基于程序的异常检测研究综述

黄金钟 朱淼良
(浙江大学计算机科学与技术学院 杭州 310027)

摘 要 以程序正常行为描述方法为线索,将利用系统调用数据检测程序异常行为的各种技术分类为基于规范的方法、基于频率的方法、控制流分析方法、数据流分析方法。详细介绍了这些方法的基本思想、使用的各种模型以及最新研究进展,指出并分析了现有技术中存在的问题和不足,正式提出了基于程序的异常检测技术应该以各种服务器程序为研究对象的观点,介绍了一个经过初步实验验证了的、基于服务器程序运行踪迹层次结构的异常检测原型系统,该原型系统利用了服务器程序请求-应答式工作特征和一些关键系统调用的语义信息以及运行时的动态信息,通过结构模式识别技术在识别服务器程序正常行为过程中发现异常并具备分析异常、提供入侵相关详细信息的能力,而这种能力正是异常检测技术进一步研究发展的方向之一。

关键词 入侵检测,异常检测,异常分析,系统调用,服务器程序,结构模式识别
中图法分类号 TP393 **文献标识码** A

Overview of Anomaly Detection Based on Program

HUANG Jin-zhong ZHU Miao-liang
(College of Computer Science and Technology, Zhejiang University, Hangzhou 310027, China)

Abstract In terms of methods describing normal program behavior, anomaly detection based on program can be grouped into several broad categories: specification-based, frequency-based, control-flow-based, and data-flow-based. After reviewing systematically the basic ideas and various models used in these approaches, discussing the new advances of the technique, pointing out and analyzing some problems and weaknesses which exist in current research, this paper formulated a notion that anomaly detection based on program should focus attention on various server programs. A system prototype based on the hierarchical structure of server programs' traces and validated by a preliminary experiment was simply introduced. The prototype is capable of analyzing anomalous events and providing detailed information with respect to intrusion, and these abilities are just the trend for more research of anomaly detection.

Keywords Intrusion detection, Anomaly detection, Anomaly analysis, System call, Server programs, Structural pattern recognition

1 引言

随着网络应用的不断深入,入侵与防卫的矛盾越演越烈。作为网络安全领域的一个研究方向,入侵检测技术已经将网络技术、模式识别、人工智能、软件工程等多种技术和理论集成在一起,而异常检测因为适应性强、具备检测未知类型入侵的能力,更是受到了普遍关注。

传统上,异常检测以用户行为或网络流量、通信数据包为监测对象,但这些对象行为随意性很强,很难准确描述其正常行为轮廓,这就导致了基于这些对象的异常检测技术的虚警率居高不下。实际上,网络入侵和网络病毒侵袭,无论采用什么样的技术手段,最终都要通过目标计算机系统中的一些关键程序来实施。在UNIX/Linux中,这些程序称为特权程序,即对系统资源拥有完全控制权限的setuid类程序。基于程序的异常检测就是通过对特权程序进行监视,看其行为是否背离预期行为或正常模式,从而发现潜在的网络攻击与入侵事件。

基于程序的异常检测研究起源于20世纪90年代中期,是从传统的"沙箱(sandboxing)"技术演变而来的一种新的研究方向。所谓沙箱技术,是为了保护网络计算机系统的敏感资源不受恶意代码的非法访问而出现的一种安全机制,其核心在于建立了一个受限的应用程序执行环境"沙箱",即通过对不安全代码中的系统调用制定一套严密的安全策略来为程序运行提供一个静态的相对安全的环境。在"沙箱"技术的基础上,Calvin. Ko提出了以监视特权程序动态运行过程中产生的系统调用来检测程序异常行为这一基本思想[1]。

基于程序的异常检测具有如下特点:

(1) 直接对位于应用层的程序进行监控。应用层安全是困扰网络的薄弱环节,利用应用层协议和特权程序中的安全漏洞进行攻击是网络入侵的主要手段。

到稿日期:2010-07-07 返修日期:2010-10-23 本文受国家自然科学基金(60773182)资助。
黄金钟 男,博士,主要研究方向为人工智能、模式识别、网络安全,E-mail:huangjz@zju.edu.cn;**朱淼良** 男,教授,博士生导师,主要研究方向为人工智能、智能机器人、计算机视觉、计算机网络。

(2) 程序行为空间相对狭小和稳定，便于精确描述其正常行为轮廓。程序功能是预先设计好了的，其活动范围一般都有较为严格的限制，不像用户行为和网络通信流量那样有相当大的不确定性。

(3) 数据源单一。程序运行时产生的系统调用序列，也称为程序的运行踪迹，很好地刻画了程序的实际行为，这使得原本很复杂的建模工作变得简单和清晰。

(4) 数据结构简单。系统调用数据只包含 3 方面的属性：名称或者编号、参数、返回结果，而且现有方法中的大部分只用到了系统调用编号这一种属性。处理这种简单数据结构的计算量小、复杂度低，这对大规模、高带宽及分布式网络计算环境中的入侵检测技术相当重要。

如何描述程序的正常行为模式，是基于程序的异常检测技术的关键，本文将围绕这个问题展开讨论。

2 基于规范的方法

程序在完成自己的功能时总要访问一些系统资源，但存取哪些资源并以何种方式存取，这方面应该有个约束。例如 Web 服务器程序可以应某个用户要求读取网页文件并传回给该用户，但任何篡改或删除网页文件的操作都是不允许的。Ko 把这种约束称为程序行为规范(specification)[1]，它与系统调用及其参数密切相关。

为了定义程序行为规范，Ko 使用了一种基于正则式与谓词结构的语言 PPSL，其中的正规式用来表示对象的名字，例如带路径的文件全名，谓词代表可允许的操作，例如 read，谓词的参数包括对象名字和程序运行的状态变量组(例如程序一旦创建了一个临时文件就用一个变量记下当时的状态)。于是，程序在某个状态下可以存取哪些对象，就可以用类似于 Prolog 语言的谓词结构来定义。对于并行分布式环境下的程序行为规范，Ko 拓广了 PPSL 语言，使用了一种带状态变量的上下文无关文法 PE-Grammars 来进行定义[2]。PE-Grammars 综合考虑了系统资源对象存取、相关操作的先后顺序、进程间的同步和竞争等方面的安全属性。

但是，Ko 的方法是通过手工分析审计日志数据来定义程序行为规范的，检测引擎也是对审计数据进行分析，缺乏实时检测能力，类似的原型系统还有 DTE[9]，LIDS[8]等。

