 시스템에서 구현되었다. 각 모듈들은 커널과 마찬가지로 C 언어로



<그림 3> ALOGS의 구성 및 동작

작성되었으며, 응용시스템을 로깅하기 위한 모듈도 마찬가지로 C 언어로 작성되었다. 시스템 호출을 발생(entry.S)시키는 부분에서는 어셈블리 언어를 이용하여 수정하였다.

3.1 LSM 모듈과 ALOGS

ALOGS는 LSM 커널 모듈과는 독립적으로 상위의 로깅 데몬인 LSM 데몬의 일부분으로 구현되어 응용에 대한 선택 폭을 넓힐 수 있도록 하였고, 커널 안에 삽입되는 모듈의 크기를 줄여서 커널을 융통성 있게 유지할 수 있도록 하였다.

따라서 LSM 모듈의 기능을 최대한 활용하면서, 기능제한 없이 구성하기 위하여 LSM의 자료를 읽어오기 위한 가상 장치 드라이버(/dev/audit)를 통하여 읽혀진 자료를 등록된 응용 프로그램별로 분리된 파일에 저장할 수 있도록 구현하였다.

LSM 모듈의 적재는 insmod를 통하여 이루어지며, rmmod를 통하여 커널에서 제거할 수 있다. ALOGS의 동작은 LSM 데몬이 시작될 때 읽어온 환경설정 파일에 해당 응용이 등록되었을 때만 로그 파일 저장위치(/var/log/audit)에 분리되어 저장된다. 이러한 일련의 명령어들의 사용 권한은 루트 사용자에게만 부여되어 있다[6].

3.2 ALOGS의 기능제어

LSM 모듈의 기본 기능은 모든 시스템 호출에 대한 관련정보를 생성하는 것이다. 모듈이 커널 내에 위치한 상태에서 LSM 데몬(auditd)이 로그명세서(audit.event, audit.group, audit.opt)에 기록된 정보를 토대로 User별, System Call을 Class로 구성한 System Call Group별로 로그 관련정보를 생성해 낸다. 이후에 LSM 데몬 내에 삽입되어 있는 ALOGS가 등록된 응용 별로 로그 관련정보를 분리하여 저장하게 되는 것이다.

단, 위와 같은 동작순서로 인해 루트 사용자만이 동작시킬 수 있는 특정 응용(서버 데몬 등)에 대해서는 루트 사용자의 시스템 호출 생성을 광범위하게 기록할 수 있도록 설정하여 응용들의 동작을 세부적으로 살펴볼 수 있도록 로그 명세서를 변경하는 것이 좋다. 반대로 가장 많은 시스템 호출정보를 발생시키는 루트 사용자의 시스템 호출의 발생량을 증가시키기 때문에 시스템의 성능저하를 초래할 수 도 있다.

3.3 응용프로그램 로그 정보의 저장 및 분석

리눅스 커널과 LSM 모듈에 의해 생성된 시스템

호출 관련 자료를 보관하기 위해서는 모듈의 임시 기억장소로 접근하여 로그 파일에 저장하여야 한다. 또한 이러한 정보들을 분석하여 유용한 보안 활동을 할 수 있다.

ALOGS는 리눅스 커널 내부에 위치한 LSM 모듈의 임시 기억장소에 접근하여야 한다. 그러나 사용자 프로세스가 커널 내부에 직접 접근할 수 없기 때문에 모듈에 의해 제공되는 가상 커널 장치드라이버(/dev/audit)를 통해 시스템 호출 관련정보를 읽어 로그파일에 저장하게 된다. 이 때 관리자가 로깅을 하고자 하는 응용에 대한 정책파일인 audit.app에 따라 시스템 호출 관련정보를 선별하여 각 응용 로그 파일에 저장하게 된다.

ALOGS 제어 프로그램은 시스템 호출 관련정보를 제어한다. 즉, 필요에 따라 시스템의 정보 생성 기능을 취소하여 LSM 모듈이 더 이상 자료를 생성하지 않도록 하면 이에 따라 ALOGS도 로그 파일에 더 이상 자료를 기록하지 않게 된다. ALOGS 제어 프로그램은 본 논문에서 생성한 새로운 시스템 호출인 audit_on과 audit_off를 실행하여 동작한다.

이렇게 ALOGS에 의해 응용 로그파일에 저장된 자료는 시스템 호출 모니터링 도구에 의해 사용자, 프로세스, 시스템 호출로 분류하여 체계적으로 분석할 수 있다. 따라서 관리자는 audit.opt와 audit.app 파일을 기술하고, ALOGS 제어 프로그램으로 로깅을 시작한 후에 X-Windows 기반 시스템 모니터링 도구인 xalogs에 의해 로그 파일을 분석할 수 있다.

4. 결론 및 향후 연구과제

본 논문에서는 LINUX 시스템의 보안성 향상을 위해 시스템 호출기반 로깅 패키지인 LSM을 이용하여 ALOGS를 설계하고 제안하였다.

현재 많은 서버들의 구성은 여러 기능을 복합적으로 하는 서버가 아닌 한 대의 서버가 한 개의 서비스 만을 전담하는 형태로 제공됨으로써 시스템에 대한 전체 시스템 호출을 로깅한다는 것은 비효율적일 수 밖에 없다. 이에 구현된 각 모듈은 다양한 분류 옵션에 의하여 특정한 사용자의 특정한 시스템 호출만을 필터링하여 로깅할 수 있고, 또 특정 응용에 대해서만 로깅할 수 도 있어 시스템 오버헤드에 대한 문제를 해결하였다. 그리고 커널의 부담을 최소화하기 위하여 모듈로 구성함으로써, 필요시 모듈을 삽입하고 제거하는 방식으로 시스템의 효율성을 향상시킬 수 있었다. 그러나, LSM 모듈이 추가됨으로써 시스템에 부하가 늘어 시스템의 속도 저하를 가져오게 된다는 문제가 있으므로 성능에 따라 로깅이나 재구성 측면에서 차이가 많이 나게 된다.

향후에는 생성된 ALOGS를 통하여 생성된 로그파일을 토대로 침입탐지를 위해 필요한 최소한의 집합을 계산하여 자료를 축약하고, 응용 프로그램에 대한 침입을 탐지하기 위한 패턴을 정의하는 등의 응용프로그램에 대한 침입탐지 연구가 필요하다.

참고문헌

[1] Sun Microsystems, Inc. SunSHIELD Basic Security Module Guide, Sun Microsystems, Inc. 2000

[2] Daniel P. Bovet, Macro Cesati, Understanding the LINUX Kernel, OReilly & Associates, Inc, 2001

[3] D. Verworner, M. Beck, M. Bohme, M. Dziadzka, R. Magnus and U. Kunitz, LINUX Kernel Internals 2/E, Addison-Wesley Publishing Company, Inc. 1997

[4] Remy Card, Eric Dumas, Franck Mevel, The LINUX Kernel book, John Wiley & Sons, 1998

[5] Anonymous, Maximum LINUX Security, SAMS Publishing, 2000

[6] Ori Pormerantz, LINUX Kernel Module Programming Guide version 1.1.0, LINUX Document Project, Apr., 1999

[7] 전남대학교, 리눅스 보안 소프트웨어 개발에 관한 연구, 한구전자통신연구원, 1999년 12월

[8] Ben Laurie, Peter Laurie Daniel, Apache: The Definitive Guide, 2nd Edtion, OReilly & Associates, Inc, 1998