

# 基于字节码图像的 Android 恶意代码家族分类方法

杨益敏, 陈铁明

(浙江工业大学计算机科学与技术学院, 浙江 杭州 310023)

**摘要:** 面对 Android 恶意代码高速增长的趋势, 提出基于字节码图像的 Android 恶意代码家族分类方法, 通过将 Android 恶意应用的字节码转化为 256 阶灰度图形式的字节码图像, 利用 GIST 算法提取图像的纹理特征, 并结合随机森林算法对特征进行分类。对常见的 14 种 Android 恶意代码家族的样本进行了实验验证, 并与 DREBIN 方法进行比较, 实验结果表明, 该方法可有效进行 Android 恶意代码家族分类, 具有检测精度高且误报率低的优点。

**关键词:** 安卓; 恶意代码家族; 图像纹理; 字节码

**中图分类号:** TP301

**文献标识码:** A

**doi:** 10.11959/j.issn.2096-109x.2016.00066

## Android malware family classification method based on the image of bytecode

YANG Yi-min, CHEN Tie-ming

(College of Computer Science and Technology, Zhejiang University of Technology, Hangzhou 310023, China)

**Abstract:** An Android malware family classification method based on the image of bytecode was proposed according to the exponential growth of Android malware. A bytecode file of Android malware was converted to a 256-level grayscale image and texture features was extracted from the image by GIST. The random forest algorithm was applied to classify the extracted features. The method by the experimental data of 14 kinds of common Android malware families was verified and was compared against the DREBIN on the same dataset. The experimental results show that the proposed method has high detection precision and low false positive rate.

**Key words:** Android, malware family, image texture, bytecode

### 1 引言

随着移动互联网的蓬勃发展, 移动应用的种类与数量快速增长, 同时, 移动应用的恶意代码规模也呈现指数级增长。阿里聚安全的 2015 移动安全病毒年报指出, Android 平台约 5.6 台设备中就有 1 台染毒, 设备感染率高达 18%, 约 95% 的

应用都存在病毒仿冒<sup>[1]</sup>。

虽然恶意代码的数量每年都在增加, 但大部分新的变种是在原有恶意代码的基础上经过一些变换操作得到的<sup>[2]</sup>。恶意软件编写者在生成恶意软件的过程中, 往往采用模块重用或自动化工具等手段来编写变种。这些同源的恶意代码家族很大程度上共用祖先的大部分代码, 具有一定的内在关联性、

收稿日期: 2016-04-18; 修回日期: 2016-05-23。通信作者: 陈铁明, tmchen@zjut.edu.cn

基金项目: 国家自然科学基金资助项目 (No.U1509214); 浙江省自然科学基金资助项目 (No.LY16F020035)

**Foundation Items:** The National Natural Science Foundation of China (No.U1509214), The Natural Science Foundation of Zhejiang Province (No.LY16F020035)

相似性。因此，对恶意代码的分类研究尤为重要，是影响恶意代码检测效果的一个关键因素。

恶意代码可视化方法在学术领域已经有不少研究<sup>[3-5]</sup>，该类方法的基本思路是将恶意代码的特征转换为图像层面的特征，通过分析图像的特征来间接实现检测恶意代码或对恶意家族进行分类的效果。

本文采用了一种恶意代码可视化的方法对 Android 恶意代码家族进行分类，先将 APK 中的 Dalvik 字节码转换为一种灰度图形式的字节码图像并用 GIST 算法提取图像中的特征，最后结合机器学习算法对特征进行分类。

## 2 相关工作

在 PC 平台，恶意代码的可视化方法已有一定的研究。例如，2011 年，Nataraj 等<sup>[6]</sup>提出了一种将 Windows 平台的恶意代码转化为纹理图像的方法来进行恶意代码分类，该方法具有无需执行代码、无需反汇编等优点。2012 年，Wu 等<sup>[7]</sup>提出一种点图可视化方法并对恶意代码进行聚类。2013 年，Han 等<sup>[8]</sup>将恶意代码通过反汇编后提取出 OpCode，并用散列函数将 OpCode 序列生成像素点的坐标信息与 RGB 信息，最后，通过计算图像的相似度对恶意代码家族进行分类。2014 年，Shaid 等<sup>[9]</sup>用不同的色系表示不同的 API 调用风险程度，生成彩图来可视化恶意代码的行为，此方法对于识别恶意代码的变种有较高的精准度。国内也有学者进行了相关研究，韩晓光等<sup>[10]</sup>将恶意代码映射为图片，并用灰阶共生矩阵算法提取图片的纹理指纹特征作为恶意代码的指纹，该方法对恶意代码变种有较好的识别能力。目前，相对于 PC 平台，Android 平台利用可视化方法来检测恶意代码变种的相关研究尚属空白。

2012 年，北卡罗莱纳州立大学 Zhou 等<sup>[11]</sup>第一次对 Android 恶意家族进行了系统性的研究，他们的研究团队从 2010 年 8 月到 2011 年 10 月收集了 Android 平台共计 1 200 个恶意样本，并对 Android 恶意家族的特征与演化过程进行了系统化的研究。2014 年，Arp 等<sup>[12]</sup>收集了 5 560 个恶意样本，并提出 DREBIN 方法，该方法将申请的权限、可疑的 API 函数调用和网络地址等作为静态分析的特征，不仅能对 Android 恶意代码与正