为了解决实时检测问题，Sekar 对检测引擎进行了改进，使之能从程序实际运行中收集系统调用数据，并在程序行为规范定义中附加了入侵响应动作[4,5]，从而加强了实时保护能力。但是，Sekar 的规范定义方法还是依靠安全专家分析审计数据、用一种称为 ASL 的语言手工编写的。

一般来说，手工编写的程序行为规范精致易读，而且方便添加修改，但这只适合中小型程序，对于 Apache 这类大型程序，手工编写它们的行为规范是难以想象的。为此，Ko[3]利用归纳逻辑编程 ILP(Inductive Logic Programming)方法从程序的实际运行例子中自动学习行为规范。BlueBox 原型系统[7]则采取了一种折衷的方法：对于 Apache 等大型服务器程序，用一种模板机制从其配置文件及审计数据中自动生成规则库，而对于 CGI 小程序则还是采用手工编写方式。

程序行为规范最终要通过编译转换成一组规则库。在实时检测过程中，规则库与检测引擎一起被装载进内核，对程序产生的系统调用进行检查。为了减轻系统开销，文献[7]在定义程序行为规范时对系统资源及其存取方法按安全相关程度进行了分类，结果发现有相当一部分系统调用不会涉及安全问题。对这些无害系统调用，没必要为它们定义规范。Xu[10]进一步按功能和危险程度对系统调用进行了分类，例如在进程管理一类系统调用当中，execve，setuid 等具有最高危险等级等等。Xu 把这些危险程度最高的系统调用称为“关键系统调用”，在机器学习过程中只对这些关键系统调用构建行为规范，从而有效减小了规则库，缩减了检测过程中的规则匹配时间。

基于规范的方法只为单个系统调用制订规范，忽略了程序运行踪迹中的时序信息，无法检测符合这些彼此孤立的规范约束的攻击行为。顺便一提，近年来出现了一种将“沙箱”技术与基于规范的方法相结合来加强安全措施的方法，称之为“动态沙箱”技术[12,13]。

3 基于频率的方法

系统调用的频率特性也可以作为评判程序行为是否异常的依据。进程正常运行时各种系统调用出现的频率是稳定的，出现异常时，频率会发生变化。

Warrender 用系统调用短序列的频率分布来描述程序正常行为，并称其为 t-stide 方法[14]。一个入侵行为的系统调用序列出现时，若偏离了正常行为的频率分布，可通过设定阈值来检测。Surekha 则用不同序列集中的不同系统调用出现的频率来描述程序正常行为[15]，通过为这些序列集建立频率分布模型，用 Bayesian 网络实现异常行为的检测。Luo[80]在对样本进行切分的情况下，构建了一个对系统调用序列发生频率敏感的基于支持向量的描述模型，利用发生频率定义样本的“重要性”，使分类器更加倾向于这些重要的样本。

Liao 采用文本分类技术，将程序运行踪迹看作是一篇文档并用一个向量表示，向量长度是操作系统提供的系统调用个数，向量元素值为各种系统调用在运行踪迹中出现的频率，检测时使用 K-NN 分类器对目标进程进行判断[16,28]。Kang 借鉴文本分类中的词袋(Bag of words)模型，用“系统调用袋”[17]表示程序的运行踪迹。在系统调用袋中，系统调用的次序信息可以忽略不计，仅考虑每个系统调用出现的次数，接下来的问题就是一般的训练和分类了，例如使用单类朴素 Bayes 算法、C4.5 决策树、RIPPER 算法、支持向量机等。Paek 提出了 sC4.5 算法，其不但保持了 C4.5 算法的分类精度，而且使决策树相对减小[18]。

另外，也有研究者将系统调用频率与其它方法结合起来，例如 Wang 在 Sekar 的 FA 模型工作基础上引入系统调用频率统计模型[19]。在模型建立的过程中，除了建立自动机之外，还记录下自动机上每个转换所对应的系统调用执行的频率，在检测过程中，若监视到某些系统调用发生频率异常升高，即发生频率超过正常执行时频率的平均值的一定倍数时，将这些系统调用标记为异常。

基于频率的方法存在一个很大的缺点：由于只考虑了系统调用出现的频率，因此入侵者很容易躲过检测，例如可以通过少量操作来获取敏感信息但并没有引起频率的巨大变化，或者通过加入一些无关系统调用来保持频率的一致。

4 控制流分析方法

控制流分析方法考察的是程序运行踪迹中系统调用之间

的时序先后关系，以各种数据模型模拟和逼近程序的实际控制流程。这种方法有两条基本路线：静态分析、机器学习。

4.1 静态分析

静态分析(Static analysis)通过对程序代码的执行路径进行分析来建立程序正常行为模式。因为采用的模型是从程序代码中抽象出来的，即它的模型只接受符合程序控制流程的行为，所以这种方法的虚警率为零。

目前，在利用静态分析技术构建程序正常行为模式的各种方法中，广泛引用的是 D. Wagner 提出的调用图模型与抽象栈模型[31]。

4.1.1 调用图(Call Graph)模型

调用图实际上是一个不确定的有穷自动机(NFA)，它以程序源码中的函数调用位置来定义图中的状态节点：函数入口点 Entry(f)及出口点 Exit(f)。另外，每遇到一个函数调用，引入两个新节点 v 及 v'：从 v 进入 Entry(f)，从 Exit(f)出来到 v'。程序源码与其调用图的例子如图 1 所示。

```
f(int x) {
  x ? getuid():geteuid();
  x++;
}
g() {
  fd=open("foo",O_RDONLY);
  f(0);close(fd);f(1);
  exit(0);
}
```

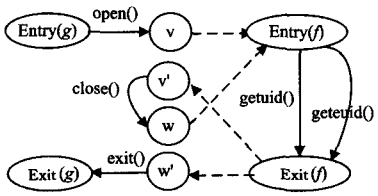

图 1 程序源码及其调用图

调用图模型简单易行，但其中包含有实际情况中不可能出现的执行路径，例如图 1 中的执行路径“v→Entry(f) →…→Exit(f) →w'”是实际上不可能出现的——对照源码可以看出：该路径是从 $f(0)$调用的入口进，而却是从 $f(1)$调用出口出来。显然，这将使得沿着这条不可能路径进行的入侵行为逃脱检测。

