시스템 호출 추적을 통한 Linux 시스템에서의 침입 탐지

신 동 철, 조 성 제
단국대학교 전산통계학과

Intrusion Detection on Linux System via System Call Traces

Dongcheol Shin, Seongje Cho
Dept. of Computer Science and Statistics, Dankook University

요 약

최근 Linux 시스템의 사용이 급격히 증가함에 따라 Linux 시스템에서의 보안이 큰 문제로 대두고 있으며 그에 대한 대응책으로 침입 탐지 및 방지를 위하여 여러 가지 방안들이 나오고 있다. 그러나 대부분의 침입 탐지 및 방지 시스템이 쉘이나 응용 수준에서 이루어지며 이 경우 침입자가 그 쉘 또는 응용 등을 회피하면 그 효용성이 없어진다는 단점이 있다. 따라서 본 논문에서는 Linux 시스템의 커널 수준에서 명령들의 시스템 호출 과정을 추적하여 침입을 탐지하고 보안을 강화시켜 주는 시스템인 LiSID(Linux System Intrusion Detector)를 제안하였다.

1.서론

중소 규모의 서버에서 Linux 시스템의 사용이 증가함에 따라 Linux 시스템을 대상으로 하는 침입 사례도 증가하고 있다. 침입이란 "자원의 무결성(integrity), 기밀성(confidentiality), 가용성 (availability)을 훼손시키려 하는 일련의 행동"이라 정의된다[7]. 시스템에 접근한 사용자에 의한 침입을 탐지(detection)하는 방법은 오용 탐지(Misuse Detection)와 변칙 탐지(Anomaly Detection), 두 가지로 나누어 질 수 있다. 오용 탐지란 이전에 이미 알려진 시스템의 취약점이나 침입 방법을 이용하여 시스템에 침입하는 것을 탐지해 내는 것을 말하며, 변칙 탐지란 기존의 정상적인 행위(normal behavior)에서 벗어나는 행위(anomaly behavior)가 있을 경우 이것이 침입인지 아닌지를 결정하는 것을 말한다[3].

오용 탐지 시스템으로는 COPS, SATAN, STAT, swatch 등이 이미 존재하나 이들 시스템은 침입이 발생하여도 다음번에 탐지 시스템이 실행될 때까지 탐지가 안 되고(COPS, SATAN 등) 이미 알고 있거나 새로 발견된 시스템의 취약점을 탐지 시스템에게 전달하여야만 그 침입을 탐지해 낼 수 있으므로, 탐지 시스템에 저장되어 있는 것과 다른 방법의 침입이 있을 경우 알아내기가 어렵다는 단점이 있다[5][6]. 변칙 탐지 시스템으로는 IDES 등이 있으며 IDES의 경우 사용자나 프로그램이 사용하는 CPU, I/O등 시스템 특성을 기반으로 하기 때문에 서로 다른 시스템에 대해서는 적용하기가 어렵다는 단점이 있다. 또한 쉘이나 프로그램 수준의 침입 탐지 및 방지 시스템의 경우 침입자가 해당 쉘이나 프로그램을 회피하면 그 효용성이 상실된다는 단점을 가진다.

따라서 본 논문에서는 Linux 시스템에서 침입자가 침입 시 사용하게 되는 주요 시스템 호출을 커널 수준에서 관리함으로써 오용 탐지를 수행하고, 시스템 침입에 사용 될 수 있는 프로그램에 대하여 시스템 호출을 추적, 기록함으로써 변칙 탐지를 수행할 수 있는 시스템인 LiSID(Linux System Intrusion Detector)를 구성하였다.

2. LiSID(Linux System Intrusion Detection)의 구조 및 기능

LiSID는 기존에 알려진 방법을 사용하여 침입을 시도할 경우 이를 시스템 호출 수준에서 감지, 차단하는 오용 탐지 시스템인 LMD(Linux Misuse Detector)와 의심이 가거나 침입에 사용될 수 있는 프로그램의 시스템 호출을 추적, 기록 후 감사를 통하여 변칙 탐지를 수행하는 LAD(Linux Anomaly Detector)로 이루어진다.

