计算机研究与发展
Journal of Computer Research and Development
DOI:10.7544/issn1000-1239.2020.20180846
57(3):631-638,2020

基于最大频繁子图挖掘的动态污点分析方法

郭方方[1] 王欣悦[1] 王慧强[1] 吕宏武[1] 胡义兵[1] 吴 芳[1] 冯光升[1] 赵 倩[2]
[1](哈尔滨工程大学计算机科学与技术学院 哈尔滨 150001)
[2](哈尔滨商业大学计算机与信息工程学院 哈尔滨 150028)
(guofangfang@hrbeu.edu.cn)

A Dynamic Stain Analysis Method on Maximal Frequent Sub Graph Mining

Guo Fangfang[1], Wang Xinyue[1], Wang Huiqiang[1], Lü Hongwu[1], Hu Yibing[1], Wu Fang[1], Feng Guangsheng[1], and Zhao Qian[2]
[1](*College of Computer Science and Technology, Harbin Engineering University, Harbin* 150001)
[2](*College of Computer and Information Engineering, Harbin University of Commerce, Harbin* 150028)

Abstract The malicious code recognition method on traditional dynamic stain analysis technology has many problems such as huge number of malicious code behavior dependency graphs (MBDG) and long time of matching process.According to the common characteristics of each malicious code family, the behavior dependency graph is represented by some common sub graph parts. Therefore, this paper proposes a malicious code behavior dependency graph mining method based on maximum frequent sub graphs. The method mines the largest frequent sub graphs which can represent the significant common features of the family from the malicious code family behavior dependency graph. The maximum frequent sub graph that is mined can represent the most significant common feature among the variants of this type of malicious code. The target behavior dependency graph just needs to be matched with the largest frequent sub graph after mining.Besides, the method reduces the number of behavior dependency graphs and improves the recognition efficiency without losing the characteristics of malicious code behavior. Compared with the traditional dynamic stain analysis method for malicious code recognition, when the minimum support is 0.045, the number of behavior dependency graphs decreases by 82%, the recognition efficiency increases by 81.7%, and the accuracy rate is 92.15%.

Key words malicious code recognition; malicious code family; dynamic taint analysis; behavior dependency graph; maximal frequent sub graph mining

摘 要 目前,传统面向恶意代码识别的动态污点分析方法广泛存在行为依赖图数量巨大、匹配时间消耗长的问题.提出一种动态污点分析方法——基于最大频繁子图挖掘的动态污点分析方法.该方法从恶意代码家族行为依赖图挖掘出代表家族显著共性特征的最大频繁子图,被挖掘出的最大频繁子图即为某类恶意代码家族以及该家族所有变种之间最为突出的共有特征,使用挖掘出的最大频繁子图与被测

收稿日期:2018-12-24;**修回日期**:2019-04-17
基金项目:国家自然科学基金项目(61502118);黑龙江省自然科学基金项目(F2016028);中央高校基本科研业务费专项资金(HEUCF180602,HEUCFM180604);国家科技重大专项基金项目(2016ZX03001023-005)
This work was supported by the National Natural Science Foundation of China (61502118), the Natural Science Foundation of Heilongjiang Province of China (F2016028), the Fundamental Research Funds for the Central Universities (HEUCF180602, HEUCFM180604), and the Major National Science and Technology Program (2016ZX03001023-005).
通信作者:冯光升(fengguangsheng@hrbeu.edu.cn)

行为依赖图进行比较匹配即可.既能够保证原有恶意代码特征无丢失又削减了行为依赖图数量,并在此基础上进一步提升了识别效率.经实验分析,提出的这种新的动态污点分析方法相比于传统方法,当最小支持度为 0.045 时,行为依赖图数量减少了 82%,识别效率提高了 81.7%,准确率达到了 92.15%.

关键词 恶意代码识别;恶意代码家族;动态污点分析;行为依赖图;最大频繁子图挖掘

中图法分类号 TP393

近年来由于恶意代码攻击事件发生频率越来越高,危害性日益凸显.高效率的恶意代码识别方法对于保障主机运行时安全必不可少.现有识别恶意代码方法按检测时代码是否在内存中真实运行可分成 2 类:1)静态分析方法;2)动态分析方法.动态分析方法相比较于静态分析方法提取到的特征更加准确.因为,即使经过混淆处理后的代码,代码在运行时特征是无法隐藏和改变的,具有更高的识别准确率.因此,使用动态分析方法识别恶意代码已成为研究热点.其中,最常用的技术是动态污点分析技术[1-2].动态污点分析方法首先通过污点标记、污点传播、应用程序接口(application programming interface, API)进行中间截取污点文件,污点文件就是把污点传播路径记录下来,最后使用得到的污点文件构建恶意代码的行为依赖图(malicious code behavior dependency graph, MBDG).目前具有代表性的动态分析工具包主要有 CWSandbox[3], TTAnalyze[4], Norman Sandbox[5], Anubis[6]等.