Lam 提出一种称为 Paid 的机制[32]，其能从程序源码中自动导出 DFA 模型。该方法中的 Paid 由两部分组成：一个编译器，它捕获程序源码中的系统调用点、系统调用次序以及部分控制流程；一个嵌入系统内核的运行时核验器，由它执行实时检测。Paid 将每个系统调用点与 int 0x80 指令地址相关联，将程序调用图中具有多个调用点的函数调用与该函数本身的调用图相关联，这样就解决了自动机的部分确定化问题。对于函数指针、信号处理等无法静态确定的问题，Paid 引入“通报(notice)”机制，由编译器将这一情况告诉运行时的核验器以实现动态确定。

文献[27] 在调用图基础上提出 IMA(Inline Model of Automaton)模型，其基本思想是先构造单函数的局部自动机，然后由这些局部自动机通过直接嵌入(Inline)来构造全局自动机，即在每个调用点把局部 NFA 直接嵌入到全局 NFA 中。此方法有个缺点，即递归调用函数的嵌入会导致死循环以至系统资源耗尽。

4.1.2 抽象栈(Abstract Stack)模型

对调用图模型进行扩展，加入一个下推栈用于保存与函数调用相对应的返回位置，便得到一个非确定的下推自动机以及相应的上下文无关文法，Wagner 称之为“抽象栈模型”。

抽象栈模型中，由于与函数调用相对应的返回位置信息被保存在堆栈里，函数返回时就能根据栈顶确定下一个位置，因此不可能路径问题基本得以消除。但是，由于静态分析阶段无法判断条件分支应该走哪条路径，因此栈状态存在不确定性。这种 NPDA 的运行代价相当高，很大程度上影响了检测系统的实时性。

为了解决 NPDA 的确定化问题，Feng 对抽象栈模型进行了改进[33]，通过分析程序的二进制代码，不仅可以得到程序的控制流图，还可以找到函数调用与系统调用出现处的地址。于是压栈时用地址信息替代抽象栈模型里的位置信息，而且对每个系统调用也在堆栈里保存返回地址。这种方法实际上是模拟了程序运行栈机制，只是它比实际的运行栈简单许多(略去了参数单元、局部变量等部分)。由于增加了表示栈状态的符号(系统调用处的地址及返回地址)，因此栈的状态被细化了，每次转换都能确定唯一的下一个状态。

J. Jiffen 采用二进制代码重写方法[34]来解决 NPDA 的确定化问题，其分以下 4 个步骤：

1. 分析二进制代码，为每个函数调用构建控制流图；

2. 将每个控制流图转换成类似于调用图的局部模型，该局部模型是一个非确定的有穷自动机，它接收原函数产生的函数调用与系统调用序列；

3. 重写二进制代码，在每个函数调用的前后位置都加入空系统调用(null-call)或称“哑”系统调用，哑调用不做任何事情，仅仅起指示作用，目的在于运行栈的一致性检查；

4. 将所有局部模型组合起来构成一个全局模型，组合方法与抽象栈模型类似。该全局模型就是重写后的二进制程序的 PDA 模型，而且这个下推自动机是确定的。

与文献[33]方法相比，文献[34]方法没有引进多余的栈状态符号，但重写二进制代码引进了哑调用，特别地，在函数递归调用及循环结构中，这种附加的空调用引起的开销是比较大的。为此，Jiffen 采用了“空调用压制(Null Call Squelching)技术”[34]，即对递归函数以及不产生系统调用的函数，不添加空调用，这其实是以牺牲精度换取性能的一种折衷做法。

静态分析方法虽然没有虚警问题，但也存在其它一些局限性：第一，从程序代码导出的模型比较保守，代码中一些可能的路径在程序正常行为中从来不会执行或者非常罕见，而入侵者却可以利用它们，最典型的例子是由于配置错误导致程序可以以调试模式运行。第二，处理非标准控制流程比较困难。非标准控制流包括 C 语言里的函数指针、goto 语句等 setjmp 类转移指令，还有非同步的信号处理机制等。另外，库函数以及动态连接代码问题对静态分析方法也是一个挑战，语言与平台差异也是一个问题。第三，开放源代码的程序毕竟有限，现实中的大部分程序代码不一定能得到，特别是一些行业软件出于各种考虑要求保密。另外，代码重写也会带来新的问题[37,39]。

4.2 机器学习

采用机器学习技术，从程序的实际运行中学习归纳出程序正常行为模式，是基于程序的异常检测研究中极其活跃的一个分支。这种方法无需程序代码，不受平台限制，不需要深入的专家经验，具有相当强的适应性和广泛的应用前景。

目前，基于机器学习的方法使用的基本模型有：短序列模型、有穷自动机模型、虚拟路径模型和运行图模型。

4.2.1 短序列(N-gram) 模型

短序列模型是 S. Ferrest 在其“人工免疫系统”研究项目

中定义“自我(Self)”[21]时使用的一种方法。Ferrest 认为程序正常行为由大量的局部模式组成,这些局部模式可以用固定长度的系统调用短序列,即 N-gram 来描述,通过列举程序运行踪迹中出现的所有这种 N-元组,可以构建一个描述程序正常行为整体模式的数据库,而异常检测就是一个简单的模式匹配过程。

文献[21]使用一种“滑动窗口”机制从系统审计文件中学习程序的正常行为,并称这种方法为 stide(Sequence time-delay embedding)”[14]。Kymie 对大量入侵数据进行分析,发现至少需要 6 个以上的系统调用才能区分入侵短序列与局部正常模式[22]。Debar[46]认为长模式更能体现进程的某些特性,模式越长,由入侵产生的系统调用序列与局部正常模式匹配的概率就越低,从而越容易被发现,由此提出用变长短序列来描述程序局部正常模式。采用变长短序列[40]覆盖训练序列所需要的模式要少很多,并能减小检测的计算量,提高模式匹配的效率。Wespi [20,44]在此基础上采用 Teiresias 算法来发现变长短序列模式,模式抽取按提取训练集中所有最大模式与模式约简两个步骤进行。文献[35]则利用静态分析技术从程序的二进制代码中自动发现局部短序列模式。

在短序列模型中,为了保存系统调用之间的时序信息,需要对出现在局部模式中的一些系统调用进行重复存储。为了减小存储开销和提高检测效率,Lee[23,24]提出利用数据挖掘技术从审计数据中提取短序列的预测规则,其基本思想是:对相近的短序列类进行关联分析,用少量的几个具有代表性的系统调用来表示这一类短序列,并采用 Ripper 算法建立这种规则库,规则形式诸如:如果短序列中第 m 个系统调用是 S_m,那么第 n 个系统调用应该是 S_n。Cai 等[42,41]将粗糙集理论应用到局部模式的挖掘方法中,进一步精简了预测规则。模糊理论和方法也在这里得到了应用[60],另外,Zhang 将系统调用按它们的作用效果进行划分[50],提出只监视那些具有“写”性质的子集,局部模式也只使用子集中的系统调用,大大降低了存储开销和计算处理时间。Lookahead pairs 方法[45,47]则使用一种与正规式类似的表达式结构描述局部模式,紧凑而高效。

