文章编号:1007-5321(2019)04-0089-07 **DOI**:10.13190/j.jbupt.2018-318

一种基于行为集成学习的恶意代码检测方法

胥小波[1,2], 张文博[1], 何 超[1], 罗 怡[1]
(1. 中国电子科技网络信息安全有限公司, 成都 610041;
2. 中国电子科技集团公司第三十研究所, 成都 610041)

摘要: 为了解决变种恶意代码、未知威胁行为恶意分析等问题,研究了基于梯度提升树的恶意代码分类方法,从大量样本中学习程序行为特征和指令序列特征,实现了智能恶意代码分类功能. 将 GBDT 算法引入恶意代码检测领域,使模型结果行为序列具有可解释性,对恶意代码的检测能力大幅提高. GBDT 算法能够客观地反映恶意代码的行为和意图本质,能够准确识别恶意代码.

关 键 词: 恶意代码; 未知威胁; 梯度提升树; 行为特征
中图分类号: TP302 **文献标志码**: A

A Malicious Code Detection Method Based on Ensemble Learning of Behavior

XU Xiao-bo[1,2], ZHANG Wen-bo[1], HE Chao[1], LUO Yi[1]
(1. China Electronics Technology Cyber Security Company Limited, Chengdu 610041, China;
2. China Electronic Technology Group Corporation Thirtieth Research Institute, Chengdu 610041, China)

Abstract: In order to solve the problem of variant malicious code and behavior analysis of unknown threat, a method for malware classification based on gradient boosting decision tree (GBDT) algorithm is researched, which learns the characteristics of code behavior and instruction sequence from a large number of samples, and realizes the intelligent malicious code classification function. GBDT algorithm is introduced into the field of malicious code detection, so that the behavior sequence of the model is interpretable, and improves its ability to detect malicious code significantly. GBDT algorithm can reflect the nature of the behavior and intention of malicious code objectively, and identify malicious code accurately.

Key words: malware code; unknown threat; gradient boosting decision tree; behavior characteristics

基于恶意代码特征码技术无法有效解决恶意代码变种、代码混淆等问题,基于沙箱技术可以跟踪代码的执行行为,却无法有效评估行为整体是否具有恶意. 针对这些问题,提出了基于行为的恶意样本集成学习方法,该方法能够从大量样本中学习区分正常和恶意的行为特征组合. Jacob 等[1]提出基于代码行为分析的恶意判定方法,可以在一定程度上避免恶意代码混淆的影响,但是这种方法无法解决行为层混淆和系统调用重排的干扰. 为克服恶意代码行为层混淆问题,Christodorescu 等[2]提出一种基于语义的特征检测方法. 该方法匹配过程比较复杂,且本质上是一种静态检测方法. 王蕊等[3]提出一种使用动态污点传播技术,结合代码语义进行恶意代码检测的方法,计算复杂性较高. 韩晓光等[4]提出一种基于纹理指纹的恶意代码特征提取及检测方法,不具备对加壳或加密恶意程序的检测能力. Fan Yujie 等[5-6]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. Hiromu Yakura 等[7]结合卷积神经网络和注意力机制解决恶意代码分类问题. Cui

收稿日期: 2018-12-22
作者简介: 胥小波(1985—), 男, 博士, E-mail: xxb0620@163.com.

等[8]同样将恶意代码转换为灰度图像,利用卷积神经网络(CNN,convolutional neural networks)实现对恶意代码的分类.

提出的基于GBDT算法的恶意代码检测方法具有较好的可解释性,更有利于与安全业务场景融合.且结合了行为特征和指令流特征建立模型,能够更加精确地检测恶意代码. 该算法不依赖于行为的时间序列和先后顺序,可以避免行为层混淆和系统调用重排的干扰. 使用样本行为特征、应用程序编程接口(API,application programming interface)特征、指令序列特征设计出一种快速准确的恶意样本分类方法. 利用人工智能及机器学习算法,自动化、智能化学习恶意样本行为的内在规律,达到对样本进行智能分类的目的.