目前,国内外研究重点关注的对象是动态污点分析方法的实现问题.Mchaisen 等人[7-8]提出一种 AMAL 分类器,该分类器能够根据恶意代码的行为进行自动的分析.但该分类器忽视了程序的控制依赖关系,易造成漏报现象.Fattori 等人[9]设计了一套恶意代码行为检测系统 AccessMiner,实时检测恶意行为,但该系统存在数据量过大、匹配时间过长的问题.Alam 等人[10-11]提出了一种 MARD 框架,该框架主要是针对各种变异恶意代码的检测,能够对变异恶意代码实时快速检测.此外,他们还提出了一种 SWOD-CFWeight 方法,该方法能够根据恶意代码的控制流语义对恶意代码进行实时捕获.但该方法依然存在存储空间消耗过大问题.Ghiasi 等人[12]提出一种基于寄存器内容的恶意代码检测框架,但该框架受环境限制较大且存在存储空间爆炸问题.Salehi 等人[13]使用分类算法对恶意代码的 API 调用和参数一起作为输入特征对恶意代码进行检测,但该方法提取的特征较少,准确率不高.Qiao 等人[14]假设频繁 API 调用序列可以准确反映恶意代码行为,通过聚类算法探索恶意代码之间的内部规律,此外,提出了一种使用聚类和分类相结合的技术来识别恶意代码,该方法主要是计算各个二进制形式的文件来进行预测[15].Toderici 等人[16]提出一种 Chi-Squared 方法,将隐 Markov 模型与卡方检验的统计框架相结合,用来检测变形恶意代码.但这 2 个方法都存在存储空间消耗过大的问题.

综上所述,目前的研究成果对依赖图数量多、存储空间消耗过大的问题描述较少甚至未加说明,而此问题已经逐渐成为影响动态污点分析技术发展的重要瓶颈.因此,以减少依赖图数量为切入点,力图保证特征完整性,并削减行为依赖图数量,从而进一步提高识别效率.

1 行为依赖图的定义与生成

本文提出了基于最大频繁子图挖掘的动态污点分析方法.即根据每个恶意代码家族具有共性的特点[17-19],把每个恶意代码家族的最大频繁子图挖掘出来,各个恶意代码家族的依赖图数量会大幅度减少.恶意代码家族及其变种恶意代码的主要特征都包含在挖掘出的最大频繁子图中,待测代码只需和挖掘出的这些最大频繁子图进行匹配就可得出结论,这样提高了识别效率.最大频繁子图方法采用基于生成树的挖掘(spanning tree based maximal graph mining, SPIN)方法.下面对该方法进行详细介绍.

1.1 行为依赖图定义

在介绍具体方法之前,先给出行为依赖图的定义:$G_{beh}=\{V,DE,CE,\varphi,L\}$,$G_{beh}$表示行为依赖图.其中,$V$ 表示图的顶点,DE($DE\subseteq V\times V$)表示数据关联边,控制关联边用 CE($CE\subseteq V\times V$)表示,标号集为 φ,集合内部有相应 API 名称、必要的输入性参数、某些输出参数和结果的返回值,L 为 V 和 φ 间的一种映射关系,具体为 $L:V\rightarrow\varphi$.动态污点分析方法主要依靠分析污点文件,污点文件中包括在执行一段恶意代码过程中,该段代码在执行过程中调用的重要 API 以及相应重要指令返回值,当执行一

段程序后,根据污点文件生成许多行为依赖图,这些依赖图被记入一个集合中,称为总行为依赖图集合,表示为 GG,在这里把集合 GG 记为

$$GG=\{G_{beh1},G_{beh2},\cdots,G_{behi},\cdots,G_{behn}\},1\leqslant i\leqslant n.$$

1.2 生成行为依赖图

生成行为依赖图包括 4 个主要部分:顶点添加部分、数据关联边添加部分、控制关联边添加部分和生成结束判断部分.其中行为依赖图采用邻接矩阵的形式存储,而顶点间的数据关联边用 1 表示,控制关联边用 2 表示,无相应依赖边用 0 表示.详细步骤描述为:

1) 添加顶点.首先分析污点文件,对于某一 API 来说,污点参数存在该 API 中,那么该 API 为邻接矩阵中的一个顶点.