短序列模型特别适合用人工智能中的一些技术来处理[26],例如将 N-元组作为人工神经网络[30,49,53]、支持向量机[43]的输入,利用 ANN 和 SVM 强大的学习与自适应能力来提高检测性能。另外,隐马尔可夫模型在基于短序列模型的异常检测中也得到了广泛应用[36,48,52,54,55]。文献[25]还将遗传算法应用到局部模式库的构建过程中。文献[51]则利用基因规划技术自动繁殖局部正常模式,使得训练周期大大缩短,检测的准确率也得到很大提高。

短序列模型没有从全局考虑程序的执行流程,只能检测局部不匹配的情形,对于跨度大的入侵序列的漏检率较高。另外,模式匹配中的阈值很难掌握,阈值取得太大会漏检,取得过小会导致虚警率升高。

4.2.2 有穷自动机(FA)模型

程序运行踪迹是系统调用的时间序列,可以看成是一个由系统调用构成的“串”,而有穷自动机是描述这种离散序列的有效工具。

最早的程序正常行为 FA 模型是 Kosoresow 通过分析审计数据,用手工构造的[56]。Michael 也提出过一种从程序运行踪迹构造 FA 的方法[57],但其性能不尽人意。最有效的是 Sekar 提出的利用程序运行时的动态信息构建精确 FA 模型的方法[58]。

在 Sekar 的 FA 模型中,有穷自动机的状态由程序执行系统调用时的程序计数器 PC 值来表示,状态之间的转换由相应的系统调用来标记。学习过程中,每遇到一个系统调用,记下当前的 PC 值与相应的系统调用,用偶对(PC,Syscall)表示,同时给 FA 增加一个状态“PC”和一条转换弧,该转换弧从前一个状态“PC_{prev}”出发指向当前状态,并标记为 $Syscall_{prev}$。在学习程序正常行为的 FA 模型时,有 3 个问题需要解决:

1. 动态连接问题。动态连接库中的同一个函数可能会装载到程序的不同位置,此时,同一系统调用的 PC 值就会不同。

2. 库函数产生的系统调用问题。函数调用时改变了主程序的控制流程,即,程序运行进入到库函数时,程序计数器 PC 的值已经不再与主程序相关了。

3. fork/exec 问题。进程派生子进程时应该如何处理 FA。

对于第一个问题,在 Linux 中,动态连接代码是由系统调用 mmap 加载的,因此利用 mmap 系统调用的参数和返回值就可以计算动态连接代码的地址范围;对于第二个问题,可以从程序的运行栈中取出函数的返回地址,从而保证主程序 PC 值的连贯性;对于第三个问题,如果子进程只是父进程的克隆,则父子进程公用一个 FA;如果子进程通过 exec 执行另外一个程序,则单独为子进程再构建一个 FA。

Sekar 的 FA 模型利用了程序运行时的一些动态信息(PC 值和运行栈中的返回地址信息),能准确地反映程序的实际控制流程,检测性能也得到大幅提高。但是,该模型不能解决“不可能路径”问题,因而会导致一些漏检。

4.2.3 虚拟路径(Vtpath)模型

为了解决 FA 模型中存在的“不可能路径”问题,H. Feng 提出了虚拟路径模型[59]。虚拟路径,是指两个系统调用位置之间的转换序列。模型建立过程如下:

每当程序运行产生一个系统调用时,记下该系统调用的名称和当前 PC 值,同时,从程序运行栈中抽取所有返回地址并构成一张虚拟栈列表 A,即:

$A=\{a_0,a_1,a_2,\cdots,a_{n-1}\}$,$n$ 是运行栈的栈帧编号,a_{n-1} 是最后一个被调用函数的返回地址。

然后,将当前 PC 值作为 a_n 加到 A 中,于是,虚拟栈列表 A 记录了所有还未返回的函数调用。

假设 $A=\{a_0,a_1,a_2,\cdots,a_n\}$ 与 $B=\{b_0,b_1,b_2,\cdots,b_n\}$ 分别对应当前系统调用和上一个系统调用,并考虑到这两个系统调用可能会出现在不同的函数中。从 a_0,b_0 开始逐个比较 A 和 B 中的各项,直到 $a_l\neq b_l$,如图 2 所示。

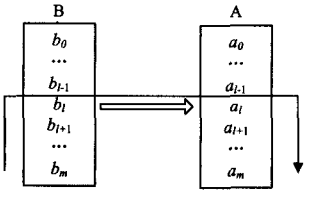

图 2 虚拟路径

此时,这两个系统调用之间的虚拟路径即被定义为:

$$P=b_m\rightarrow Exit;\cdots;b_{l+1}\rightarrow Exit;b_l\rightarrow a_l;$$
$$Entry\rightarrow a_{l+1};\cdots;Entry\rightarrow a_n$$

其中，*Entry* 与 *Exit* 是对应于函数调用入口与出口处的 PC 值。

虚拟路径模型抽象表达了两个系统调用之间的执行路径：从前一个系统调用开始，程序依次从一些函数返回（与返回地址 $b_{m-1}\cdots b_l$ 对应），然后顺序进入另外一些函数（与返回地址 $a_l\cdots a_{n-1}$ 对应），直到当前系统调用。

虚拟路径模型以返回地址记录对函数的一次调用，因此不会出现“不可能路径”问题，因为两次调用时的返回地址不可能相同。

4.2.4 运行图（Execution graph）模型

Gao 在 VtPath 模型的基础上进行了扩展，提出了程序正常行为的运行图模型[38]，它实际上是程序关于函数调用以及系统调用的控制流图。

运行图模型建立在“观察（Observation）”与“运行（Execution）”两个基本概念之上：“观察”被定义成一个由正整数组成的向量$(r_1,r_2,\cdots,r_k)$，其中 r_k 是当前的系统调用号，r_{k-1}，r_{k-2}，$\cdots$，r_1 是当前程序运行栈中的返回地址序列，包括函数调用与系统调用的返回地址。因此，r_{k-1} 是与当前系统调用执行完毕后将要返回的地址。特别地，r_1 与 C 语言程序中的第一个函数即 main() 函数中的某处地址相对应。程序每产生一个系统调用，就有一个相应的“观察”，而连续的“观察”则揭示了程序中的函数调用结构。“运行”是指由任意长度的“观察”构成的序列。

