doi:10.11835/j.issn.1000-582X.2023.03.012

基于动态行为特征加权聚类的加壳恶意软件未知变种检测方法

陈 岑，李暖暖，蔡军飞，郭志民，吕 卓
（国网河南省电力公司 电力科学研究院，郑州 450000）

摘要：攻击者为了逃避检测，常利用加壳技术对恶意软件进行加密或压缩，使得安全分析人员以及传统基于静态分析的恶意软件检测方法在恶意软件运行前难以利用反汇编等逆向工具对其进行静态分析。为检测加壳恶意软件，当前主要采用动态分析方法检测加壳恶意软件，然而受限于加壳工具种类和样本规模，以及恶意软件加壳行为带来的混淆噪声，导致传统基于机器学习检测方法存在准确率不足等问题。研究提取并分析加壳恶意软件运行时的系统调用行为特征，识别并筛选出敏感行为，旨在过滤脱壳行为噪声产生的影响；通过对系统调用行为特征加权降维，提升行为特征的有效性；通过对加权降维的行为特征进行聚类分析，最终实现加壳恶意软件未知变种检测和检测模型增量更新。实验结果表明，提出的基于动态行为特征加权聚类的加壳恶意软件未知变种检测方法检测误报率 3.9%，相较几种典型机器学习检测方法呈显著降低。

关键词：恶意软件变种检测；动态行为分析；主成分分析；密度聚类
中图分类号：TP391 **文献标志码**：A **文章编号**：1000-582X(2023)03-129-008

A packed malware variants detection method based on weighted dynamic behaviour feature clustering

CHEN Cen, *LI Nuannuan*, *CAI Junfei*, *GUO Zhimin*, *LYU Zhuo*
(State Grid Henan Electric Power Research Institute, Zhengzhou 450000, P. R. China)

Abstract: In order to avoid malware detection, attackers often use packing techniques to encrypt or compress malware binaries, which makes it difficult for security analysts and malware detectors based on traditional static analysis to use reverse tools, such as disassembly tools, to statically analyze malware before it runs. Currently, to detect packed malware, dynamic analysis methods are mainly used. However, due to the limitation of the types of packing tools and packed samples, as well as the confusion noise caused by malware packers, traditional machine learning based detection methods have insufficient accuracy. In this paper, to filter the packing behavior, the system call behavior features of packed malware are extracted and analyzed, and then sensitive behaviors are identified and filtered out. Next, the feature dimensions of system call behaviours are reduced by weighting to improve the contribution of each feature. Finally, these behaviours are analyzed by using density-based clustering, realizing the detection of unknown variants of packed malware and the update of the detection model. The experimental results show that the proposed

收稿日期：2022-05-12
基金项目：国家电网有限公司科技资助项目(5700-202124182A-0-0-00)。
Supported by the Science and Technology Project of State Grid Corporation of China (5700-202124182A-0-0-00)
作者简介：陈岑(1990—)，女，工程师，主要从事电子科学与技术，电力信息安全方向研究，(E-mail)1020065011@qq.com。

packed malware variants detection method based on weighted clustering of sensitive behavior features achieves 3.9 % false alter rate and significantly reduces the false alter rate compared with that of some other machine learning-based detection methods.

Keywords: malware variants detection; dynamic behaviour analysis; principal component analysis; density-based clustering

恶意软件(变种)是当今最主要的网络安全威胁之一,中国2021互联网应急响应中心报告表明[1]:境内全年捕获恶意软件数量超1×10^9个,VirusTotal 2021年度恶意软件趋势报告[2]指出:已采集恶意文件约500×10^9个,因此传统基于Hash特征码匹配的检测方法难以应对当今恶意软件威胁形势。目前学术界和工业界主要通过逆向分析提取恶意软件特征,结合机器学习方法,构建恶意软件检测模型,以提升恶意软件检测覆盖率和效率,已取得了一定成果。例如,采用反汇编工具如IDA Pro等提取恶意软件操作码,基于Bi-gram等模型表征恶意软件特征,采用多种机器学习方法利用恶意/良性样本特征训练检测模型。不少研究表明,这类方法在限定的数据集内,可以有效检测恶意软件未知变种。