2) 添加数据关联边.遍历 $Dlist$ 双向链表,查看污点传递路径,将会得到某 2 个 API 间的调用 T_i^{API} 和 T_j^{API}.若 T_i^{API} 和 T_j^{API} 存在数据关联关系且 T_i^{API} 调用 T_j^{API},则在邻接矩阵中 T_i^{API} 和 T_j^{API} 间记录 1,表示它们两者间存在数据依赖关系.

3) 添加控制关联边.某段代码在运行时,会出现新的 API 被调用的情况,这时必须考虑这个 API 是否存在于某个污点数据使用控制转移指令所能抵达范围之内.如果使用控制转移指令可以到达,那么邻接矩阵某对应位置记为 2,代表控制关联关系存在于它们之间.

4) 判断结束条件.当有 2 种情况出现时,行为依赖图被终止生成:①污点文件分析完成.对污点文件不断分析,行为依赖图中的顶点数量以及边的数量都不断增多.当某污点文件被分析完成,邻接矩阵中所有空位被补上 0,据此邻接矩阵生成最终行为依赖图.②当未污染的数据重新覆盖目前已存在的污点数据时.

由邻接矩阵生成对应行为依赖图的方法由算法 1 表示:

算法 1. 生成行为依赖图.

输入:邻接矩阵 $\boldsymbol{A}$;

输出:行为依赖图 G.

① for each API in $\boldsymbol{A}$
② 向 G 中添加顶点;
③ end for
④ for each T_i^{API} in $\boldsymbol{A}$ do
⑤ for each T_j^{API} in $\boldsymbol{A}$ do
⑥ if $((T_i^{API},T_j^{API})=1)$ then
⑦ 在 T_i^{API} 和 T_j^{API} 间添加数据依赖边;
⑧ end if
⑨ if $((T_i^{API},T_j^{API})=2)$ then
⑩ 在 T_i^{API} 和 T_j^{API} 间添加控制依赖边;
⑪ end if
⑫ end for
⑬ end for

2 最大频繁子图生成

最大频繁子图生成算法是核心与重点,其名称是 SPIN-MGM(MBDG mining method on spin).该方法首先从获得的行为依赖图集中使用 FFSM(fast frequent subgragh mining)方法获得候选频繁子图集,再通过扩展方法进行候选数据关联边和控制关联边的添加生成最大频繁子图集.

2.1 FFSM 方法

FFSM 方法使用规范邻接矩阵 CAM(canonical adjacency matrix, CAM),通过该矩阵能够得到规范编码,通过使用 CAM 矩阵就不需要直接计算同构子图,既可以唯一的表示图,又能降低时间复杂度.要想得到有向图的规范编码需要遍历邻接矩阵,这是因为行为依赖图是有向图.

FFSM 方法首先使用 $FFSM\_Join$ 函数和 $FFSM\_Extension$ 函数产生候选子图.其次,对产生的候选子图进行剪枝,这是为了筛选掉非频繁子图以及 CAM 不是次优的子图,最终达到减少候选子图数量的目的.FFSM 方法具体过程由算法 2 表示:

算法 2. FFSM.

输入:行为依赖图集 GG、最小支持度 SUP_{min};

输出:候选频繁子图集 W.

① $S\leftarrow${频繁顶点的 CAM 集合};
② $P\leftarrow${频繁边的 CAM 集合};
③ for each $\boldsymbol{p}\in P$ do
④ if ($\boldsymbol{p}$ is CAM) then
⑤ $W\leftarrow W\cup\{\boldsymbol{p}\},D\leftarrow\varnothing$;
⑥ for $\boldsymbol{q}\in P$ do
⑦ $D\leftarrow D\cup FFSM\_Join(\boldsymbol{p},\boldsymbol{q})$;
⑧ end for
⑨ $D\leftarrow D\cup FFSM\_Extension(\boldsymbol{p},\boldsymbol{q})$;
⑩ 剪枝去掉候选子图中非频繁子图或 CAM 不是次优的;
⑪ $FFSM(D,W)$;
⑫ end if
⑬ end for

2.2 SPIN-MGM 方法

SPIN-MGM 方法对由 FFSM 方法产生的候选频繁子图集进行剪枝和扩展处理来获得最大频繁子图集.SPIN-MGM 方法由算法 3 表示:

算法 3. SPIN-MGM.

输入:行为依赖图集 GG、最小支持度 $SUP_{\min}$;

输出:最大频繁子图集 EG.

① $Trees \leftarrow \{T \mid T$ 是 GG 中的一棵频繁树$\}$;

② $F \leftarrow \{EG \mid EG \in Expansion(T)$ and $T \in Trees\}$;

③ return $\{EG \mid EG \in F$ and EG is maximal$\}$.