运行图是一种有向图(V,E)，其结点集合 V 由“观察”序列中的 r_i 组成。运行图中的边有以下 3 种：

E_{call}：调用边，表示从函数中的某个位置调用其它函数或进入系统调用。显然，被调用目标执行完毕后应该返回到相应位置的下一条语句。

E_{rtn}：返回边，表示程序从被调用目标中返回。

E_{crs}：运行边，表示程序在一个函数中的运行控制流，即在同一个函数内部的语句之间跨越。

利用“观察”和“运行”，通过一个学习过程可以构建程序的运行图。Gao 还证明了运行图与通过静态分析从程序代码中抽象出来的控制流图是一致的。

机器学习方法也存在一些问题：1）训练不够完备则会导致虚警，而有些学习算法要求的训练周期比较长；2）在线训练期间容易受到入侵者的干扰破坏；3）程序正常行为模式会变化，例如系统环境重新配置的时候，这就需要重新学习。

机器学习与静态分析是可以互补的，例如，先用静态分析构建一个基本轮廓，然后利用机器学习来进行提炼和补充[61,62,29]。

控制流分析方法对于入侵者常用的缓冲区溢出、破坏堆内存等企图运行外来代码的攻击手段非常有效，但对于 Mimicry[63-65] 等非控制流攻击手段[66]则检测能力有限。尽管有些研究者尝试在控制流分析中加入简单数据流分析方法[67-69]，但迄今为止进展不大。

5 数据流分析方法

数据流分析方法，就是对存在于程序运行踪迹中的系统调用参数及返回值进行分析，总结出程序正常运行时的数据规律，以此为根据来检测对系统资源的异常操作。

利用静态分析技术可以抽出程序代码中的常数数据及环境依赖关系并进行一些简单的数据流分析[11,33,71]，但这种方法存在两个严重不足：1）只能手工分析；2）只能分析程序代码中的静态数据，而程序运行时处理的大部分数据是动态的。因此，数据流分析比较适合采用基于机器学习的方法。

一种简单的数据流分析方法是学习程序运行踪迹中系统调用参数的统计特征。Kruegel[72] 针对系统调用的字符串类型参数，提出了 4 个统计模型：1）串长度模型，对程序正常运行时系统调用的串类型参数的长度分布进行统计学习，根据串长度的平均值与平均方差，利用切比谢夫不等式来计算串类型参数是否异常的概率。2）串字符分布模型，将参数中可能出现的 256 种字符的相对频率按递降排序，并把通过训练得到的这种序列集合称为“理想字符分布”，检测时，利用皮尔森 χ^2-测试计算实际参数样本与理想字符分布的接近概率。3）串结构模型，用形式语言理论中的正规语言文法来描述字符串参数的结构组成，对于程序正常运行时可能出现的所有串类型参数，为它们建立一个正规语言文法描述其结构，同时对每条文法产生式计算其 Bayes 概率。训练完成后，正规文法用 Markov 模型表示，如果参数与正规文法不匹配，则 Markov 模型输出异常信号；4）记号（token）发现模型，记号指诸如 FLAG，MODE 之类的标志型参数，学习算法将运行踪迹中出现的所有这种记号放入一个有穷集合中。检测时如果出现的记号能从集合中找到则输出 1，否则输出 0。

Maggi 在上述 4 个模型基础上引入“聚类”概念[73]，用聚类参数自动推断同一系统调用的不同使用方式以便建立精确的参数模型。聚类还用于捕获并构建同一系统调用的不同参数之间的相关性。实现时用一个库 LibAnomaly 管理这些参数模型，用一个算法学习产生程序正常行为的整体 Markov 模型。Mutz[75,76] 则使用 Bayesian 网络对不同系统调用的参数模型进行组合输出。另外，Sufatrio[67] 提出给参数制订规范的方式来实现异常检测，Tandon[68,73,77] 则用 if-then 规则来描述系统调用序列中的控制流与数据流组合特征，但运行耗费相当高。

与学习单个系统调用参数的统计特性不同，Bhatkar[39] 提出学习不同系统调用的参数在时间上的特性，从而捕获贯穿整个程序的、与安全相关的数据流。数据流特性涉及程序运行踪迹中的系统调用参数数据及其在不同系统调用间的流动，它们被表示为参数数据上的一元和二元关系：一元关系描述单个系统调用参数特性，二元关系则描述两个不同系统调用参数之间的约束与依赖。学习这些关系需要借助控制流上下文信息，为此，Bhatkar 提出一种声称可以运行于现有基于机器学习的控制流模型之上的学习算法，该算法用一种统一的方式从程序运行踪迹中提取和确定系统调用参数特性以及它们之间的关系，从而提高了这些模型和技术的检测精度，但文献中仅将该算法应用于简单的 DFA 模型上，且运行费用较高。Li[78] 则通过合并静态分析与动态学习，用一种两阶段分析方法来构造程序的数据依赖图并以此识别无关参数和无用关系。

数据流分析技术弥补了控制流分析方法难以检测 Mimicry等非控制流攻击手段的不足，但对于隐藏在正常数据处理流程里的入侵同样也无法判断。另外，大型程序处理的数据

往往是海量的，其中的关系也错综复杂，理清并存储这些关系本身就要付出极大代价。

6 目前研究存在的问题及一种解决方法

经过十余年的探索，基于程序的异常检测技术在提高检测能力、降低虚警率等性能方面都得到了很大进展，但也应该看到，这种技术离实际应用还有一段距离，还有不少问题需要解决：1)还没有一种方法对所有类型的入侵都能做出反应，为了进一步提高检测能力，应该尽可能集众家之长；2)异常检测的虚警问题很难避免，现有的技术只能给出简单的报警，而且绝大部分检测方法属于黑箱操作，检测分析过程缺乏透明度，这便使得安全管理人员对模糊的报警信号产生疑惑而无所适从；3)检测系统的效率，特别是对入侵行为的快速反应机制需要加强。虽然安全报警关联技术研究正在展开[81]，但随着技术的进步，入侵手段也日新月异，高级入侵方式都是以入侵程序自动进行的，从攻破防线到造成破坏后果往往只需几秒时间。因此，及时准确地发现异常并迅速找到入侵来源就显得非常重要，而现有的基于程序的异常检测方法在这方面几乎无所作为。