然而,随着恶意软件逆向分析和检测技术不断发展,攻击者为了逃避检测,常利用加壳技术对恶意软件进行加密或压缩,使得安全分析人员在恶意软件运行前难以利用反汇编等逆向工具对其进行静态分析。现有脱壳工具主要针对指定壳进行处理,对于未知壳/新型壳难以脱壳处理。因此,目前对于加壳恶意软件多采用动态检测方法,通过提取并分析恶意软件运行时的行为特征,作为判定未知软件为恶意软件的依据。然而,加壳恶意软件运行时的行为除了包含原恶意软件行为,还包含常见于良性加壳软件中的脱壳行为,容易混淆提取和判定恶意软件的行为特征;另一方面,受限于加壳恶意软件样本规模,采用有监督学习的检测方法,容易产生检测模型过拟合问题,造成对未知样本检测精度不足或误报率较高问题;不仅如此,对于新增未知样本,由于缺少可信标签,不利于检测模型通过有监督学习提升准确性和泛化性。

研究基于动态分析的加壳恶意软件检测方法,提取并分析加壳恶意软件运行时的系统调用行为特征,识别并筛选出敏感行为,旨在过滤脱壳行为噪声产生的影响;通过对系统调用行为特征加权降维,提升行为特征的有效性;通过对加权降维的行为特征进行聚类分析,最终实现加壳恶意软件未知变种动态分析检测。主要贡献包括:1)提出一种基于动态行为特征加权聚类的加壳恶意软件未知变种检测方法,通过识别敏感系统调用行为并进行加权降维,最后对待检测样本进行聚类分析,实现加壳恶意软件未知变种检测和模型增量更新;2)针对加壳恶意软件检测时误报率较高问题,提出一种加壳恶意软件特征提取方法,基于信息增益计算非加壳恶意/良性软件中不同系统调用权重,采用主成分分析对系统调用特征向量进行降维处理,以降低加壳/脱壳程序中系统调用特征对于加壳恶意软件检测的干扰;3)采用密度聚类算法对已标注样本特征进行聚类,标记恶意样本占比大于阈值的簇为恶意样本簇,计算新增未知样本与已标注样本的欧式距离,根据最小欧式距离样本的标签标注新增未知样本,如果该样本在恶意样本簇中,则将该样本判定为恶意样本并更新恶意样本簇,以实现检测模型增量更新检测;4)为了能快速检测加壳恶意软件,避免因传统密度聚类一一比较所有簇中的所有样本,而导致加壳恶意软件检测时间开销较大的问题,采取一种快速密度聚类簇搜索方法,通过计算每个簇的中心样本,首先搜索最邻近的恶意/非恶意簇中心样本,再依次搜索该簇内最小欧式距离标注样本,以降低比较次数提升搜索效率;5)实验结果表明:提出的基于动态行为特征加权聚类的加壳恶意软件未知变种检测方法相较于传统方法而言误报率显著降低,同时漏报率未见明显升高。

1 国内外研究现状

目前恶意软件检测方法多基于反汇编等软件逆向分析技术提取数据特征,并结合机器学习方法进行检测。汪嘉来等人[3]近年来对基于机器学习的Windows恶意软件智能检测技术进行综述;Wang等人[4]采用逆向分析技术提取恶意软件字符串特征和其他结构化特征,并通过融合多种特征检测恶意软件变种;杨鸣坤等人[5]基于静态分析技术提取API调用和Permission混合特征进行恶意软件检测;类似工作还包括文献[6-7]。然而,针对加壳恶意软件检测,由于软件加壳技术阻碍软件逆向分析,导致传统基于静态分析的检测

方法难以适用于检测加壳恶意软件。

因此,研究考虑采用动态分析技术,通过监测恶意软件运行时的异常行为对加壳恶意软件进行研判。Yan 等人[8]较为全面的调查并总结了基于动态分析的恶意软件检测方法;Suaboot 等人[9]提取动态运行时的 API 调用行为,采用马尔科夫链对局部 API 调用序列进行表征,并对恶意软件 API 调用行为进行检测;林鑫等人[10]分别提出基于沙箱技术提取恶意软件运行时的行为特征并进行检测;陈志锋等人[11]通过提取软件运行时与操作系统交互的内核特征以检测恶意软件,然而将上述方法直接用于检测加壳恶意软件时,容易被加壳恶意软件运行时的脱壳进行干扰,影响恶意软件行为特征的提取和判定,导致误报或漏报。Zhang 等人[12]提出一种基于恶意软件运行时的系统调用行为,并结合多层感知器实现恶意软件检测;Wang 等人[13]通过分析运行时 Permission 准入行为风险,检测恶意软件变种;Zhang 等人[14]提取可执行程序动态运行时的 API 调用行为特征以及静态代码的操作码二元组特征,并融合反馈神经网络和卷积神经网络方法对恶意软件进行检测,类似的方法还包括文献[15-16]。目前主流的恶意软件检测方法多为有监督学习方法方法,在较小规模的加壳恶意软件样本下,容易产生过拟合的问题而造成较高的误报率。

