基于行为分析和特征码的恶意代码检测技术 *

李 华[1,2], 刘 智[3], 覃 征[1], 张小松[3]
(1. 清华大学 计算机科学与技术系, 北京 100084; 2. 78155 部队, 成都 610036; 3. 电子科技大学 计算机科学与工程学院, 成都 611731)

摘 要: 提出一种新的恶意代码检测技术,能自动检测和遏制(未知)恶意代码,并实现了原型系统。首先用支持向量机对恶意代码样本的行为构造分类器,来判断样本是否是恶意代码,同时对恶意代码提取出特征码。运行在主机的代理利用特征码识别恶意代码并阻断运行。为了精确分析程序行为,将程序放入虚拟机运行。实验结果表明,相对于朴素贝叶斯和决策树,系统误报率和漏报率均较低,同时分布式的系统架构加快了遏制速度。

关键词: 恶意代码; 行为分析; 特征码; 虚拟机
中图分类号: TP309.5 **文献标志码**: A **文章编号**: 1001-3695(2011)03-1127-03
doi:10.3969/j.issn.1001-3695.2011.03.094

Technique of detecting malicious executables via behavioral and binary signatures

LI Hua[1,2], LIU Zhi[3], QIN Zheng[1], ZHANG Xiao-song[3]
(1. *Dept. of Computer Science & Technology, Tsinghua University, Beijing* 100084, *China*; 2. *Unit* 78155 *of PLA, Chengdu* 610036, *China*; 3. *College of Computer Science & Engineering, University of Electronic Science & Technology of China, Chengdu* 611731, *China*)

Abstract: This paper proposed a new approach that could effectively detect and restrict (unknown) malware, and implemented a prototype system. First, used support vector machine to build classifier, which could judge whether a program was malicious or not, and extracted the malware's signature. Agents running in host could detect malware and stop its execution. To analyze precise behaviors, put samples in virtual machines for executions. Experiment results show compared with naive Bayes and decision tree, our system yields low false positives as well false negatives, and the distributed architecture accelerates restriction.

Key words: malware; behavioral analysis; binary signatures; virtual environment

恶意程序是从软件系统中增加、修改或删除的任意代码,用于蓄意的对目的主机造成破坏[1]。通常恶意代码根据传播途径和存在方式划分为三类:病毒、特洛伊木马和蠕虫,前两种属于二进制形式的恶意代码。由于软件同构性和互联网模型对访问缺乏有效控制,它们造成的危害越来越大,波及范围越来越广,已造成了严重威胁。在本文中,将基于二进制代码的恶意代码简称为恶意代码。

恶意代码检测系统必须符合下列三个特征:准确性、扩展性和适应性[2]。在近十年中,入侵检测系统(IDS)和反病毒软件被广泛应用于恶意代码检测,但它们大都依赖于特征码库,因此对未知恶意代码和 0-day 攻击无能为力。研究表明基于人工分析的检测和遏制技术效率低,而且大大滞后于攻击事件。业界和研究机构已将注意力转向自动化的检测机制研究。然而,目前提出的检测模型仅仅依赖于单一检测机制,很难达到上述三个目标。

本文提出一种新的恶意代码检测与遏制模型,从行为和二进制特征两个层面刻画恶意代码,克服了传统技术的不足。相对于系统调用序列和内存使用,基于操作系统的事件能提供更丰富的信息,因此把它们作为行为分析的粒度。该检测模型的核心思想是利用数据挖掘技术检测出未知恶意代码,并将提取出的二进制特征码分发到各主机代理;后者依据特征码对恶意代码进行阻断。整个过程无须人工操作。用于实验的恶意代码来自 VX Heavens[3] 及采用蜜罐搜集的样本。在原型系统中,通过 ROC 表明具有较高的检测精度。

1 传统检测技术

误用检测通过特征码匹配方式检测攻击,如 Snort[4] 是目前流行的开源网络入侵检测系统。误用检测的最大缺陷在于需要维护庞大的特征码库,而且无法检测未知攻击。然而最近几年研究表明,基于特征码检测能被轻易绕过[5]。另一方面,基于源码分析通常是静态检测,但商业软件或恶意代码往往并不提供源码。异常检测的最大优点在于能检测未知攻击,但难点在于如何建立正常模式库。基于行为的分析可划分为基于系统调用序列的分析和基于系统事件的分析。1996 年,美国新墨西哥州立大学的 Forrest[6,7] 首次将人工免疫原理用于计算机安全检测,提出利用系统调用序列的入侵检测。其出发点是程序在正常执行条件下其短系统调用序列恒定,而且与其他

收稿日期: 2010-07-01; **修回日期**: 2010-09-08 **基金项目**: 国家自然科学基金资助项目(60673024);国防"十一五"预研基金资助项目(102060206,402040202)

作者简介:李华(1973-),女,四川仪陇人,博士研究生,主要研究方向为信息安全(lihua05@mails.tsinghua.edu.cn);刘智(1985-),男,四川南充人,硕士,主要研究方向为信息安全;覃征(1956-),男,湖南石门人,教授,博士,主要研究方向为知识工程、软件体系结构、计算机系统集成与电子商务、复杂信息处理、移动计算等;张小松(1968-),男,四川双流人,副教授,博士,主要研究方向为信息安全.