1 行为监控及提取

行为监控及提取通过机器代码跟踪技术,实现对API系统调用层和指令层的监控. 前者通过拦截记录所有样本的系统调用行为,然后集中分析样本行为特征(如文件、注册表、进程、服务、网络、其他行为)实现动态行为检测. 后者则通过指令流(如内核状态、寄存器读写、输入/输出(IO,input/output)操作、内存读写执行、中央处理器(CPU,central processing unit)状态等)分析样本行为特征.

分析环境的构建技术包括仿真环境的搭建、运行规则的设定、运行环境的监控、模拟按钮点击、运行结果的封装和传输等. 通过虚拟化技术,建立代码跟踪环境,支持Windows XP、Windows 7(32位和64位)运行环境,对运行过样本的环境进行还原,并且确保每次分析环境干净,无干扰的运行环境和网络环境. 运行规则的设定,自动识别机器代码类型,针对不同类型的动态分析样本,提供不同的启用方式和启用参数. 对仿真环境300多个底层API进行监控,监控其系统API情况. 在运行过程中模拟鼠标移动和点击,自动进行对话框对话操作,确保全面的动态行为的监控.

支持静态和动态分析,静态分析包括机器代码结构分析、资源检测、证书链校验等,动态分析包括API监控和指令分析,对系统底层300多个API进行监控,根据恶意程序的特征对API调用方式、参数、返回结果等进行分析,建立100多个行为分析策略.

2 算法模型

2.1 算法框架

基于行为的恶意样本集成学习算法建模的思路如图1所示. 在基础数据分析的基础上,需要对特征进行抽取,特征分析,特征筛选等操作,再进行模型算法选择,学习器构建、生成等,整个过程可以视为模型建立. 从数据的抽取,到数据清洗,特征处理

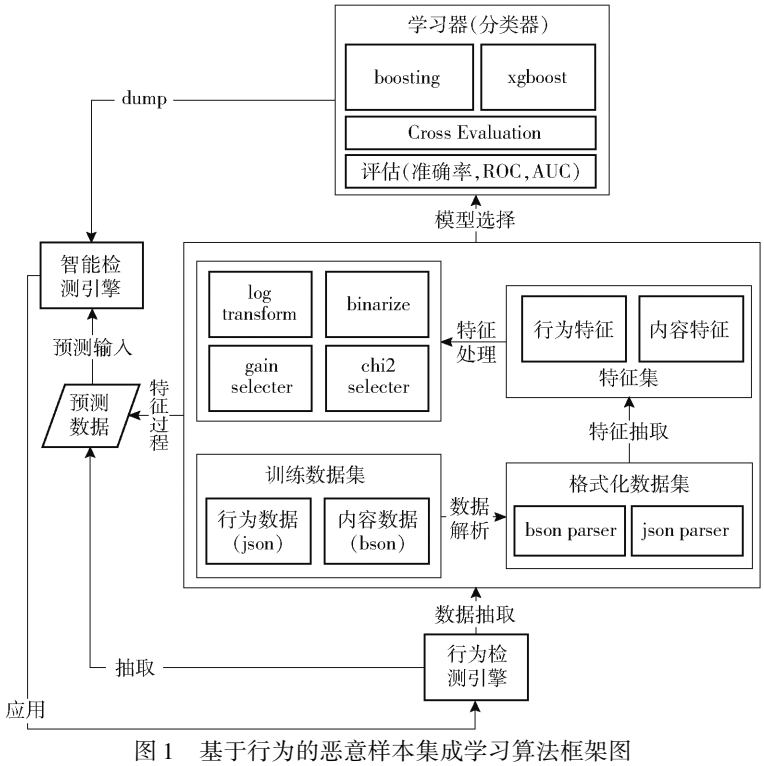

图1 基于行为的恶意样本集成学习算法框架图

到学习器生成，最终到模型预测.

数据清洗：在数据清洗阶段主要对一些特殊样本、特殊特征进行清洗；特征过程：从原始数据中抽取特征，并对特征进行分析、处理、转化、筛选等，在后期预测同样会对新样本数据进行特征抽取处理，再通过学习器对新样本进行预测；模型建立：建立学习器主要使用机器学习算法，形成集成模型作为学习器输出，作为下一步预测；模型预测：将学习器应用到行为检测引擎中，对新样本进行检测和预测，实现智能恶意检测、未知威胁发现的目的.

