基于 LSM 的程序行为控制研究*

张 衡[1+], 卞洪流[1], 吴礼发[2], 张毓森[2], 崔明伟[1], 曾庆凯[3]

[1](解放军理工大学 通信工程学院,江苏 南京 210007)

[2](解放军理工大学 指挥自动化学院,江苏 南京 210007)

[3](南京大学 计算机科学与技术系,江苏 南京 210093)

Study on Program Behavior Control Based on LSM

ZHANG Heng[1+], BIAN Hong-Liu[1], WU Li-Fa[2], ZHANG Yu-Sen[2], CUI Ming-Wei[1], ZENG Qing-Kai[3]

[1](Institute of Communication Engineering, PLA University of Science and Technology, Nanjing 210007, China)

[2](Institute of Command Automation, PLA University of Science and Technology, Nanjing 210007, China)

[3](Department of Computer Science and Technology, Nanjing University, Nanjing 210093, China)

+ Corresponding author: Phn: +86-25-80828020, E-mail: e_zhheng@sohu.com

Received 2003-09-09; Accepted 2004-06-10

Zhang H, Bian HL, Wu LF, Zhang YS, Cui MW, Zeng QK. Study on program behavior control based on LSM. *Journal of Software*, 2005,16(6):1151−1158. DOI: 10.1360/jos161151

Abstract: Program behavior control is an active detection mechanism. The research of program behavior control mainly focuses on four aspects: audit data selection, behavior description, the establishment of normal behavior and behavior matching. This paper investigates the event sequence model and proposes the use of LSM(Linux security modules) as an alternative data source to system calls. Based on the data quality analysis and execution results from real systems, the efficiency of the LSM data source is verified from both theoretical and practical points of view. Results show that, because of its more refined granularity and its better security relevance, LSM data source is more suitable for the audit events used in event sequence models.

Key words: program behavior control; model of short sequence; LSM(Linux security modules)

摘 要: 程序行为控制作为一种主动检测机制,主要在 4 个方面进行研究:审计数据源选择、行为描述、正常行为模式的建立与行为匹配.对事件序列模型作了深入研究,提出了采用另外一种与系统调用完全不同的数据源——LSM(Linux security modules,简称 Linux 安全模块)截获点,并从理论和实践两个方面来验证 LSM 数据源的有效性,即基于信息理论的数据质量分析和实际系统的运行结果分析.结果表明,由于LSM数据源的粒度更细以及和安全更相关,使得它更适合作为事件序列模型的审计事件.

* Supported by the National High-Tech Research and Development Plan of China under Grant No.2002AA141090 (国家高技术研究发展计划(863))

作者简介: 张衡(1977－),男,安徽巢湖人,博士生,主要研究领域为安全操作系统,人工免疫系统;卞洪流(1976－),男,工程师,主要研究领域为计算机网络;吴礼发(1968－),男,副教授,主要研究领域为网络管理,网络安全;张毓森(1949－),男,教授,博士生导师,主要研究领域为指挥自动化系统,系统仿真,信息安全;崔明伟(1978－),男,硕士生,主要研究领域为计算机安全;曾庆凯(1963－),男,教授,主要研究领域为网络安全,分布计算.

关键词: 程序行为控制;短序列模型; Linux 安全模块(LSM)
中图法分类号: TP309 文献标识码: A

随着社会信息化、网络化进程的逐渐深入,信息系统的安全变得越来越重要.程序行为控制是针对当前系统安全问题的重新思考而引入的.安全信息系统中,访问控制占据着极其重要的地位,它主要对系统中的主体为完成某项功能而赋予的权限进行限制;程序行为控制则保证程序能够按照预期设计的方式运行.访问控制从主客体的角度来设置权限,对特定程序的实际可能的动作并不了解,因此,尽管有 DTE,Capability 等模型,访问控制的监控粒度比行为控制要粗.因此,对程序的行为进行监控,防止它造成危害是十分必要的,而且也是访问控制所不能完成的.程序行为控制是一种主动的检测方式,它监控系统中的关键进程,在其异常时及时地予以报警,并通过人工或自动的方式采取对应的响应措施.

