doi:10.3969/j.issn.1000-436x.2013.08.019

基于特征聚类的海量恶意代码在线自动分析模型

徐小琳[1,2,3,4]，云晓春[1,2,3,4]，周勇林[4]，康学斌[5]

(1. 中国科学院 计算技术研究所，北京 100190；2. 中国科学院大学，北京 100049；3. 中国科学院 信息工程研究所，北京 100093；
4. 国家计算机网络应急技术处理协调中心，北京 100029；5. 安天实验室，黑龙江 哈尔滨 150040)

摘 要：针对传统海量恶意代码分析方法中自动特征提取能力不足以及家族判定时效性差等问题，通过动静态方法对大量样本行为构成和代码片段分布规律的研究，提出了基于特征聚类的海量恶意代码在线自动分析模型，包括基于 API 行为和代码片段的特征空间构建方法、自动特征提取算法和基于 LSH 的近邻聚类算法。实验结果表明该模型具有大规模样本自动特征提取、支持在线数据聚类、家族判定准确率高等优势，依据该模型设计的原型系统实用性较强。

关键词：恶意代码；在线自动分析；快速聚类；特征提取

中图分类号：TP393.08 **文献标识码**：B **文章编号**：1000-436X(2013)08-0146-08

Online analytical model of massive malware based on feature clusting

XU Xiao-lin[1,2,3,4], YUN Xiao-chun[1,2,3,4], ZHOU Yong-lin[4], KANG Xue-bin[5]

(1. Institute of Computing and Technology, Chinese Academy of Sciences, Beijing 100190, China;
2. University of Chinese Academy of Sciences, Beijing 100049, China;
3. Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China;
4. National Computer Network Emergency Response Technical Team/Coordination Center of China,
Beijing 100029, China; 5. Antiy Lab, Harbin 150040, China)

Abstract: In order to improve the effectiveness and efficiency of mass malicious code analysis, an online analytical model was proposed including feature space construction, automatic feature extraction and fast clustering. Our research focused on the law of malware behavior and code string distribution by dynamic and static techniques. In this model, a sample was described with its API and key code fragment. This model proposed a fast clustering approach to identify group samples that exhibit similar feature when applied this model to real-world malware collections. The result demonstrates that the proposed model is able to extract feature automatically, support streaming data clustering on large-scale, and achieve better precision.

Key words: malware; on-line analytical; fast clustering; feature extraction

1 引言

随着互联网的迅猛发展，伴随而来的网络安全局面日益复杂，网络用户受到的安全威胁日益加大，用户利益受到损害的安全事件不断增多，其中由恶意代码引发的用户信息泄漏、网银被盗、主机被控等问题层出不穷。恶意代码主要包括计算机病毒、蠕虫、木马、僵尸程序等。据迈克菲发布的 2012 年第四季度威胁报道，恶意代码不仅数量大幅上升，而且呈现变种多、更加智能化的特点，因此在

收稿日期：2013-05-07；**修回日期**：2013-06-30

基金项目：国家高技术研究发展计划(“863”计划)基金资助项目(2013AA014700)；国家科技支撑计划基金资助项目(2012BAH46B02)；中国科学院战略性科技先导专项基金资助项目(XDA06030200)

Foundation Items：The National High Technology Research and Development Program of China(863 Program)(2013AA014700)；The National Science and Technology Planning Project(2012BAH46B02); Strategic Priority Research Program of the Chinese Academy of Sciences(XDA06030200)

线自动分析恶意代码变得更加困难也更加重要。

恶意代码分析一直是业界最为关注的热点，病毒源代码公开、自动生成机的普及、黑客工具的肆意传播使得其已经形成灰色产业链，导致恶意代码数量爆炸式增长，为分析工作带来严峻挑战，主要表现在 2 个方面，一是如何能自动提取反映代码本质的特征，为下一步进行自动分析或人工分析提供更为全面描述信息；一是在面对实时出现的大量未知恶意代码，如何能更加快速对未知样本作出家族判定，从而提升处置速度或提高人工分析效率。