2.1. 오용 탐지(Misuse detection)

오용 탐지란 이미 알려져 있는 공격 방법을 이용하여 시스템의 취약부분을 찾아 침입하는 행위를 탐지해 내는 것을 말한다. 기존에 존재하는 오용 탐지 시스템의 경우 직관적이고 사용자가 사용하기 쉽다는 장점이 있으나 모든 사례에 대하여 각각 지정해야 하는 복잡함이 있었다. 예를 들어 네트워크를 통한 침입에 흔히 사용되는 .rhosts에 대한 비합법적인 접근의 경우 sendmail이나 passwd 프로그램의 취약점을 사용하는 등의 침입 방법이 있는데 통상의 오용 탐지 시스템은 이들에 대하여 각각 별개의 정보를 지정하여 주어야 한다. 다시 말해서 sendmail의 취약점을 이용하여 .rhosts에 접근하는 경우에 대비하기 위하여 sendmail의 취약점을 오용 탐지 시스템에 알려주는 것과는 별도로 passwd의 취약점 역시 탐지 시스템이 가지고 있어야 한다는 것이다.

본 연구에서는 커널 수준에서 시스템 호출을 감시하여 오용 행위(misuse behavior)를 탐지하는 시스템인 LMD를 제안한다. 위의 경우 .rhosts 파일에 대한 open 시스템 호출이 발생하였을 경우 그 호출자(시스템 호출을 호출한 프로그램)와 호출대상(즉 열려야 할 파일, .rhosts) 및 그 외 정보(사용자의 uid, euid, gid, 시간 등)를 참조하여 시스템 호출의 실행이 허가되어 있는 상황이 아닌 경우 봉쇄하거나 사전에 지정되어 있는 동작을 하도록 유도함으로써 침입 방법에 관계없이 결과적으로 침입을 탐지, 봉쇄하게 된다. 또한 허용되지 않은 시스템 호출을 사용한 프로그램은 이후 다시 또 다른 방식의 침입에 사용될 수 있다고 간주되어, LMD는 그 프로그램에 대한 정보를 LAD에 전달, 이후 침입에 대응한다.

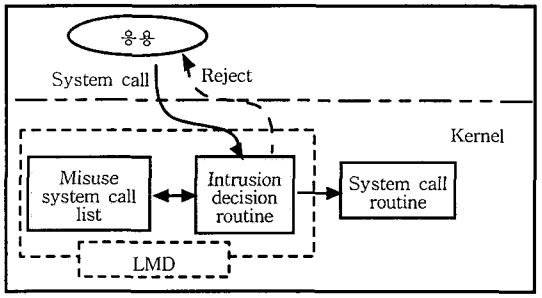

그림 1. 오용 탐지(LMD)의 동작

2.2. 변칙 탐지(Anomaly detection)

변칙 탐지란 일반적인 행위에서 벗어난 일련의 행위가 발생하였을 때 그러한 행위가 침입인지 정상 행위의 한 부분인지를 판단하는 것을 말한다. 변칙 탐지는 오용 탐지처럼 발생 즉시 알 수 있다기보다는, 의심스러운 행위가 미리 추출된 정상 행위의 특성에서 얼마나 벗어나 있는가를 통하여 추정할 수 있는 경우가 대부분이다. 침입에 대한 대응 방법은 실시간으로 처리되는 것이 가장 좋다. 침입이 발생하였을 때, 그 즉시 침입자에 대하여 대응할 수 있다면(침입자의 움직임을 봉쇄한다던가 극단적으로는 시스템을 shut down 시키는 등) 피해를 최소화할 수 있다. 그러나 오버 헤드로 인해 실시간으로의 처리가 여의치 않다면 차후에 (침입 중 또는 후에) 관리자가 시스템 감사를 통하여 침입 사실을 발견하고 조치할 수 있도록 그 기반이 되는 정보를 기록하는 것이 차선책이라 할 수 있다. 이 경우 관리자는 기록된 정보를 토대로 침입 사실, 침입 방법, 피해 정도 등을 알아내어 같은 방법의 침입이 다시 발생하는 것을 방지하여 더 이상의 피해를 막을 수 있다[4].

LAD는 시스템 내에서 사용자가 사용 가능한 프로그램 중 변칙 행동에 의한 침입에 사용될 수 있는 프로그램들의 시스템 호출을 기록함으로써 이후 시스템 관리자가 침입 여부를 판단하고 그 방법을 찾아내는데 필요한 정보를 제공한다. LAD는 침입에 사용될 수 있는 취약한 프로그램을 선정하여 등록한 후 시스템 호출이 발생하면 그 시스템 호출을 사용한 프로그램이 취약 프로그램일 경우 그 시스템 호출을 기록한다. 취약 프로그램 명세는 관리자가 취약한 프로그램이라고 판단하여 등록한 것과 허용되지 않은 시스템 호출을 호출하여 오용 탐지 시스템인 LMD가 취약 프로그램으로 선정한 것들로 이루어진다.