程序行为控制的基本原理是根据程序的行为或资源使用状况的正常程度来判断是否被用户恶意使用.它一般通过对程序行为建模,监控其行为是否符合行为模型.其关键问题在于正常行为模型的建立.行为控制基于这样一个假设:无论是程序的执行还是用户的行为,在系统特性上都呈现出紧密的相关性.这些带有强一致性的行为特征正是我们希望进行统计的行为模式[1].

本文中程序行为控制在操作系统中实质上是对进程的行为进行监控,是动态概念,为了统一,文中均采用程序行为控制.

1 相关工作

目前,行为控制的研究主要有 4 个方面:审计数据源选择、行为描述、正常行为模式的建立、行为匹配.在这 4 个方面中,行为描述是核心.

1.1 审计数据源选择

程序行为控制实质上归结为对所选取的被监控程序审计数据的处理.因此,审计数据源质量的好坏直接影响到行为控制的效果.

目前,审计数据源主要来源于两个方面:

- 系统调用:它是操作系统内核与应用程序的接口,任何对客体的访问都必须通过系统调用;同时,UNM 的研究小组[2,3]发现:对一个特定的程序,其系统调用序列片段是相当稳定的.因此,较常采用系统调用作为数据源.
- 系统审计数据:由于审计数据大部分有较为丰富的语义,不少研究者采用它作为数据源.具体实现时,大部分使用 SUN 的 BSM(基本安全模块),它为实现提供了方便.

将上面两者结合使用,同时加入系统调用的参数,已成为一种趋势.这种方法一般把基础数据源赋予语义,定义成审计事件,使得它具有更丰富的语义,在执行时具有更高的效率[4].

1.2 行为描述

行为描述是行为控制的核心,它提供描述程序行为的方法.描述程序行为的关键点是:使用一组稳定的特征来定义程序的行为.目前,主要有 3 种行为描述方法:

- 事件发生序列[5]:即程序状态发生的顺序,程序的正常行为都可以由该程序正常运行时产生的事件序列来描述;
- 有限自动机[6,7]:这种方法将审计数据源的每个审计事件看成一个状态,通过分析程序,可以得到程序的状态转换图;
- 高层描述[8−10]:这种方法一般使用有确切语义的语言来表达程序能做什么,不能做什么,具有确切的意义.

1.3 安全行为模式的建立与行为匹配

选定数据源和行为描述方式后,需要得到大量程序运行后的审计数据.一些组织和研究小组公开了在他们的实验环境下得到的数据.

有了大量的审计数据后,需要对这些数据进行学习和处理,以得到程序安全行为的模式.目前,安全行为的建立及其行为匹配主要有以下几种方法:马尔可夫模型方法[11]、Bayesian 概率网络方法[12]、基于规则的方法[8−10,13]、决策树方法[14]、数据挖掘方法[15,16]、隐马尔可夫模型[17]、自动机的方法[6]、系统调用短序列方法[2,3,18]、神经网络方法[19]等.

经过对比实验[19]发现,除了隐马尔可夫模型比系统调用短序列好些以外,其他模型相差不大,系统调用短序列方法甚至在某些方面更突出,而且其实现难度、对性能的影响均非常小.这说明由于系统调用序列能够表达系统的规律,简单模型一样可以工作得很好.我们的工作主要借鉴了系统调用短序列方法的思想,通过监控系统的特征序列来实现程序行为控制,但所用数据源不是系统调用,而是 LSM(Linux security modules,简称 Linux 安全模块)数据源.

2 短序列模型描述

下面形式化地描述短序列模型[20,21].

定义 1. 审计事件.系统中所有被记录的事件为审计事件.所有审计事件集合 $E=\{e_1,e_2,e_3,\ldots,e_n\}$.

定义 2. 系统踪迹.在系统 S 中,产生的审计事件序列 $v_1,v_2,\ldots,v_i$,称为系统踪迹,其中 $v_i\in E$.