其中扩展方法 Expansion 由算法 4 表示:

算法 4. 扩展方法 Expansion.

输入:频繁子树 T;

输出:频繁子图集 EG'.

① $D \leftarrow \{de \mid de$ 是 T 中的候选数据关联边$\}$;

② $F \leftarrow SearchGraphs(T, D)$;

③ $C \leftarrow \{ce \mid ce$ 是 T 中的候选控制关联边$\}$;

④ $F \leftarrow SearchGraphs(T, C)$;

⑤ Return$\{EG' \mid EG' \in F, EG'$ 是频繁的,EG' 和 T 有相同的正则生成树$\}$.

其中 SearchGraphs 方法由算法 5 表示:

算法 5. SearchGraphs 方法.

输入:频繁子树、候选控制依赖边(T, C)或候选数据依赖边(T, D);

输出:M.

① $M \leftarrow \varnothing$;

② for each $e_i \in C$ or each $e_i \in D$ do

③ $M \leftarrow M \cup SearchGraphs(G \oplus e_i, \{e_{i+1}, e_{i+2}, \cdots, e_n\})$;

④ end for

⑤ return M.

2.3 最大频繁子图的应用

最大频繁子图的作用主要体现在实际匹配过程中.设计的匹配方法主要实现过程是:挖掘出所有恶意代码家族行为依赖图中的最大频繁子图后,生成特征库,然后实时跟踪恶意代码,绘制出依赖图.通过匹配方法来得到行为依赖图之间的相似度.

匹配方法将目标行为依赖图 G_{target} 与特征库中行为依赖图集 GG 中每个最大频繁子图都进行匹配,得到 1 组匹配结果,得到的结果被存储到数组中.当进行匹配时,对数组进行一次遍历,遍历得到的最大值就是最终的匹配结果.由算法 6 表示:

算法 6. 匹配方法.

输入:GG,G_{target};

输出:匹配结果.

① for each $g \in GG$

② $m \leftarrow 0$;

③ $n \leftarrow 0$;

④ $i \leftarrow 0$;

⑤ for each $e \in g$

⑥ if $e \in G_{\text{target}}$

⑦ $m++$;

⑧ else

⑨ $n++$;

⑩ end if

⑪ end for

⑫ $array[i++] \leftarrow m/(m+n)$;

⑬ end for

⑭ 返回数组中最大的值.

3 实验结果与分析

3.1 实验指标

实验对提出的基于最大频繁子图挖掘的动态污点分析方法进行实验,并通过实验数据对该方法可行性进行研究.此外,与传统面向恶意代码识别的动态污点分析法以及文献[9,16]中提到的一些方法进行性了对比实验,突出本文所提出方法的有效性.

实验结果的测量使用准确率、误报率、漏报率、最小支持度和识别时间进行度量.这些度量标准主要是由下列 4 个检验指标来计算,它们的含义如表 1 所示:

Table 1 Inspection Metric

表 1 涉及的检验指标

Metric	*TP*	*TN*	*FP*	*FN*
Sample Type	Malice	Normal	Normal	Malice
Inspection Results Type	Malice	Normal	Malice	Normal

SUP,即为支持度,在这里把图集设定为 $GG = \{G_1, G_2, \cdots, G_i, \cdots, G_n\}$,图 G_i 的支持度记为 $A_{G_i}^{\text{SUP}}$,当图集 GG 中某个图 G_i 的支持度大于或等于最小支持度,即有 $A_{G_i}^{\text{SUP}} \geqslant SUP_{\min}$ 时,则该图为频繁子图.1 个图集中某子图满足最小支持度并且是最大的公共部分,该子图被称为最大频繁子图.

$$A_{G_i}^{\text{SUP}} = \frac{GG \text{ 中与 } G_i \text{ 子图同构的图数}}{GG \text{ 中总的图数}}. \quad (1)$$

PR(precision rate),代表能够按样本真实类别分类的样本个数占总样本个数的比率,值越高,代表方法分类效果越好:

$$PR=\frac{TP+TN}{TP+TN+FN+FP}\times 100\%. \quad (2)$$

FPR(false positive rate),代表把正常样本错误地识别为恶意样本的比率,对样本进行了误判.如果误报率越低,那么代表这个方法识别效果就越好:

$$FPR=\frac{FP}{FP+TN}\times 100\%. \quad (3)$$

FNR(false negative rate),是方法做出了错误的判断,把原本为正常的恶意样本,错误地识别为恶意的.误报率越低代表误判的样本越少,方法效果更好:

$$FNR=\frac{FN}{FN+TP}\times 100\%. \quad (4)$$

识别时间,表示从方法开始运行到识别实验所用全部样本所需的时间.