本文针对恶意代码分析工作中自动特征提取能力不足和家族判定时效性差问题，通过对大量样本的深入研究，提出了基于特征聚类的海量恶意代码在线自动分析模型(OAM, online analytical model of massive malware)。该模型主要由特征空间构建、自动特征提取及快速聚类分析 3 个部分组成，其中特征空间构建部分提出了基于统计和启发式的代码特征空间构建方法，自动特征提取部分提出了由 API 行为和代码片段构成的样本特征向量描述方法，快速聚类分析部分提出了基于位置敏感散列（LSH, locality-sensitive hashing）的快速近邻聚类算法。该模型可自动完成未知样本的行为及代码特征分析，并快速识别该样本是否为已知家族或新增家族。依据该模型设计的原型系统在海量样本数据集上进行了测试，结果表明可自动提取更为全面的行为和代码特征，家族判定准确率高，聚类速度快。

2　国内外研究现状

分析恶意代码的功能和危害是恶意代码分析的基础，目前主要有静态分析法和动态分析方法。静态分析方法指不运行恶意代码而是通过文件结构分析、反汇编、反编译等方法对恶意代码的可执行文件进行分析的方法。该方法可以了解恶意代码的程序流程和功能，获取用于检测和查杀恶意代码的静态特征。动态分析方法指在一个可控的环境内运行恶意代码，分析恶意代码与运行环境之间交互行为的方法。该方法通过捕捉环境在运行恶意代码前后发生的变化，在不同层次上给出恶意代码的指令或系统调用描述，从而近似还原恶意代码实际功能。Egele[1]总结并分析了当前主流的动静态分析工具，例如静态分析方法常用的反汇编工具 IDA 和 SAFE 开发的 PE 文件解析器，动态分析方法常用的 Anubis、CWSandbox、Norman Sandbox、Joebox 等均可以从系统调用、API 操作、文件系统操作等角度对恶意代码进行分析。

为了能从本质刻画恶意码特点和行为模式，研究人员采用了不同的恶意代码特征提取方法用于表示恶意代码样本。Kephart Griffin[2]等人最早提出了从样本提取二进制字符串作为样本特征的方法，该方法基于 5-gram 马尔可夫模型选出 48 byte 字符串代表恶意代码并用其检测终端是否感染恶意代码。SATHYANARAYAN V S、谷歌[3,4]等提出用 API 作为恶意代码特征的方法，即通过分析源码或者监测系统调用提取恶意代码样本的 API 集合或序列，辅以调用次数或参数构成表示样本的特征向量。Kolbitsch Clemens[5]提出用指令间调用关系作为表示样本特征的方法。随着恶意代码网络行为的增多，一些研究人员采用网络行为[6,7]作为恶意代码特征。Morales[8]针对 2 000 余个样本分析了 DNS、NetBIOS、TCP、UDP、ICMP 等网络协议，分析发现其和正常代码的网络行为有较大不同。Perdisci[9]则捕获恶意代码发出的 HTTP 请求并从中提取 URL 结构信息作为恶意代码特征。

随着恶意代码的不断增多和演变，为了便于描述和定性，主流安全厂商对其可检测的恶意代码均赋予一个名字，虽然各自的命名规则不同，但其相同点都是将有类似行为或特点的恶意代码归为一个家族，即大量恶意代码间存在类别关系[10,11]。Rieck Konrad[12]等人针对新增代码通过 SVM 分类器判断其是否属于已知家族或不属于恶意代码。SATHYANARAYAN V S [13]通过统计恶意代码和白代码样本关键 API 及其频度的卡方值，从而判断未知代码是否为恶意代码。

随着恶意代码数量的急剧增加，以前主要依靠人工分析的方法无法应对快速检测处置的现实需求，因此出现了一些融合实时捕获、个体分析、群体聚类或家族特征提取功能为一体的自动分析系统[14]。Yanfang[15]等人设计实现了 AMCS 系统用于恶意代码自动分类，同时根据分类结果能自动生成家族特征用于主机检测。该系统通过静态分析方法提取样本指令序列及频度，通过集成 tf-idf 及 k-medoids 的聚类方法实现分类并提炼出家族特征。Perdisci[9]设计了一个针对具有 HTTP 行为的恶意代码进行自动聚类的系统，该系统提取 URL 行