每个审计事件有两个属性:时间属性和属主属性.时间属性标为 $C(v_i)$,表示事件发生的时间,所有事件都是按时间排序的,即对所有的 $i\geq 1, C(v_i)<C(v_{i+1})$;属主属性标为 $O(v_i)$,表示事件的发起主体,一般用进程 pid 或(user,pid)表示.

定义 3. 进程踪迹.对某一进程 p_i,其事件序列 $v_{1,i},v_{2,i},\ldots,v_{i,i},v_{i+1,i}$,称为进程踪迹.进程踪迹是系统踪迹的子串.

定义 4. 踪迹合并.给定 V 的两个子串 V_1 与 V_2,它们的合并记为 $V_1\odot V_2$,按事件的时间属性将子踪迹事件合并排列.

定义 5. 短序列.设 k 为滑动窗口的大小,被监控程序在运行时产生审计事件序列 $V:v_1,v_2,\ldots,v_m(m\geq k)$,用滑动窗口在审计事件序列上滑动,滑动窗口内的 k 个事件序列$(v_i,v_{i+1},\ldots,v_{i+k-1})$构成短序列.

定义 6. 正常短序列模式库.设被监控程序在正常运行时产生审计事件序列 $V:v_1,v_2,\ldots,v_m(m\geq k)$,$V$ 可由被监控程序在系统中的进程踪迹合并而成.

设 P 为正常短序列模式库,k 为窗口大小,则:

$$P=\{(s_i,s_{i+1},\ldots,s_j)|s_i,\ldots,s_j\in E, i\geq 1, j\leq m, j-i+1=k, s_i=v_i,s_{i+1}=v_{i+1},\ldots,s_j=v_j\}.$$

正常模式库的完备性是由正常运行的完备性决定的,因此应尽量考虑各种不同情况.

定义 7. 短序列匹配.设窗口为 k 的审计事件序列 $u=v_i,v_{i+1},\ldots,v_{i+k-1}$,若 $u\in P$,则定义此短序列为匹配;否则为异常.

Forrest 的短序列方法采用的审计事件为程序运行时产生的系统调用.我们采用了另外一种完全不同的数据源——LSM 截获点,下面对 LSM 作简单描述.

3 LSM 概述

LSM[22](Linux security modules,Linux 安全模块)项目是由 Wirex 发起、开发的一个框架结构,目前已有 2.4 内核和 2.5 内核的补丁.LSM 为 Linux 内核的访问控制提供统一的支持.

这种框架对系统需要保护的资源进行分析,确定哪些是要保护的客体,在对客体进行访问的最终的函数中插入 hook,截获访问,调用安全机制,并通过另外一些 hooks 修改客体对应的数据结构,以满足安全机制的需要.LSM 不改变 Linux 原有的系统调用过程以及原有的安全机制,只是在对客体进行操作的函数中加 hook,以截获对客体的访问,hook 将指向安全机制模块(security module).安全模块是独立起作用的,不会影响 Linux 原有的安全机制.

从上面的介绍可以看出,LSM 的截获点在系统调用或内核函数内,它的粒度比系统调用还细,而且都位于对各种资源的访问点上,与安全关系密切;系统调用记录进程与内核交互的动作序列,LSM 记录进程访问资源的序

列,更安全相关;同时,作为一个框架,其验证、实现都很方便;最后,作为开放源代码的 Linux 操作系统适合我们对其作修改.因此,非常适合作为行为控制的数据源.

4 LSM 数据质量的理论分析

4.1 实验数据的获取

在理论上,我们主要采用基于信息理论的数据分析方法,检验基于 LSM 的数据源与系统调用数据的优劣.通过人工构造实验环境,模拟各种可能的情况,收集验证数据:在 Linux 平台上,正常的运行 wu-ftp2.6,这时分别从系统调用监控器和 LSM 监控器得到相同行为的不同数据.实际运行时,我们考虑了其中的一些可变的因素,最后收集的数据是较为完备的.实验内容如下所示.