2.2　算法模型

基于行为的恶意样本集成学习算法是基于 boosting 和梯度结合的集成学习算法. 算法模型如下：

输入：训练样本集合

$$T=(x_1,y_1),(x_2,y_2),\cdots,(x_m,y_m).$$

输出：强学习器 $f(x)$.

1）初始化：

n_estimators：迭代次数，弱分类器的个数；learning_rate：学习率；loss_function：损失函数选择. 在中，建模时使用的损失函数为对数似然损失函数，如

$$L(y,f(x))=\log(1+\exp(-yf(x))) \tag{1}$$

2）迭代训练过程：

第 1 轮迭代：

使用式(1)对数据进行一次训练，得到 $L(y,f(x_i))$ 的损失函数值：

$$(c_1,y_1),(c_2,y_2),\cdots,(c_m,y_m)$$

第 2 – T 轮迭代：

① 对于样本 $i=1,2,3,\cdots\cdots,m$，计算负梯度. 在迭代过程中，对损失函数的残差或负梯度进行拟合. 式(2)是第 t 轮迭代的第 i 个样本的损失函数的负梯度.

$$r_{ti}=-\left[\frac{\partial L(y,f(x_i))}{\partial f(x_i)}\right]_{f_t(x)=f_{t-1}(x)} \tag{2}$$

② 利用 (x_i,r_{ti}) 拟合一棵分类与回归树(CART，classification and regression tree)，得到第 t 棵回归树，对应的叶子结点区域为 R_{ij}. 其中 j 为叶子结点的个数.

③ 按照式(3)计算最佳拟合值，针对所有样本求出损失最小. 其中，c 为每轮迭代需要进行拟合的目标，$L(y,f_t(x))$ 为 t 轮迭代的损失函数.

$$c_{ij}=\arg\min_{c}\sum_{x_i\in R_{ij}}L(y_i,f_{t-1}(x_i)+c) \tag{3}$$

④ 按照式(4)更新强学习器.

$$f_t(x)=f_{t-1}(x)+\sum_{j=1}^{J}c_{ij}I(x\in R_{ij}) \tag{4}$$

3）迭代结束，得到最终的学习器如式(5)所示.

