ISCAS 中国科学院软件研究所学术年会'2022
暨计算机科学国家重点实验室开放周
学术论文

MalGraph: Hierarchical Graph Neural Networks for Robust Windows Malware Detection

基于层次化图神经网络的鲁棒Windows恶意软件检测

凌祥[1,2]，吴凌飞[3]，邓伟[2]，曲振清[2]，张江瑜[2]，张晟[2]，马腾飞[4]，王滨[5]，吴春明[2]，纪守领[2]

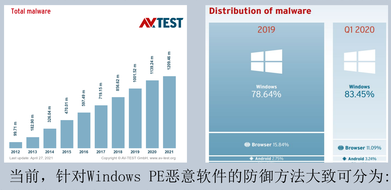
[1]中国科学院软件研究所，[2]浙江大学，[3]京东硅谷研究院，[4]IBM研究院，[5]海康威视

发表在 IEEE INFOCOM 2022 （CCF-A类推荐会议）

主要联系人：凌祥　　**电话**：13116760356　　**邮箱**：lingxiang@iscas.ac.cn

背景与动机

随着软件工程技术的快速发展与应用，软件的大小和规模持续增加，恶意软件也随之呈现指数级的增长，并逐渐成为当前网络空间中最严重的安全威胁之一。据AV-TEST统计，当前出现在各类操作系统中恶意软件的总数已经从2015年的4.7亿激增到2020年的11.4亿。此外，由于Windows操作系统已经在个人和企业用户中被普遍使用，所以当前有超过80%恶意软件的攻击目标是Windows，并且大部分恶意软件的格式是**可执行文件**（Portable Executable，简称“PE”）。虽然恶意软件还存在许多其他文件格式，但是我们认为针对一种格式的恶意软件检测方法通常经过少量的调整就可以应用到针对其他文件格式的恶意软件检测。本研究主要关注**Windows PE恶意软件检测**。



当前，针对Windows PE恶意软件的防御方法大致可分为：①**基于静态分析的检测方法**主要通过提取软件的静态特征来判断该软件是否为恶意软件；②**基于动态分析的检测方法**则是通过分析软件在隔离环境中运行的各种行为特征来判断该软件是否为恶意软件。因为基于动态分析的检测方法所需要的计算资源和时间消耗通常都非常大，并且必须在恶意软件运行后才能被检测出来，所以这类检测方法很难被扩展到不同的场景中，从而无法在现实世界中被广泛地使用和部署。因此，本研究主要研究**基于静态分析的恶意软件检测方法**。

传统的基于静态分析的检测方法最早可以追溯到经典的**基于签名的恶意软件检测方法**。然而，这种检测方法的一个致命的缺点是完全依赖于已知恶意软件的签名库，从而只能检测出已知的恶意软件。近年来，随着机器学习和深度学习的不断发展和广泛应用，大量的研究人员提出各种**基于传统机器学习或者深度学习的恶意软件检测方法**。尽管这些检测方法可以利用机器学习模型或者深度学习模型的学习能力在一定程度上检测出未知的恶意软件，但是我们认为这些检测方法存在**两个关键问题**影响恶意软件检测的**准确性**和**鲁棒性**，即：

- **准确性有限**：当前检测方法通常采用**简单特征**或者**浅层特征**来表示可执行文件，忽略了其中更深层次的结构特征和语义信息，从而很难获得恶意软件中最本质的特性。
- **鲁棒性极差**：当前检测方法几乎都建立在一些很容易被修改的特征上，因此所能提供的鲁棒性非常有限。例如，在PE恶意软件末尾添加精心构造的字节序列，可以很容易绕过一些基于原始字节的恶意软件检测模型。

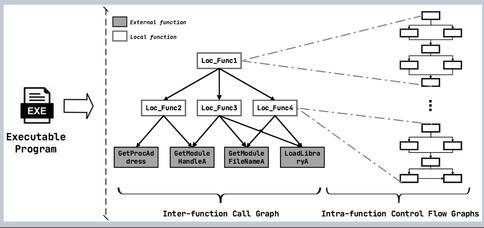
为了弥补当前恶意软件检测方法的局限性，本研究首次利用可执行软件的函数调用图和函数的控制流图构建可执行软件的**层次图表示**，并在层次图表示的基础上提出一种新的**基于层次化图神经网络的高鲁棒性Windows恶意软件检测方法**。

层次图表示方法

本研究提出一种新的可执行软件表示方法 - **层次图**表示方法，首次将可执行软件的函数调用图和函数的控制流图相结合，可以有效地获取其中丰富的语义信息和结构信息。

