ISSN 1004-9037,CODEN SCYCE4
Journal of Data Acquisition and Processing Vol. 32,No. 3,May 2017,pp. 612－620
DOI:10.16337/j.1004-9037.2017.03.021

http://sjcj.nuaa.edu.cn
E-mail:sjcj@nuaa.edu.cn
Tel/Fax:＋86-025-84892742

软件代码的恶意行为学习与分类

范宇杰[1,2]　陈黎飞[1,2]　郭躬德[1,2]

(1.福建师范大学数学与计算机科学学院,福州,350007; 2.福建师范大学网络安全与密码技术福建省高校重点实验室,福州,350007)

摘　要:传统的静态特征码检测方法无法识别迷惑型恶意代码,而动态检测方法则需要消耗大量资源;当前,大多数基于机器学习的方法并不能有效区分木马、蠕虫等恶意软件的子类别。为此,提出一种基于代码恶意行为特征的分类方法。新方法在提取代码恶意导向指令特征的基础上,学习每种代码类别特有的恶意行为序列模式,进而将代码样本投影到由恶意行为序列模式构成的新空间中。同时基于新特征表示法构造了一种近邻分类器对恶意代码进行分类。实验结果表明,新方法可以有效地捕捉代码的恶意行为并区分不同类别代码之间的行为差异,从而大幅提高了恶意代码的分类精度。

关键词:恶意代码分类;序列模式;恶意行为;特征提取;分类方法

中图分类号:TP391　　**文献标志码**:A

Learning and Classification of Malicious Behaviors in Software Code

Fan Yujie[1,2], Chen Lifei[1,2], Guo Gongde[1,2]

(1. School of Mathematics and Computer Science, Fujian Normal University, Fuzhou, 350007, China; 2. Key Laboratory of Network Security and Cryptography, Fujian Normal University, Fuzhou, 350007, China)

Abstract: Traditional signature-based method fails to identify the obfuscated malicious codes, while the dynamic method consumes a large amount of resources. Currently, most machine-learning-based detection methods cannot effectively distinguish trojan horses, worms and other malwares. Hence, we propose a new classification method based on malicious behavior features. The new method first learns specific malicious behavior sequential pattern of each malware category on the basis of the extraction of malicious-oriented instruction. The sample is projected to the new space which is composed of sequential patterns. Based on the new feature representation, a nearest neighbor classifier is constructed to classify the malicious codes. Experimental results show that the proposed method can effectively capture the malicious behavior and distinguish the differences among the behaviors of different malware categories, so as to improve the classification precision sharply.

Key words: malware classification; sequential pattern; malicious behavior; feature extraction; classification method

---

基金项目:国家自然科学基金(61175123)资助项目。
收稿日期:2015-05-18;**修订日期**:2015-06-19

引 言

恶意代码分类是恶意代码分析和入侵检测领域中的核心问题[1]。随着互联网技术的不断发展,各种类型的恶意代码正不断危害着计算机网络以及系统的安全,正确地对这些恶意代码进行分类有利于提高分析效率,从而缩短应急响应时间[2]。根据恶意代码检测时代码状态的区别,可以分为静态检测和动态检测[3]。静态检测[4,5]在不执行代码的情况下,通过静态扫描文件内容来判断其类别。其中,最具代表性的是基于特征码的检测方法。这类方法简单有效,却存在诸多缺陷,其中恶意代码混淆技术和加壳技术给静态分析带来巨大障碍[6,7]。动态检测[8]则通过在虚拟机或仿真器中执行代码来分析其行为,从而进行检测并分类。相比于静态检测,此类方法开销较大且占用较多系统资源[3]。近年来,研究者开始利用数据挖掘和机器学习技术来检测恶意代码[9-11]。这些方法首先提取 API 调用、字符串信息等特征,进而利用决策树、朴素贝叶斯等分类算法实现代码的自动检测。如:Schultz 等[9]提取系统信息、字符串信息及字符序列特征作为分类器 Ripper, Naive Bayes 和 Multi-Naive Bayes 的输入进行恶意代码检测,实验结果表明基于静态特征码的检测方法效果最差而基于机器学习的方法都具有较高的准确性。然而,上述方法仅局限于判断样本是否属于恶意代码却不能有效预测恶意代码的类别。为此,本文提出一种基于恶意行为特征的恶意代码分类方法。该方法主要分为恶意行为特征提取(Malicious behavior feature extraction, MBFE)和分类两个过程。MBFE 过程以机器指令作为静态特征,首先提取具有倾向性的恶意导向指令,在此基础上通过改进的序列挖掘算法依次在全局层和类别层学习每种恶意代码类别特有的恶意行为序列模式。分类过程则通过将样本投影到由这些序列模式组成的新空间中构造了一种新的近邻分类器,从而实现恶意代码的自动分类。新方法的主要特点有:(1)使用序列挖掘方法提取每种类别特有的恶意行为序列模式,这些模式刻画了每种恶意代码类别特有的恶意行为;(2)在静态检测的基础上结合了动态检测捕捉代码行为的优势,不仅能提高识别率而且有效减小了系统开销。