为中的结构字段作为代码特征，采用单链接层次聚类算法对代码进行聚类。Cesare 等人[16]设计了 Malwise 系统，该系统利用信息熵检测恶意代码是否加壳，针对脱壳后的代码从中提取控制流程图作为样本特征，再通过近似图匹配算法实现恶意代码的分类。

以上方法或系统均选取了不同的恶意代码特征提取方法，并通过分类或聚类方法实现未知代码的家族判定，但也存在一些问题，主要表现在：1) 提取的特征过多依赖人工经验，在适应更多尤其是未知恶意代码类型时有一定局限性；2) 提取方法和流程较为复杂不利于将其自动化，无法较快完成大规模样本的自动特征提取；3) 大多数分类或聚类算法不适合海量高维的样本集，无法快速完成大规模未知样本的家族判定。针对上述问题，在分析了大量恶意代码后，提出了恶意代码特征由 API 行为和代码片段构成，其中 API 行为作为决定恶意代码功能的主要因素往往也是区分不同类别（即家族）的主要因素，而代码片段特征由于其反应了代码编写环境或编写习惯则在更精确地区分变种方面有较好表现。针对 API 行文和代码特征，基于动态行为和 DUMP 文件分析方法，结合统计技术，设计了自动化程度更高的特征提取算法。由于需要对大规模的未知样本进行家族判定，采用聚类分析方法就要求尽可能只对数据处理一次，因此要求聚类算法具备较低的时间空间复杂度、能够处理高速数据流[17~19]，在 Bayer 等人提出[20]高速层次聚类算法基础上，结合划分聚类提出了基于 LSH 近邻聚类算法。综合上述三点，本文设计了基于特征聚类的海量恶意代码在线自动分析模型。

3 模型设计及特征构建

3.1 系统架构

OAM 模型主要包括自动特征提取、特征空间构建、快速聚类分析 3 个主要模块。

自动特征提取模块：该模块自动完成单个样本的特征提取，主要采用动态分析方法，即在虚拟机中执行恶意代码，通过在 Ring0 和 Ring3 层上的 API hook 机制获取 API 及部分参数集合，同时针对 dump 文件通过信息熵值计算提取代码片段。

特征空间构建模块：该模块完成整体样本的特征空间构造，针对海量黑白样本进行无差异化的行为和代码片段提取，通过概率统计来量化每个特征在黑白样本集中的区分度，从而确定提取的行为特征集合和代码片段的熵值范围。

快速聚类模块：该模块完成未知样本的实时聚类，针对新增样本通过 LSH 快速定位其最近邻，再计算其与最近邻所在类的距离，如果有超过阈值的距离将新增样本归为其所在类，否则形成新类。

OAM 模型的框架如图 1 所示。

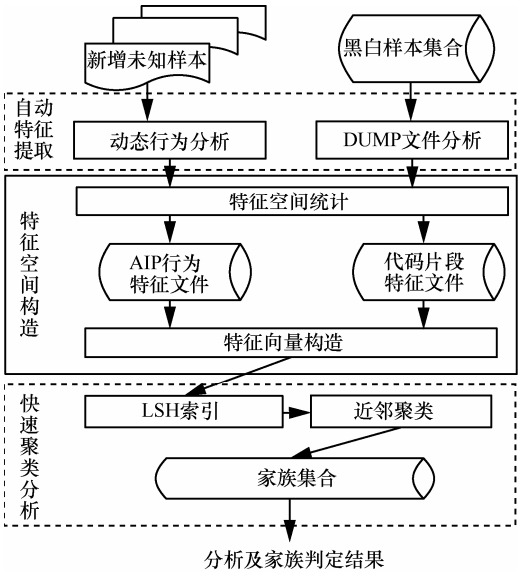

图 1 基于特征聚类的在线自动分析模型

3.2 特征空间构造

通常恶意代码的行为特征主要依靠启发式方法依靠人工和经验总结，其确定的可疑 API 相对固定也缺乏在大数据集上的验证。为了能更加合理地确定 API 行为集合和代码片段，基于统计方法设计了通过大量黑白样本集进行特征集合自动选择的构造方法。

