**检测应用是否滥用权限**

**摘要：**

我们要如何知道一个程序是否滥用权限？通过基于API对应用进行聚类，我们通过每个聚类中的权限调用情况鉴别出异常应用。例如，一个使用了通讯录权限的天气应用、一个使用了地图定位权限的短信应用等等。本文提出了一个检测滥用权限的方法：首先，提取样本应用的API调用情况，为每个应用生成API特征向量。其次，根据应用的API特征向量对应用进行聚类。再次，在每个聚类中提取应用的权限调用情况，生成权限特征向量。最后，在每个聚类中使用SVM检测出滥用权限的应用。通过实验证明，该方法能够较为有效的检测出滥用权限的应用，其正确率可达71.1%。

**关键词：**

Android、API、应用权限

**1.引言**

**1.1智能手机安全现状**

智能手机上网的用户群正在快速增长，Android病毒木马产业链一开始就是以追逐经济利益为目的，使得Android恶意软件的行为更隐蔽。在Android手机平台，只有手机恶意广告让中招用户的感受很直接，Android后门、窃取隐私、后台推广软件等行为难以被普通网民察觉。

**1.2国内外研究现状**

国内在基于权限方面的智能手机安全研究上，朱佳伟、喻梁文等人对目前的权限机制做了相关的研究工作，介绍说明了权限机制的实现原理，并对其进行了有效性分析，提出了对权限机制的改进方案，以及未来关于不依赖用户进行安全决策的研究发展方向[1]。杨晶、金伟信等人提出了一套基于Android系统的权限配置方案，通过Permission Manager来对权限配置方案进行实现，当程序进行安装时，对配置权限设限，同时可以禁止高危权限运行，只有当用户授权使用高危权限时才可继续，该方法在一定程度上优化了Android的权限系统[2]。

国外的Di Cerbo Franeeseo和Girardello Andrea通过提取Android平台上应用程序的权限信息作为基础来发现恶意应用，并在Google电子市场上的应用程序进行测试取得了较好的效果[3]，这正是通过提取了Android应用文件的特征来对应用程序数据进行安全评估。

**1.3预设条件**

严格来说，应用的功能可能并不完全和其描述相同，除此之外，应用一般会动态申请权限，这在一定程度上会增加我们检测的困难程度。因此在这里我们假设：

(1)应用在APK中静态声明的API就是其实际会调用的API，并以此刻画应用的功能特征；

(2)应用在manifest中静态声明的权限就是其实际使用的权限。

**2.通过API进行聚类**

**2.1生成API特征向量**

提取出描述了每个应用API的调用情况的数据，即该应用调用了那些API以及API中的哪些函数。本文介绍的方法只关心应用是否调用了某个API，而非具体到调用API的次数或者函数个数。考虑到应用可能对API的名字进行混淆处理，这会影响我们程序的准确率和效率，因此需要尽可能筛除那些被混淆的API。另外，被调用次数过少的API也会在一定程度上影响准确率和效率。因此，我们先对每个API计算出调用它的应用个数，如果个数少于10，则不再将该API纳入后续流程的考虑之中。

在筛除被调用次数非常少的API后，我们分析每个应用调用其余API的情况，生成应用的API调用特征向量，这一特征向量的每个维度只有0和1两种赋值，意为未调用/调用对应的API。

**2.2根据API特征向量对应用进行聚类**

这里使用的工具是Kmeans算法：给定度量空间中的一组元素，并给出数字K表示聚类的个数，Kmeans会为每个聚类选择一个质心，然后为数据集中的每个元素选取距离其最“近”的一个质心，从而识别元素所属的聚类。在这里元素就是指第三步生成的应用API调用向量。

读取第二步得出的应用API特征向量表，根据每个聚类中平均距离以及总数据量，设定Kmeans的聚类数为15后直接对应用使用Kmeans算法进行聚类。

**3.SVM分类**

SVM分类是算法中的第二步，目的是在每个聚类中根据permission的调用情况找出异类，即使用了大多数程序没有使用的permission的程序。SVM分类主要分为以下几步：① 生成每个聚类中的程序的permission表；②生成每个的程序的特征向量；③用SVM模型进行训练；④选取训练集，用训练好的模型进行预测。

①提取permission的方法采用apkanalyzer，命令为”analyzer manifest permission“+程序路径。提取出的文件将放在同一个文件夹中。生成permission表时，遍历每个程序的permission表，将他们放在一起，去掉重复的项，就成为了permission表。

②生成特征向量：对于每个程序，找出它属于的聚类的permission表，并建立一个向量，维数等于permission表的表项数。接下来遍历里面的所有表项，与程序本身的permission列表进行比对，如果程序用到了第i个permission，就把向量的第i维设为1，否则设为0.这样会生成一系列由0,1构成的向量，作为该程序的特征向量。每个程序自身带有属于良性程序或是恶意程序的标记，1表示良性程序，0表示恶意程序，最后一步处理会把这个标记加到特征向量的最后一维。这样就把程序的permission特征转化成了向量，可以使用svm模型进行计算。

