一种采用免疫原理的恶意软件检测方法

张福勇 齐德昱
(华南理工大学计算机系统研究所 广州 510640)

摘 要 针对现有恶意软件检测方法的不足,提出一种采用免疫原理的恶意软件检测方法。该方法采用程序运行时产生的IRP请求序列作为抗原,定义系统中的正常程序为自体、恶意程序为非自体,通过选定数量的抗体,采用人工免疫原理识别非自体。实验结果表明,此方法在恶意软件的检测方面具有较高的准确率,且误报和漏报率较低。

关键词 人工免疫,恶意软件,恶意软件检测,反病毒
中图法分类号 TP393 **文献标识码** A

Immune-based Method for Malware Detection

ZHANG Fu-yong QI De-yu
(Research Institute of Computer Systems, South China University of Technology, Guangzhou 510640, China)

Abstract In order to solve the problems existing in the current malware detection, a new malware detection method based on immune was proposed. In this method, the IRP request sequences created by running programs are regarded as antigen, and the normal programs in operating system are self, malwares are nonself. The nonself will be detected by some antibodies using artificial immunology. Experimental results reveal that this model has high true positive rate, and low false positive and false negative rate.

Keywords Artificial immune, Malware, Malware detection, Anti-virus

随着恶意软件复杂度的提高,要求恶意软件检测方法不仅能实现高效的检测,而且要具有很好的鲁棒性以应对可能出现的迷惑检测策略。当前恶意软件的检测技术主要有:基于特征码的检测方法、行为监测法和启发式方法。特征码检测法是目前应用最广泛的检测方法。通过采集恶意软件样本,提取其特征码,检测时将特征码与检测样本比较,判断是否有样本片段与此特征码吻合。这种方法只能检测已知病毒,对于目前广泛传播的变形病毒、多态病毒等无能为力。行为监测法是利用病毒的特有行为特征性来监测病毒的方法。通过对病毒多年的观察、研究,有一些行为是病毒的共同行为,而且比较特殊,在正常程序中比较罕见。当程序运行时,监视其行为,如果发现了病毒行为,则报警。行为监测法可以识别未知病毒,但误报率较高,且实现困难。启发式方法是通过仿真运行程序,寻找可疑的代码组合,如果可疑代码组合超过一定的阈值,则认为是恶意程序。启发式方法针对不同类型的病毒,需要用完全不同的规则来构建启发式分析器的判断逻辑,通常情况下误报率也较高。

生物免疫系统(Biological Immune System, BIS)具有良好的多样性、分布性、自学习、自适应和鲁棒性等特点,引起了研究人员的广泛关注。1974年,丹麦学者Jerne提出了第一个免疫系统的数学模型[1]。Forrest的否定选择算法[2]和计算机免疫学[3]概念的提出,推动了人工免疫系统在计算机安全领域的发展。Paul K. Harmer等人[4]运用人工免疫原理建立了一个计算机安全领域的应用框架,介绍了免疫原理在病毒检测、入侵检测等领域的应用。

在病毒检测方面,Dhaeseleer[5]使用阴性选择算法来检测被保护数据和程序文件的变化,该方法可以检测未知病毒,但只能针对静态数据和软件进行检测。Kephart[6]提出了一种新的病毒检测方法,它采用已知病毒的特征代码序列来检测已知病毒,而未知病毒则通过系统中发现的异常行为加以检测。Forrest[7]提出了一种基于系统调用序列的恶意行为检测方法。Lee[8]通过从程序入口点开始提取一系列字符串来区分自体与非自体,以实现病毒检测。Li Tao[9]提出了一种基于免疫的动态病毒检测模型,给出了自体、非自体的动态描述及其演化过程。

本文提出一种新的基于免疫学原理,采用IRP请求序列的恶意软件检测方法(Malware Detection Method based on Immune, MDMI)。此方法通过合理的抗原、抗体定义实现了较高的检测率,且误报、漏报率均较低。本文第1节给出了方法理论及抗原、抗体(检测器)的定义;第2节给出了实验结果;最后进行了总结。