2　基于系统调用特征加权聚类的加壳恶意软件检测

2.1　总体方案概述

笔者提出一种基于动态行为特征加权聚类的加壳恶意软件检测方法,实现加壳恶意软件未知变种动态分析检测。该方法针对加壳恶意软件,首先利用沙箱提取加壳恶意/良性软件运行时的系统调用序列日志,包含脱壳程序和原程序运行时的系统调用序列日志;采用概率分布表示系统调用特征,基于信息增益计算非加壳恶意/良性软件中不同系统调用权重,以筛选对于区分非加壳恶意/良性软件信息量更大的系统调用特征,从而尽量降低加壳/脱壳程序中系统调用特征对于加壳恶意软件检测的干扰;由于系统调用特征维度较大,存在系统调用特征向量稀疏,影响检测模型的准确性,因此采用主成分分析对系统调用特征向量进行降维处理,获得非稀疏表征的系统调用特征;采用密度聚类算法对已标注样本特征进行聚类,获得不同样本的簇,如果该簇恶意样本占比大于阈值,那么标记为恶意样本簇;计算未知样本与已标注样本的欧式距离,搜索最小欧式距离标注样本,如果该样本在恶意样本簇中,则将该样本判定为恶意样本,否则判定为正常样本,实现加壳恶意软件未知变种的检测,同时,将该样本作为恶意样本添加至恶意样本簇中,以实现检测模型增量更新检测。技术路线如图 1 所示。

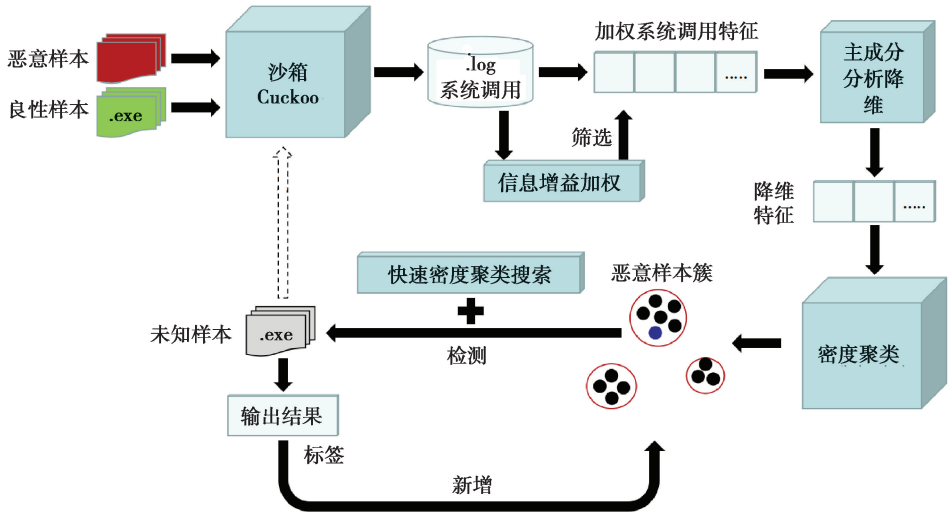

图 1　技术路线

Fig. 1　The work flow of the method

2.2 基于信息增益的加权系统调用特征表示

基于 Cuckoo 沙箱提取软件运行时的系统调用日志，将其构造为系统调用特征向量，并基于信息增益方法计算每个系统调用特征对恶意软件/良性软件分类的作用，作为权重对系统调用特征进行加权，得到软件运行时的系统调用加权特征向量，以表征软件运行时行为，同时降低加壳恶意软件脱壳行为的干扰。