程序执行的系统调用序列相异。文献[8]利用自动机对攻击进行检测。基于系统调用序列的检测主要针对 UNIX 下的特权进程,即程序本身是正常的,因而不适用于恶意代码检测。恶意代码常常与操作系统交互,且事件能提供丰富的信息,因此本文以操作系统的事件作为行为粒度。文献[9]利用模拟器进行恶意代码检测,但效率较低。文献[10]是与本文工作较接近的文献,其主要缺点在于监控的行为和深度不尽人意,且没有实现特征码提取。

2　系统结构

2.1　概述

本文的检测系统具有三层架构,如图 1 所示。三个层次对应不同的功能实体:代理、控制器和分析中心。在一个检测区域(通常是一个局域网)内,存在代理和控制器两种实体。每台主机都具有一个主机代理,它负责监视主机异常活动以及根据规则采取相应措施,如下载恶意代码特征库。一个检测区域只有一个网络代理,它负责监视网络异常活动,如端口扫描等。在系统中有三种数据流:原数据、事件和指纹,分别对应于三个层次的实体。原数据来自代理收集到的数据,未经任何处理。事件经控制器产生,是对原数据的处理和加工,然后被发送到分析中心。恶意代码特征码包括二进制特征和行为特征,前者用来阻断其运行,后者用来检测未知恶意代码。控制器是代理和分析中心的接口。在每个检测区域只有一个控制器,向上传递代理搜集的数据,向下发送控制命令,如启动或停止代理。同时也发送配置规则给代理。在整个检测系统中有一个唯一的分析中心,它是系统的核心部分,接受用户的配置输入,将分析结果展示给用户。此外,分析中心最重要的功能是对恶意代码行为进行深度分析,检测出未知恶意代码;同时提取出已知恶意代码的二进制特征。首先,利用数据挖掘算法对恶意行为进行分析,如果精确度不够或人工认为分析不精确,则将恶意代码放入虚拟机中执行。由于虚拟机是一个受控、干净的执行环境,许多在特定环境下的执行路径能被观察到。

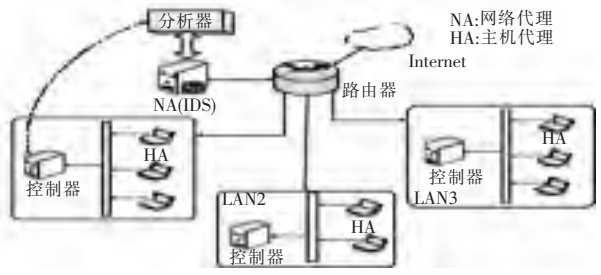

图 1　系统框架图

2.2　代理

代理分为主机代理(HA)和网络代理(NA)。代理相对独立,彼此不通信,仅仅将收集到的数据转发给上层的控制器,由控制器根据配置信息决定对代理的控制,如启动或停止代理。主机代理不仅监视主机异常活动,上报日志,同时根据二进制特征码对恶意代码进行阻断运行。因此,结合了行为和二进制的指纹不仅能对已知恶意代码进行阻断,而且能检测未知的恶意代码。代理的功能分为:

a)数据收集。主机代理和网络代理将异常事件或“感兴趣”的事件发送给对应的控制器。本文将网络代码放置在局域网层次而不是主机上,因为运行在主机上的代理会对主机造成比较大的负载影响。核心问题是哪些数据需要收集,也就是说通过哪些行为来刻画恶意代码。

b)响应控制。该模块由两部分组成,即内核模块和控制模块。控制模块根据上层实体下发的控制信息采取相应行为,如启动、停止代理,更新指纹库。内核模块含有由分析中心下发的二进制特征码库来检测已知恶意代码并阻断其运行。本文采用文件过滤驱动(file system filter driver)[7]来监控并阻断恶意代码运行。文件过滤驱动实质上是对文件操作系统访问的拦截,从而改变操作行为。概括地说,文件过滤驱动能使人们对文件的访问(如读、写、删除)进行拦截并进行相应操作。文件过滤驱动工作在操作系统内核态,在应用层有对应的控制程序。每当过滤驱动拦截对文件的打开操作,将文件信息(如文件路径)发送给控制程序,后者提取程序的特征码并进行比对,若二进制特征符合,则发送命令给驱动阻断其运行。文件过滤驱动是 Windows 官方推荐的方法,而且本文采用了高效的哈希表,因此对操作系统效率影响极小。