1 恶意软件检测方法

1.1 抗原定义

到稿日期:2009-10-26 返修日期:2010-01-19 本文受国家技术创新基金项目(08C26214411198),粤港关键领域重点突破项目(2008A011400010),广州市创新基金项目(2007V41C0301)资助。

张福勇(1982—),男,博士生,主要研究方向为信息安全等,E-mail:fuyong1681@163.com;齐德昱(1959—),男,教授,博士生导师,主要研究方向为信息安全、计算机体系结构等。

在运行时恶意软件检测方面，很多研究人员选择采用API调用序列进行检测。但一些研究人员使用的API调用捕获工具运行在用户态[12]，只能捕获用户态的API调用，对于内核态的API调用就无能为力，因此不能检测采用驱动技术进行内核API调用的恶意软件。

鉴于此，我们选择采用程序运行时产生的IRP请求序列作为抗原。为了捕获程序运行时产生的IRP请求，开发了一个基于内核驱动技术的IRP请求捕获工具(MBMAS)。它可以捕获程序运行时创建的进程信息以及每个进程针对文件和注册表操作的IRP请求等信息，并将捕获的信息以XML格式进行存储。图1是IRP请求捕获工具的用户态界面。

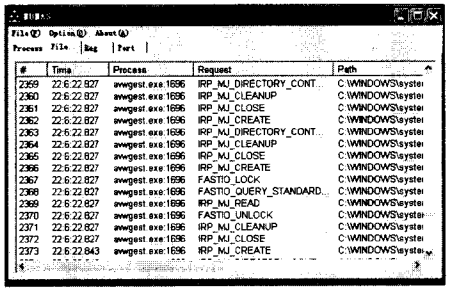

图1 IRP请求捕获工具

通过对捕获的IRP请求的统计分析发现共有30种不同的IRP请求。为了方便匹配，用ASCII字符0－M代表这30种类型。定义抗原域为$U\in\{0-M\}^l$，$l\in$自然数。定义抗原$Ag=\{x|x=IRP$请求序列对应的字符序列，$x\in U\}$。定义自体为正常代码产生的IRP请求序列对应的字符序列$S\subseteq Ag$，非自体为恶意代码产生的IRP请求序列对应的字符序列$N\subseteq Ag$，且满足$S\cup N=Ag$，$S\cap N=\varnothing$。

1.2 抗体(检测器)定义

使用免疫系统的抗体模拟恶意软件检测器，定义检测器集合$Ab=\{\alpha_1,\alpha_2,\cdots,\alpha_i\}$，$\alpha_i\in\{0-M\}^p$，$p\in$自然数，且$p\leqslant l$，$i\in$自然数。

为了有效检测恶意代码，需要将检测器按免疫原理进行学习和进化，将检测器分为未成熟检测器、成熟检测器和记忆检测器。未成熟检测器是由系统随机生成，需经过自体耐受才可以进行检测；经过自体耐受后存活的检测器称为成熟检测器，具有检测能力和一定的生命周期；若在生命周期内未匹配到任何非自体抗原，将被抛弃。记忆检测器是成熟检测器匹配到非自体抗原且达到一定阈值后激活产生的，原则上拥有无限的生命周期。但由于系统资源有限，不可能让记忆检测器集合无限增长，因此当记忆检测器数量达到最大值时，采用LRU算法淘汰最近最久未匹配到非自体的检测器。淘汰后的记忆检测器将进入成熟检测器集合，重新设定生命周期。

1.3 模型框架

系统模型的架构如图2所示。模型首先采用大量的自体抗原学习检测系统，得到一定数量的成熟检测器，而后将待检测序列交给检测系统进行检测，成熟检测器检测到非自体并达到一定阈值后成为记忆检测器，并对此记忆检测器进行克隆，克隆副本放入成熟检测器集合，同时将克隆副本进行变异，以适应外部环境的动态变化。本模型采用Hamming空间单点变异法[10]进行变异，变异后的检测器放入未成熟检测器集合，重新进行自体耐受。