现有的大多数检测恶意代码的工作都更注重提取字符串、代码块等特征,如:Tian 等[12]利用字符串信息作为特征向量输入分类器,实现木马和病毒类别的家族分类。然而,与这些静态特征相比,指令特征更易于提取且其表征样本行为的能力更强[13],而程序代码(可执行文件)的指令特征可由第三方反汇编工具 C32Asm[14]对其反汇编得到。图 1 表示木马样本(Trojan-Downloader. Win32. Hmir. abm)的反汇编及解析过程。中间部分为反汇编后输出的部分机器指令集合,如第 1 条机器指令由操作码 SUB, CPU 寄存器 ESP 以及 16 进制操作数 120 组成。而右边部分表示解析过程,即剔除机器指令中的操作数而保留操作码并将其编码为一个整数(称为指令 ID)。图 1 中间部分的反汇编结果被解析为 369→197→309→222→…这样的序列,其中 369,197 和 309 分别表示操作码 SUB, LEA 和 PUSH。Han 等[15]根据上述指令特征出现的频率来计算样本之间的相似度从而实现恶意代码分类。然而,Han 等并没有考虑指令之间的相关性,仅通过频率单独挑选出的无序特征并不足以表达样本的行为。Ahmadi 等[16]则利用迭代模式挖掘[17]方法提取频繁迭代模式作为分类特征,实验结果表明利用序列挖掘方法提取的特征具有良好的表达代码行为的能力,在提高检测率的同时降低了误报率。然而该方法是建立在动态检测的基础上,需要在虚拟环境中收集 API 序列,效率低且难以实现,更重要的是其并不能判断恶意代码的类别。本文使用的分类特征是在静态指令特征的基础上,利用序列挖掘算法提取恶意行为特征。每一个特征刻画了一个能区分不同恶意代码类别的恶意行为。新特征充分考虑了指令之间的顺序关系,具有良好的表达能力。

1 代码恶意行为特征提取

代码恶意行为提取过程主要分为恶意导向指令提取、全局层及类别层恶意行为序列模式提取 3 个步骤。本文使用正常代码样本及 4 个类别的恶意代码样本作为实验数据,为方便表述,用 N 表示正常

图 1 Trojan-Downloader. Win32. Hmir. abm 的反汇编及解析过程

Fig. 1 Process of disassembling and parsing of Trojan-Downloader. Win32. Hmir. abm

代码样本(Normal),B 表示后门样本(Backdoor),T 表示木马样本(Trojan Horse),V 表示病毒样本(Virus),W 表示蠕虫样本(Worm),并用 A 表示 4 种恶意代码的集合,即 $A=\{B,T,V,W\}$,另外取 $M\in A$ 。

1.1 恶意导向指令提取

恶意代码与正常代码在指令的使用上往往存在明显的差别,主要表现在有些指令更倾向于被恶意代码使用,将这样的指令称为恶意导向指令。受 Zhang 等[18]的启发,使用定义 1 来衡量一条指令倾向恶意代码的程度。

定义 1 用 i 表示指令 ID,其倾向度定义为

$$t(i)=\begin{cases}\dfrac{f_M(i)}{f_M(i)+f_N(i)} & f_M(i)\neq 0\\ 0 & f_M(i)=0\end{cases} \tag{1}$$

式中:$f_M(i)$ 和 $f_N(i)$ 分别为指令 i 在 M 类恶意代码样本以及正常代码样本中的加权频率。定义 1 表明,若某个类别中第 i 条指令的倾向度越大,则它相对于该类别的恶意导向性则越强,本小节的目标就是挑选出恶意导向性强的指令。利用文献[18]的方法计算各个恶意代码类别中每条指令的倾向度,并根据公式 $t(i)>t$ (t 为用户给定的阈值)挑选出属于每个类别的恶意导向指令集合。为方便理解,用木马类别作为例子阐述下面步骤。按上述方法为木马类别挑选恶意导向指令并加入指令库 I_T 。随后,每个木马样本被表示为一条由 I_T 库中指令组成的序列,所有这样的序列组成木马类恶意导向序列库 S_T 。对后门、病毒和蠕虫类别做同样处理,分别得到指令库 I_B , I_V 和 I_W 及序列库 S_B , S_V 和 S_W 。由于这些恶意导向指令序列剔除了大量无效信息,仅保留一小部分带有倾向性的恶意指令,因此,在此基础上进行序列模式挖掘效率更高且更具说服力。