1) 针对行为特征采用的构造方法

定义 1 白样本为一个软件其静态特征与其所在的系统环境及其活动对系统不会产生威胁，不会对系统造成破坏、没有信息窃取行为的软件样本。

定义 2 黑样本为一个软件对其所在环境如系统和网络产生威胁，比如常见的病毒、木马蠕虫、间谍件、广告件等。

定义 3 样本特征 S 是指所有行为特征和关键代码片段特征的集合，s 指集合中的某个元素。

定义 4 $PM(s)$ 指某个特征 s 在黑样本集合 M 中出现的概率。

定义 5　$PT(s)$ 指某个特征 s 在白样本集合 T 出现的概率。

定义 6　A 表示某个样本的特征集合 $A=\{s_1,s_2,s_3,\cdots\}$，$s_i \in S$,S 表示向量空间中所有特征的集合。

$$\sigma_i = (PM(s_i) - PT(s_i))/(PM(s_i) + PT(s_i)) \quad (1)$$

σ_i 为该特征在白样本集与黑样本集中的差异度。若为正，则表明该特征更多的出现在黑样本中，若 σ_i=1，则该特征只出现在白名单集中。大量数据集可提出的总行为数量很多，在实现中对大量的黑白名单样本集合进行了统计，丢弃了 σ_i 值接近 0 的特征，因为这样的特征在恶意样本集和白名单样本集中差异很小，不能作为分类依据且获取该类特征需要耗费时间，所以通过统计确定出了区分度大的特征即满足 $\sigma_{ij}<-0.2 \cup \sigma_{ij}>0.2$ 的 S_i。

2) 针对关键代码片段特征采用的构造方法

提取代码片段的方法借鉴了语言统计模型思想，代码片段即二进制文件里一段连续的二进制串。其想法源于静态分析方法，源代码分析是静态分析中最常用的方法，分析人员通过 IDA 进行反汇编后查看字符串可以迅速了解程序的流程和功能，但是该方法依赖反汇编手段，针对加壳样本或非 PE 文件的样本等问题都无法很好解决。为了能够尽可能自动提取可用于聚类的代码片段，基于统计语言模型的思想，自动提取样本中由 26 个英文字母和数字组成的连续字符串，分别计算其在可理解和难理解字符串集合中的熵值。易理解代码片段和难理解代码片段集合通过选取一定数量的文件进行代码片段提取，然后对结果进行人工标注。通过实验发现易理解代码片段熵主要分布在 2~5 之间，表明熵值过低的代码片段确实无意义，但熵值过高也不是易于理解的，因此在提取时选取熵值在 2～5 之间的易理解代码片段。

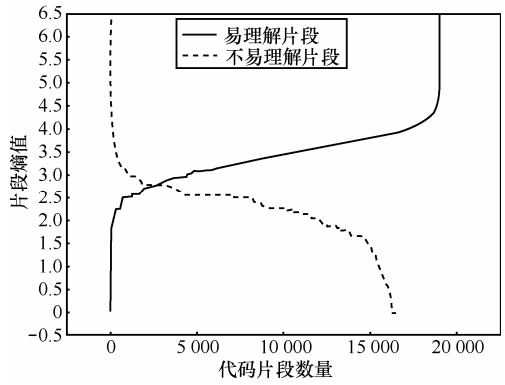

图 2　代码片段的熵值分布

3.3　样本特征向量化

根据上述方法，在十万规模的黑白样本集上针对 API 行为进行训练后，遴选出相对固定的 API 特征空间集合。针对代码片段在万级规模的样本机上训练后，确定熵的取值范围。比如样本 A 和样本 B 的字符串提取结果分别为 A（TObject，Sender，Integer，kernel32.dll，SelfDel）、B（Integer，kernel32.dll，Boolean，Height，GetProcAddress，SelfDel），X(kernel32.dll，Boolean，Height，GetProcAddress，SelfDel)，由 API 行为和代码片段构成样本的描述文件。

由于描述文件不利于后续聚类计算，因此针对所有字符串形式的 API 和代码段采用 CRC64 进行转换排序构成总的向量空间，每个样本的描述文件可以转换成 01 的向量。实际计算时由于其占用空间过大，实际采用索引形式向量。那么由 A、B 构成的总结果集将以“字符串－索引”形式存储，即（TObject－0，Sender－1，Integer－2，kernel32.dll－3，Boolean－4，Height－5，GetProcAddress－6，SelfDel-7， copyself-8）。

