基于PE静态结构特征的恶意软件检测方法

白金荣[1,2] 王俊峰[1] 赵宗渠[1]

(四川大学计算机学院 成都 610065)[1] (云南玉溪师范学院信息技术工程学院 玉溪 653100)[2]

摘 要 针对现有检测方法的不足,提出了一种通过挖掘PE文件结构信息来检测恶意软件的方法,并用最新的PE格式恶意软件进行了实验。结果显示,该方法以99.1%的准确率检测已知和未知的恶意软件,评价的重要指标AUC值是0.998,已非常接近最优值1,高于现有的静态检测方法。同时,与其他方法相比,该检测方法的处理时间和系统开销也是较少的,对采用加壳和混淆技术的恶意软件也保持稳定有效,已达到了实时部署使用要求。此外,现有的基于数据挖掘的检测方法在特征选择时存在过度拟合数据的情况,而该方法在这方面具有较强的鲁棒性。

关键词 恶意软件检测,结构特征,数据挖掘,PE

中图法分类号 TP309 **文献标识码** A

Malware Detection Approach Based on Structural Feature of PE File

BAI Jin-rong[1,2] WANG Jun-feng[1] ZHAO Zong-qu[1]

(College of Computer Science and Technology, Sicuan University, Chengdu 610065, China)[1]

(School of Information Technology and Engineering, Yuxi Normal University, Yuxi 653100, China)[2]

Abstract In order to solve the problems existing in malware detection, we proposed a novel malware detection approach by mining structural features of PE (Portable Executable) files and conducted the against recent Win32 malwares. Experimental results indicate that the accuracy of our method is 99.1% and the value of the AUC is 0.998 which is close to 1(The AUC value of the best possible classifier) and better than that of other static approaches. Compared with other static approaches, our method achieves higher detection accuracy with less detection time, is hard to evade by malware which applies the obfuscation and packing technique, and is real-time deployable. Most malware detection approaches using data mining may overfit experimental data in feature selection, but our experiments show that our method overcomes this problem.

Keywords Malware detection, Structural features, Data mining, PE

1 引言

恶意软件是指在未经用户许可的情况下,对用户计算机进行破坏,侵犯用户合法权益的软件,包括病毒、蠕虫、木马等。国内网络安全厂商金山安全实验室报告显示,2009年,金山毒霸共截获新增恶意软件2000多万个,与5年前新增恶意软件数量相比,增长了400倍[1]。与此同时,尽管研究人员对恶意软件的检测方法进行了大量的研究,但是目前恶意软件的检测技术依然有很大的局限性。当前的反病毒软件主要是利用恶意软件或被感染文件中的特征码(即每种恶意软件所独有的十六进制代码串)进行检测。它存在以下3个方面的缺点:(1)它几乎不能检测新的或未知的恶意软件种类,能检测的恶意软件在加壳后又不能检测,使用多态[2]和变形[3]技术的恶意软件在传播的过程中不断随机地改变着二进制文件内容,没有固定的特征,使用该方法不能检测;(2)反病毒软件特征库的大小和特征的匹配时间都呈指数级增加;(3)反病毒厂商的响应速度严重滞后,存在真空期(即从恶意软件出现到反病毒软件能查杀该恶意软件),在真空期恶意软件可能已经产生了严重的破坏。近年来,研究人员提出了一些能够检测未知恶意软件的非特征码方法。这些方法分析了恶意软件以下几个层次的特征:字节码序列、反汇编代码序列、反汇编代码的控制流图、反汇编代码中的API调用和运行时API调用序列。这些方法存在的主要问题是提取特征复杂、检测时间较长、占用系统资源较多和误报率较高。

一个可执行二进制软件都有其基本的数据格式,如Windows的PE格式、Linux的ELF格式。可信软件一般会遵循格式要求的约束,恶意软件或被恶意软件感染的可执行文件本身也遵循格式要求的约束,但静态结构在属性上表现出和正常文件的一些差异。现有的反病毒软件都在一定程度上使用了静态结构异常作为恶意软件检测的启发式规则。Szor[4]中详细总结归纳了恶意软件和被感染文件存在的结构异常,并讨论了基于结构异常的一些启发式检测方法。Weber[5]通过对PE文件的结构进行可视化,应用可视化技术来诊断可执行文件是否存在结构异常,为专业分析人员提供了快速识