令 $\mathrm{SysCall}=\{\mathrm{syscall}_1,\mathrm{syscall}_2,....,\mathrm{syscall}_i,....,\mathrm{syscall}_n\}$ 为软件运行时系统调用集合，其中 $\mathrm{syscall}_i$ 表示系统调用集合中索引为 i 的元素。令 **SysCallVec**$=[val_1,val_2,....,val_i,....,val_n]$ 为系统调用特征向量，其中 val_i 表示索引为 i 的系统调用特征值，根据公式(1)进行计算，其中 $\mathrm{Num}(val_i)$ 表示索引为 i 的系统调用在软件中出现的频次。令 **SysCallWeight**$=[\mathrm{weight}_1,\mathrm{weight}_2,....,\mathrm{weight}_i,....,\mathrm{weight}_n]$ 为系统调用特征权重，其中 weight_i 表示索引为 i 的系统调用特征权重值，采用信息增益方法，根据公式(2)进行计算，该权重值表示某系统调用特征对于恶意软件的敏感程度。其中 $p(\mathrm{syscall}_i\mid\mathrm{malware})$ 表示恶意软件中 $\mathrm{syscall}_i$ 出现的概率，$p(\mathrm{malware})$ 表示恶意软件的比例，$p(\mathrm{syscall}_i)$ 表示 $\mathrm{syscall}_i$ 出现的概率。加权系统调用特征向量根据公式(3)进行表示，为系统调用频次与权重的乘积。

$$val_i=\frac{\mathrm{Num}(val_i)}{\sum_{j=1}^{n}\mathrm{Num}(val_j)} \tag{1}$$

$$\mathrm{weight}_i=\left|\frac{p(\mathrm{syscall}_i\mid\mathrm{malware})}{p(\mathrm{syscall}_i)\cdot p(\mathrm{malware})}\cdot\log_2\frac{p(\mathrm{syscall}_i\mid\mathrm{malware})}{p(\mathrm{syscall}_i)\cdot p(\mathrm{malware})}\right|, \tag{2}$$

$$\mathrm{SysCallVec}_{\mathrm{weight}}=[val_1\cdot\mathrm{weight}_1,val_2\cdot\mathrm{weight}_2,....,val_i\cdot\mathrm{weight}_i,....,val_n\cdot\mathrm{weight}_n]。 \tag{3}$$

2.3 基于主成分分析法的加权系统调用特征降维

由于系统调用特征向量维度较大且数据稀疏，影响系统调用特征对恶意软件/良性软件分类的效果和效率，因此本文采用主成分分析法对加权系统调用特征向量进行降维，并且降低数据的稀疏性，从而效率的同时不影响特征向量精确性。令 CoVarianceMatrix 为系统调用特征向量的协方差矩阵，用于放大样本特征向量的差异，其中 $avg(\mathrm{SysCallVec}_{\mathrm{weight}})$ 表示加权系统调用特征向量期望，根据公式(4)进行计算。通过计算该协方差矩阵的特征值和特征列向量，获取特征值最大的前 $t(t=100)$ 个特征列向量形成特征矩阵，如公式(5)所示。最后根据公式(6)计算得出降维后的加权系统调用特征向量。该特征向量保留了原特征向量的主成分，降低维度的同时几乎不损失精度。

$$\mathrm{CoVarianceMatrix}=\frac{\sum(\mathrm{SysCallVec}_{\mathrm{weight}}-avg(\mathrm{SysCallVec}_{\mathrm{weight}}))}{\mathrm{Num}(\mathrm{malware})+\mathrm{Num}(\mathrm{malware})}, \tag{4}$$

$$\mathrm{EigenMatrix}=|\lambda\cdot E-\mathrm{CoVarianceMatrix}|_{t,t<n}, \tag{5}$$

$$\mathrm{SysCallVec}_{\mathrm{weight+reduceDim}}=\mathrm{SysCallVec}\cdot\mathrm{EigenMatrix}。 \tag{6}$$

2.4 基于密度聚类的加壳恶意软件检测模型构建

为了进一步提升加壳恶意软件检测的精度，基于上述降维后的加权系统调用特征向量，采用密度聚类算法 DBSCAN 设计加壳恶意软件检测模型。之所以采用密度聚类算法构建加壳恶意软件检测模型，是因为恶意软件特征和良性软件特征在 N 为空间中的分布并非是“球”型集中式分布，而是任意形状分布，并且密度聚类比较适用于处理该类数据聚类问题。针对训练集中的恶意软件样本和良性软件样本，根据公式(7)计算两两样本 $\mathrm{exe}_i,\mathrm{exe}_j$ 间欧式距离 $\mathrm{Dist}(\mathrm{exe}_i,\mathrm{exe}_j)$。对于距离 $\mathrm{Dist}(\mathrm{exe}_i,\mathrm{exe}_j)$ 小于等于阈值 $d(d=0.8)$ 且密度 $\rho(\mathrm{cluster})$ 大于等于阈值 $\alpha(\alpha=1)$ 样本聚类成簇，公式(8)判定样本是否可以聚类成簇。另外，孤立样本单独成簇。在所有聚类的簇中筛选出恶意样本簇 $\mathrm{cluster}_{\mathrm{malware}}$，即一个簇(密度大于等于 2 的簇)中恶意样本占比大于 $\beta(\beta$ 大于等于 0.85)，公式(9)用于判定该簇是否为恶意样本簇。