1.2 恶意行为序列模式挖掘

恶意代码的恶意行为表现为代码内具有关联性的多个关键指令的共同作用,可以理解为是某些恶意导向指令的有序排列。这种排列频繁存在于某一类恶意代码中,却极少出现在正常代码及其他类别恶意代码中。本文所需提取的正是这些能够体现区别的有序排列,因此本小节主要在全局层和类别层这两个层次上进行序列模式挖掘。

1.2.1 全局层恶意行为序列模式

广义序贯模式(Generalized sequential pattern,GSP)算法[19]是目前普遍使用的序列模式挖掘算法,然而,算法效率低下是其主要弊端之一。为此,本文在 GSP 的基础上加入“面向目标[20]”的概念,使算法倾向于发现带有恶意性质的序列模式;同时使用置信度来过滤所产生的序列模式,从而使搜索空间锐减。在进行序列模式挖掘前,首先应明确“面向目标”中的一些定义。

定义 2 已知两个序列 $\alpha=a_1,a_2,\cdots,a_n$ 和 $\beta=b_1,b_2,\cdots,b_m$,如果存在整数 $1\leqslant j_1<j_2<\cdots<j_n\leqslant m$ 使得 $a_1=b_{j_1},a_2=b_{j_2},\cdots,a_n=b_{j_n}$,则称序列 α 为序列的子序列,表示为 $\alpha\subseteq\beta$ 。

定义 3 设 α 为 S_M 中指令序列 β 的子序列,则 α 的支持度和全局层置信度定义为

$$s_\alpha=\frac{|\{\beta|(\beta\in S_M)\wedge(\alpha\subseteq\beta)\}|}{|S_M|}\times 100\% \tag{2}$$

$$g_\alpha=\frac{|\{\beta|(\beta\in S_M)\wedge(\alpha\subseteq\beta)\}|}{\sum_{t\in\{M,N\}}|\{\beta|(\beta\in S_t)\wedge(\alpha\subseteq\beta)\}|}\times 100\% \tag{3}$$

式中:$|S_M|$ 和 $|S_N|$ 分别为 M 类恶意代码和正常代码的指令序列个数(1.1 节已经将每个样本表示为由恶意导向指令组成的指令序列)。

定义 4 给定最小支持度 s,若子序列 α 满足 $s_\alpha\geqslant s$,则称 α 为序列模式。

定义 5 给定全局层最小置信度 c,若序列模式 α 满足 $g_\alpha\geqslant c$,则称其为全局层恶意序列模式。

支持度和全局层置信度分别表示一条序列在恶意代码中存在的普遍性以及它在正常代码和恶意代码中的差异性,因此,具有高支持度、高置信度的序列正是需要挖掘的能区分两类代码的序列模式,即定义 5 中的全局层恶意序列模式。基于如上定义,序列模式算法的基本步骤如下:

(1) 扫描 S_M,根据式(2,3)计算每个项的支持度和全局层置信度,并根据定义 4 得到长度为 1 的序列模式 L_1,作为初始种子集。

(2) $k=2$。

(3) 根据长度为 $k-1$ 的种子集 L_{k-1},通过自连接和修剪操作生成长度为 k 的候选序列集 C_k。

(4) 扫描 C_k 并计算每个候选序列模式 C_c 的支持度和全局层置信度,根据定义 4 和式(4)得到长度为 k 的序列模式 L_k,并将 L_k 作为新的种子集。在式(4)中,C_c' 表示 $C_c\in C_k$ 的所有长度为 $k-1$ 的子序列,则