1	上传	不同大小文件 9 种(1K~409.6M)
		连续上传文件的数量 5 种(1 个~128 个)
		不同文件类型(24 种)
	下载	不同大小文件 9 种(1K~409.6M)
		连续下载文件的数量 5 种(1 个~128 个)
		不同文件类型(24 种)
2	执行其他命令	共 73 条命令
3	不同用户各完成 1、2	4 个用户,一个匿名用户

4.2 基于信息理论的数据分析

在异常检测中,正常行为的学习依赖于学习数据的质量,但数据的质量难以评估.针对这个问题,Wenke Lee[23]提出了利用信息论的某些概念:熵、条件熵、相对熵和信息增益,可以定量地描述一个数据集的特征,分析数据源的质量.下面给出一些用于数据源质量分析的定义.

定义 8. 给定数据集合 X,对任意 $x\in C_x$,定义熵为 $H(X)=\sum_{x\in Cx}P(x)\log\frac{1}{P(x)}$.

在数据集中,每个唯一的记录代表一个类,熵越小,数据也就越规则,根据这样的数据集合建立的模型的准确性越好.根据实验数据,计算出这两种数据源的熵值如图 1 所示.

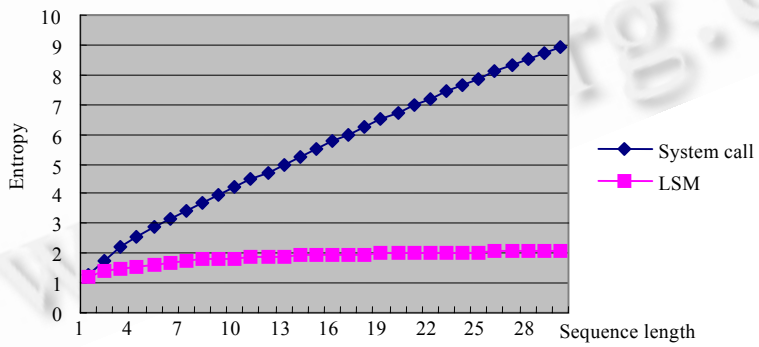

Fig.1 Entropy of system call and LSM

图 1 熵对比图

定义 9. 定义条件熵为 $H(X|Y)=\sum_{x,y\in C_x,C_y}P(x,y)\log\frac{1}{P(x|y)}$.

其中,$P(x,y)$为 x 和 y 的联合概率,$P(x|y)$为给定 y 时 x 的条件概率.安全审计数据通常都具有时间上的序列特征,条件熵可以用来衡量这种特征,按照上面的定义,令 $X=(e_1,e_2,\ldots,e_n)$,令 $Y=(e_1,e_2,\ldots,e_k)$,其中 $k<n$,条件熵 $H(X|Y)$可以衡量在给定 Y 以后,剩下的 X 的不确定性还有多少.条件熵越小,表示不确定性越小,从而通过已知预测未知的

可靠性越大.

根据实验数据,假设滑动窗口的大小为 n,窗口中从 1~n 有 n 个系统调用,用 X 表示,其中以从 1~n–1 个系统调用作为属性,用 Y 表示,第 n 个系统调用作为分类.计算条件熵 $H(X|Y)$.同样,对 LSM 数据计算其条件熵.计算的结果如图 2 所示.

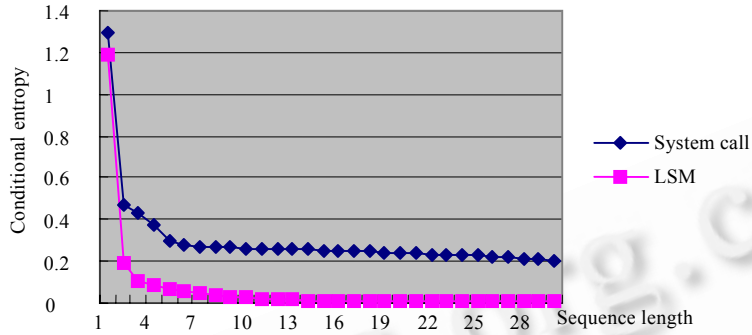

Fig.2 Conditional entropy of system call and LSM

图 2 条件熵对比图

从图 1、图 2 可以看出,LSM 数据的熵与条件熵明显小于系统调用的熵与条件熵, LSM 数据是有一定优势的.同时,从图 2 也可以看出,对系统调用,当滑动窗口大于 6 后,条件熵变化不大,因此 K 取 6 是合适的;而对于 LSM 数据源,K 取 10 较为合适,因此,LSM 数据不足之处是实际使用时,所需的窗口较大,计算复杂度有所增加.