$$Dist(\mathrm{exe}_i,\mathrm{exe}_j)=(\mathrm{SysCallVec}_{\mathrm{weight+reduceDim},\mathrm{exe}_i}-\mathrm{SysCallVec}_{\mathrm{weight+reduceDim},\mathrm{exe}_j})^2\leqslant d, \tag{7}$$

$$\mathrm{exe}_i\in\mathrm{cluster}\mid\mathrm{exe}_j\in\mathrm{cluster}\&\mathrm{Dist}(\mathrm{exe}_i,\mathrm{exe}_j)\leqslant d\&\rho(\mathrm{cluster})\geqslant\alpha, \tag{8}$$

$$\mathrm{cluster}_{\mathrm{malware}}\mid\rho(\mathrm{malware})\geqslant\beta。 \tag{9}$$

2.5 基于快速密度聚类簇搜索加壳恶意软件检测

根据上述方法构建恶意样本簇，利用该恶意样本簇检测加壳未知可执行程序样本。对于未知可执行程

序样本,基于 Cuckoo 沙箱提取该样本加权降维后的系统调用特征向量,输入至聚类训练后的检测模型中,采用欧式距离公式,计算与已标注样本的加权降维后的系统调用特征向量,搜索最小欧式距离标注样本,如果该样本存在恶意样本簇中,那么将未知样本判定为恶意软件,否则判定为正常软件,公式(10)表明搜索出恶意样本簇中欧式距离最小的标注样本。

$$\text{arc min}_{\text{exe}_i}(\text{Dist}(\text{exe}_i,\text{exe}_u))\&\text{exe}_i \in \text{cluster}_{\text{malware}}。\tag{10}$$

由于传统密度聚类簇搜索需要一一比较所有簇中的所有样本,导致加壳恶意软件检测时间开销较大,为此采取一种快速密度聚类簇搜索方法:1)在进行已标注样本密度聚类过程中,计算每个簇的中心样本;2)在进行加壳恶意软件检测时,首先搜索最邻近的恶意样本簇中心样本,和最邻近的非恶意样本簇中心样本;3)再依次计算该恶意样本簇中心样本所属的簇内样本以及该非恶意样本簇中心样本所属的簇内样本之间的欧式距离,搜索最小欧式距离标注样本,如果该样本存在恶意样本簇中,那么将未知样本判定为恶意软件,否则判定为正常软件。

2.6　检测模型更新

对于新增样本,根据上述已标记的恶意样本簇,根据公式(7)计算新增未知样本与已标注样本的欧式距离,如果最小欧式距离样本在恶意样本簇中,则认为该样本为较高置信度样本,因此将该样本判定为恶意样本,否则判定为正常样本,即满足公式(10),同时将该样本作为恶意样本添加至恶意样本簇中,以更新恶意样本簇,从而实现检测模型增量更新检测,如公式(11)所示。

$$\text{cluster}_{\text{malware}} = \text{cluster}_{\text{malware}} \oplus \text{exe}_{\text{u}}。\tag{11}$$

综上所述,提取加壳恶意软件运行时的系统调用行为作为恶意软件运行时的动态特征,并通过加权、降维等方法减少特征中的噪声,提升特征的可用性,最后对恶意软件特征进行聚类分析以检测加壳恶意软件,同时增量训练更新检测模型。

3　实验分析

3.1　实验环境和数据集

所有实验均在相同计算机软硬件配置下完成,实验环境包括 CPU Intel i5-3470 @ 3.20GHz,RAM 16.0 GB,操作系统 Ubuntu 16。采集的恶意软件数据集来源于 VxHeaven,从中随机选取 2 000 个 Windows 恶意样本,包括病毒、蠕虫、木马、后门等类型;采集的 Windows 良性软件数据集来源于本地多台计算机,为保证数据集平衡,从中随机选取 2 000 个良性样本,包括 Windows 应用程序和用户应用程序。