2.3　控制器

控制器是代理和分析中心的接口。它也有两个功能:数据处理和发送命令给代理。前者接收代理发送的数据,并进行适当处理(如删除冗余数据),然后转发给最顶层的分析中心。当控制器接收到分析中心发送的控制命令时,它将命令发送给代理。

2.4　分析中心

分析中心是整个检测系统的核心部分。它含有三个模块:用户接口、恶意代码行为分析、二进制特征生成以及命令控制。用户通过用户接口查询报警信息,更改配置。分析中心含有中心数据库用来存放所有的报警日志信息和可疑程序样本。一旦恶意代码二进制指纹被提取,它将分发到所有的主机代理。底层的代理和控制器不直接通信,它们受分析中心控制。虽然集中式的分析处理可能会对效率带来一些好的影响,但模型主要以自动化方式高精度地检测恶意代码。所以,在以后工作中,可以采取完全的分布式策略。

3　实现

本文选择操作系统事件而不是系统调用序列作为行为特征。表 1 列出了需要观察和分析的特征,这些特征被大多数木马或 rootkit 具有。虚拟机环境本文选取的是 VMWARE[5]。通过虚拟机环境和主机件的管道通信,虚拟机将分析得到的恶意代码行为自动发送到主机。

表 1　恶意代码行为

对象	行为
进程	隐藏,创建
注册表	修改,创建
文件	修改系统文件
原生 API	钩子
端口	钩子

本文采用通用的二进制特征提取(本文不涉及变形技术)。对于 Windows PE 来说,从入口点(entry point)后选取 32 或 64 Byte 内容作 hash 变换。检测框架流程如图 2 所示。

4　实验结果与分析

原型系统搭建在 Windows Service Pack 2,奔腾 4 处理器,

内存 512 MB 的软硬件环境下。实验数据来源于 VXHeaven (vx. netlux. org)和本文用蜜罐收集的恶意代码,样本数量为 1 863个,加上正常样本 230 个。将这些样本(随机)分为三组,如表 2 所示,样本为正常程序和(恶意代码),后者在括号里表示,训练集为恶意代码。本文对其行为进行训练(SVM、朴素贝叶斯和决策树算法)构造分类器,测试时包含恶意代码和正常程序,并分别得到漏报率和误报率,见表 3(括号里为误报率)。

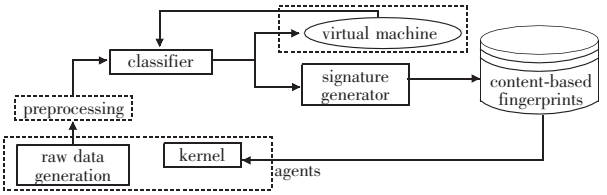

图 2　检测框架图

表 2　测试样本组成

对象	第一组	第二组	第三组
样本	650(81)	505(62)	708(87)
训练集	512	442	591

表 3　实验结果

对象	第一组	第二组	第三组
SVM	94.5(8.2)	96.1(7.4)	94.2(7.1)
贝叶斯	81.5(10.1)	83.8(8.4)	83.0(9.0)
决策树	85.9(8.8)	88.4(6.5)	90.5(9.2)

在算法实现时,本文采用 LIBSVM 实现了 SVM 算法,朴素贝叶斯和决策树用 C ++ 编写。在用 SVM 训练时,有以下步骤:a)对收集到的数据进行格式转换,使之能被 LIBSVM 分析;b)对数据简单缩放操作;c)选取 RBF 核参数;d)采用交叉验证选择最佳参数 C 与 g;e)采用最佳参数 C 与 g 对整个训练集进行训练获取支持向量机模型;f)用获取的模型进行测试与预测。本文选取的核参数为线性核参数。C 和 g 选用 LIBSVM 默认参数(相对于线性核参数)。

朴素贝叶斯的理论依据是各行为相互独立,观察到联合的概率正好是对每个单独属性的概率乘积。由于分类结果只有两种(即恶意代码和正常程序),实验设定分类标准为:恶意代码行为的概率乘积大于正常行为概率的乘积。