$$g_{C_c}\geqslant g_{C_c'} \tag{4}$$

(5) $k=k+1$。

(6) 重复第(3)步,第(4)步以及第(5)步,直到没有新的序列模式或新的候选序列模式产生为止。

(7) 根据定义 5 从所有序列模式中挑选出恶意序列模式加入全局层恶意序列模式库 GP^M。

式(4)表示长度为 k 的序列模式的置信度必须大于或等于它长度为 $k-1$ 的子序列的置信度,这是因为如果序列的长度越长,则其鉴别两类代码的能力就越强,也就是每次迭代过程产生的序列模式对恶意代码的预测应起到促进作用。算法第(4)步使用这样的裁剪策略,大幅减小了搜索空间。另外,算法第(7)步利用定义 5 来提取能体现恶意行为的全局层恶意序列模式,这正是"面向目标"的思想体现。实验中,用 S_B,S_T,S_V 和 S_W 分别与正常代码样本作为训练集,使用上述算法提取各自类别的全局层恶意序列模式并加入库 GP^B,GP^T,GP^V 和 GP^W。

1.2.2 类别层恶意行为序列模式

全局层恶意行为序列模式虽然较好地描述了恶意代码与正常代码在行为上的差别。然而,它还不足以体现不同类别恶意代码间的行为差异。因此,需要在类别层上进一步挖掘能体现类别差异的恶意行为序列模式,将这样的模式称为类别层恶意行为序列模式。

定义 6 已知 α 为 GP^M 中的全局层恶意序列模式,则 α 的类别层置信度定义为

$$c_\alpha=\frac{|\{\beta|(\beta\in S_M)\wedge(\alpha\subseteq\beta)\}|}{\sum_{t\in A}|\{\beta|(\beta\in S_t)\wedge(\alpha\subseteq\beta)\}|}\times 100 \tag{5}$$

式中:$A=\{B,T,V,W\}$,$M\in A$。与定义 3 中全局层置信度类似,类别层置信度表示一条序列在某个恶意代码类别及其余类别中的差异性。根据定义 6 分别计算库 GP^B,GP^T,GP^V 和 GP^W 中每个模式的类别层置信度。最后,为每个类别挑选类别层置信度最高的 10 个全局层恶意序列模式加入库 CP^B,CP^T,CP^V 以及 CP^W。最终的分类特征即为类别层恶意序列模式库中的模式。与 GP^M 相比,CP^M 不仅数量更少,而且具备区分不同类别恶意代码行为的能力。

2 恶意代码分类方法

2.1 恶意代码的向量空间表示模型

第 1 节已经通过序列挖掘方法为每个恶意代码类别提取出 10 个体现类别差异的恶意行为序列模式，共 40 个。现在考虑如何用这些特征来表示样本。本文使用的方法称为样本向量化，即将样本投影到一个新的特征空间，在新空间中，每个特征对应一个类别层恶意行为序列模式，而样本则被转化为一个布尔向量，向量中每个元素表示样本是否含有对应的序列模式。形式地，样本 x 被表示成 1×40 维空间中的一个向量 $\boldsymbol{V}[x]$：$\boldsymbol{V}[x]=[x_1,x_2,\cdots,x_{40}]$，其中 $x_j\in\{0,1\}$，$j=1,2,\cdots,40$。显然，使用这样的表示方法，可以很容易地描述每个样本的恶意行为并测量不同样本间的相似性。

2.2 分类算法

通过样本向量化，每个样本都被表示为新空间中的一个布尔向量，这样的表示法适用于大多数分类算法，其中基于距离的分类方法因简单有效而被广泛使用。K 最近邻(K-nearest neighbor，KNN)[21]算法是目前普遍使用的基于距离的分类方法，然而，其分类精度受 k 值的影响较大，而 k 又是一个用户设定的参数。为克服传统 KNN 算法中参数 k 难以确定的问题，本文提出一种自动确定 k 的策略，具体的步骤如下：

(1) 利用 $\mathrm{dist}(y,t)=\|\boldsymbol{V}[y]-\boldsymbol{V}[t]\|_2$ 计算待测样本 y 与每个训练样本 t 之间的欧几里得距离。

(2) 根据 $t_s=\mathrm{argmin}_t\mathrm{dist}(y,t)$ 选取与 y 距离最小的所有训练样本并组成近邻集合 t_s(这里需注意，与 y 距离最小的样本可能不止一个)。