到稿日期:2012-03-09 返修日期:2012-06-09 本文受国家"863"计划基金项目(2008AA01Z208),四川省青年基金(09ZQ026-028)资助。

白金荣(1977-),博士生,讲师,主要研究方向为网络安全、数据挖掘,E-mail:baijr223@163.com;王俊峰(1976-),男,博士,研究员,博士生导师,主要研究方向为网络安全、空间信息网络;赵宗渠(1976-),博士生,讲师,主要研究方向为网络安全、数据挖掘。

别恶意软件的方法。这些方法的局限性是只对结构存在明显异常的恶意软件或被感染文件有效。Kolter 等[6]以二进制可执行文件的 N 元字节序列(n-grams)作为特征,使用分类算法来对恶意软件和正常文件进行分类,取得了较好的实验结果。Dai[7]重现了 Kolter 的实验,并分析了过滤选择后的特征,发现大部分特征都是文件的结构特征和字符串信息,很少部分是机器码系列。

针对非特征码方法存在的问题,本文提出了一种基于二进制可执行文件静态结构属性的恶意软件检测方法。本文深入分析 Windows 平台 PE 文件静态结构属性,提取在恶意软件和正常软件间具有很好区分度的属性,使用数据挖掘分类算法进行学习,从而正确识别恶意软件和正常文件。实验结果显示,所使用分类算法能够以 99.1%的准确率检测已知和未知的恶意软件,且检测时间较短,占用系统资源较少,可实际部署于反病毒软件中使用。

2 PE 格式概要

PE(Portable Executable)格式,是 Windows 的标准可执行文件的格式。PE 文件格式被组织为一个线性的数据流,它由一个 MS-DOS 头部开始,接着是 PE 文件头,之后紧接着节头部,节头部之后跟随着所有的节实体,图 1 是 PE 文件的基本结构。

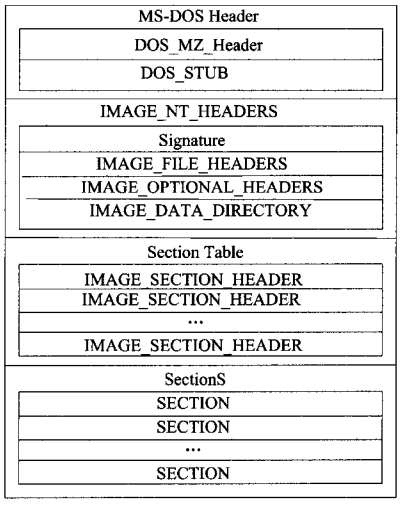

图 1 PE 文件结构

2.1 MS-DOS 文件头

每一个 PE 文件都以一个小的 MS-DOS 可执行文件开始。当可执行文件在没有安装 Windows 的机器上运行时,这个程序至少可以输出一条消息,用来指明它需要运行在 Windows 平台上。

2.2 PE 文件头

PE 文件头由 3 部分组成:Signature、IMAGE_FILE_HEADER 结构和 IMAGE_OPTIONAL_HEADER32 结构。Signature 是一个合法 PE 文件标志。IMAGE_FILE_HEADE 域包含了关于 PE 文件物理分布的一般信息,IMAGE_OPTIONAL_HEADER32 结构包含了关于 PE 文件逻辑分布的信息,该结构中还包括 IMAGE_DATA_DIRECTORY 结构数组,每个数组元素给出一个重要数据结构的相对虚地址和大小,如导入表、重定位表,通常共 16 个成员。

2.3 节表(section table)

紧跟着 PE 文件头的是节表(section table)。节表是一个 IMAGE_SECTION_HEADER 结构数组。此结构提供了与它相关的节的信息,其中包括偏移量、长度和其他属性。通过偏移量属性,可以定位相应节数据在文件中的位置。

2.4 节(section)