向量形式：A111100010、B001111110、X000111110。

索引形式：A (0, 1, 2, 3, 7)、B (2, 3, 4, 5, 6, 7)、X (3, 4, 5, 6, 7)。

4　快速聚类算法

4.1　算法描述

算法的基本思想是：每一个样本和与之最近的样本属于同一家族的概率最大。针对新增样本在已聚类样本集中查询其最近邻，计算新增样本与其所有最近邻所在家族的距离，其中最短距离如果超过设定阈值就将新增样本归为最短近邻所在家族，否则形成新家族。主要步骤如下。

step1　根据新样本特征向量计算 LSH 值。

step2　根据 LSH 值查询其最近邻集合。

step3　分别计算新样本与集合中每个样本所在类的距离。

step4　比较得到最小距离，如果其超过阈值，将新样本归为最近相似样本所在类，否则单独形成一个新类。

step5　重新计算发生变化的类中心点。

伪代码如下。

输入：新样本特征向量 FV_1，FV_2，…，FV_n。

输出：聚类后的簇和各簇的中心点。

Begin

for i = 1 to n
LSH(FV_i) = string → vector
for each FV_i do
NearestNeighbor.add (FV_i)
for each NearestNeighbor$_k$
$d_{\max}$ = Jaccard(S, Cen_1)
for j = 2 to n_2
{
if $d_{\max} \leqslant$ Jaccard(S, Cen_i)
$d_{\max}$ = Jaccard(S, Cen_i)
}
if $d_{\max} \geqslant \theta$
{
AddToCluster (S, $Cluster_i$)
（$Cen_i = S \cap Cen_i$ ）
}
else
{
$Cen_{n_2+1} = S$
CreatNewCluster (Cen_{n_2+1})
$n_2 = n_2 + 1$
}
}
end

4.2 LSH 函数描述

由于经过自动提取的样本API行为特征集合规模庞大，样本特征向量中仅 API 行为部分的维度超过 27 000，如果样本数量也较大时，采用分裂聚类法需要两两计算样本距离其计算复杂度是无法满足实时性的要求。LSH 算法的基本思想是基于散列函数无需两两计算即可快速得到某样本的近邻，从而大幅降低计算规模。需要根据度量样本相似度的距离标准选择适合的 LSH 函数，而该函数只要满足单调递增映射关系即可[21]。Charikar 提出了随机投影位置敏感散列函数族适用于高维数据同计算余弦距离判定相似度的场景[22]。

OMA 模型中确定 2 个样本是否为同一家族时通过计算其Jaccard距离的度量标准，即如果Jaccard距离小于 R 即为同一家族，否则为不同家族，由于基于抽样(bit sampling)的散列函数构建方法适用于原始特征向量每一维的取值为{0,1}的特征串，本文采用了这个方法。其散列函数是一组随机选取 d 维特征向量中的一个 K 维子向量。按照索引集中的每个索引，从每个结果对应的原始向量中取出对应的值（0 或 1），拼接成子向量。这样每个结果在一次随机子向量计算之后又会有一个对应的长度为 K 子向量，如果 2 个结果对应的子向量相同，即可认为 2 个向量相似；再选取 L 个这样的函数。例如样本 A、B 和 X 的原始 API 特征向量分别为：111100010、001111110 和 000111110，随机选取 L=4 的索引(3，4，7，8)作为子向量，A 的向量子集为 1010、B 的向量子集是 1110、X 子集为 1110，则 A 不是 X 的最近邻而 B 是 X 的最近邻。

4.3 算法分析

假设系统目前需要聚类的数据集总数 n、样本特征向量维度 d，针对每个向量求其 LSH 值的时间为 t，则预处理复杂度为 $O(nLkt)$；最近邻集合逐一计算 Jacarrd 距离的复杂度为 $O(ncd)$，其中，c 是最近邻集合的元素数目，由于随着样本集的不同 c 并不是一个固定值，但是实际测试时其远小于 n，因此总的复杂度 $O(nLkt + ncd)$，一般情况下远小于 $O(n^2)$，最坏情况下 c 趋于 n，即每个样本都差异过大独自聚成一类。