$$f(x)=f_T(x)=\sum_{t=1}^{T}\sum_{j=1}^{j}c_{ij}I(x\in R_{ij}) \tag{5}$$

2.3　算法流程

基于行为的恶意样本集成学习算法流程如图 2 所示. 整个流程主要分为特征抽取及特征工程、

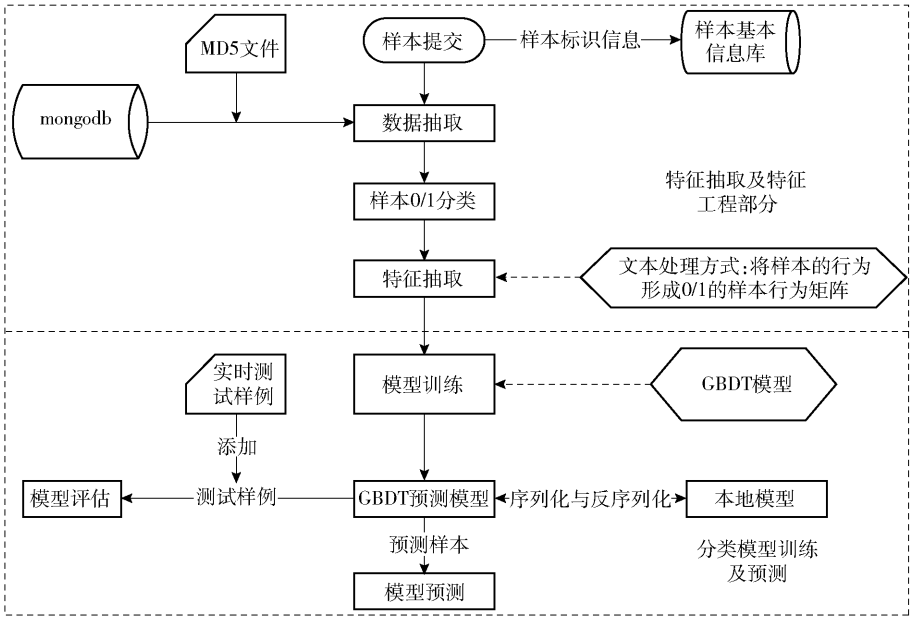

图 2　基于行为的恶意样本集成学习算法流程

GBDT 分类模型训练及预测 2 个部分. 特征抽取过程,主要将样本的原始行为进行向量化,模型原始行为是一串带分割的文本,将所有样本的行为形成矩阵的列,样本为矩阵的行,将样本在行为上的参数作为矩阵的值. 再进行部分特征的筛选,筛选后得到样本的特征矩阵. 模型的训练部分,主要是选择 GBDT 集成分类算法,GBDT 算法是基于决策树进行建立. 当进入一个或多个样本,进行预测时,会根据训练时输出的每个模型进行模型预测,然后对预测的结果进行融合,最终得到分类预测结果.

2.3.1 机器代码行为特征提取

所有训练样本利用脚本程序批量提交至机器代码行为提取分析系统进行分析. 样本在进入机器代码行为提取分析系统之前,首先计算该样本的消息摘要算法 5(MD5,message-digest algorithm 5)值,该 MD5 值为样本的唯一标识. 样本进入机器代码行为提取分析系统分析后,该系统会生成一个 task_id 来标记此次分析任务. 样本提交至机器代码行为提取分析系统后,后台会将该样本的 MD5 值和机器代码行为提取分析系统任务号 task_id 这些样本标识信息存入样本基本信息库.

机器代码行为提取分析系统将所有分析样本的行为数据、API 调用数据和指令流分析完之后,将会批量从样本基本信息库中读取样本基本信息,然后利用这些样本的基本信息去反查存放在 MongoDB 中的样本行为特征数据、样本 API 特征数据和指令数据. 将 API 序列、指令序列的名称作为横坐标,行为次数作为纵坐标,经过特征提取、特征可视化、特征描述统计、特征转化、特征正态化、特征筛选和特征降维等步骤,映射到特征矩阵.

通过分析样本执行指令流(内核状态、寄存器读写、IO 操作、内存读写执行、CPU 状态等),突破了机器代码指令级跟踪技术,实现了对 Shellcode、堆喷射漏洞、面向返回地址编程(ROP, return oriented programming)漏洞、结构化异常处理(SEH, structured exception handling)漏洞的指令流检测,作为指令流特征.

2.3.2 特征降维

在不降维的情况下,模型训练的特征数至少在 600 维以上,对于分类模型而言,特征数太多会影响模型效果. 因此需要进行降维,在模型中降维方法选择奇异值分解(SVD,singular value decomposition)方法. SVD 可以对数据进行降噪,且处理稀疏矩阵的能力较强. 降维后的维度数主要通过 SVD 分解后的特征值确定. 对训练集进行降维的过程中会输出降维后的特征矩阵,并返回训练后的模型. 在预测时用该降维的模型对数据进行转化,再进行模型输入.

2.3.3 模型训练

在部署的环境中,模型训练可以选择适当的时间,定期对模型进行更新训练. 可以实时的检测是否有新的样本加入. 在训练过程中会将之前的模型进行备份,训练完成后才会删除之前备份的模型. 在模型训练时,将训练数据分为 2 个部分,一部分数据作为训练数据,剩下一部分用于模型效果的评估. 分割的比例为 8:2,抽样的方式为随机抽样. 整个模型训练流程如图 3 所示.

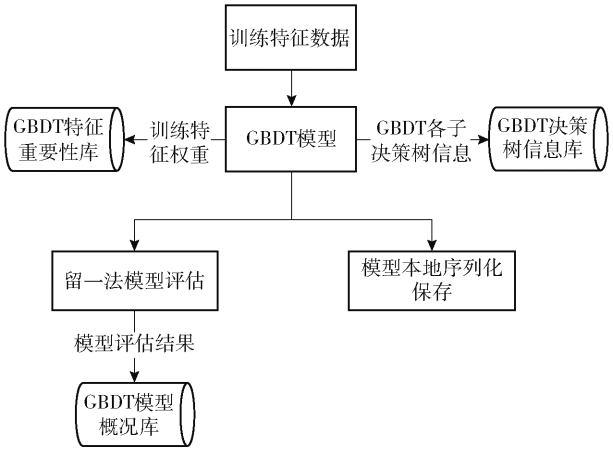

图 3 模型训练流程

恶意样本检测是一个二分类问题,因此 GBDT 的损失函数采用对数似然损失函数,对数似然损失函数对二元分离和多元分类都有比较好的优化. 经过实际调参训练,GBDT 分类器的学习率和迭代次数分别设置为 0.1 和 200.