一般来说，网络安全重点保护的对象是各种网络服务器系统，而服务器程序是网络用户与服务器系统之间的自然边界。只要在服务器程序上严格把关，恶意用户和网络病毒就无法染指系统中的其他特权程序。鉴于此，本文提出基于程序的异常检测技术的研究对象应该集中到各种网络服务器程序的观点。

服务器程序以“请求-应答”方式工作，这种特性使得服务器程序的运行踪迹呈现出层次结构特征[79]。利用一些关键系统调用的语义及其参数与返回值信息，再加上程序运行时的一些动态信息，可以把程序运行踪迹分割成具有不同层次的结构块，而且与服务器程序执行的操作内容相对应，每个结构块都有明确的语义。通过机器学习，可以建立程序正常行为模式的层次结构化描述，于是异常检测就对应于一个结构模式识别过程，只是这里识别的是程序的正常行为，而且识别粒度得到了细化。一旦发现非正常行为，即可沿着识别过程进行回溯，在语义信息的帮助下分析异常现场，找出异常点和异常原因并迅速准确定位入侵来源(入侵者的 IP 地址)，揭示入侵过程。另外，这种方法在建模及检测分析过程中都利用到了系统调用的参数和返回值数据，因此在现有机制上融入数据流分析技术就很方便和自然，例如将原有的服务器程序正常行为描述文法扩展为属性文法即可展开数据流分析，这将大大提高该方法对 mimicry 等非控制流攻击的检测能力。

结束语 基于程序的异常检测是网络入侵检测技术中的研究热点，可行、可信、可靠依然是其研究目标，这包括实时检测、及时发现、结果准确、信息丰富、快速反应等现实需求。未来值得关注的方向应该还有：对检测结果进行入侵机理分析，入侵特征抽取，与其它入侵检测系统共存、共享及互操作等等。另外，随着 Internet 技术的日新月异，高带宽、分布式处理已成为网络互联应用的发展趋势，尤其是 IPv6 网络的快速发展以及 P2P 技术的广泛应用，这些都给网络安全领域带来了新的挑战，也给基于程序的异常检测技术的关键突破提供了新的机遇。

参 考 文 献

[1] Ko C, Fink G, Levitt K. Automated detection of vulnerabilities in privileged programs by execution monitoring[C]// Proceedings of the 10th Conference on Computer Security Applications. Los Alamitos, CA: IEEE Computer Society Press, 1994: 134-144

[2] Ko C, Ruschitzka M, Levitt K. Execution monitoring of security-critical programs in distributed systems: a specification-based approach[C]// Proceedings of the 1997 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, 1997: 175-187

[3] Ko C. Logic induction of valid behavior specifications for intrusion detection[C]// Proceedings of the 2000 IEEE Symposium on Security and Privacy. IEEE Computer Society Press, 2000: 142-153

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

[5] Uppuluri P, Sekar R. Experiences with specification-based intrusion detection[C]// Proc. of the 4th Int'l Symptom on Recent Advances in Intrusion Detection. Davis: Springer-Verlag, 2001: 172-189

[6] Bernaschi M, Gabrielli E, Mancini R. A security-enhanced operating system[J]. ACM Trans. on Information and System Security, 2002, 5(1): 36-61

[7] Chari Suresh N, Cheng Pau-chen. Bluebox: A policy-driven, host-based intrusion detection system[J]. ACM Transactions on Information and System Security, 2003, 6(2): 173-200

[8] Xie H, Biondi P. The LINUX intrusion detection project[OL]. http://www.lids.org, 2002

[9] Walker K M, Daniel F S, Lee Badger M. Confining root programs with domain and type enforcement[C]// Proc. of the 6th USENIX Security Symp, Focusing on Applications of Cryptography. San Jose, USENIX, 1996: 21-36

[10] 徐明，陈纯，应晶. 基于系统调用分类的异常检测[J]. 软件学报，2004, 15(3): 391-403

[11] Lam L-C, Chiueh T-C. Automatic extraction of accurate application-specific sandboxing policy[C]// Military Communications Conference. IEEE, 2005: 17-20

[12] Mohanty H, Swamy M V, Thilak P, et al. Secured networking by sandboxing LINUX 2.6[C]// ICSMC2009. IEEE, 2009: 3669-3674

[13] Inoue H, Forrest S. Inferring Java Security Policies through Dynamic Sandboxing[C]// Proceedings of Programming Languages and Compilers. Las Vegas, 2005

[14] Warrender C, Forrest S, Pearlmutter B. Detecting intrusions using system calls: Alternative data models[C]// Proceedings of the 1999 IEEE Symposium on Security and Privacy. 1999

[15] Varghese S, Mariam J K. Process profiling using frequencies of system calls[C]// The Second International Conference on ARES. 2007: 473-479

[16] Liao Yihua, Vemuri R. Use of K-Nearest Neighbor classifier for intrusion detection[J]. Computers & Security, 2002, 21(5): 439-448

[17] Kang D K, Fuller D, Hona R V. Learning classifiers for misuse detection using a bag of system calls representation[C]// Proc. of IEEE International Conference on Intelligence and Security Informatics 2005. IEEE Computer Society, Atlanta, GA, USA, 2005: 511-516

[18] Paek S-H, Oh Y-K, Yun J B, et al. The Architecture of Host-based Intrusion Detection Model Generation System for the Frequency Per System Call[C]// ICHIT '06. vol. 2, 2006: 277-283

[19] 王宇,刘文予,罗宁. 基于扩充数据源的系统调用异常检测算法[J]. 计算机与数字工程,2006(1):13-16

[20] Wespi A, Dacier M, Debar H. Intrusion detection using variable-length audit trail patterns[C]// Proceedings of the 2000 Recent Advances in Intrusion Detection. 2000

[21] Forrest S, Hofmeyr S A, Somayaji A, et al. A Sense of Self for Unix Processes[C]// Proceedings of the 1996 IEEE Symposium on Security and Privacy. 1996: 120-127

[22] Tan K M C, Maxion R A. "Why 6?" defining the operational limits of stide, an anomaly-based intrusion detector[C]// Proceedings of the 2002 IEEE Symposium on Security and Privacy. Washington, DC, USA, IEEE Computer Society, 2002: 188

[23] Lee W, Stolfo S J. Data Mining Approaches for Intrusion Detection[C]// Proceedings of the 7th conference on USENIX Security Symposium. Volume 7, USENIX Association, San Antonio, Texas, 1998: 6-12

[24] Lee W, Stolfo S J, Mok K W. A data mining framework for building intrusion detection models[C]// Proceedings of the 1999 IEEE Symposium on Security and Privacy. 1999: 120-132