5 基于 LSM 数据源的程序行为控制系统 PB 的实现与分析

为了从实践上验证采用 LSM 作为行为控制的有效性,我们设计、实现了基于 LSM 数据源的程序行为控制系统 PB.

5.1 设计与实现

PB 的框架如图 3 所示.被监控进程列表记录需要监控程序的每个进程,每个进程对应一条记录;执行映像列表记录被监控程序的执行映像,每个执行映像对应一条记录;正常模式库是每个被监控程序的短序列集合被加载到内存后的结构,为了便于在线匹配,模式库以森林形式存在.在系统运行过程中,被监控进程列表根据进程的加载、派生、退出情况动态更新.

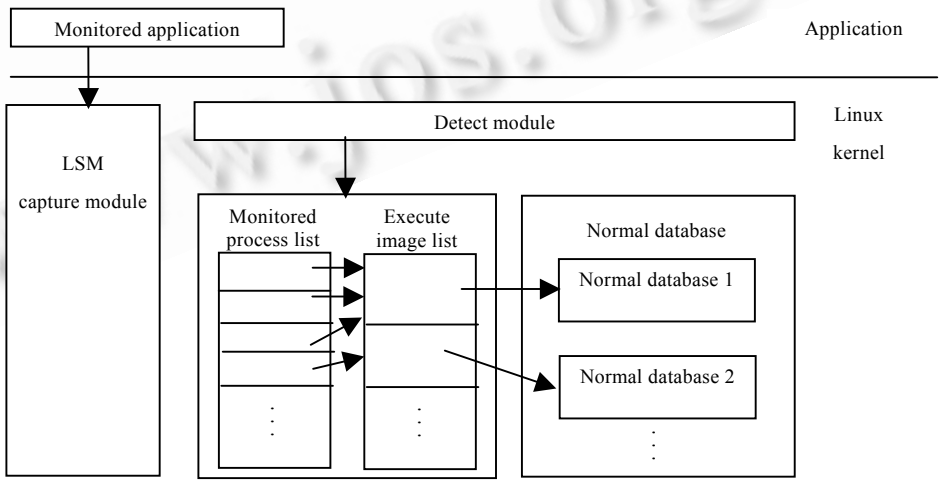

Fig.3 Framework of PB

图 3 PB 框架结构图

LSM截获模块在所有的LSM监控点上设置截获函数,调用检测模块,检测模块根据被监控进程列表决定当前进程是否属于被监控进程,若不属于则返回,否则根据执行映像列表找到对应的正常模式库,并与正常模式库匹配,如果匹配则返回,否则报警.

5.2 正常模式库的建立

正常模式库是通过学习训练所得数据建立的,学习过程中,用长度为 k 的滑动窗口依次滑过训练得到的LSM 截获点数据序列,最终形成若干条 k 长度短序列集合.在 PB 系统中,我们根据前面得到的 wu-ftp2.6 数据,针对不同的 k 值,学习得到正常模式库,其正常模式库大小与学习数据数量之间的关系如图 4 所示.

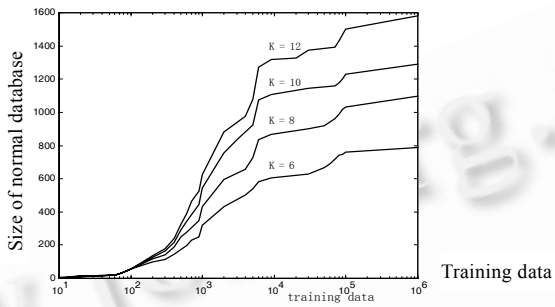

Fig.4 Relationship between size of normal database and training data

图 4 正常库大小与学习数据关系图

5.3 实验与分析

为了检测程序行为控制系统 PB 的执行性能及其对系统效率的影响,实验首先在运行 Redhat 8.0 的 PIII PC 机上对 Linux 的常用命令 ls,insmod,ps 等在运行 PB 前后进行了时间耗费上的测试,得到了实验结果(见表 1).从表 1 的实验数据可知,程序行为控制系统 PB 的运行对于系统的执行效率影响比较小,一般不超过 5%.