节(section)是 PE 文件真正内容的划分。每一节是是拥有共同属性的数据的集合。每节都包含了文件的内容,包括代码、数据、资源以及其它可执行信息。

3 检测模型

基于文件静态结构属性的恶意软件检测模型如图 2 所示。检测模型分为 2 个阶段:训练阶段和检测阶段。训练阶段用于完成分类器的构建;而检测阶段用于完成恶意软件的检测。

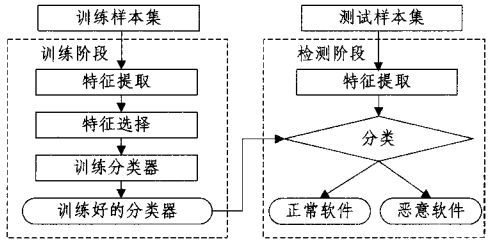

图 2 基于文件静态结构属性的恶意软件检测模型

模型从训练阶段提取样本结构属性开始,PE 文件的结构属性比较多,很多属性不能很好地区分恶意软件和正常文件。我们基于对静态结构属性的深入分析,把有可能区分的属性提取出来。大部分提取的特征并不能提高分类器的准确率,甚至可能降低分类的准确率,同时还增加了训练和检测阶段的处理开销,因此需要应用特征选择方法选择出有较好区分度的特征子集,然后使用选择出的特征子集来训练分类器,训练好的分类器能识别恶意软件和正常文件。

在检测阶段,根据训练阶段特征选择得到的特征信息,提取待检测文件的相应结构属性,使用训练阶段训练好的分类器对待检测文件分类,得到是恶意软件或是正常文件的结果。

4 实验

4.1 实验样本

实验样本分为恶意软件样本和正常文件样本,如果应用在反病毒产品中,应获取足够多的具有代表性的训练样本,考虑到只是验证方法的可行性,我们从经过杀毒软件检测无病毒的 XP 系统 windows 目录和 Program Files 目下获取了正常 PE 文件 8592 个,从 VXHeavens[8]网站下载了恶意软件 10521 个,共计 19113 个样本,恶意软件的分布如表 1 所列。

表 1 PE 格式恶意软件分布表

恶意软件类别	Backdoor	Constructor+Virtool	DoS+Nuker	Flooder	Exploit+Hacktool	Worm	Trojan	Virus	合计
数量(个)	3184	418	254	339	395	1422	3012	1497	10521

4.2 特征提取

PE 文件的静态结构属性很多，基于对恶意软件的分析和各个静态结构属性的深入认识，我们初步提取出可能与恶意软件检测相关的属性，如表 2 所列。

表 2 可能和 PE 格式恶意软件检测相关的属性

特征描述	数量(个)
信息增益最高的 DLLs	30
信息增益最高的 APIs	30
引用 DLL 的总数	1
引用 API 的总数	1
导出表中符号的总数	1
重定位节的项目总数	1
IMAGE_FILE_HEADER	5
IMAGE_OPTIONAL_HEADER	16
IMAGE_DATA_DIRECTORY	32
.text 节头	11
.data 节头	11
.rsrc 节头	11
.rdata 节头	11
.reloc 节头	11
资源目录表	22

对表 2 提取的特征简单描述如下：

①引用的 DLLs 和引用的 APIs：通过一个可执行程序引用的动态链接库(DLL)和应用程序接口(API)可以粗略地预测该程序的功能和行为。我们统计所有样本导入节中引用的 DLL 和 API 的频率，留下引用频率大于 100 次的 DLL 和 API，共剩下 46 个 DLL 和 597 个 API。然后计算每个 DLL 或 API 的信息增益，选择信息增益最高的 30 个 DLL 和 30 个 API。每个样本的导入节里若存在选择出的 DLL 或 API，以 1 表示，不存在则以 0 表示。

②PE 文件头部：PE 文件头部是定义整个 PE 文件“轮廓”的属性。我们排除了有可能误导实验结果的部分属性，如机器类型、链接器信息、操作系统信息、时间戳等(Windows 目录下的 PE 文件的这些属性都相同，可能使检测的准确率更高)，然后选择了剩下的所有字段。

③节头部：我们提取了 5 个节(.text、.data、.rsrc、.rdata 和.reloc)的节头部属性，这 5 个节在大部分 PE 文件中都存在。如果某个样本不存在相应节，该节头部的信息都以 0 表示。