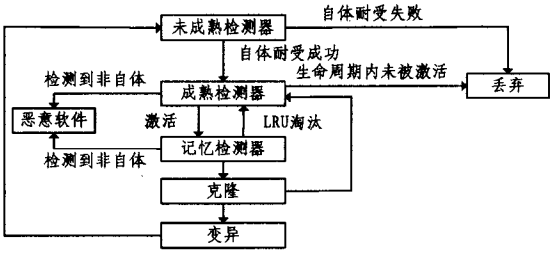

图2 基于免疫的恶意软件检测模型

1.4 匹配规则

检测器与抗原的匹配采用通用字符串匹配算法，匹配规则如图3所示。

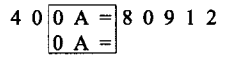

图3 匹配规则

采用检测器与抗原序列依次匹配，只要在抗原中找到与检测器匹配的字符串，即认为此检测器与抗原匹配。

1.5 自体耐受

未成熟检测器需要经过自体耐受，生成成熟检测器。本模型采用否定选择算法[2]进行自体耐受，如图4所示。

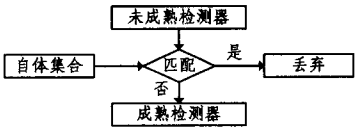

图4 自体耐受过程

自体耐受过程如式(1)描述：

$$f_{tolerance}(I)=I-\{i|i\in I\wedge\exists s\in S,f_{match}(|s|,|i|)\} \quad (1)$$

式中，I为未成熟检测器集合，S为自体集合，$|s|$为自体序列s的长度，$|i|$为未成熟检测器i的长度，f_{match}的定义如下：

$$f_{match}(|s|,|i|)=\begin{cases}1, & i \text{ match } s\\ 0, & \text{otherwise}\end{cases} \quad (2)$$

经自体耐受后的非成熟检测器将转变为成熟检测器M：

$$M=\{m|m\in Ab\wedge m\in f_{tolerance}(I),m.age<\lambda\} \quad (3)$$

式中，M为成熟检测器集合，λ为成熟检测器的生命周期阈值，$m.age$为成熟检测器m的生命周期。

定义$|M|$为成熟检测器M的数量。

自体耐受算法描述如下：

```
Begin
While 成熟检测器数量未达到值|M| do
  Begin
    随机生成一个长度为 k 的检测器;
    计算此检测器与每一个自体抗原是否匹配;
      If 不匹配
      Then 此检测器成为成熟检测器;
      Else 丢弃此检测器;
    End
End
```

1.6 检测过程

采用的检测函数如式(4)所示：

$$f_{detect}(L,k)=\begin{cases}1, & f_{match}(L,k)\wedge count>\eta\\ 0, & \text{otherwise}\end{cases} \quad (4)$$

式中，$count$为$f_{match}(L,k)=1$的检测器数量，η为匹配非自体的检测器数量阈值。

成熟检测器到记忆检测器的演变过程如式(5)—式(6)描述：

$$f_{mature}(M)=\{m|m\in M\wedge\exists n\in N, m.age<\lambda, f_{detect}(|n|,|m|)\} \quad (5)$$

$$R=\{r|r\in Ab\wedge r\in f_{mature}(M),|R|<\delta\} \quad (6)$$

式中，$|R|$为记忆检测器R的数量，δ为记忆检测器数量最大值。

恶意代码检测算法描述如下：

```
Begin
  For 每一个待检测序列 do
    Begin
      If 存在记忆检测器 r 匹配待检测序列
        Then 认为此待检测序列为非自体；
      Else
    Begin
      计算待检测序列与每个成熟检测器的匹配值 f_match，记录所有
      f_match=1 的检测器；
      If f_match=1 and count>η Then
        Begin
          认为此待检测序列为非自体；
          将此检测器放入记忆检测器集合；
          克隆此检测器，放入成熟检测器集合；
          克隆副本变异，放入未成熟检测器集合；
          End
        End
    End
End
```