[25] Guan Jian, Liu Da-xin, Cui Bin-ge. An induction learning approach for building intrusion detection models using genetic algorithms[C]// WCICA 2004. 2004: 4339-4342

[26] Idris N B, Shanmugam B. Artificial Intelligence Techniques Applied to Intrusion Detection[C]// INDICON2005. IEEE, 2005: 52-55

[27] Gopalakrishna R, Spafford E H, Vitek J. Efficient intrusion detection using automaton inlining[C]// Security and Privacy. IEEE, 2005: 18-31

[28] Liao Y, Vemuri V R. Using text categorization techniques for intrusion detection[C]// Proc of USENIX Security Symposium. San Francisco. California USA, USENIX, 2002: 51-59

[29] 陆炜. 曾庆凯. 一种基于控制流的程序行为扩展模型[J]. 软件学报,2007,18(11),2841-2850

[30] Li Min, Wang Dong-liang. Anormaly Intrusion Detection Based on SOM[C]// ICIE '09. WASE International Conference. vol. 1, 2009: 40-43

[31] Wagner D, Dean R. Intrusion detection via static analysis[C]// Proceedings of the 2001 IEEE Symposium on Security and Privacy. 2001: 156-169

[32] Lam L-C, Chiueh T-C. Extraction of accurate application- specific sandboxing policy[C]// MILCOM 2005. IEEE

[33] Feng H H, Giffin J T, Yong Huang, et al. Formalizing Sensitivity in Static Analysis for Intrusion Detection[C]// IEEE Symposium on Security and Privacy. 2004: 194

[34] Giffin J T, Jha S, Miller B P. Efficient context-sensitive intrusion detection[C]// 11th Network and Distributed System Security Symposium(NDSS). San Diego, California, 2004

[35] 苏璞睿,杨轶,基于可执行文件静态分析的入侵检测模型[J]. 计算机学报,2006,29(9):1572-1578

[36] 田新广,高立志,孙春来,等. 基于系统调用和齐次 Markov 链模型的程序行为异常检测[J]. 计算机研究与发展,2007,44(9):1538-1544

[37] Gao De-bin, Reiter M K, Song D. On gray-box program tracking for anomaly detection[C]// Proceedings of the 13th Conference on USENIX Security Symposium. San Diego, CA 2004: 8-8

[38] Gao De-bin, Reiter M K, Song D. Gray-box extraction of execution graphs for anomaly detection[C]// Proceedings of the 11th ACM Conference on Computer and Communications Security. Washington DC, USA, 2004

[39] Bhatkar S, Chaturvedi A, Sekar R. Dataflow Anomaly Detection[C]// Proceedings of the 2006 IEEE Symposium on Security and Privacy. 2006: 48-62

[40] Eskin E, Stolfo S J, Lee W. Modeling System Calls for Intrusion Detection with Dynamic Window Sizes[C]// DISCEX II'01. vol. 1, 2001: 165

[41] Li Yong-zhong, Zhao Bo, Xu Jing, et al. Anoamaly intrusion detection method based on Rough Set Theory[C]// ICWAPR '08. vol. 2, 2008: 764-770

[42] 蔡忠闽,管晓宏,邵萍,等. 基于粗糙集理论的入侵检测新方法[J]. 计算机学报,2003,26(3):361-366

[43] 饶鲜,董春曦,杨绍全. 基于支持向量机的入侵检测系统[J]. 软件学报,2003,14(4):798-803

[44] Wespi A, Dacier M, Debar H. An intrusion-detection system based on the teiresias pattern-discovery algorithm[C]// Proceedings of the 1999 European Institute for Computer Anti-Virus Research Conference. 1999

[45] Inoue H, Somayaji A. Lookahead pairs and full sequences: a tale of two anomaly detection methods[C]// Proceedings of the 2nd Annual Symposium on Information Assurance. 2007

[46] Debar H, Dacier M, Nassehi M, et al. Fixed vs. Variable-length pattern for detecting suspicious process behavior[C]// JESORICS 98. 5th European Symposium on Research in Computer Security. LNCS, Louvain-la-Neuve, Belgium, 1998: 1-15

[47] Amer S H, Hamilton J A. Investigating Intrusion Detection Systems That Use Trails of System calls[C]// SPECTS 2008. International Symposium . June 2008

[48] Hu Jian-kun. A Simple and Efficient Hidden Markov Model Scheme for Host-Based Anomaly Intrusion Detection[J]. IEEE Network, 2009, 23(1): 42-47

[49] Han Sang-jun, Cho S-B. Evolutionary Neural Networks for Anomaly Detection Based on the Behavior of a Program[J]. IEEE Transactions on systems, man, and cybernetics-part B: cybernetics, 2006, 36(3): 559-570

[50] 张相锋,孙玉芳,赵庆松. 基于系统调用子集的入侵检测[J]. 电子学报,2004,32(8):1338-1341

[51] 苏璞睿,基于基因规划的主机异常入侵检测模型[J]. 软件学报,2003,14(6):1120-1126

[52] Zeng Fan-ping, Yin Kai-tao, Chen Ming-hui, et al. A New Anomaly Detection Method Based on Rough Set Reduction and HMM[C]// ICIS 2009. IEEE, 2009: 285-289

[53] Sujatha P K, Kannan A. A Behavior Based Approach to Host-Level Intrusion Detection using Self-organizing Maps[C]// First International Conference on Emerging Trends in Engineering and Technology. IEEE Computer Society, 2008: 267-1271

[54] 林果园,郭山清. 基于动态行为和特征模式的异常检测模型[J]. 计算机学报,2006,29(9):1553-1560

[55] Khreich W, Granger E, Sabourin R. Combining Hidden Markov Models for Improved Anomaly Detection[C]// IEEE ICC 2009 Proceedings. 2009

[56] Kosoresow A P, Hofmeyer S A. Intrusion detection via system call traces[J]. IEEE Software, 1997, 14(5): 35-42

[57] Michael C, Ghosh A. Two state-based approaches to program-based anomaly detection[C]// Proceedings of the 16th Annual Computer Security Applications Conference(ACSAC'00). New Orleans, LA, 2000: 21-29

(下转第 53 页)

[2] Jiang S, Smith S, Minami K. Securing Web servers against insider attack[C]// Proceedings of 17th Annual Computer Security Applications Conference. IEEE Press, 2001: 265-276

[3] Sadeghi A R, Stüble C, Wolf M, et al. Enabling fairer digital rights management with trusted computing[C]// Proceedings of ISC'07. LNCS 4779. Springer, 2007: 53-70