Table 1 Contrast of time cost while execute commands

表 1 常用命令执行的时间耗费比较

	Ls (μs)	Insmod (μs)	ps –aux (μs)
Before running PB	93	125	209
After running PB	94	129	232

同时,为了测试 PB 对攻击行为的检测,我们做了针对 wu-ftp 的攻击性测试,测试环境为内部局域网,测试内容包括成功攻击(SITE EXEC 漏洞)与正常运行 SITE EXEC 命令.图 5 是实验结果,图中横坐标是实验过程产生的 LSM 调用,纵坐标是当前 LSM 调用之前 32 个调用中总不匹配数.

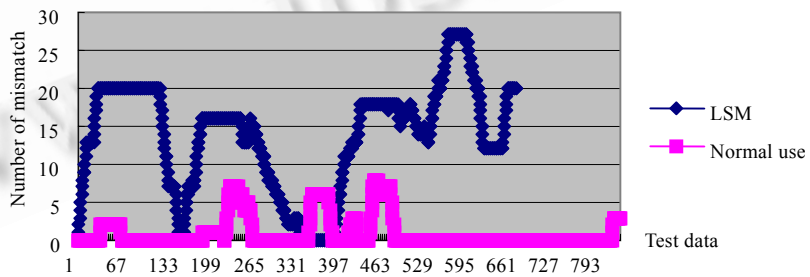

Fig.5 Result of attack test

图 5 攻击测试结果

从图 5 中可以看出,攻击行为产生的不匹配明显高于正常行为,通过这一点可以区分出正常与异常.同时我们也可以看到,由于正常库的不完备,还存在部分误报现象.我们可以通过大量的在线学习来减少误报.同时,需要更多的攻击测试来确定异常阈值,以保证减少漏报.

6 结束语

本文的工作借鉴了系统调用短序列模型的思想,采用了另外一种与系统调用完全不同的数据源——LSM 截获点,并从理论和实践两个方面来验证 LSM 数据源的有效性:基于信息理论的数据质量分析和实际系统的运行结果分析.结果表明,由于 LSM 数据源的更细粒度和更安全相关,基于 LSM 的数据源更适合作为事件序列模型的审计事件.

在实际运行中,有一定的误报率.可以通过大量的在线学习来减少误报;可以设计出算法来滤除误报;需要更多的攻击测试来确定异常阈值,以保证减少误报的同时不增加漏报.

References:

[1] Lian YF. Research of distributed intrusion detection [Ph.D. Thesis]. Hefei: University of Science and Technology of China, 2002 (in Chinese with English abstract).

[2] Forrest S, Hofmeyr SA, Somayaji A, Longstaff TA. A sense of self for UNIX processes. In: Proc. of the 1996 IEEE Symp. on Security and Privacy. Oakland: IEEE Computer Society Press, 1996. 120−128.

[3] Hofmeyr SA, Forrest S, Somayaji A. Intrusion detection using sequences of system calls. Journal of Computer Security, 1998,6(3): 151−180.

[4] Liu HF, Qing SH, Meng Y, Liu WQ. A new audit-based intrusion detection model and its implement mechanism. Acta Electronica Sinica, 2002,30(8):1167−1171 (in Chinese with English abstract).

[5] Jones A, Li S. Temporal signatures for intrusion detection. In: IEEE Computer Society, ed. Proc. of the 17th Annual Computer Security Applications Conf. New Orleans: IEEE Computer Society Press, 2001. 252−264.

[6] Wagner DA. Static analysis and computer security: New techniques for software assurance [Ph.D. Thesis]. Berkley: University of California, 2000.

[7] Ilgun K, Kemmerer RA, Porras PA. State transition analysis: A rule-based intrusion detection approach. IEEE Trans. on Software Engineering, 1995,21(3):181−199.