3.2 实验数据

实验所用的恶意代码的来源为 http://malware.lu 网站,该网站是一个巨大的恶意代码站,目前是由美国进行维护.该网站现有 4 963 698 个恶意代码样本.在这个网站上可以自由进行恶意代码的下载,并可根据恶意代码的名字和恶意代码的 Hash 值进行恶意代码搜索.

实验抽取了 6 类典型恶意代码家族,共 6 410 个恶意样本,具体如表 2 所示:

Table 2 Samples of Malicious Code Family

表 2 恶意代码家族样本

Malicious Code Family	Malicious Code Type	Variant Number	Sample Number
Mytob	Spam Worm	103	1 183
Netsky	Spam Worm	79	994
Allaple	Polymorphic Worms	120	1 169
Bagle	Spam Worm	66	1 063
Mydoom	Spam Worm	56	975
Agobot	Backdoor	54	1 026

所抽取的 6 类恶意代码家族是目前最突出且最具代表性的.这些恶意代码大多数以上都经过加壳处理,因此在进行识别之前,要先对恶意代码进行查壳与脱壳处理.

使用的正常样本数据均来自于常见的应用程序,并通过杀毒软件进行验证,如 360 云盘、QQ 音乐、网易云音乐、WPS、GoogleChrome、腾讯视频等,共 16 065 个.

3.3 实验结果分析

首先,针对 SPIN-MGM 方法的 SUP_{min} 进行实验分析,分析识别准确率 PR、误报率 FPR 和漏报率 FNR 随最小支持度 SUP_{min} 的变化情况,结果如图 1 所示:

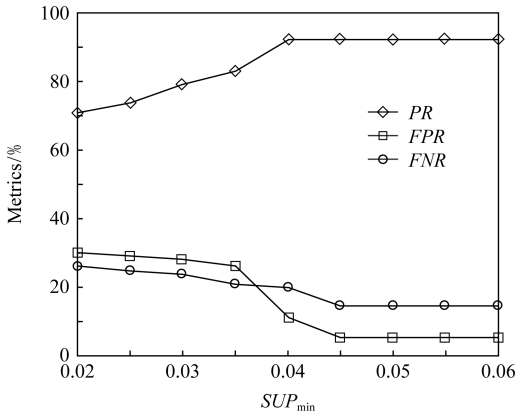

Fig. 1 Recognition results varies with SUP_{min}

图 1 识别结果随 SUP_{min} 变化情况

从图 1 可以看出,当 $SUP_{min}<0.045$ 时,准确率 PR 逐渐上升,漏报率 FNR 和误报率 FPR 都逐渐下降.这是由于支持度变大,SPIN-MGM 方法挖掘出的行为依赖图包含更多的共同特征;当 $SUP_{min}=0.045$ 时,PR 达到 92.15%,FPR 达到 5.64%,FNR 为 14.67%;当 $SUP_{min}>0.045$ 时,PR,FPR,FNR 基本保持平稳.当 $SUP_{min}=0.045$ 时,识别时间、PR、FPR 和 FNR 达到不错的结果.因此,据以上分析后,SPIN-MGM 的最小支持度设置为 0.045.

其次,研究 SUP_{min} 对识别时间和识别效率的影响.将最小支持度 SUP_{min} 依次设置为 0.02,0.025,0.03,0.035,0.04,0.045,0.05,0.055,0.06.如图 2 和图 3 所示,可以清楚地看到行为依赖图数量以及识别时间随最小支持度变化趋势.

从图 2 和图 3 中可以发现,当最小支持度 SUP_{min} 从 0.02 变化至 0.045 时,经过 SPIN-MGM 方法不断挖掘,行为依赖图数量不断地在减少.与此同时,识别时间不断削减.这种现象存在的原因是随着 SUP_{min} 逐渐增加,符合该支持度的最大频繁子图数量会越来越少,也即特征库中需要进行匹配的行为依赖图越少.因此,识别样本所需时间也越来越少;当 $SUP_{min}=0.045$ 时,行为依赖图数量为 216 个,识别时间约为 401 s;当 $SUP_{min}>0.045$ 时,行为依赖图数量和识别时间基本不在变化.由上面的分

析可以推断出这时挖掘出的行为依赖图数量不再变化.相比之下,传统未经挖掘的识别方法,其识别时间为 2 201.3 s.因此,可以得出结论,使用 SPIN-MGM 方法后可缩短样本识别时间,提高识别效率.