(3) 根据 t_s 中的类标号按多数投票原则确定 y 的类别。

上述算法与传统 KNN 的最大区别在于算法的第(2)步，即：使用与待测样本具有最小距离的样本数作为 k；通常，这样的样本数大于 1；若仅存在 1 个这样的训练样本，此时将退化为 KNN($k=1$)。因此，本文将这种策略称为 Auto-k。为便于理解，下面给出一个真实例子来验证该策略的有效性。考虑木马样本 Trojan-Downloader. Win32. Hmir. cbb 作为待测样本，如果使用 KNN 分类器，则不同的 k 值将产生不同的分类结果，若 $k=3$，其被错分为正常代码类，而 $k=9$，则正确分类。然而，若采用结合 Auto-k 策略的近邻分类器，则算法第(2)步将会得到 861 个训练样本，这些样本与待测样本的距离最小，其中 831 个样本属于恶意代码类而其余 30 个属于正常样本类，此时 861 就被自动设置为 k 的值。最终，根据第(3)步投票原则，待测样本被正确分为恶意代码类。

3 恶意代码检测与分析

实验在 CPU 2.2 GHz，RAM 4 GB 的计算机上进行，操作系统为 Microsoft Windows XP。通过分析恶意行为序列模式在结合 Auto-k 策略的近邻分类器上的二分类及多类别分类结果来评估新分类方法的有效性。

3.1 实验数据

根据恶意代码传输机制的不同，大体可以将其分为以下 4 个大类[22]：后门、木马、病毒及蠕虫。为此，从 http://vxheaven.org/ 获取了 4 451 个后门样本，2 605 个木马样本，2 602 个病毒样本以及 3 708 个蠕虫样本同时从一台新安装 Window XP 的计算机上获取 1 460 个正常代码作为实验数据。每次实验从中随机抽取 4 个类别的恶意代码样本 1 200 个，正常样本 800 个作为数据集。

3.2 参数选择

由于 t,s,c 这 3 个参数具有一定的耦合性，难以根据最后的检测结果调整参数。本文通过分析每个

参数的直接影响对象来确定最佳参数选择。不同的倾向度阈值 t 对应不同的恶意导向指令,从而产生不同数量的恶意导向指令序列。由于筛选出的恶意导向指令具有强鉴别力,所以在保证所有的恶意样本都能被转化为恶意导向指令序列的情况下,最佳的 t 应该尽可能让正常样本保留较少的指令。因此,引入 cov(M)和 cov(N)来表示恶意代码样本和正常代码样本被转化为恶意导向指令序列的转化率,则

$$\mathrm{cov}(M)=\frac{|S_M|}{|N_M|} \tag{6}$$

$$\mathrm{cov}(N)=\frac{|S_N|}{|N_N|} \tag{7}$$

式中:$|N_M|$ 和 $|N_N|$ 分别为 M 类恶意代码和正常代码的样本总数。以木马类别为例,如表 1 所示,当 $t=0.90$ 时,所有木马样本都可以被转化为由 I_T 库中指令组成的序列,而正常代码仅有 690 个样本可以被表示为这样的序列(其他 110 个样本被转化为空序列)。利用同样的方法就可以确定其他 3 类恶意代码参数 t 的值。对于 s 和 c,高支持度、高置信度的全局层恶意行为序列模式意味着它广泛存在于恶意代码中,却极少出现在正常代码中。因此,若 s 和 c 的值越大,则恶意序列模式的鉴别力就越强。表 2 表示木马类别在不同 s 和 c 情况下所产生的全局层恶意行为序列模式数目。从表中可以看出,恶意行为序列模式随着 s 和 c 的减小而急剧增加。根据平衡特征冗余原则,当 $s=97\%$,$c=96\%$ 时,所提取的4 397 个特征数目适中。同样地,可以确定其他 3 个类别 s 和 c 的值。

表 1 木马类别在不同 t 值下恶意导向指令序列的转化率

Tab. 1 Conversion rate of instruction sequence of Trojan Horse with different t

t	cov(T)/%	cov(N)/%
0.80	100	94.50
0.85	100	87.50
0.90	100	86.25
0.95	98.33	74.00

表 2 木马类别在不同 s 和 c 下恶意序列模式数目

Tab. 2 Number of malicious sequential patterns of trojan horse with different s and c

s/%	c/%	序列模式个数
98	97	0
	96	46
97	97	0
	96	4 397
96	97	0
	96	70 818

3.3 实验结果与分析

实验采用十折交叉验证法,即将原数据集随机等分为 10 份,每一折都使用其中的 9 份作为训练集,剩余的 1 份作为测试集,10 折的平均结果作为最终的结果。另外,使用如下几个指标对结果进行评价:

(1) 检测率(Detection rate, DR):$\mathrm{DR}=\dfrac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$

(2) 误判率(False positive rate, FPR):$\mathrm{FPR}=\dfrac{\mathrm{FP}}{\mathrm{FP}+\mathrm{TN}}$