5 实验结果分析

5.1 实验样本集

该模型实验时主要用到两类数据集，基础数据集和测试数据集。基础数据集用于特征向量空间构建，主要包含白样本和黑样本，其中，白样本集主要从 Windows 系统提取相关文件，黑样本集主要通过反病毒厂商样本交换及网络蜜罐捕获积累的样本，主要是·exe 类型的文件，如表 1 所示。

表 1 基础数据集

名称	样本组成	数量
黑样本集	活体病毒库	838 864
白样本集	Windows XP 系统文件、软件	8 244
	Windows 2003 系统文件、软件	8 298
	Windows Vista 系统文件、软件	15 832

测试数据集用于测试该模型的分析效果，主要由黑样本集中的样本组成。为了保证实验效果，从黑样本库中随机抽取样本，如与基础数据集中黑样本一致(MD5 值相同)，则重新抽取，进行聚类测试的黑样本集共包含 19 534 个样本。

5.2　聚类结果分析

在基础数据集的对特征空间构造训练的基础上，用该模型设计的原型系统对测试样本集进行了聚类实验，由于目前尚无对家族分类有较为权威的判别标准，主要还是针对准确率进行比较。实验总共得到 905 个家族，识别率（即准确率）为 78%。同时为了对比验证本文的聚类效果，同时使用卡巴斯基扫描引擎扫描（引擎版本 13.0.1.4273，以下简称卡巴）对测试数据集进行了对照扫描，根据其命名信息进行了家族分类得到 483 个家族，其识别率为 69%，如表 2 所示，可见 OMA 模型的识别率高于卡巴斯基。但也可以看出 OMA 模型得到的家族数约为卡巴斯基的一倍，说明 OMA 模型在相似度阈值选为 0.5 的情况下，家族聚类比较细。

表 2　　聚类结果对比

聚类方法	聚类样本	未聚类样本	家族数	准确率
OMA	15 765	3 769	905	78%
卡巴斯基	13 385	6 149	483	69%

针对利用 OMA 模型聚类的结果进一步分析发现该聚类结果中有多个家族包含了被卡巴识别为不同家族或者无法被识别的样本，表 3 列举了 4 个典型样例。

编号为 1～3 的典型样例都包含 3 种被识别为同一家族的样本：被卡巴识别为一种家族、被卡巴识别为不同家族、卡巴无法识别。出于篇幅考虑选取案例 1 涉及的样本进行人工分析，选取样本如表 4 中所列。

从中抽取编号为 1.1、1.4 和 1.6 这 3 个样本进行人工分析发现它们确实具有相似行为。

1) 浏览器中间人劫持行为；

表 3　　聚类结果典型案例

案例号	聚类 ID	数量	卡巴识别	家族样本数量
1	174	71	Packed.Win32.Krap	9
			Trojan-Spy.Win32.Zbot	44
			NULL	17
2	303	25	Trojan.Win32.BHO (num : 10)	10
			Trojan-Dropper.Win32.Agent (num : 2)	2
			NULL (num : 13)	13
3	73	12	Trojan.Win32.Buzus (num : 7)	7
			NULL (num : 5)	5
			HEUR:Trojan.Win32.Generic	1
4	49	92	Trojan.Win32.StartPage.balfnot-a-virus	70
			AdWare.WinLNK.Clicker.c	4
			HEUR:Trojan.RAR.Clicker.a	3
			Backdoor.Win32.Ruskill.fix	1
			UPX'}}	1
			Virus.Win32.Neshta.b	1
			Trojan-Downloader.Win32.Agent.usxo	1
			Trojan.Win32.Hosts2.gen	1
			Trojan-Downloader.Win32.Small.cmj	1

2) 内联 hook 函数个数 56 个相同位置的 API；

3) 解密释放文件 Application Data 目录子目录下，随机命名；

4) 通过释放 BAT 文件自删除，bat 文件名 tmp（8 bit 随机）；

5) 读取自身；

6) 创建 3 个与磁盘卷相关的 Global\开头的 guid。

通过人工确认都应该是 zeus 家族的变种，该结果表明聚类对具有相似行为的样本更加准确的识别，能够把样本行为基本一致的样本划分到一个类别里，并能够识别杀毒软件未识别的变种。如果只从这些样本的二进制文件看其差别较大，通过简单的静态比较很难识别为一类。案例 2 和案例 3 的分析结果基本类似。