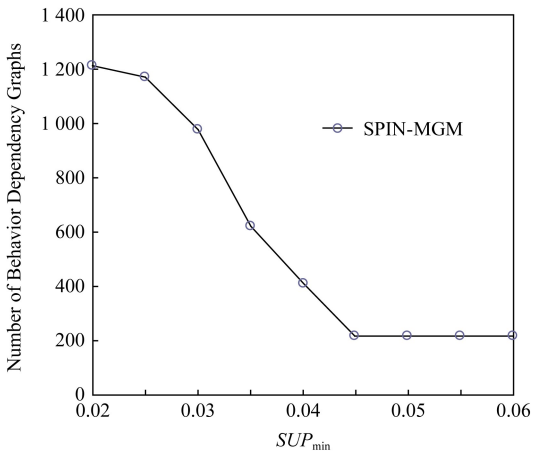

Fig. 2 Number of behavior dependency graphs varies with SUP_{min}

图 2 行为依赖图数量随 SUP_{min} 变化情况

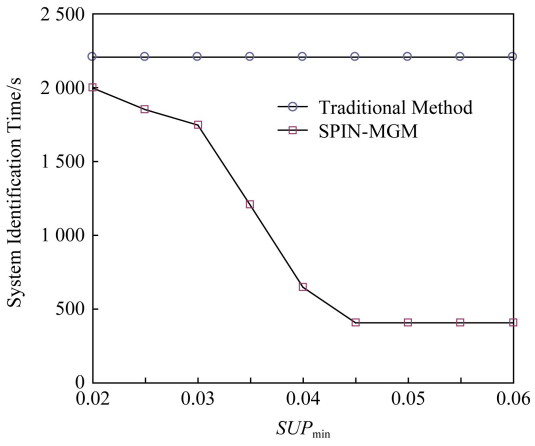

Fig. 3 Recognition time varies with SUP_{min}

图 3 识别时间随 SUP_{min} 变化情况

传统基于动态污点分析技术的恶意代码识别方法,以跟踪 API 参数的传播路径,找出 API 之间的依赖关系,挖掘出恶意代码的行为为目的,生成行为依赖图,对恶意代码进行分类.即使用 API 函数参数提取和基于函数参数的污点分析方法.但是,由于需要与恶意代码家族进行频繁的比对,导致这种传统的方法识别时间过长,识别效率较低.从图 2 可以看出当 $SUP_{min}=0.045$ 时,SPIN-MGM 方法特征库中的行为依赖图数量减少到 216 个.但是,传统行为依赖图污点分析方法中的行为依赖图始终为 1 210 个左右.因此,SPIN-MGM 方法也解决了文献[9,16]中传统动态污点分析方法存储空间消耗过大的问题.

此外,就准确率、误报率和漏报率把提出的 SPIN-MGM 方法与文献[9,16]中提到的一些恶意代码识别方法进行了性能对比,如图 4～6 所示.Access-Miner[9]是 1 个以系统为中心的行为恶意软件检测器,提供了 1 个通用的检测解决方案,不需要对恶意样本进行训练,因此它的准确率不及 SPIN-MGM.Chi-Squared[16]将隐 Markov 模型与卡方检验的统计框架相结合,用来检测变形恶意代码.SVM[20]是一种常见的分类学习方法,但是这种方法对变种恶意代码的识别准确率相对较差,因为该方法通过样本训练模型,一旦出现一种新型变种恶意代码,它的识别准确率就会降低.