3.2　验证方法

随机选取 80%的恶意软件和良性样本用于构建聚类检测模型,剩余 20%的恶意软件和良性样本采用当前对外公开的加壳工具如 ASPack、ZProtect、UPX 等以及企业编写的私有加壳工具进行加壳处理,用于测试评估聚类检测模型的性能。实验均采取 K-折交验证法,生成 10 组实验数据进行训练和测试,测试结果去 10 组实验平均值。评估指标包括准确率(ACC)、误报率(PRE)、漏报率(UP)、$F1$ 值(F1-score),如公式(12)、(13)(14)和(15)所示。其中 TP 表示样本为加壳恶意样本且被正确检测,FP 表示样本为加壳恶意样本但被错误检测,TN 表示样本为加壳良性样本且被正确检测,FN 表示样本为加壳良性样本单被错误检测,精度表示当检测结果判定样本为加壳恶意样本时其中正确结果的比例,召回率表示检测结果中加壳恶意样本中被正确检测出来的比例。

$$\text{ACC} = \frac{TP + TN}{TP + TN + FP + FN},\tag{12}$$

$$\text{FAR} = \frac{FP}{TP + FP},\tag{13}$$

$$UP = \frac{FN}{TP + FN},\tag{14}$$

$$F1 - \text{score} = 2 \times \frac{(1 - \text{FAR}) \times (1 - UP)}{(1 - \text{FAR}) + (1 - UP)}。\tag{15}$$

3.3 性能评估与对比分析

通过对比提出方法和基于几种典型机器学习方法包括 K 近邻、支持向量机、朴素贝叶斯检测加壳恶意软件的准确率、精度、召回率、$F1$ 值，以验证提出方法的效果和优势，实验结果如表 1 所示。实验结果表明：1)提出方法的误报率约为 3.9%，对比其他几种典型机器学习方法有显著降低，是因为仅当未知样本属于恶意样本簇时才会被判定为恶意样本，即具备明显恶意样本特征且与一定数量恶意样本特征相似时才会被判定为恶意样本，所以提升了恶意样本检测精度，降低了误报率；2)提出方法的准确率约为 89.4%、$F1$ 值约为 88.6%，对比其他方法均有一定提升；3)提出方法的漏报率约为 17.8%，虽然对比其他方法相对较高，但总体性能表现相对较好。

表 1 性能评估与对比分析结果

Table 1 Performance evaluation and comparison %

方法	误报率	漏报率	准确率	$F1$ 值
文本方法	3.9	17.8	89.4	88.6
K 近邻	18.9	7.8	85.4	86.3
支持向量机	17.2	7.5	86.6	87.4
贝叶斯分类	20.6	13.1	82.2	83.0

3.4 检测模型批量训练时不同训练规模的性能对比分析

由于不同规模的训练样本对准确性的影响较大，因此通过测试提出方法在不同规模训练样本下(批量训练比例：50%、60%、70%、80%)的准确率、精度、召回率、$F1$ 值，分析提出的方法在不同训练样本规模下的性能。如图 2 所示，实验结果表明：1)随着训练样本规模的增加，研究方法的各项性能指标呈现逐步提升的趋势，说明模型准确性和泛化性得到进一步提升。2)随着训练样本规模不断增加，模型的各项性能指标呈现平缓趋势，实验结果符合模型训练预期。