表 4　　典型案例 1 样本

样本编号	样本 MD5	卡巴对照命名
1.1	845DF5E5821477BDAE21B1F7BC90AD41	NULL
1.2	359971A23F67694CB20F6F107E245139	NULL
1.3	E94E167187F6BCC700CEE5A32295B967	Packed.Win32.Krap.iu
1.4	480C18B817BD344FE441BAFA7805508C	Packed.Win32.Krap.iu
1.5	C9D42C22F8F6B02C7C504E02443B9733	Trojan-Spy.Win32.Zbot.efaw
1.6	CE3FAEBDCFC702FCE49C943957D376F2	Trojan-Spy.Win32.Zbot.eibr
1.7	3AC709F98C7CBDAC073FA84BFE5C2D06	Trojan-Spy.Win32.Zbot.eeoj

当然在实验结果中也发现该分析模型存在一定误报率，比如案例 4 中也将多个卡巴命名为不同家族的样本聚为一类。针对该案例中的样本进行人工分析发现其主要是 Sfx 自解压格式类型文件，由于动态分析得到的 API 行为和代码片段比较相似，因此被聚为一类，但其实聚成不同类更为合理。分析后发现如果在行文特征提取是增加对释放体的描述信息才能解决这个问题。

5.3 性能测试

为了验证该聚类算法的复杂度远远低于 K 中心聚类，针对相同样本集同时采用 2 种方法对比其聚类速度，从图 3 可以看出随着总样本数量增加和家族数的增加，K 中心聚类方法的聚类速度不断下降，随着样本数量的增加逐渐趋于零，而 OMA 模型基本维持在匀速。

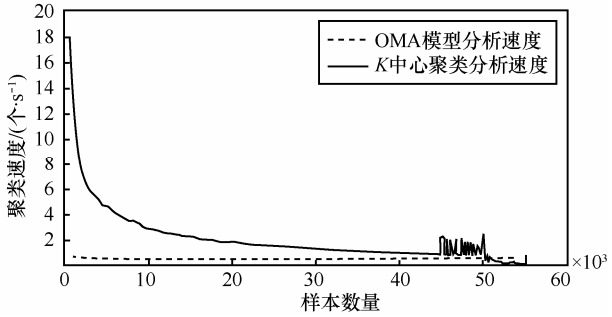

图 3 聚类效率对比曲线

6 结束语

本文深入研究了当前主要的恶意代码分析方法，针对其存在的问题，尤其是传统方法中自动特征提取能力不足以及家族判定时效性差等问题，提出了基于特征聚类的海量代码在线自动分析模型。该模型基于海量样本统计构建了适应性更好的特征空间，通过基于 LSH 的近邻算法实现了接近实时的聚类功能，这使得利用该模型设计的系统可客观快速地对未知样本进行自动行为分析和家族判定，从而可以大幅提高未知样本的分析效率，减少需要人工分析样本的数量。本文通过实验证明了基于该模型设计的原型系统在有大规模样本自动特征提取、支持在线数据聚类、家族判定准确率高等优势，依据该模型设计的原型系统实用性较强。

参考文献：

[1] 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): 1-42.

[2] KEPHART J O, ARNOLD W C. Automatic extraction of computer virus signatures[A]. Proceedings of the 4th Virus Bulletin International Conference[C] 1994.178-184

[3] SATHYANARAYAN V S, KOHLI P, BRUHADESHWAR B. Signature generation and detection of malware families[A]. Information Security and Privacy[C]. Springer Berlin Heidelberg, 2008. 336-349.

[4] SATISH S, PEREIRA S. Behavioral Signature Generation Using Clustering: WIPO Patent 2011137083[P]. 2011.

[5] KOLBITSCH C, COMPARETTI P M, KRUEGEL C, *et al*. Effective and efficient malware detection at the end host[A]. Proceedings of the 18th Conference on USENIX Security Symposium USENIX Association[C]. 2009. 351-366.

[6] RAMACHANDRAN A, FEAMSTER N. Understanding the network-level behavior of spammers[J]. ACM Sigcomm Computer Communication Review, 2006, 36(4):291-302.