변칙 탐지를 올바르게 수행하려면 먼저 취약 프로그램들 각각의 정상 행위에 대한 정보를 유지하면서 비정상적인 행위가 발생했을 경우 비교 기준으로 삼아 침입 여부를 결정해야 한다. LAD에서는 관리자가 다양한 조건과 변수를 가지고 취약 프로그램을 실행하여 일종의 데이터 베이스로 구축하는 선 처리 과정이나 취약 프로그램의 실행 시 기록된 추적 파일을 관리자가 선별하는 과정을 통하여 정상 행위에 관한 정보를 획득하게 된다. 또한 변칙 탐지를 위하여 사용되는 감사 모듈의 성능을 고려하여 정상 행위에 대한 정보의 양을 결정하여야 한다. 정상 행위에 대한 정보가 너무 적으면 올바른 침입 탐지가 어려우며 너무 많은 경우 저장공간의 낭비가 생기므로 올바른 판단을 할 수 있는 최소의 양으로 결정한다.

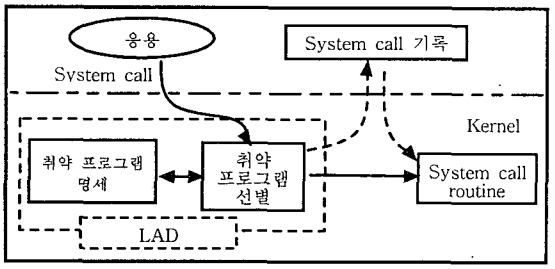

그림 2. 변칙 탐지(LAD)의 동작

3.구현

LiSID는 Linux 커널 2.2.9(Red Hat 5.2) 시스템에서 구현되었다. 각 모듈들은 커널과 마찬가지로 C 언어로 작성되었으며 일부는 어셈블리 언어를 사용하였다[8][9].

3.1. LMD

LMD는 새로운 커널 수준 모듈인 LMD_Mod와 새로운 시스템 호출인 LMD_syscall로 구성된다. LMD_Mod은 시스템 호출이 사용되었을 경우 그 시스템 호출이 침입에 사용된(또는 사용될 수 있는) 시스템 호출인가를 확인하고 적절한 조치를 취하는 기능을 하며 LMD_syscall은 LMD_Mod 모듈에게 어떤 시스템 호출이 어떠한 조건에서 호출되었을 경우 그 시스템 호출을 봉쇄하거나 감시하여야 하는 가를 알려주는데 사용된다. 또한 비정상적인 시스템 호출을 시도한 프로그램에 대한 정보는 LAD의 LAD_syscall을 사용하여 LAD에 전달된다(3-2 참조).

3.2. LAD

LAD는 커널 수준에서 시스템 호출을 추적하므로 추적해야 할 프로그램을 커널에 알려주는 시스템 호출인 LAD_syscall과 시스템 호출을 추적하고 기록하는 모듈인 LAD_Mod 그리고 기록된 내용이 정상 행위인지 침입인지를 구별하는 모듈인 LAD_Dis로 이루어진다.

취약 프로그램으로 선정된 프로그램 명들은 LAD_syscall을 통하여 LAD_Mod에 등록된다. 사용자가 취약 프로그램을 실행하게 되면 LAD_Mod는 해당 프로세스에 대한 정보를 담고 있는 구조체의 flags 변수에 PF_TRACE 비트를 설정한다. 그리고 이후 이 프로세스가 사용하는 시스템 호출을 기록할 파일을 만들고(프로그램명.pid 형태) 해당 프로세스에 대

한 정보를 기록한다. 시스템 호출이 발생하면 LAD_Mod는 시스템 호출을 사용한 프로세스의 구조체의 PF_TRACE 비트가 설정되어 있는지를 확인하고, 설정되어 있을 경우 미리 준비되어진 파일에 그 시스템 호출을 기록한다. LAD_Dis는 사용자 수준의 모듈로 구현되었다. LAD_Dis는 기 구축되어 있는 정상 행위 정보와 취약 프로그램이 기록한 정보를 비교하여 침입으로 판정될 경우 관리자에게 알리고, 그렇지 않은 경우 해당 프로그램의 정상 행위 데이터 베이스에 추가한다. passwd의 정상 행위와 변칙 행위 그리고 buffer overflow 공격을 추적한 것에 대하여 단순 비교를 사용하는 LAD_Dis를 적용하였을 경우의 결과가 아래 표에 나와 있다. 각 숫자는 정상 행위(case 1,2,3)에 대하여 각각이 얼마나 차이가 나는가를 백분율로 나타낸 것이다.