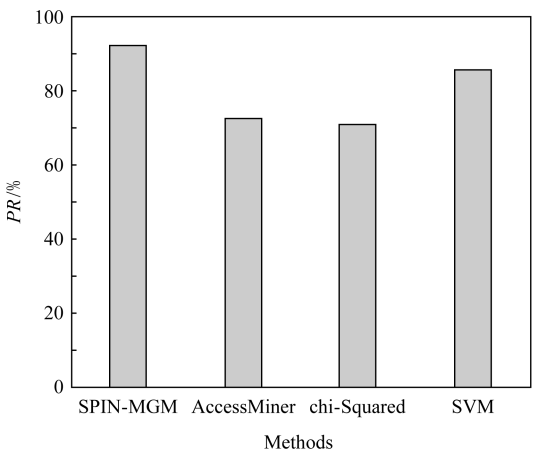

Fig. 4 Comparison of recognition accuracy PR of four analysis methods

图 4 4 种分析方法的识别准确率对比

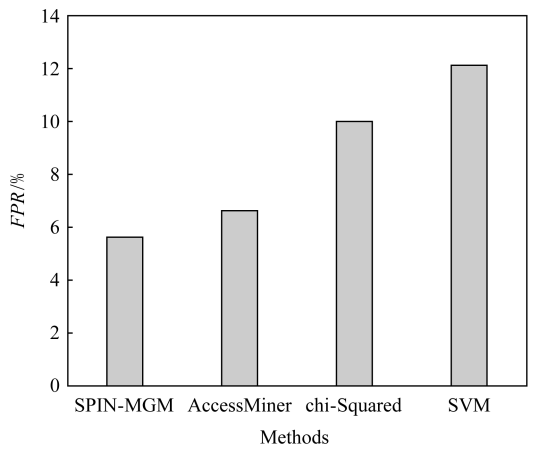

Fig. 5 Comparison of alarm failure FPR of four analysis methods

图 5 4 种分析方法漏报率 FPR 对比

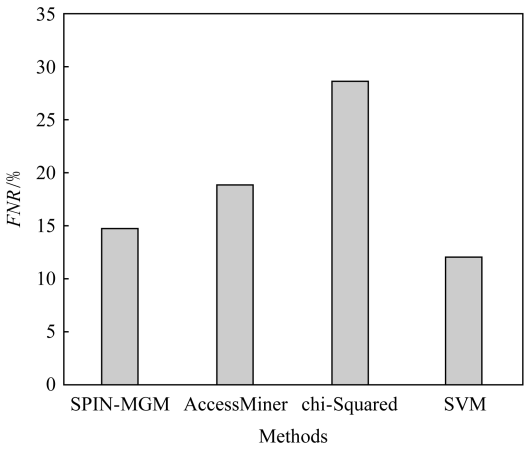

Fig. 6 Comparison of false alarm rate *FNR* of four analysis methods

图 6 4 种分析方法误报率 *FNR* 对比

从图 4～6 中可以看出,SPIN-MGM 方法在准确率、误报率上相比于 AccessMiner,Chi-Squared,SVM 方法达到了更好的效果,在漏报率上优于 AccessMiner 和 Chi-Squared 方法.SVM 在漏报率上低于 SPIN-MGM 方法.

4 结 论

恶意代码识别方法是抵御恶意代码攻击的重要方法.动态污点分析方法是现阶段研究的热点,但存在依赖图数量大的问题.据此,本文提出了基于最大频繁子图挖掘的动态污点分析方法,减少了行为依赖图的数量,解决了上述问题,提高了识别速度.1)SUP_{min}越大,满足该支持度的最大频繁子图越少,即特征库中行为依赖图越少,识别样本集时间也随之变短;2)支持度越大,SPIN-MGM 方法挖掘出的行为依赖图越能代表恶意代码家族间的共性特征;3)在本文的实验中,当最小支持度 $SUP_{min}=0.045$ 时,行为依赖图数量减少了 82%,识别效率提高了 81.7%,准确率 *PR* 达到 92.15%,误报率 *FPR* 达到 5.64%、漏报率 *FNR* 为 14.67%.下一步的工作将采用更多类型的恶意代码,进一步提升识别准确率.

参 考 文 献

[1] Fu Jianming, Tang Yi, Liu Xiuwen, et al. A memory boundary access detection framework based on dynamic blot [J]. Journal of Wuhan University, 2016, 62(5): 401-410 (in Chinese)
(傅建明, 汤毅, 刘秀文, 等. 一种基于动态污点的内存越界访问检测框架[J]. 武汉大学学报, 2016, 62(5): 401-410)

[2] Zhu Zhengxin, Zeng Fanping, Huang Xinyi. Dynamic symbolized blot analysis of binary programs [J]. Computer Science, 2016, 43(2): 155-158 (in Chinese)
(朱正欣, 曾凡平, 黄心依. 二进制程序的动态符号化污点分析[J]. 计算机科学, 2016, 43(2): 155-158)

[3] Joo J U, Shin I, Kim M. Efficient methods to trigger adversarial behaviors from malware during virtual execution in SandBox [J]. International Journal of Security and Its Applications, 2015, 9(1): 369-376

[4] Nogoorani S D, Jalili R. TIRIAC: A trust-driven risk-aware access control framework for grid environments [J]. Future Generation Computer Systems, 2016, 55: 238-254

[5] Seideman J D, Khan B, Vargas C. Quantifying malware evolution through archaeology [J]. Journal of Information Security, 2015, 6(2): 101-110

[6] Lebiere C, Bennati S, Thomson R, et al. Functional cognitive models of malware identification [C] //Proc of Int Conf on Cognitive Modeling (ICCM 2015). Groningen, Netherlands: ICCM, 2015: 9-11