③&④这两个部分相对于前面处理数据的部分比较简单，我们对于每个程序聚类都会训练一个svm模型，然后对训练集中属于该类的程序进行预测，并计算准确度。

**4.实现和评估**

**1.数据集的规模**

程序的总数为4389个，其中有3400个良性程序和989个恶意程序。在数据集中随机挑选10%的良性程序和10%的恶意程序组成测试集（340个良性程序和98个恶性程序），剩下的作为训练集（3060个良性程序和891个恶性程序）。

**2.基于API调用的应用聚类**

2.1 API特征提取

针对训练集一共提取得到了6400种API，排除了被调用次数少于10次的API后剩下2339中API参与后续计算。

2.2 使用Kmeans进行聚类

聚类个数的确定主要参考两个方面：其一，是每个聚类中元素聚类中心点的距离平均值；其二，是总的应用个数，如果聚类个数过多，会导致某些聚类的规模过小，从而增大后续SVM分类出错的可能性。

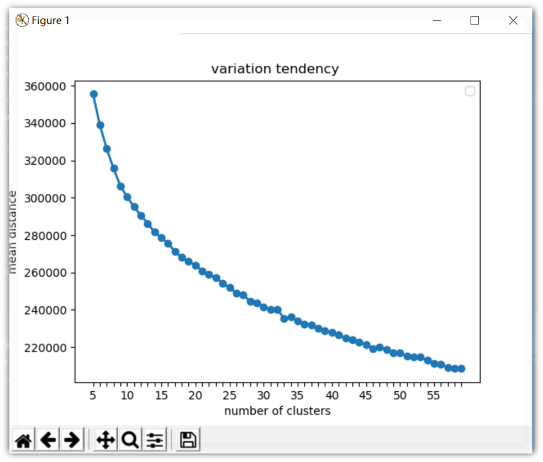


图1.

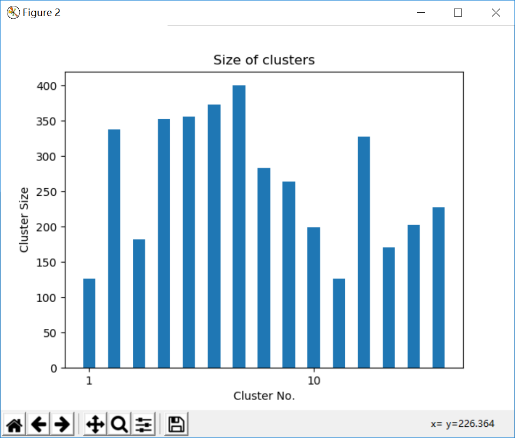


图2

图1是各个聚类数（Kmeans中的K值）对应的聚类情况，参与聚类的应用个数是3951，横坐标表示聚类数，纵坐标刻画了聚类的“密度”，意为每个聚类中元素距离中心点的距离的平均值。

图2是以K=15,的聚类结果，3951个应用聚成的15个类中，最大的聚类大小是400 ，最小的聚类大小是126，平均的大小是263.4。

**3.svm分类**

在对Kmeans聚类的结果进行处理的过程中，我们发现有的聚类只有良性程序，而并没有恶意程序。这是因为数据集的不均衡，即并没有覆盖到所有的类别，导致聚类结果分布的不均衡。由于svm分类需要训练集同时包含两种类别的程序，所以我们只针对那些有两种类型程序的聚类进行训练。最终我们对测试集50%左右的程序进行了预测，而剩下的程序由于属于全为良性程序的聚类，所以根据经验我们把它们默认预测为良性。在参数的选择上，我们选择惩罚系数C=0.8，gamma=20，核函数kernel=rbf.最后预测的准确率，成功进行预测的部分准确率是71.1%，如果加上默认预测良性的程序，总准确率为89%.

**5.结论**

我们的两步检测方法准确率在70%左右，而市面上应用的检测算法准确率一般在90%左右。所以我们还有很大的提升空间，在很多角度还可以进一步探索，比如特征向量的生成是否还有更好的方式，比如参数的设置是否能更优，比如聚类数的选择。另外，数据集的分布也是影响实验结果的重要因素。由于精力原因，我们没有再进一步研究。

另外，我们在本次实验中做的一个假设就是：恶意程序即为使用不该使用的permission的程序，反过来也是一样。这样才能把对permission向量的svm分类同恶意程序对应起来。

**参考文献：**

[1] 朱佳伟，喻粱文，关志，等．Android权限机制安全研究综述．计算机应用研究，2015，4(1O)：2880～2885

[2]杨晶，金伟信 ，吴作顺 ．基于Android系统的权限管理优化方案研究 电子质量，2015，(3)：4—1O

[3] DiCerboFrancesco，GirardelloAndrea，MichahellesFlorian，Voronkova

Svetlana．Detection of malicious applications on Android OS[C]//IEEE

Computer Society．2010，(11)：138—149．

[4] Alessandra Gorla,Ilaria Tavecchia Florian Gross,Andreas Zeller. Checking App Behavior Against App Descriptions, International Conference on Software Engineering. 2014:1025-1035.