④资源目录表：提取了较常见的 21 种资源类型的个数，如果没有相应类型的资源，该资源的个数以 0 值表示。此外还提取了资源节中资源个数的总和。

4.3 特征选择

通过以上方法提取特征，包含很多冗余特征，从中选取有利于区分软件类型的特征是必要的。特征选择是寻找可以准确描述原始事例的信息量最大的特征的一个过程。特征选择的目的就是在这些特征中选择最相关的一组特征，同时剩下较少的特征，以提高分类的性能。对学习算法来说，有效的特征选择可以降低学习问题的复杂性，提高学习算法的泛化性能，简化学习模型。特征选择和后续学习算法的结合方式可分为嵌入式(Embedded)、过滤式(Filter)和 Wrapper 式 3 种。

在嵌入式特征选择中，特征选择算法本身作为组成部分嵌入到学习算法里。最典型的是决策树算法，如 Quinlan 的 ID3 和 C4.5 算法等。算法在每一结点选择分类能力最强的特征，然后基于选中的特征进行子空间分割，继续此过程直到满足终止条件，可见决策树生成的过程也就是特征选择的过程。

过滤式(Filter)特征选择的评估标准独立于学习算法，直接由数据集求得。过滤式特征选择的评估依赖于数据集本身，通常是选择和目标函数相关度大的特征或者特征子集，一般认为相关度较大的特征或者特征子集会使对应后续学习算法获得较高的准确率。过滤式特征选择的评估方法很多，如类间距离、信息增益、关联度(Correlation)以及不一致度等等。但寻找和类别相关的特征子集和选择可最优化分类准确率的特征子集是两个不同的任务。

Wrapper 特征选择算法的核心思想是：和学习算法无关的过滤式特征评价会与后续的分类算法产生较大的偏差。不同的学习算法偏好不同的特征子集，既然特征选择后的特征子集最终将用于后续的学习算法，那么该学习算法的性能就是最好的评估标准。因此在 Wrapper 特征选择中将学习算法的性能作为特征选择的评估标准。由于采用学习算法的性能作为特征评估标准，Wrapper 特征选择算法比过滤式特征选择算法准确率高，但效率较低。

对于像 J48[9]、Random Forest[10] 等决策树算法，特征选择已经嵌入分类学习算法，理论上不需要使用其它特征选择算法，但考虑到检测阶段提取的特征太多，增加了单个文件的检测时间，仍然在分类学习之前进行了特征选择。我们使用了 Weka[9] 数据挖掘软件中实现的两种特征选择算法：CfsSubsetEval[9](过滤式特征选择)和 Wrapper。过滤式特征选择的结果评价简单，速度较快，Wrapper 特征选择算法对尝试的每种选择都要进行分类学习来评价选择结果，速度较慢，但只影响学习训练阶段的性能，对检测阶段没有影响。选出的特征的均值如表 3 所列。

表 3 选出特征的均值

特征名	均值	
	恶意软件	正常文件
NumberofAPIs①②	65.1	97.1
mscoree.dll①②	0.002	0.189
wsock32.dll①②	0.259	0.016
imm32.dll②	0.001	0.013
lstrlenW①	0.005	0.296
DisableThreadLibraryCalls①	0.003	0.253
__adjust_fdiv①	0.035	0.415
GetModuleHandle②	0.001	0.23
CreateFileW②	0.002	0.204
_initterm②	0.036	0.416
RegDeleteKey②	0	0.1
Numberofsections②	4.8	4.1
NumberOfSymbols②	3.1×10^{6}	15.9
Characteristics①②	1.5×10^{4}	7.1×10^{3}
BaseOfCode②	7.3×10^{4}	7.8×10^{3}
ImageBase①②	1.7×10^{7}	7.6×10^{8}
SizeOfHeapReserve①	1.5×10^{6}	1.0×10^{6}
DllCharacteristics①②	29.8	3.6×10^{3}
SizeOfCertificateTable①②	8.87	2.8×10^{3}
AddressOfDebugData①	3×10^{3}	1.4×10^{5}
SizeOfDebugData①②	0.982	22.6
SizeOfLoadConfigurationTable①	1.0×10^{5}	22.7
.text.Characteristics②	1.4×10^{9}	1.5×10^{9}
.rsrc.characteristics①	1.4×10^{9}	1.0×10^{9}
.reloc.characteristics①②	7.4×10^{8}	9.3×10^{8}
.rdata.Characteristics②	6.2×10^{8}	3.5×10^{8}
NumberOfMESSAGE TABLE①	0.002	0.057
NumberOfGROUP ICON①	0.879	1.204
NumberOfVERSION①②	0.408	0.95