[4] Stumpf F, Tafreschi O, Röder P, et al. A robust integrity reporting protocol for remote attestation[C]// 2nd Workshop on Advances in Trusted Computing, WATC'06. 2006: 1-12

[5] Gasmi Y, Ahmad-Reza S, Patrick S, et al. Beyond Secure Channels[C]// Proceedings of the 2nd ACM Workshop on Scalable Trusted Computing, STC 2007. Alexandria, VA, USA, November 2007: 30-40

[6] Zhou Lingli, Zhang Zhenfeng. Trusted Channels with Password-based Authentication and TPM-based Attestation [C]// Proceedings of International Conference on Communication and Mobile Computing. 2010: 223-227

[7] Armknecht F, Gasmi Y, Sadeghi A-R, et al. An Efficient Implementation of Trusted Channels based on OpenSSL[C]// Proceedings of STC'08. Fairfax, Virginia, USA, October 2008: 41-50

[8] 林宏刚. 可信网络连接若干关键技术的研究[D]. 成都: 四川大学, 2006

[9] Trusted Computing Group. TCG Specifi_ cation Architecture Overview Specification's Revision1. 2[S]. https: // www. trustedcomputinggroup. org. Apr. 2004

[10] Davis C R. IPSec: VPN 的安全实施[M]. 北京: 清华大学出版社, 2002

[11] 范红. 互联网密钥交换协议及其安全性分析[J]. 软件学报, 2003, 14(3): 600-605

[12] Trusted Computing Group. TCPA Main Specification V1. 1b [S]. http: // www. trustedcomputinggroup. org/specs/TPM/TCPA_Main_TCG_Architecture_v1_1b. pdf, Sep. 2003

[13] 徐锐, 王震宇, 康新振. 可信计算环境证书机制中 SKAE 扩展项的分析[J]. 信息工程大学学报, 2008, 3(9): 90-93

(上接第 13 页)

[58] Sekar R, Bendre M, et al. A Fast Automaton-Based Method for Detecting Anomalous Program Behaviors[C]// Proceedings of the 2001 IEEE Sympo- sium on Security and Privacy. May 2001: 144

[59] Henry H F, Kolesnikov O M, et al. Anomaly Detection Using Call Stack Information[C]// Proceedings of the 2003 IEEE Symposium on Security and Privacy. 2003: 62-68

[60] Sekeh M A, Maarof M A. Fuzzy Intrusion Detection System via Data Mining Technique with Sequences of System Calls[C]// Information Assurance and Security, 2009, IAS. 2009: 154-157

[61] Zhen L, Bridges S M, Vaughn R B. Combining static analysis and dynamic learning to build accurate intrusion detection models [C]// Proceedings of the 3rd IEEE International Workshop on Information Assurance. March 2005

[62] 李闻, 戴英侠, 连一峰, 等. 基于混杂模型的上下文相关主机入侵检测系统[J]. 软件学报, 2009, 20(1): 138-151

[63] Parampalli C, Sekar R, Johnson R. A practical mimicry attack against powerful system-call monitors[C]// Proceedings of the 2008 ACM symposium on Information, computer and communications security. Tokyo, Japan, March 2008

[64] Wagner D. Mimicry attacks on host-based intrusion detection systems[C]// Proceedings of the 9th ACM conference on computer and communications security. Washington, DC, USA, 2002

[65] Kruegel C, et al. Automating mimicry attacks using static binary analysis[C]// Proceedings of the 14th Conference on USENIX Security Symposium. Baltimore, MD, 2005: 11-11

[66] Chen Shuo, et al. Non-control-data attacks are realistic threats [C]// Proceedings of the 14th Conference on USENIX Security Symposium. Baltimore, MD, 2005: 12

[67] Sufatrio, Yap R. Improving host-based ids with argument abstration to prevent mimicry attacks[C]// Proceedings of the International Sypmosium on Recent Advances in Intrusion Detectino(RAID). 2006: 146-164

[68] Tandon G, Chan P. Learning useful system call attributes for anomaly detection[C]// Proceedings of the 18th International FLAIRS Conference. 2005

[69] Li Peng, et al. Bridging the Gap between Data-flow and Control-Flow Analysis for Anomaly Detection[C]// Computer Security Applications Conference, 2008. ACSAC 2008. Annual, 2008: 392-401

[70] Pang Jian-jing, Peng Xin-guang. Detection of Programs Behaviors on Context Dependency[C]// Networks Security, Wireless Communications and Trusted Computing. 2009: 382-385

[71] Giffin J T, Dagon D, et al. Environment-sensitive intrusion detection[C]// Recent Advances in Intrusion Detection(RAID). September 2005

[72] Kruegel C, Mutz D, Valeur F, et al. On the detection of anomalous system call arguments[C]// Proceeding of ESORICS 2003. Berlin, Heidelberg: Springer-Verlag, 2003: 326-343

[73] Tandon G, Chan P. Learning rules from system call arguments and sequences for anomaly detection[C]// ICDM Workshop on Data Mining for Computer Security(DMSEC). 2003: 20-29

[74] Maggi F, Matteucci M, Zanero S. Detecting Intrusions through System Call Sequence and Argument Analysis[J]. Dependable and Secure Computing, 2010, 7(4): 381-395

[75] Mutz D, Robertson W. Exploiting Execution Context for the Detection of Anomalous System Calls[C]// Proceedings of the International Symposium on Recent Advances in Intrusion Detection(RAID). Gold Coast, Australia, 2007: 1-20

[76] Mutz D, Valeur F, Vigna G, et al. Anomalous system call detection[J]. ACM Trans, Inf. Syst. Secur., 2006, 9(1): 61-93

[77] Tandon G, Chan P. On the learning of system call attributes for host-based anomaly detection[J]. International Journal on Artificial Intelligence Tools, 2006, 15(6): 875-892

[78] 李红娇, 李建华. 基于程序行为异常检测的数据流属性分析[J]. 上海交通大学学报, 2007, 41(11): 1778-1782

[79] 黄金钟, 朱淼良, 郭晔. 基于文法的异常检测[J]. 浙江大学学报: 工学版, 2006, 40(2): 243-248

[80] 罗隽, 丁力, 潘志松, 等. 异常检测中频率敏感的单分类算法研究[J]. 计算机研究与发展, 2007, 44(Suppl.): 235-239

[81] 伏晓, 谢立. 安全报警关联技术研究[J]. 计算机科学, 2010, 37(5): 9-14