2 仿真实验

实验分3个阶段进行：

(1)建立自体数据库。实验采用10台计算机进行自体数据的收集。计算机的操作系统均为Windows XP，系统中安装MBMAS进行IRP请求的捕获。自体数据收集工作进行了30天，共收集到自体序列43862个，平均每台机器每天146个。这里每一个进程从创建到消亡过程中产生的IRP请求为一个序列。将收集到的IRP序列转换为相应的字符序列。

(2)学习过程。实际是对未成熟检测器的自体耐受过程，设定成熟检测器数为10000、记忆检测器数为200。

(3)检测过程。本文收集了100个恶意软件和100个正常Windows可执行文件。恶意软件包括病毒、木马和蠕虫，所有恶意软件均为Win32 portable executable(PE)格式。

IRP请求的捕获是在一台新安装的Windows XP虚拟机中进行的，并且每运行完一个样本即将虚拟机恢复到新安装时的状态。在捕获IRP请求时，仅捕获程序运行过程中新创建进程的IRP请求，运行结束后将各进程的IRP请求连成一个序列。

设定初始检测器长度$k=3$，匹配非自体的检测器数量阈值$\eta=1$，采用检测率(TP，非自体被检测出的概率)、误报率(FP，自体被误认为非自体的概率)、漏报率(FN，非自体被误认为自体的概率)来评估检测性能。

当检测器长度$k=2,3,4$时得到的检测结果如表1所列。

表1 检测器长度与检测性能的关系

长度k	检测率/%	误报率/%	漏报率/%
2	72	1	28
3	98	2	2
4	100	12	0

由表1可以看出，当$k=3$时，检测率可以达到98%，而误报率仅为2%。原因为采用了大量的自体数据对系统进行学习，实际检测时将自体误认为非自体的几率较小；仅有少数非自体，它们的序列长度很短，没有产生过多的行为，与自体非常相似，才会被误认为是自体。实验结果表明，正常程序跟异常程序在IRP请求上确实存在一定差异。采用4位的检测器达到了100%的检测率，但是存在较大的误报。

图5、图6显示了匹配非自体的检测器数量阈值η与检测性能的关系。结果表明，阈值$\eta=1$时检测率为98%，误报率为2%。当$\eta=2$时得到96%的检测率，低于$\eta=1$时得到98%的检测率，但其误报率为1%。因此在实际应用中可根据需要调整η值的大小。

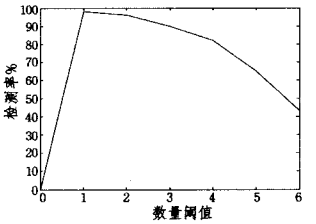

图5 数量阈值与检测率的关系

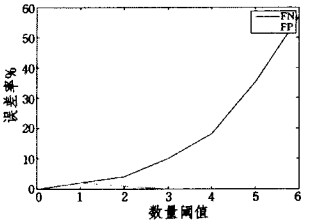

图6 数量阈值与误差率的关系

为体现算法性能，将算法与Forrest等人提出的ARTIS模型[2,11]进行比较。ARTIS采用的r-连续位匹配法[11]进行匹配。为了应用r-连续位匹配法，将每一种IRP请求转变为一个唯一的5位二进制字符串。表2是取不同r值时的检测结果。当$r=13$时得到了96%的检测率，但误报率高达35%，原因是将IRP请求转化为二进制串后不能真实反映IRP请求的顺序。图7是$k=3$，$r=13$时MDMI与ARTIS检测结果的比较，可以看出MDMI达到的检测率要高于ARTIS。原因为ARTIS采用的r-连续位匹配法不能很好地区分样本中的自体与非自体。