注：①：过滤式(Filter)方法选出的特征；②：Wrapper(Random Forests)选出的特征

从表 3 可以看出，恶意软件引用的动态库(DLL)和 API

都明显地比正常文件的少，这主要是恶意软件为了隐藏其功能和行为，通常不在导入节引用动态库和 API，而是使用 LoadLibrary 显示地加载动态库。此外，恶意软件功能也相对简单，引用的动态库和 API 也明显较少。但是存在一个明显的特例，25.9%的恶意软件引用了 wsock32.dll，而仅仅 1.6%正常文件引用了该动态库，再者，很多恶意软件使用 LoadLibrary 加载动态库，说明大部分恶意软件通过网络进行传播和破坏。在 PE 文件头部，恶意软件有 6 个属性，与正常文件有明显的差异。恶意软件的证书表(证书表是软件厂商的可认证的声明)也远远小于正常文件的，说明恶意软件很少有证书表，而正常文件大部分都有软件厂商可认证的声明。恶意软件的调试数据也明显小于正常文件的，这是因为恶意软件为了增加调试的难度，很少有调试信息。恶意软件 4 个节(.text，.rsrc，.reloc，和.rdata)的 Characteristics 属性和正常文件的也有明显差异，Characteristics 属性通常指定该节是否可读、可写、可执行等等，部分恶意软件的代码节存在可写异常，只读数据节和资源节存在可写、可执行异常等。恶意软件资源节的资源个数也明显小于正常文件的，如消息表、组图表、版本资源等，这是因为恶意软件很少使用图形界面资源，也很少有版本信息。

4.4 分类学习

为了使恶意软件检测系统愈趋完善，将数据挖掘技术引入到恶意软件检测系统中是一个重要趋势。近年来，研究人员已经提出了一些应用数据挖掘技术来检测未知恶意软件的方法。识别一个未知程序是恶意软件还是正常文件是一般的分类问题，研究人员已经提出了大量的具备较高准确率的分类算法。我们使用了 Weka 数据挖掘软件中实现的两种决策树分类算法：J48(C4.5 第 8 版本)和 Random Forest(随机森林)[10]，同时也使用了集成学习方法(Boosting[11] 和 Bagging[12])来改进 J48 的性能。Boosting 和 Bagging 通过组合任意的弱分类器，而成为一个强分类器，从而提升任意给定分类算法的准确率。

5 实验结果与分析

5.1 评价方法和指标

为了评价检测方法，我们使用了下面 3 个经典的评价指标：准确率(Accuracy Rate)，即所有被正确分类的数量在待测集合中所占比例；检测率(TPR)，即被正确分类的恶意软件数目在待测集合的所有恶意软件中所占比例；误报率(FPR)，即被错误分类成恶意软件的正常软件在待测集合的所有正常软件中所占比例。此外，还使用了 ROC 曲线和 AUC。ROC 曲线广泛用于机器学习和数据挖掘领域，用于描述一个分类器在检测率和误报率之间的折中。ROC 分析给选择最好的模型和在类分布中独立地抛弃一些较差的模型提供了工具。AUC 是通过 ROC 曲线计算检测率的一种方法，AUC 的优势在于其不敏感于类先验分布及错分代价，且能度量算法的概率和排序输出特性。

5.2 实验结果

我们使用过滤式方法选择出的子集作为特征，使用 WEAK 数据挖掘软件的 4 种分类算法(J48，Random Forest，Bagging(J48)，AdboostM1(J48))来训练 4 种分类器，训练后的 4 种分类器检测恶意软件的准确率相差不大，都是 99%左右。使用 Wrapper 方法选择出的子集作为特征，使用随机森林(Random Forest)分类算法来训练分类器，训练后的分类器的准确率是 99.1%。实验结果见表 4，分类算法的 ROC 曲线见图 3。