(3) 准确率(Accuracy, ACC):$\mathrm{ACC}=\dfrac{\mathrm{TP}+\mathrm{TN}}{\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}}$

式中:TP 为被正确分类的正样本数目,TN 为被正确分类的负样本数目,FP 为被错误分类为正样本的负样本数目,FN 为被错误分类为负样本的正样本数目。

3.3.1 二分类实验

实验目的主要有:(1)验证恶意行为序列模式是否具备鉴别恶意代码和正常代码的能力;(2)评估 Auto-k 策略的有效性。此时,相当于一个二分类问题(4 个类别恶意代码看成一个恶意代码大类),分类特征为类别层恶意序列模式库中的模式(40 个)。如图 2 所示,这些恶意序列模式频繁存在于恶意代码样本中,却仅在少数正常代码样本中出现,如编号为 1 的模式在 1 118 个恶意代码样本中出现,而仅有

35 个正常代码样本包含该模式。图 2 从侧面反映了这些模式具备区分两类样本的能力。利用分类实验可以进一步验证它们的鉴别力。另外，为了评估 Auto-k 策略的性能，选取了不同 k 值情况下的 KNN (k=3,5,7,9)算法进行对比实验。表 3 表示恶意行为序列模式在不同近邻分类器上的二分类结果。可以看出其在结合 Auto-k 策略的分类器中的检测率和精确度都高于 KNN 且都保持在 92%以上。图 2 和表 3 的结果说明恶意行为序列模式较好地刻画了恶意代码的恶意行为，而这些恶意行为正是需要挖掘的在两个类别中具有明显差异的特征。与此同时，实验还验证了 Auto-k 策略的有效性，并再次体现其无需考虑 k 值的优点。

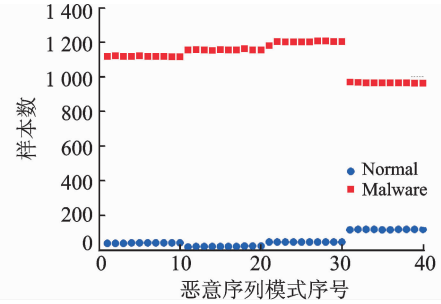

图 2 恶意代码及正常代码中包含恶意序列模式的样本数

Fig. 2 Number of malware and normal samples containing malicious sequential pattern

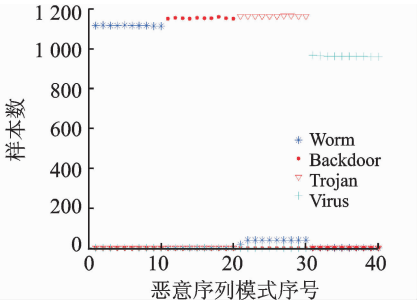

图 3 4 类恶意代码中包含恶意序列模式的样本数

Fig. 3 Number of malware samples of four categories containing malicious sequential pattern

表 3 恶意行为序列模式在不同近邻分类器上的二分类结果

Tab. 3 Binary classification result in different nearest neighbor classifiers of malicious sequential pattern

分类器	k	TP	TN	FP	FN	DR/%	ACC/%
KNN	3	4 455	720	80	345	92.81	92.41
	5	4 451	719	81	349	92.73	92.32
	7	4 444	719	81	356	92.58	92.20
	9	4 442	714	86	358	92.54	92.07
Auto-k		4 462	729	71	338	92.96	92.70

3.3.2 多类别分类实验

本小节评估上述恶意行为序列模式是否具有区分 4 类恶意代码的能力。如图 3 所示，这 40 个恶意序列模式都带有明显的类别倾向性，例如：编号为 1～10 的 10 个模式在 1 200 个蠕虫样本中的平均覆盖率达到 93.13%，而在其他类别的 3 600 个样本中的平均覆盖率仅为 1%，即这 10 个模式更倾向于被蠕虫代码使用。下面通过多分类实验来验证恶意序列模式的类别鉴别力，同时为了更好地体现恶意行为序列模式的优势，选取了 3 种常用的特征选择方法：信息增益(Information gain, IG)，最大相关(Max-relevance, MR) 以及卡方测试(Chi-square)[23]进行对比实验。分别使用这 3 个方法对所有指令排序，选取前 100 个指令作为各自方法最终的分类特征，恶意行为序列模式则依然使用上述经过 MBFE 方法得到的 40 个特征，同样在 Auto-k 策略的近邻分类器上进行实验。实验结果如表 4 和图 4 所示，表 4 中数值表示 4 个类别恶意代码检测率和误判率的平均值。从表 4 可知，恶意行为序列模式在结合 Auto-k 策略的近邻分类器上的检测率高达 97%且误判率不足 1%，显然，其分类效果要远优于其他 3 个特征选择方法。图 4 表示不同特征在 Auto-k 策略的近邻分类器上正确分类及错误分类的样本数。从中可以看