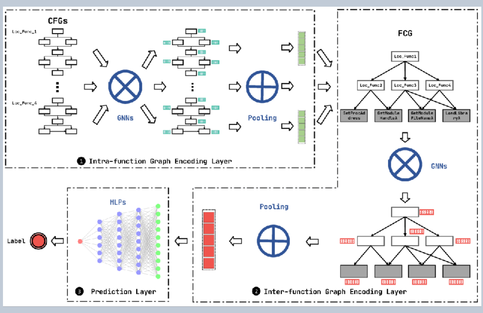
1. **函数间调用图**（Inter-Function Call Graph）：函数调用图FCG由多个函数（节点）以及函数之间的“调用者－被调用者”的关系（边）共同构成，从而表示每个可执行软件中不同函数间的交互信息。FCG中函数（节点）类型主要分为外部函数（通常是系统函数或者库函数）和内部函数（通常开发人员精心设计实现的函数）。为了兼顾准确性和效率，本研究采用混合策略对FCG中每个函数进行编码：FCG中代表**外部函数**的节点用函数名称的独热编码来表示，而代表**本地函数**的节点则被进一步反汇编采用函数内控制流图表示。

2. **函数内控制流图**（Intra-function Control Flow Graph）：为了进一步表达和学习**本地函数**内部的结构特征和语义信息，我们采用控制流图CFG的形式来表示每个本地函数。具体而言，CFG 由多个基本块（节点）以及基本块之间的控制路径（边）共同构成，在很大程度上表达了经过反汇编后本地函数的功能特征。



层次化图神经网络

在层次图表示方法的基础上，本研究提出了一种基于层次化图神经网络（MalGraph）的鲁棒Windows 恶意软件检测方法，如下图所示，MalGraph主要包括以下三个模块：



❶函数内图编码模块（Intra-function Graph Encoding Layer）：利用图神经网络和池化网络来学习一个可执行软件中每个本地函数对应的控制流图的全局图向量；

❷函数间图编码模块（Inter-function Graph Encoding Layer）：首先采用本地函数的全局图向量和外部函数名称的独热编码向量作为函数调用图中所有节点的初始化向量，然后利用图神经网络和池化网络来学习该可执行软件的向量表示；

❸恶意软件预测模块（Prediction Layer）：利用多层感知机网络来预测该可执行软件的恶意概率，并判断其是否为恶意软件。

实验分析

为了验证MalGraph在检测Windows恶意软件任务上的准确性和有效性，本研究在包含十八万多的恶意软件和善意软件的真实数据集上，系统地比较了MalGraph 与三个现有最好的恶意软件检测模型（即EMBER、MalConv、MAGIC）的准确性和鲁棒性。

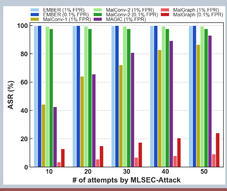
准确性评估

- 为了公平地比较TPR 和balance Accuracy （bACC），本研究在保证FPR 分别为1%和0.1%的情况下计算相应的TPR和bACC；
- MalGraph在同一测试数据集上表现出比所有基线模型更好的检测有效性，特别是在更严格的FPR=0.1%情况下，MalGraph能够检测出恶意软件的准确性远远高于所有的基线模型；

Models	AUC (%)	FPR = 1%		FPR = 0.1%	
		TPR (%)	bACC (%)	TPR (%)	bACC (%)
EMBER	99.89	99.35	99.18	83.53	91.72
MalConv-1	99.90	99.13	99.06	-	-
MalConv-2	96.07	14.68	56.84	2.24	51.07
MAGIC	98.48	90.81	94.91	-	-
MalGraph	**99.97**	**99.46**	**99.23**	**96.45**	**98.18**

鲁棒性评估

- 攻击成功率ASR越大，模型的鲁棒性越差；
- MalGraph在FPR=1%和0.1%的条件下所评估的ASR分别是9.2%和24.1%，远低于所有的基线恶意软件检测模型的ASR值；



研究总结

- ◆ 本研究首次提出层次图表示方法，将可执行软件的函数调用图和函数的控制流图相结合，有效地获取丰富的语义和结构信息；
- ◆ 本研究提出一种新的基于层次化图神经网络MalGraph的鲁棒性恶意软件检测方法，不仅可以有效地检测系统中的恶意软件，而且可以提高恶意软件检测模型的鲁棒性；
- ◆ 本研究在包含十八万多的恶意软件和善意软件的真实数据集上进行了广泛的实验评估，结果证明了MalGraph 比现有基线恶意软件检测模型展现了更好的准确性和鲁棒性；
- ◆ 本研究代码开源地址：https://github.com/ryderling/MalGraph

论文PDF请扫描　　代码地址请扫描