表 4 分类算法的检测结果(从整个样本集选择特征)

特征选择方法	分类算法	检测率(%)	误报率(%)	准确率(%)	AUC
Filter	J48	98.9	1.4	98.7	0.994
	Random Forest	99.1	1.4	98.9	0.996
	AdboostM1(J48)	99.0	1.0	99.0	0.998
	Bagging(J48)	98.9	1.3	98.8	0.997
Wrapper	Random Forest	99.1	1.0	99.1	0.998

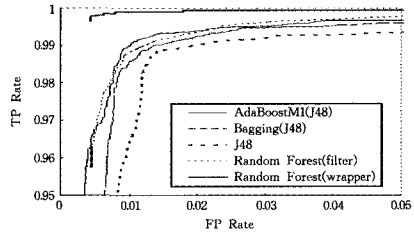

图 3 分类算法的 ROC 曲线(从整个样本集选择特征)

现有的使用数据挖掘技术进行恶意软件检测的方法都普遍存在一个问题：虽然分类实验时训练样本和测试样本没有交集，但特征选择时是从整个样本集里进行特征选择，这在一定程度上存在过度拟合数据。我们进行了另一种实验，把整个样本集随机地分成两部分，训练样本占 80%，测试样本占 20%。特征选择和训练分类器都只使用训练样本，然后使用测试样本来测试训练好的分类器。实验的结果见表 5，分类算法的 ROC 曲线见图 4。

表 5 分类算法的检测结果(从训练样本集选择特征)

特征选择方法	分类算法	检测率(%)	误报率(%)	准确率(%)	AUC
Filter	J48	98.8	1.2	98.8	0.989
	Random Forest	99.2	1.5	98.9	0.997
	AdboostM1(J48)	99.0	0.9	99.0	0.999
	Bagging(J48)	99.0	0.9	99.0	0.999
Wrapper	Random Forest	99.3	0.9	99.2	0.998

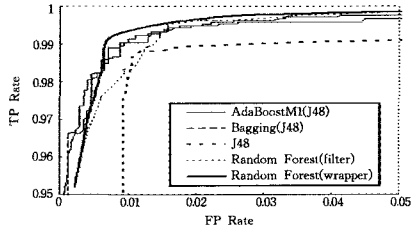

图 4 分类算法的 ROC 曲线(从训练样本集选择特征)

5.3 结果分析

从表 4 的实验结果可以看出，所使用的分类方法能识别 99%的恶意软件，误报率在 1%左右。所用算法的性能大致相同。J48 分类算法的性能相对稍差，准确率是 98.7%。使用集成学习方法(Boosting 和 Bagging)一定程度上改进了

J48 的性能。随机森林(Random Forest)的性能优于其他分类算法,准确率是 99.1%。就特征选择方法来说,Wrapper 特征选择方法优于所有过滤式特征选择方法。虽然 Wrapper 特征选择方法速度较慢,但只影响训练阶段,对检测阶段没有影响。在表 4 的评价指标中,最重要的指标是 AUC 值,很多文献都只给出了实验结果的 AUC 值,因为 AUC 与准确率实际上统计一致并且分辨率要比准确率高。本文所使用的分类算法的 AUC 值都大于 0.994,随机森林和 AdboostM1(J48)分类算法的 AUC 值是 0.998,已经非常接近最优值 1。从图 3 的 ROC 曲线可以明显看出,使用 Wrapper 特征方法的结果作为特征的随机森林(Random Forest)分类算法明显优于其他方法,J48 分类算法也明显弱于其他分类算法。

对于大部分基于数据挖掘的恶意软件检测方法,如果只从训练集里进行特征选择,会明显降低检测的准确率。从表 5 的实验结果可以看出,只使用训练集来选择特征,我们的检测方法的性能并没有受影响,只是 J48 分类算法的评价指标稍有下降,其他分类算法的评价指标还稍有提高。从图 4 可以看出,J48 和随机森林分类算法的 ROC 曲线稍有下降,其他分类算法的 ROC 曲线没有变化。综合来看,本文方法无论是从整个样本集选择特征,还是只从训练集选择特征,对实验结果影响不明显。和其它使用数据挖掘技术的静态检测方法相比,我们的检测方法具有较强的鲁棒性。