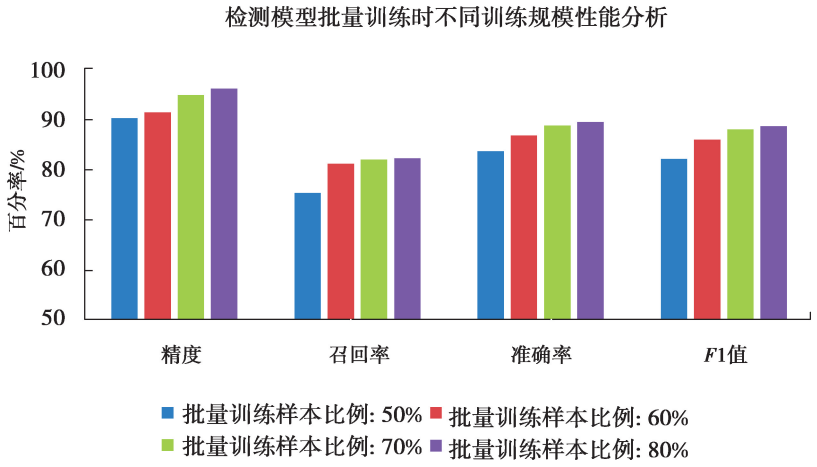

图 2 监测模型批量训练性能分析

Fig. 2 Performance analysis of detection model with different training scale

3.5 检测模型增量更新准确率分析

研究方法基于无监督降维、聚类实现加壳恶意软件检测和检测模型增量更新，为了测试检测模型增量更新的效率，采用提出的模型增量更新方法，测试随着检测样本增量训练(增量训练比例：50%+10%、50%+20%、50%+30%)时检测模型的性能。如图 3 所示，测试结果表明：1)随着检测样本的不断增加，被检出的恶意样本更新至恶意样本簇中，能提升模型的检测性能，说明被检出的恶意样本是准确的，能保障模型更新后的准确性；2)相较于检测模型批量训练，增量训练的性能略低于批量训练的准确率。

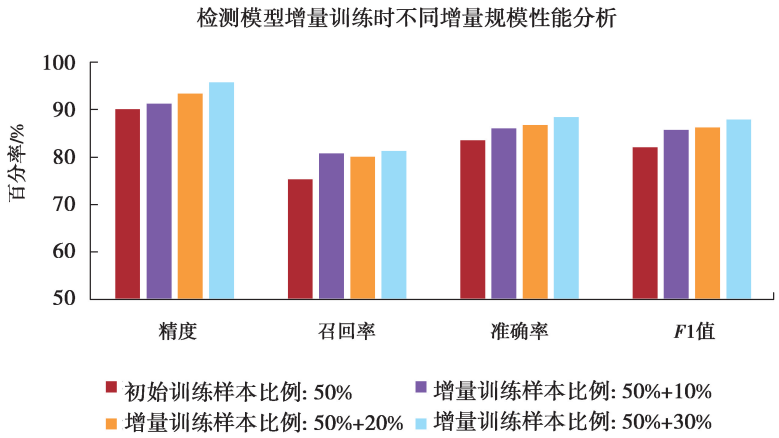

图 3　检测模型增量训练性能分析

Fig. 3　Performance analysis of detection model update

3.6　特征维度对性能影响分析

为了分析研究方法进行主成分分析降维时特征维度对准确率和检测时间开销的影响,选取不同的特征维度(t 值取值分别为 10,50,100,200,300),通过对比分析不同特征维度取值下准确率的变化趋势,选取最优特征维度,如表 2 所示。实验结果表明:1)当特征维度>100 时,准确率基本变化不大;2)当特征维度<100 时,准确率呈明显下降趋势;3)特征维度越小,检测时间开销越小,反之亦然。

表 2　特征维度对性能影响分析

Table 2　Performance evaluation of different feature dimensions

T 值	准确率/%	平均检测时间开销/s
10	73.4	0.09
50	81.2	0.77
100	89.4	1.60
200	89.4	3.31
300	89.6	5.93

3.7　训练时间开销与检测时间开销对比分析

通过对比分析研究方法与其他对比方法的训练时间开销和检测时间开销,以评估方法训练与检测效率,如表 3 所示。实验结果表明:1)研究方法与 K 近邻方法仅需比较检测样本与已知样本的特征相似度,因此无需提前训练模型,支持向量机与贝叶斯分类需要提前训练模型,其中支持向量机训练时间开销较大;2)研究方法平均检测时间开销小于 K 近邻方法,且研究快速密度聚类簇搜索时间开销小于传统密度聚类搜索时间开销,说明研究方法提升了检测效率;3)尽管支持向量机平均检测时间开销较小,但训练时间开销较大,而贝叶斯分类训练和检测时间开销均较小,但检测准确性较差。

表 3　训练时间开销与检测时间开销评估

Table 3　Evaluation of training and detection time cost

方法	训练时间开销/s	平均检测时间开销/s
文本方法(快速密度聚类簇搜索)	0.00	1.60
研究方法(传统密度聚类搜索)	0.00	6.59
K 近邻	0.00	13.44
支持向量机	973.17	< 0.01
贝叶斯分类	6.67	< 0.01