[8] Ko C, Fink G, Levitt K. Automated detection of vulnerabilities in privileged programs by execution monitoring. In: Proc. of the 10th Annual Computer Security Applications Conf. Orlando: IEEE Computer Society Press, 1994. 134-144.

[9] Ko C, Ruschitzka M, Levitt K. Execution monitoring of security-critical programs in distributed systems: A specification-based appoach, In: Los Alamitos, ed. Proc. of the 1997 Symp. on Security and Privacy. Oakland: IEEE Computer Society Press, 1997. 175-187.

[10] Sekar R, Bowen T, Segal M. On preventing intrusions by process behavior monitoring. In: Proc. of the USENIX Intrusion Detection Workshop. Santa Clara: USENIX, 1999. 29-40.

[11] Ye N. A markov chains model of temporal behavior for anomaly detection. In: Proc. of the 2000 IEEE Workshop on Information Assurance and Security. United States Military Academy, West Point: IEEE Press, 2000. 171−174.

[12] Ye N, Xu MM, Emran SM. Probabilistic networks with undirected links for anomaly detection. In: Proc. of the 2000 IEEE Workshop on Information Assurance and Security United States Military Academy. West Point: IEEE Press, 2000. 175−179.

[13] Guo JL, Zhang WM, Cao Y, Xu L. Constructing an expert system rule of intrusion dtection using machine learning. Computer Engineering, 2002,28(7):69−71 (in Chinese with English abstract).

[14] Li X, Ye N. Decision tree classifiers for computer intrusion detection. Journal of Parallel and Distributed Computing Practices, 2001,4(2):179−190.

[15] Lee W, Stolfo SJ. Data mining approaches for intrusion detection. In: Proc. of the 7th USENIX Security Symp. San Antonio: USENIX, 1998. 6−9.

[16] Lee W. A data mining framework for constructing features and models for intrusion detection systems [Ph.D. Thesis]. New York: Columbia University, 1999.

[17] Lane T. Hidden Markov models for human/computer interface modeling. In: Proc. of the Int'l AI Society, ed. Proc. of the IJCAI-99 Workshop on Learning about Users. Stockholm: International AI Society, 1999. 35−44.

[18] Forrest S, Hofmeyr SA. Immunology as information processing. In: Segel LA, Cohen I, eds. Design Principles for the Immune System and Other Distributed Autonomous Systems. New York: Oxford University Press, 2000. 361−387.

[19] Warrender C, Forrest S, Pearlmutter B. Detection intrusion using system calls: Alternative data models. In: Gong L, Reiter MK, eds. Proc. of the 1999 IEEE Symp. on Security and Privacy. Oakland: IEEE Computer Society Press, 1999. 133−145.

[20] Sekar R, Uppuluri P. Synthesizing fast intrusion prevention/detection systems from high-level specifications. In: Proc. of the 8th USENIX Security Symposium. Washington: USENIX, 1999. 63−78.

[21] Somayaji A. Operating system stability and security through process homeostasis [Ph.D. Thesis]. Albuquerque: University of New Mexico, 2002.

[22] Wright C, Cowan C, Morris J, *et al*. Linux security modules: General security support for the Linux kernel. In: Proc. of the 11th USENIX Security Symp. San Francisco, 2002. 17−31. http://www.usenix.org/events/sec02/full_papers/wright/wright_html

[23] Lee W, Dong X. Information-Theoretic measures for anomaly detection. In: Needham R, Abadi M, eds. Proc. of the 2001 IEEE Symp. on Security and Privacy. Oakland: IEEE Computer Society Press, 2001. 130−143.

附中文参考文献:

[1] 连一峰.分布式入侵检测系统研究[博士学位论文].合肥:中国科学技术大学,2002.

[4] 刘海峰,卿斯汗,蒙杨,刘文清.一种基于审计的入侵检测模型及其实现机制.电子学报,2002,30(8):1167−1171.

[13] 郭建龙,张维明,曹阳,徐磊.应用机器学习制定的入侵检测专家系统规则集.计算机工程,2002,28(7):69−71.