[7] INOUE D, YOSHIOKA K, ETO M, *et al*. Malware behavior analysis in isolated miniature network for revealing malware's network activity[A]. IEEE International Conference on Communications[C]. 2008. 1715-1721.

[8] MORALES J A, AL-BATAINEH A, XU S, *et al.* Analyzing and exploiting network behaviors of malware[A]. Security and Privacy in Communication Networks[C]. Springer Berlin Heidelberg, 2010. 20-34.

[9] PERDISCI R, LEE W, FEAMSTER N. Behavioral clustering of HTTP-based malware and signature generation using malicious network traces[A]. Proceedings of the 7th USENIX Conference on Networked Systems Design and Implementation USENIX Association[C]. 2010. 26-26.

[10] RIECK K, HOLZ T, WILLEMS C, *et al.* Learning and classification of malware behavior[A]. Detection of Intrusions and Malware, and Vulnerability Assessment[C]. Springer Berlin Heidelberg, 2008. 108-125.

[11] BAILEY M, OBERHEIDE J, ANDERSEN J, *et al.* Automated classification and analysis of internet malware[A]. Recent Advances in Intrusion Detection[C]. Springer Berlin Heidelberg, 2007. 178-197.

[12] RIECK K, HOLZ T, WILLEMS C, *et al.* Learning and classification of malware behavior[A]. Detection of Intrusions and Malware, and Vulnerability Assessment[C]. Springer Berlin Heidelberg, 2008. 108-125.

[13] SATHYANARAYAN V S, KOHLI P, BRUHADESHWAR B. Signature generation and detection of malware families[A]. Information

Security and Privacy[C]. Springer Berlin Heidelberg, 2008. 336-349.

[14] BRANCO R R, SHAMIR U. Architecture for automation of malware analysis[A]. The 5th International Conference on Malicious and Unwanted Software (MALWARE)[C]. 2010. 106-112.

[15] YE Y, LI T, CHEN Y, *et al*. Automatic malware categorization using cluster ensemble[A]. Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining[C]. ACM, 2010. 95-104.

[16] CESARE S, XIANG Y, ZHOU W. Malwise——an effective and efficient classification system for packed and polymorphic malware[J]. IEEE Trans on Computer, 2013,62(6):1193-1206.

[17] MASUD M M. AL-KHATEEB T M, HAMLEN K W, *et al.* Cloud-based malware detection for evolving data streams[J]. ACM Transactions on Management Information Systems (TMIS), 2011, 2(3):1-27.

[18] PAULEVE L, JEGOU H, AMSALEG L. Locality sensitive hashing: a comparison of hash function types and querying mechanisms[J]. Pattern Recognition Letters, 2010, 31(11):1348-1358.

[19] KOGA H, ISHIBASHI T, WATANABE T. Fast agglomerative hierarchical clustering algorithm using locality-sensitive hashing[J]. Knowledge and Information Systems, 2007, 12(1):25-53.

[20] BAYER U, COMPARETTI P M, HLAUSCHEK C, *et al.* Scalable, behavior-based malware clustering[A]. Network and Distributed System Security Symposium (NDSS)[C]. 2009.8-11.

[21] INDYK P, MOTWANI R. Approximate nearest neighbors: towards removing the curse of dimensionality[A]. Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing[C]. 1998. 604-613.

[22] MIN K, YANG L, WRIGHT J, *et al.* Compact projection: simple and efficient near neighbor search with practical memory requirements[A]. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)[C]. 2010. 3477-3484.

作者简介：

徐小琳（1976-），女，陕西宝鸡人，中国科学院计算技术研究所博士生、高级工程师，主要研究方向为网络安全、数据分析等。

云晓春（1971-），男，黑龙江哈尔滨人，博士，中国科学院计算技术研究所研究员、博士生导师，主要研究方向为信息安全、计算机网络等。

周勇林（1974-），男，辽宁鞍山人，国家计算机网络应急技术处理协调中心高级工程师，主要研究方向为信息安全、计算机网络等。

康学斌（1982-），男，黑龙江哈尔滨人，安天实验室高级工程师，主要研究方向为恶意代码自动化分析行为、数据挖掘等。