Anomaly behavior / Normal behavior	정상 동작 (사용상의 오류 등)				침입 시도	
	case A	case B	case C	case D	case E	case F
case 1	36.21	43.78	48.09	30.54	66.82	67.58
case 2	56.54	35.21	55.37	43.00	62.91	63.71
case 3	48.46	48.31	43.32	29.63	65.05	65.83

표 1. LAD를 이용한 passwd의 추적 결과

4. on-line 침입 탐지 시스템 모델

on-line 침입 탐지 시스템은 취약 프로그램이 실행되면 LAD에 의하여 추적이 되며 각 시스템 호출은 LMD에 의하여 허용 여부가 판단되어 진다. 허용되지 않은 시스템 호출을 한 프로그램에 대한 정보는 LAD의 취약 프로그램 명세에 추가된다. LAD는 일정 주기마다 취약 프로그램의 기록을 감사하여 침입 여부를 판단하며, 침입의 경우 사용된 시스템 호출과 그 조건을 LMD에 전달하고 정상 행위일 경우 정상 행위 데이터 베이스에 추가한다.

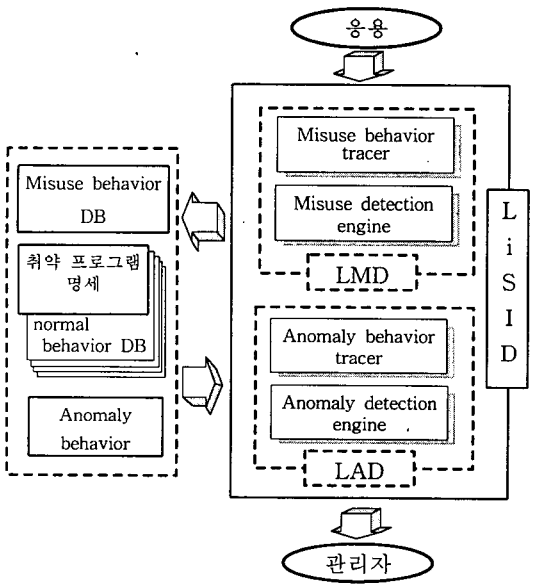

그림 3. on-line 침입 탐지 시스템

5. 결론 및 향후 연구

본 논문에서는 Linux 시스템의 보안성 향상을 위하여 시스템 호출을 추적, 감시하여 오용 탐지와 변칙 탐지를 어떻게 수행할 것인가에 대하여 LiSID 시스템을 제안하였다. 상황에 따라 시스템 호출의 허용/비허용을 결정함으로써 오용 탐지를 수행하였으며 취약 프로그램이 호출한 시스템 호출을 추적함으로써 변칙 탐지를 수행할 수 있도록 하였다.

그러나 시스템 호출에 추적하는 루틴이 추가됨으로서 인하여 시스템에 부하가 늘어 시스템의 속도 저하를 가져오게 된다는 문제가 있으며 LAD_Dis 모듈의 성능에 따라 속도나 탐지 효율이 차이가 많이 나게 된다. 향후에는 LAD_Dis 모듈의 성능을 개선시키고 커널에 추가되는 루틴의 성능을 향상시킴으로서 시스템 스스로가 다양한 침입에 대하여 빠르게 대응할 수 있는 탐지 시스템에 관한 연구가 필요하다.

참고 문헌

[1] Andrew P. Kosoresow and Steven A. Hofmeyr, "Intrusion Detection via System Call Traces," IEEE Software, pp. 35-42, September/October 1997.

[2] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff, "A Sense of self for Unix Processes," In Proceedings of the 1996 IEEE Symposium on Security and Privacy, pages 120-128, Los Alamitos, CA, 1996. IEEE Computer Society Press.

[3] W. Lee and S. J. Stolfo, "Data Mining Approaches for Intrusion Detection," In Proceedings of the 7th USENIX Security Symposium. San Antonio, Texas, 1998

[4] S. Garfinkel and G. Spafford, Practical UNIX and Internet Security, O'Reilly & Associates, Inc., 101 Morris Street, Sebastopol, CA 95472, 1996.

[5] SATAN (Security Administrator's Tool for Analyzing Networks) satan-1.1.1.tar.gz from http://www.fish.com/satan

[6] P. A. Porras, "STAT: A State Transition Analysis Tool For Intrusion Detection," M.S. thesis, Univ. of California, Santa Barbara, July 1992.

[7] R. Heady, G. Luger, A. Maccabe, and M. Servilla, "The architecture of a network level intrusion detection system," Technical report, Dept. of Computer Science, Univ. of New Mexico, Auguest 1990.

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

[9] W. R. Stevens, Advanced Programming in the UNIX Environment, Addison-Wesley Publishing Company, Inc. 1992