训练特征数据进入 GBDT 模型训练后,会生成 200 棵不同的子决策树. GBDT 分类模型训练完之后,各子决策树的详细信息会存储在 GBDT 决策树信息库中. GBDT 分类模型自带特征权重评价方法,特征权重描述了模型训练时各个特征的重要程度,对训练样本及模型的可解释有一定程度的表现. 模型训练完成后,会利用另外的 20% 测试数据对模型进行评估. 评估结果存入 GBDT 模型概况库. 训练好的模型会序列化保存在本地,下次使用该模型时直接从本地载入即可.

采用主动学习的机制,当检测到出现新型的行

为样本时则会将新型的行为加入到学习器中进行学习. 主动学习过程可以自动实现模型的扩展,实现自我学习,主动学习技术框架如图 4 所示. 框架中首先定义定时器会在周期性的发起任务—检测模块,去检测数据系统中是否存在新的数据行为数据. 并更新整个系统的状态,如果存在新型的行为,就会获取到新型行为样本的 ID,并推送给数据抽取模块. 数据抽取模块进行数据抽取,抽取完成后将数据输入到第一部分中的学习系统,并触发学习任务,对新型的行为进行学习. 学习完成后更新模型,完成一次主动学习.

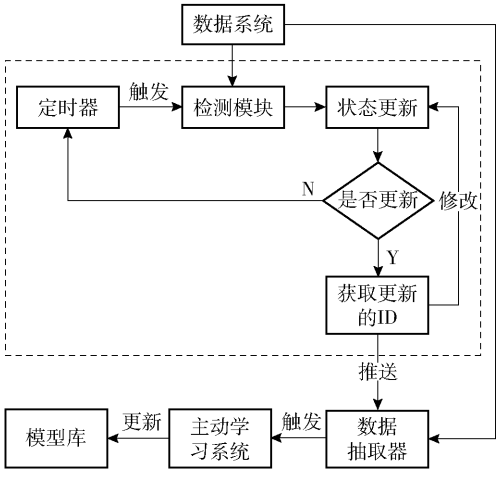

图 4　主动学习技术框图

3　性能测试与分析

训练样本来自于安全产品积累、安全公司共享、威胁情报、互联网采集等各个途径,训练正例样本收集 70 802 个. 恶意样本收集如表 1 所示,总计 80 786个样本.

表 1　训练样本情况表

黑客工具	灰色软件	病毒	木马	蠕虫	风险软件
468	3 622	12 329	47 897	15 704	766

基于沙箱技术可以跟踪代码的执行行为,却无法有效评估行为整体是否具有恶意. 在未采用机器学习算法的情况下,分别研究了基于行为规则、行为加权打分等技术,基于这些技术进行恶意研判. 但此类方法效果较差,主要是由于基于规则或行为加权打分的方法,无法完全总结出恶意行为的内在规律,而且更新规则较为困难. 在个人电脑端和移动端都存在精度较低、对安装文件误报率较高等问题.

在中筛选后的特征为 478 维,总共 15 万样本量进行模型训练,基于 GBDT 算法,使用 200 个弱分类器,损失函数选择对数似然损失. GBDT 算法的核心为上轮子决策树的偏差为下轮子决策树训练的目标. 由于各子决策树的偏差不一样,所以导致各子决策树的树型结构也不一致. 图 5 所示为 GBDT 某子决策树的部分结构,可以清晰发现,该决策树模型首先采用 CreatWindowExA 这个 API 调用次数的 9.5 值作为判断的根阈值,形成行为属性恶意识别链,可以较好地融入安全产品中.

特征重要性排名分析可以描述 GBDT 模型在进行预测样本时,哪些特征在起作用及贡献力为多少. 当样本提交至 GBDT 模型进行预测时,GBDT 会根据预测时的特征贡献力对各特征进行重要程度计算. 如图 6 所示,在进行对该样本的预测时,CreatWindowExW、GetProcAddress、LdrGetDllHandle 等 API 特征对该样本的预测较为重要.