图8是$k=3$，$r=13$时MDMI与ARTIS检测效率的比较。可以看出，MDMI的检测效率要明显高于ARTIS，因为ARTIS中采用的待检测序列的长度5倍于MDMI的序列长度，且ARTIS的检测器长度也高于MDMI的检测器。

表2 r长度与检测性能的关系

长度k	检测率/%	误报率/%	漏报率/%
11	88	14	12
12	93	22	7
13	96	35	4

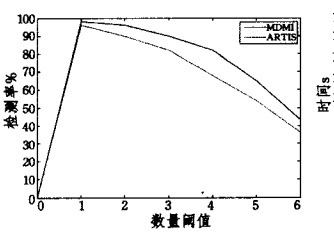

图7 MDMI与ARTIS检测率比较

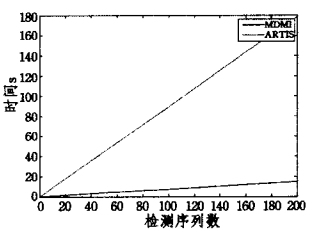

图8 MDMI与ARTIS检测效率比较

(下转第217页)

相应的虚拟传感器数据。将控制机和嵌入式系统与仿真机相连进行仿真实验，无人艇的基础运动控制试验结果如图 6 所示。

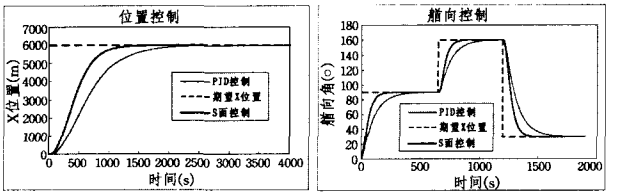

图 6　无人艇基础运动控制试验结果

由仿真试验结果可知，两种控制方法都具有良好的控制能力。相比 PID 控制器，S 面控制器具有更好的动态控制响应特性，其收敛速度、超调量等多种性能都明显优于 PID 控制器。S 面控制器与 PID 控制器不同的是，前者采用非线性函数来拟合具有强非线性特性的控制对象，控制效果好于 PID 控制器。然而，S 面控制和 PID 控制、模糊控制一样，存在难以解决运动耦合的问题，以及不具备在线学习的能力。

结束语　仿真试验结果表明，基础运动控制系统能够正常工作，并验证了系统的软件逻辑、软件体系结构、硬件体系结构、数据接口及其系统集成的合理性和可行性。试验表明，文中嵌入式运动控制系统的体系结构和数据接口等满足设计要求，这为今后的相关研究提供了技术基础。

无人艇控制系统在软件上采用简单实用的控制算法；硬件上选用基于 PC/104 总线的多板嵌入式系统；同时采用了 VxWorks 实时操作系统。这些充分保证了控制系统的实时性，能满足无人艇航速快、机动性强的要求。

在数据滤波、控制算法、故障诊断与紧急处理等方面仍然需要完善，尤其是无人艇在复杂海洋环境中的运动控制性能等需要验证。这些将是以后运动控制系统的研究重点和需要不断改进的内容。

参 考 文 献

[1] 徐玉如，苏玉民，庞永杰. 海洋空间智能无人运载器技术发展展望[J]. 中国舰船研究，2006，1(3)：2-4

[2] Manley J E. Unmanned Surface Vehicles，15 Years of Development[C]// Proc. Oceans 2008 MTS/IEEE Quebec Conference and Exhibition. Ocean'08，Quebec City，2008：1-4

[3] Veers J，Bertram V. Development of the USV Multi - Mission Surface Vehicle III[C]// 5th Int. Conf. Computer and IT Application in the Maritime Industries. COMPIT，2006：345-355