5.4 和相关静态方法对比

和相关静态方法的对比见表 6。从准确率和 AUC 值可以看出,本文方法优于其他两种方法。更重要的是,这两种方法在选择特征时训练集和测试集没有分开,一定程度上存在过度拟合数据。从所使用的特征数量来看,本文方法明显少于其他两种方法,相应的训练好的分类器也要更简单,并且提取特征的时间会更短。

表 6 与相关静态方法对比

方法名	特征数	准确率(%)	AUC
本文方法	20	99.1	0.998
字节序列 n-grams	500	—	0.996
操作码 n-grams	100	94.4	—

基于字节系列 n-grams 的检测方法经过加壳或混淆后,其字节序列完全改变了,基于操作码 n-grams 的检测方法加壳后反汇编很难成功,需要先进行脱壳处理,如果经过混淆后,其操作码系列会有很大的改变,这两种方法都不同程度地受加壳和混淆技术的影响。对于加壳或混淆处理过的恶意软件,它们仍然遵守 PE 格式的约束,仍然有其静态结构属性,我们的检测方法对这种类型的恶意软件仍然有效。

结束语 本文提出了基于 PE 文件结构信息的恶意软件检测方法,并用最新的 PE 格式恶意软件进行了实验。实验结果显示,我们的方法以 99.1%的准确率检测已知和未知的恶意软件,评价的重要指标 AUC 值是 0.998,已非常接近最优值 1,高于现有的静态检测方法,对加壳和混淆过的恶意软件也保持稳定有效。同时,和其他方法相比,我们的检测方法的处理时间和系统开销也是较少的,已达到了实时部署使用要求。此外,现有的基于数据挖掘的检测方法在特征选择时存在过度拟合数据的情况,实验显示,我们的方法在这方面具有较强的鲁棒性。

Moser 等[13]已经指出了静态检测方法的一些局限性,单独使用静态检测方法的性能指标已经很难进一步提高。恶意软件加载运行后,也有其动态结构属性,我们计划融合文件的静态结构属性和动态进程结构属性来提高我们检测方法的准确率和鲁棒性,进一步降低误判率。

参 考 文 献

[1] 马菁泽. 电脑病毒疫情及互联网安全报告综述 [J]. 网络与信息,2010(3):39-41

[2] Nachenberg C. Computer Virus-Antivirus Coevolution [J]. Communications of the ACM,1997,40(1):46-51

[3] Szor P,Ferrie P. Hunting for Metamorphic [C]// Proc of 11th International Virus Bulletin Conference. Oxfordshire, IEEE, 2001:123-143

[4] Szor P. 计算机病毒防范艺术[M]. 段海新,杨波,王德强,译. 北京:机械工业出版社,2007:308-312

[5] Weber M,Schmid M,Schatz M,et al. A toolkit for detecting and analyzing malicious software [C] // Proc of the 18th Annual Computer Security Applications Conference. Las Vegas, Nevada,IEEE,2002:423-431

[6] Kolter J, Maloof M. Learning to detect and classify malicious executables in the wild [J]. Machine Learning Research, 2006, 7 (1):123-140

[7] Dai Jian-yong. Detecting Malicious Software by Dynamic Execution [D]. Orlando, Florida: University of Central Florida, 2009

[8] http:// http://vx. netlux. org

[9] Witten I H,Frank E,Hall M A. Data mining:Practical machine learning tools and techniques [M]. USA, Morgan Kaufmann, 2011

[10] Breiman L. Random Forests [J]. Machine Learning, 2001, 45 (1):5-32

[11] Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting [J]. Journal of Computer and System Sciences,1997,55(1):119-139

[12] Breiman L. Bagging Predictors [J]. Machine Learning,1996,24 (2):123-140

[13] Moser A,Kruegel C,Kirda E. Limits of static analysis for malware detection [C]// Proc of the 23rd Annual Computer Security Applications Conference. Miami Beach, Florida, IEEE, 2007: 421-430