进一步采用 GBDT 和 Adaboost 分别训练了 2 个

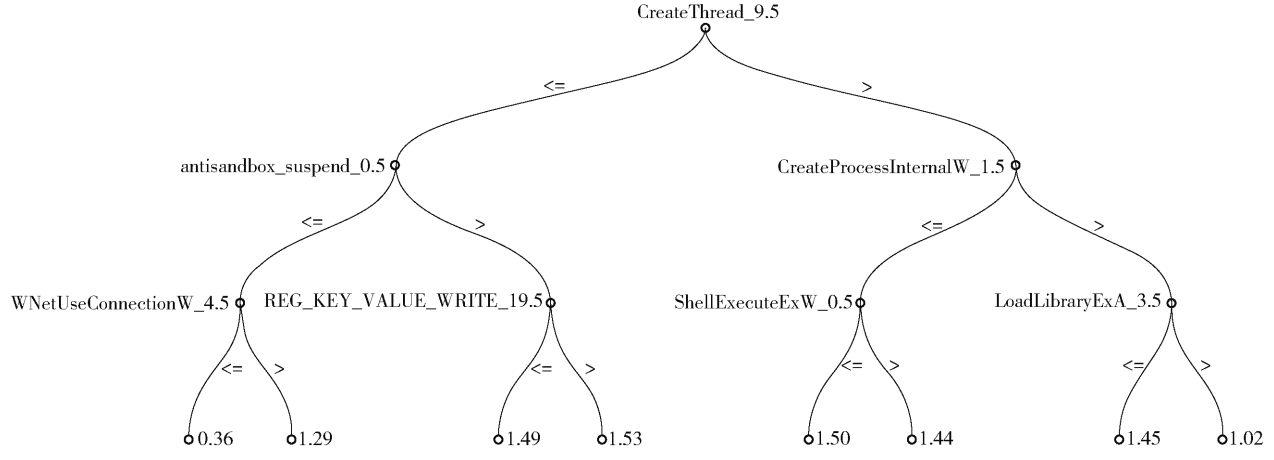

图 5　GBDT 某子决策树结构

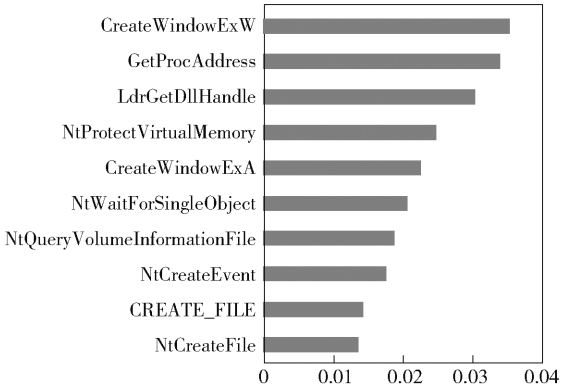

图6 测试样本特征重要性排名

集成算法模型. Adaboost 主要是从分类目标上进行模型构建,采用多弱分类器加权的形式得到整个目标. GBDT 主要是对损失项进行拟合,采用 stack 的形式向目标进行逼近. 利用 F1 和 roc_auc 对模型效果进行评估,在参数一致的情况下测试结果如表2所示,GBDT 模型比 Adaboost 模型恶意检测效果更好.

表2 模型测试结果

模型	F1	roc_auc
GBDT	0.961	0.969
Adaboost	0.944	0.954

表3所示为不同检测算法的准确率,实验结果表明,基于行为的恶意样本集成学习算法能够综合学习行为类型及参数、API 调用、指令流等特征,检测准确率和效果更好.

表3 算法准确率比较

方法	论文发表年份	准确率
决策树[9]	2009	0.950
基于置信度的算法[10]	2012	0.900
K-临近[11]	2014	0.941
循环神经网络+卷积神经网络[12]	2016	0.894
神经网络[13]	2017	0.905
使用的算法	—	0.961

综上所述,基于 GBDT 的恶意代码检测方法具有如下特点:检测精度较高;结果具有可解释性;上线运行速度较快;有助于安全分析专家较好理解行为关系;支持行为规则的提取,并能够将提取的规则整合到基于规则防护的安全产品中,形成未知威胁行为模式发现、规则提取、规则实施的闭环体系.