[4] 高双，朱齐丹，李磊. 基于神经网络的高速无人艇模糊 PID 控制[J]. 系统仿真学报，2007，19(4)：776-777

[5] 高双，朱齐丹，李磊. 滑行艇高速运动建模与姿态控制仿真[J]. 系统仿真学报，2008，20(16)：4461-4462

[6] 陈慈发，叶祥明，宋亚萍，等. 基于多智能体技术的水面无人艇测控系统研究[J]. 舰船科学技术，2008，30(3)：88-89

[7] 甘永，王丽荣，刘建成，等. 水下机器人嵌入式基础运动控制系统[J]. 机器人，2004，26(3)：246-249

[8] 甘永. 水下机器人运动控制系统体系结构的研究[D]. 哈尔滨：哈尔滨工程大学，2007

[9] 孔祥营，柏桂枝. 嵌入式实时操作系统 VxWorks 及其开发环境 Tornado[M]. 北京：中国电力出版社，2002：1-42

[10] 王丽荣，徐玉如. 水下机器人传感器故障诊断[J]. 机器人，2006，28(1)：25-26

[11] Omerdic E，Roberts G. Thruster Fault Diagnosis and Accommodation for Open-frame Underwater Vehicles[J]. Control Engineering Practice，2004，12：1575-1598

[12] 刘学敏，徐玉如. 水下机器人运动的 S 面控制方法[J]. 海洋工程，2001，19(3)：81-84

(上接第 163 页)

结束语　本文提出了一种采用 IRP 请求序列的恶意软件检测方法，相比传统的病毒检测方法在检测率及检测速度上均具有很大优势。本文方法不仅可以实现静态检测，还可以实现动态检测，采用 MBMAS 动态监视系统 IRP 请求，实时将请求序列送给 MDMI 进行检测，一旦达到匹配阈值即报警。MDMI 的动态检测能力为计算机病毒免疫系统的设计提供了一个新的方向。

参 考 文 献

[1] Jerne N K. Towards a network theory of the immune system [J]. Annual Immunology，1974，125C(1/2)：373-389

[2] Forrest S，Perelson A S，Allen L，et al. Self-nonself discrimination in a computer[C]// The IEEE Symposium on Research in Security and Privacy. Oakland：IEEE，1994：202-212

[3] Forrest S，Hofmeyr S A，Somayaji A. Computer immunology [J]. Communications of the ACM，1997，40(10)：88-96

[4] Harmer P K，Williams P D，Gunsch G H，et al. An artificial immune system architecture for computer security applications [J]. IEEE Transactions on Evolutionary Computation，2002，6(3)：252-280

[5] Dhaeseleer P，Forrest S，Helman P. An immunological approach to change detection：algorithms，analysis and implications[C]// IEEE Symposium on Security and Privacy. Oakland：IEEE，1996：110-119

[6] Kephart J O，Sorkin G B，Swimmer M. An immune system for cyberspace[C]// IEEE International Conference on Systems，Man，and Cybernetics. Orlando：IEEE，1997：879-884

[7] Forrest S，Hofmeyr S A，Somayaji A，et al. A sense of self for unix processes[C]// IEEE Symposium on Security and Privacy. Oakland：IEEE，1996：120-128

[8] Lee H，Kim W，Hong M P. Biologically inspired computer virus detection system[C]// 1st International Workshop on Biologically Inspired Approaches to Advanced Information Technology. Lausanne：Springer，2004：153-165

[9] Li Tao. Dynamic detection for computer virus based on immune system [J]. Science in China，Series F-information Sciences，2008，51(10)：1475-1486

[10] 李涛. 计算机免疫学 [M]. 北京：电子工业出版社，2004：60-62

[11] Hofmeyr S，Forrest S. Architecture for an artificial immune system [J]. Evolutionary Computation，2000，8(4)：443-473

[12] Manzoor S，Shafiq M Z，Tabish S M，et al. A sense of 'danger' for windows processes[C]// LNCS，ICARIS. Heidelberg：Springer，2009，5666：220-233