出,相比于指令特征,恶意行为序列模式具有更强的区分 4 类恶意代码的能力,这是因为 MBFE 方法同时考虑了指令的频率以及指令之间的相关性。此外,MBFE 仅使用了 40 个(每个类别 10 个)特征,也小于其他方法的 100 个指令特征。为了更进一步验证恶意序列模式的有效程度,图 5 使用混淆矩阵来表示恶意序列模式在 Auto-k 策略的近邻分类器中每个类别的具体分类结果。如图所示,4 个恶意代码类别的检测率都高达 95%,仅有少数的恶意代码被分到错误的类别。上述实验结果表明,MBFE 善于发现不同类别恶意代码行为上的差异,其挖掘出的恶意序列模式具有较强的类别鉴别力。

表 4 不同特征在 Auto-k 策略的近邻分类器上的多分类结果

Tab. 4 Multi-classification result in auto-k nearest neighbor classifier of different features

特征	方法	DR/%	FPR/%
指令特征	IG	0.788	0.071
	MR	0.795	0.068
	Chi-square	0.792	0.069
恶意序列模式	MBFE	0.975	0.008

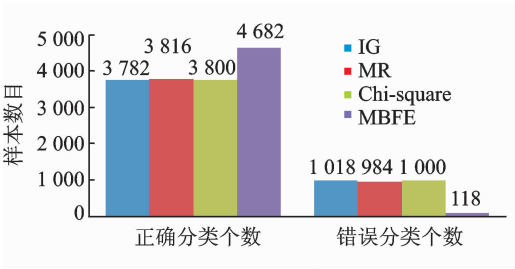

图 4 不同特征在 Auto-k 策略的近邻分类器上正确及错误分类的样本数

Fig. 4 Number of correct and error classification samples in auto-k nearest neighbor classifier of different features

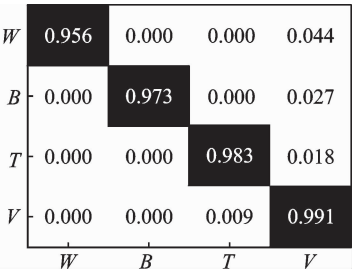

图 5 恶意序列模式在 Auto-k 策略的近邻分类器下的混淆矩阵

Fig. 5 Confusion matrix in auto-k nearest neighbor classifier of malicious sequential pattern

4 结束语

为了应对各类型恶意代码不断危害计算机网络及系统的问题,本文提出了一种新的恶意代码分类方法,即基于恶意行为特征的恶意代码分类方法。新方法首先为每个恶意代码类别提取倾向该类别的恶意导向指令,在此基础上利用序列挖掘技术分别在全局层和类别层学习每个类别特有的恶意行为序列模式并作为分类特征。每个模式刻画了该类别恶意代码的一个潜在恶意行为,进而将恶意代码投影到由这些分类特征构成的空间中。最后利用新的基于距离的分类算法进行分类。新方法同时考虑了静态检测快速的优点以及动态检测有效的优势。与现有方法相比,不仅大量减少了由无效特征引起的冗余信息,而且充分利用了指令之间的顺序关系,在有效提高类别预测精度的同时减小了误判率。

参考文献:

[1] 朱克楠,尹宝林, 冒亚明,等. 基于有效窗口和朴素贝叶斯的恶意代码分类[J]. 计算机研究与发展,2014,51(2):373-381.
Zhu Kenan, Yin Baolin, Mao Yaming, et al. Malware classification approach based on valid window and naive Bayes[J]. Journal of Computer Research and Development, 2014, 51(2): 373-381.

[2] 金然,魏强,王清贤. 基于分支序列距离的恶意代码分类[J]. 计算机研究与发展,2007,44(2):52-57.
Jin Ran, Wei Qiang, Wang Qingxian. Malware classification based on distance between branch sequences[J]. Journal of Computer Research and Development, 2007, 44(2): 52-57.

[3] 孔德光,谭小彬,奚宏生,等. 提升多维特征检测迷惑恶意代码[J]. 软件学报,2011,22(3):522-533.
Kong Deguang, Tan Xiaobin, Xi Hongsheng, et al. Obfuscated malware detection based on boosting multilevel features[J]. Chinese Journal of Software, 2011, 22(3): 522-533.