4 结束语

通过研究 GBDT 算法实现了智能恶意代码分类功能. 将 GBDT 算法引入恶意代码检测领域,基于程序行为特征、API 调用特征和指令特征,使得恶意代码的检测能力大幅提高. 该方法能够客观地反映恶意代码的本质,准确识别恶意代码,并且具有较低的误报率. 现有算法在对行为序列的可解释性上较差,所提出的基于 GBDT 的恶意代码检测方法具有较好的可解释性,能够更好地分析恶意代码的行为,较好地解决具有多重系统威胁行为的安装软件误判问题,更有利于与安全业务场景融合. 该方法实现了自主学习功能,使得系统可以实时更新行为特征和模型.

参考文献:

[1] Jacob G, Debar H, Fillol E. Behavioral detection of malware: from a survey towards an established taxonomy [J]. Journal in Computer Virology, 2008, 4(3): 251-266.

[2] Christodorescu M, Jha S, Seshia S A, et al. Semantics-aware malware detection [C]// Proc of the 2005 IEEE Symposium on Security and Privacy. California: [s. n.], 2005: 32-46.

[3] 王蕊, 冯登国, 杨轶等. 基于语义的恶意代码行为特征提取及检测方法[J]. 软件学报, 2012, 23(2): 378-393.
Wang Rui, Feng Dengguo, Yang Yi, et al. Semantics-based malware behavior signature extraction and detection method [J]. Journal of Software, 2012, 23(2): 378-393.

[4] 韩晓光, 曲武, 姚宣霞等. 基于纹理指纹的恶意代码变种检测方法研究[J]. 通信学报, 2014, 35(8): 125-136.
Han Xiaoguang, Qu Wu, Yao Xuanxia, et al. Research on malicious code variants detection based on texture fingerprint [J]. Journal on Communications, 2014, 35(8): 125-136.

[5] Fan Yujie, Chen Lifei, Guo Gongde. Learning and classification of malicious behaviors in software code [J]. Journal of Data Acquisition and Processing, 2017, 32(3): 612-620.

[6] 胥小波, 郑康锋, 李丹等. 新的混沌粒子群优化算法[J]. 通信学报, 2012, 33(1): 24-33.
Xu Xiaobo, Zheng Kangfeng, Li Dan, et al. New chaos-particle swarm optimization algorithm [J]. Journal on

Communications, 2012, 33(1): 24-33.

[7] Hiromu Yakura, Shinnosuke Shinozaki, Reon Nishimura, et al. Malware analysis of imaged binary samples by convolutional neural network with attention mechanism[C]// CODASPY'18 Proceedings of the Eighth ACM Conference on Data and Application Security and Privacy. Dallas, Texas, USA. ACM Press the 10th ACM Workshop. 2018: 127-134.

[8] Cui Zhihua, Xue Fei, Cai Xingjuan, et al. Detection of malicious code variants based on deep learning[J]. IEEE Transactions on Industrial Informatics, 2018(14): 3187-3196.

[9] 黄全伟. 基于 N-Gram 系统调用序列的恶意代码静态检测[D]. 哈尔滨:哈尔滨工业大学, 2009.

[10] Ravi C, Manoharan R. Malware detection using windows API sequence and machine learning [J]. International Journal of computer Applications, 2012, 43 (17): 12-16.

[11] 廖国辉, 刘嘉勇. 基于数据挖掘和机器学习的恶意代码检测方法[J]. 信息安全研究, 2016(1):74-79. Liao G H, Liu J Y. A malicious code detection method based on data mining and machine learning [J]. Journal of Information Security Research, 2016(1):74-79.

[12] Tobiyama S, Yamaguchi Y, Shimada H, et al. Malware detection with deep neural network using process behavior [C]//2016 40th Annual IEEE Conference on Computer Software and Applications (COMPSAC). Atlanta: IEEE, 2016: 577-582.

[13] Dahl G E, Stokes J W, Deng L, et al. Large-scale malware classification using random projections and neural networks [C]//2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Vancouver: IEEE, 2013: 3422-3426.