决策树采用 Microsoft SQL Server Analysis Service 提供的算法实现。Microsoft 决策树算法通过在树中创建一系列拆分来生成数据挖掘模型。这些拆分以节点来表示。每当发现输入列与可预测列密切相关时,该算法便会向该模型中添加一个节点。该算法确定拆分的方式不同,主要取决于它预测的是连续列还是离散列。本文的实验针对的是离散型变量。

从表 3 可以看出,三组实验中 SVM 的漏报率和误报率均明显高于朴素贝叶斯和决策树。贝叶斯要求各属性相互独立,但各个恶意代码行为并不完全满足这一要求,因而漏报率和误报率均较高。决策树结果比 SVM 低,但比贝叶斯高。SVM 的实验结果最好,因为它对样本没有特定要求,核参数选取适当,特别是在实验样本数目较大时,优势明显。

5　结束语

本文提出了一种新型的结合行为特征和二进制特征的恶意代码检测技术。它能快速发现未知恶意代码并即时作出响应。但它存在如下缺陷:a)无法对抗变形技术;b)对网络攻击缺乏有效检测机制。在以后的工作中,笔者将在上述两方面做出改进。

参考文献:

[1] MCGRAW G, MORRISETT G. Attacking malicious code: a report to the infosec research council[J]. IEEE Software, 2000, 17(5): 33-41.

[2] LEE W, STOLFO S J, MOK K W. A data mining framework for building intrusion detection models[C]//Proc of IEEE Security and Privacy. 1999: 120-132.

[3] VX heavens[EB/OL]. (2010)[2007]. http://vx. netlux. org.

[4] ROESCH M. Snort-lightweight intrusion detection for networks[C]// Proc of the 13th USENIX Conference on System Administration. 1999: 229-238.

[5] WAGNER D, SOTO P. Mimicry attacks on host based intrusion detection systems[C]//Proc of the 9th ACM Conference on Computer and Communications Security. New York: ACM, 2002: 255-264.

[6] HOFMEYR S A, FORREST S, SOMAYAJI A. Intrusion detection using sequences of system calls[J]. Journal of Computer Security, 1998, 6(3): 151-180.

[7] JON H, PASCAL M. Can source code auditing software identify common vulnerabilities and be used to evaluate software security[C]// Proc of the 37th Annual Hawaii International Conference on System Sciences. 2004: 4405-4414.

[8] SEKAR R, BENDRE M, DHURJATI D, *et al*. A fast automaton-based method for detecting anomalous program behaviors[C]//Proc of IEEE Symposium on Security and Privacy. Washington DC: IEEE Computer Society, 2001: 144-155.

[9] BAYER U, KRUEGEL C, KIRDA E. TTAnalyze: a tool for analyzing malware[C]//Proc of the 15th European Institute for Computer Antivirus Research Annual Conference. 2006: 1-12.

[10] 刘巍伟,石勇,郭煜,等. 一种基于综合行为特征的恶意代码识别方法[J]. 电子学报, 2009, 37(4): 696-700.

(上接第 1126 页)

[10] TAN Xiao-bin, XI Hong-sheng. Hidden semi-Markov model for anomaly detection[J]. Applied Mathematics and Computation, 2008, 205(2): 562-567.

[11] KURAMOCHI K, KARYPIS G. An efficient algorithm for discovering frequent subgraphs[J]. IEEE Trans on Knowledge and Data Engineering, 2004, 16(9): 1038-1051.

[12] KETKAR N S, HOLDER L B, COOK D J. Subdue: compression-based frequent pattern discovery in graph data[C]//Proc of the 1st International Workshop on Open Source Data Mining: Frequent Pattern Mining Implementations. Chicago: ACM Press, 2005: 71-76.

[13] PADMANABHAN S, CHAKRAVARTHY S. HDB-subdue: a scalable approach to graph mining[C]//Proc of the 11th International Conference on Data Warehousing and Knowledge Discovery. Linz: Springer-Verlag, 2009: 325-338.

[14] System call data sets from the computer science department of UNM [EB/OL]. (2003-05-27) [2010-02-04]. http://www. cs. unm. edu/ ~ immsec/data/.

[15] WANG Wei, ZHANG Xiang-liang, GOMBAULT S. Constructing attribute weights from computer audit data for effective intrusion detection [J]. Journal of Systems and Software, 2009, 82(12): 1974-1981.

[16] HOANG X, HU Jian-kun, BERTOK P. A Program-based anomaly intrusion detection scheme using multiple detection engines and fuzzy inference[J]. Journal of Network and Computer Applications, 2009, 32(6): 1219-1228.

[17] TAN K M C, MAXION R A. "Why 6?" defining the operational limits of stide, an anomaly-based intrusion detector[C]//Proc of IEEE Symposium on Security and Privacy. Pittsburgh: [s. n], 2002: 188-201.