4 总 结

研究提出一种基于动态行为特征加权聚类的加壳恶意软件未知变种检测方法，基于信息增益计算非加壳恶意/良性软件中不同系统调用权重，采用主成分分析对系统调用特征向量进行降维处理，最后基于密度聚类算法对其进行聚类分析，实现加壳恶意软件未知变种检测。实验结果表明：提出的基于运行时系统调用特征加权聚类的加壳恶意软件未知变种检测方法相较于传统方法而言误报率显著降低，同时漏报率未见明显上升。未来将会进一步改进密度聚类检测方法以降低漏报率，改进恶意样本簇更新方法以确保检测模型时效性的同时保障检测模型的准确性。

参考文献：

[1] 国家计算机网络应急技术处理协调中心.2018年中国互联网网络安全报告[R].2020.
CNCERT/CC. 2018 China Internet cyber-security report[R].2020.

[2] VirusTotal.VirusTotal's 2021 malware trends report[R/OL]. (2022-03-01)[2022-05-01]//https://assets.virustotal.com/reports/2021trends.pdf.

[3] 汪嘉来，张超，戚旭衍，等. Windows平台恶意软件智能检测综述[J]. 计算机研究与发展，2021，58(5)：977-994.
Wang J L，Zhang C，Qi X Y，et al. A survey of intelligent malware detection on windows platform[J]. Journal of Computer Research and Development，2021，58(5)：977-994.(in Chinese)

[4] Wang W，Gao Z Z，Zhao M C，et al. DroidEnsemble：detecting android malicious applications with ensemble of string and structural static features[J]. IEEE Access，2018，6：31798-31807.

[5] 杨鸣坤，罗锦光，欧跃发，等. 基于API和Permission的Android恶意软件静态检测方法研究[J]. 计算机应用与软件，2020，37(4)：53-58，104.
Yang M K，Luo J G，Ou Y F，et al. Android malware static detection method based on api and permission[J]. Computer Applications and Software，2020，37(4)：53-58，104.(in Chinese)

[6] Fan M，Liu J，Luo X P，et al. Android malware familial classification and representative sample selection via frequent subgraph analysis[J]. IEEE Transactions on Information Forensics and Security，2018，13(8)：1890-1905.

[7] Tian K，Yao D F，Ryder B G，et al. Detection of repackaged android malware with code-heterogeneity features[J]. IEEE Transactions on Dependable and Secure Computing，2020，17(1)：64-77.

[8] Yan P，Yan Z. A survey on dynamic mobile malware detection[J]. Software Quality Journal，2018，26(3)：891-919.

[9] Suaboot J，Tari Z，Mahmood A，et al. Sub-curve HMM：a malware detection approach based on partial analysis of API call sequences[J]. Computers & Security，2020，92：101773.

[10] 林鑫. 基于沙盒的Android恶意软件检测技术研究[J]. 电子设计工程，2016，24(12)：48-50，53.
Lin X. Malware detection technology research of Android platform based on sand box[J]. Electronic Design Engineering，2016，24(12)：48-50，53.(in Chinese)

[11] 陈志锋，李清宝，张平，等. 基于数据特征的内核恶意软件检测[J]. 软件学报，2016，27(12)：3172-3191.
Chen Z F，Li Q B，Zhang P，et al. Data characteristics-based kernel malware detection[J]. Journal of Software，2016，27(12)：3172-3191.(in Chinese)

[12] Zhang J X，Zhang K H，Qin Z，et al. Sensitive system calls based packed malware variants detection using principal component initialized MultiLayers neural networks[J].Cybersecurity，2018，1(1)：1-13.

[13] Wang W，Wang X，Feng D W，et al. Exploring permission-induced risk in android applications for malicious application detection[J]. IEEE Transactions on Information Forensics and Security，2014，9(11)：1869-1882.

[14] Zhang J X，Qin Z，Yin H，et al. A feature-hybrid malware variants detection using CNN based opcode embedding and BPNN based API embedding[J]. Computers & Security，2019，84：376-392.[LinkOut]

[15] Darem A A，Ghaleb F A，Al-Hashmi A A，et al. An adaptive behavioral-based incremental batch learning malware variants detection model using concept drift detection and sequential deep learning[J]. IEEE Access，2021，9：97180-97196.

[16] Won D O，Jang Y N，Lee S W. PlausMal-GAN：plausible malware training based on generative adversarial networks for analogous zero-day malware detection[J]. IEEE Transactions on Emerging Topics in Computing，2022，PP(99)：1.

（编辑 侯 湘）