[7] Mohaisen A, Alrawi O, Mohaisen M. AMAL: High-fidelity, behavior-based automated malware analysis and classification [G] //Information Security Applications. Berlin: Springer, 2014

[8] Jang J W, Yun J, Mohaisen A, et al. Detecting and classifying method based on similarity matching of Android malware behavior with profile [J]. Springerplus, 2016, 5(1): 273-296

[9] Fattori A, Lanzi A, Balzarotti D, et al. Hypervisor-based malware protection with AccessMiner [J]. Computers & Security, 2015, 52(1): 33-50

[10] Alam S, Qu Zhengyang, Riley R, et al. DroidNative: Automating and optimizing detection of android native code malware variants [J]. Computers & Security, 2017, 65: 230-246

[11] Alam S, Horspool R N, Traore I, et al. A framework for metamorphic malware analysis and real time detection [J]. Computers & Security, 2015, 48(2): 212-233

[12] Ghiasi M, Sami A, Salehi Z. Dynamic VSA: A framework for malware detection based on register contents [J]. Engineering Applications of Artificial Intelligence, 2015, 44(1): 111-122

[13] Salehi Z, Sami A, Ghiasi M. MAAR: Robust features to detect malicious activity based on API calls, their arguments and return values [J]. Engineering Applications of Artificial Intelligence, 2017, 59: 93-102

[14] Qiao Yong, He Jie, Yang Yuexiang, et al. Analyzing malware by abstracting the frequent itemsets in API call sequences [C] //Proc of the 12th Int Conf on Trust, Security and Privacy in Computing and Communications. Piscataway, NJ: IEEE, 2013: 265-270

[15] Ki Y, Kim E, Kim H K. A novel approach to detect malware based on API call sequence analysis [J].International Journal of Distributed Sensor Networks, 2015, 11(6): 4-13

[16] Toderici A H, Stamp M. Chi-squared distance and metamorphic virus detection [J]. Journal of Computer Virology and Hacking Techniques, 2013, 9(1): 1-14

[17] Han Xiaoguang, Yao Xuanxia, Qu Wu, et al. Malicious code family tagging based on image texture clustering technology [J]. Journal of PLA University of Science and Technology, 2014, 15(5): 440-449

[18] Wang Yi, Tang Yong, Lu Zexin, et al. Research on feature selection in malicious code clustering [J]. Information Network Security, 2016(9): 64-68 (in Chinese)
(王毅, 唐勇, 卢泽新, 等. 恶意代码聚类中的特征选取研究[J]. 信息网络安全, 2016(9): 64-68)

[19] Qi Fazhi, Sun Zhihui. Fast analysis method of malicious code based on feature threshold [J]. Computer Science, 2016, 43(S2): 342-345 (in Chinese)
(齐法制, 孙智慧. 基于特征阈值的恶意代码快速分析方法[J]. 计算机科学, 2016, 43(增刊 2): 342-345)

[20] Dong Lihua, Liu Qiang, Chen Haiming, et al. A lightweight real-time motion recognition algorithm based on time window [J]. Journalof Computer Research and Development, 2017, 54(12): 2731-2740 (in Chinese)
(董理骅, 刘强, 陈海明, 等. 一种基于时间窗口的轻量级实时运动识别算法[J]. 计算机研究与发展, 2017, 54(12): 2731-2740)

Guo Fangfang, born in 1974. PhD, associate professor of the College of Computer Science and Technology of Harbin Engineering University. His main research interest is network and information security.

Wang Xinyue, born in 1995. Master candidate. Her main research interest is network and information security. (13159829285@163.com)

Wang Huiqiang, born in 1960. PhD, professor of the College of Computer Science and Technology of Harbin Engineering University. His main research interets include network security and trusted computing. (wanghuiqiang@hrbeu.edu.cn)

Lü Hongwu, born in 1983. PhD, associate professor of the College of Computer Science and Technology of Harbin Engineering University. His main research interests include network and information security and mobile computing. (lvhongwu@hrbeu.edu.cn)

Hu Yibing, born in 1991. Master. His main research interest is network and information security. (huyibing@hrbeu.edu.cn)

Wu Fang, born in 1992. Master. Her main research interest include network and information security. (wufang@hrbeu.edu.cn)

Feng Guangsheng, born in 1980. PhD, professor of the College of Computer Science and Technology of Harbin Engineering University. His main research interets include mobile computing edge computation and IoT.

Zhao Qian, born in 1980. PhD, associate professor of the College of Computer and Information Engineering of Harbin University of Commerce. Her main research interest include trusted network and mobile computing.