[4] Christodorescu M, Jha S. Static analysis of executables to detect malicious patterns[C]// 12th Conference on USENIX Security Symposium. Berkeley, CA: USENIX Association, 2003: 169-186.

[5] Linn C, Debray S. Obfuscation of executable code to improve resistance to static disassembly[C]// 10th ACM Conference on Computer and Communications Security. New York: ACM, 2003: 290-299.

[6] Szor P. The art of computer virus research and defense[M]. Boston: Addison-Wesley Professional, 2005: 59-98.

[7] Popov I V, Debray S K, Andrews G R. Binary obfuscation using signals[C] // 16th Conference on USENIX Security Symposium. Berkeley, CA: USENIX Association, 2007: 275-290.

[8] Egele M, Scholte T, Kirda E, et al. A survey on automated dynamic malware-analysis techniques and tools[J]. ACM Computing Surveys (CSUR), 2012, 44(2): 6.

[9] Schultz M G, Eskin E, Zadok E, et al. Data mining methods for detection of new malicious executables[C] // 2001 IEEE Symposium on Security and Privacy. Los Alamitos, CA: IEEE Computer Society Press, 2001, 38-49.

[10] Kolter J Z, Maloof M A. Learning to detect malicious executables in the wild[C] // 10th International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2004: 470-478.

[11] Wang J H, Deng P S, Fan Y S, et al. Virus detection using data mining techniques[C] // 37th Annual International Carnahan Conference on Security Technology. Piscataway, NJ: IEEE, 2003: 71-76.

[12] Tian R, Batten L, Islam M R, et al. An automated classification system based on the strings of Trojan and virus Families[C]// 4th International Conference on Malicious and Unwanted Software (MALWARE). Piscataway, NJ: IEEE, 2009: 23-30.

[13] Ye Y, Li T, Chen Y, et al. Automatic malware categorization using cluster ensemble[C]// 16th International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2010: 95-104.

[14] PLL621. C32Asm, Version 1. 0. 9. 0[EB/OL]. https://tuts4you. com/download. phpview. 1130, 2011-02-17/2015-02-27.

[15] Han K S, Kang B, Im E G. Malware classification using instruction frequencies[C] // 2011 ACM Symposium on Research in Applied Computation. New York: ACM, 2011: 298-300.

[16] Ahmadi M, Sami A, Rahimi H, et al. Malware detection by behavioural sequential patterns[J]. Computer Fraud & Security, 2013, 2013(8): 11-19.

[17] Lo D, Cheng H, Han J, et al. Classification of software behaviors for failure detection: A discriminative pattern mining approach[C] // 15th International Conference on Knowledge Discovery and Data Mining. New York: ACM, 2009: 557-566.

[18] 张健飞,陈黎飞,郭躬德. 检测迷惑恶意代码的层次化特征选择方法[J]. 计算机应用,2012,32(10):2761-2767.
Zhang Jianfei, Chen Lifei, Guo Gongde. Hierarchical feature selection method for detection of obfuscated malicious code[J]. Journal of Computer Applications, 2012, 32(10): 2761-2767.

[19] Srikant R, Agrawal R. Mining sequential patterns: Generalizations and performance improvements[C]// International Conference on Extending Database Technology. Berlin: Springer, 1996: 1-17.

[20] Shen Y D, Zhang Z, Yang Q. Objective-oriented utility-based association mining[C] // International Conference on Data Mining'02. Los Alamitos, CA: IEEE Computer Society Press, 2002: 426-433.

[21] Han J, Kamber M, Pei J. Data mining: Concepts and techniques[M]. San Francisco: Morgan Kaufmann, 2006: 422-426.

[22] Nissim N, Moskovitch R, Rokach L, et al. Novel active learning methods for enhanced PC malware detection in windows OS [J]. Expert Systems with Applications, 2014, 41(13): 5843-5857.

[23] Yang Y, Pedersen J O. A comparative study on feature selection in text categorization[C] // 14th International Conference on Machine Learning. New York: ACM, 1997: 412-420.

作者简介:

范宇杰(1991-),男,硕士研究生,研究方向:数据挖掘,E-mail:kobefyj@126. com。

陈黎飞(1972-),男,教授,研究方向:数据挖掘、机器学习。

郭躬德(1965-),男,教授,研究方向:人工智能、数据挖掘。