常代码进行分类，也能对不同家族的 Android 恶意代码进行分类，该方法具有较高的检测率，但误报率较高。

综上所述，本文首次将恶意代码可视化的方法引入到 Android 平台，并根据 Android 平台的特点提出一种可快速生成 Dalvik 字节码图像的方法，该方法与 GIST 算法、随机森林算法相结合，用于分类 Android 恶意代码家族，具有精度高且误报率低的特点。

## 3 本文方法

### 3.1 系统模型

本文的基本思路是将 Android 恶意应用的 Dalvik 字节码转化为一幅 256 阶灰度图；然后，用 GIST 算法提取图像的纹理特征；最后，再结合随机森林算法对特征进行分类。

整个系统模型如图 1 所示，将 APK 中的 DEX 文件提取出来，该 DEX 文件包含 Dalvik 字节码，可以被 Dalvik 虚拟机执行；然后，将字节码构建成一个合适宽度的像素点矩阵，矩阵中的像素点为一个字节，取值范围为 0x00~0xFF，即有 256 种取值，刚好对应 256 阶灰度，由像素矩阵可生成 png 格式的灰度图；最后，利用 GIST 特征算法提取出灰度图中的纹理特征，得到一个有 512 维的特征向量，再用 GIST 特征结合随机森林算法对 Android 恶意代码家族进行分类。

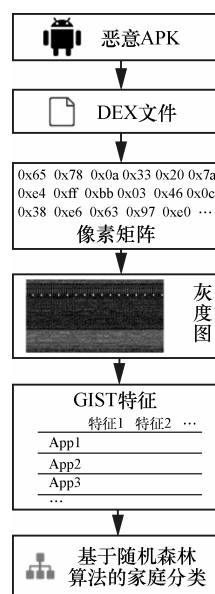


图1 系统模型

### 3.2 字节码图像的原理

APK 格式的文件通常都包含一个名为 classes.dex 的文件, 该 DEX (Dalvik executable) 文件封装了可被 Dalvik 虚拟机执行的所有 Dalvik 字节码。DEX 文件的结构由多个结构体组合而成, 包括 dex header、string\_ids、type\_ids、proto\_ids、field\_ids、method\_ids、class\_def、data 等多个部分。DEX 文件头指定了 DEX 文件的一些属性, 并记录了其他 6 部分数据结构在 DEX 文件中的物理偏移<sup>[13]</sup>。

若将 DEX 文件的字节码构建成一个宽度固定的像素点矩阵, 矩阵中的像素点为一个字节, 取值范围为 0x00~0xFF, 刚好对应 256 阶灰度, 能够由像素矩阵生成一张灰度图形式的字节码图像。图 2 显示了一个恶意代码的 DEX 文件结构与字节码图像的对应关系, 灰度图的纹理特征是对字节码数据结构与数据的综合反映。

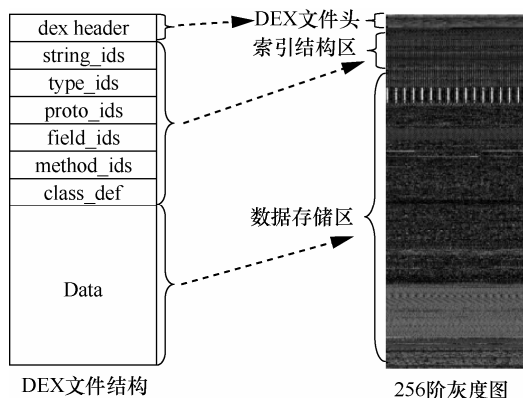


图 2 DEX 文件结构与字节码图像的对应关系

由于同源恶意代码家族很大程度上都共用了祖先的大部分代码, 具有内在关联性、相似性, 因此, 同一个家族恶意代码的字节码图像的纹理也会表现出一定的相似性。如图 3 所示, Android 恶意家族 DroidKungFu、Plankton、Opfake、Kmin 各有 5 个变种样本, 同一个家族的字节码图像的相似性要大于不同家族的字节码图像。另外, 图 4 给出了 14 个常见 Android 恶意家族较为典型的字节码图像, 不难发现, 各家族的图像具有各自的视觉特点。

### 3.3 图像特征的提取

常见的图像纹理特征提取方法包括 GLCM、边界频率法、GIST 模型、Gabor 滤波法等。本文采用 GIST 算法, 该算法常用于图像识别系统, 具体细节可参考文献[14]。接下来, 简单介绍如何利用 GIST 算法提取字节码图像特征的流程。

1) 按照原始图像的像素点总数, 通过计算可得到一个高度与宽度等长的像素点矩阵, 该矩阵存放的仍是原始图像的像素点, 并可利用矩阵生成一个正方形图像。

2) 将正方形图像均匀划分成  $4 \times 4$  个互不重叠的子区域。

3) 对每个子区域进行 8 个方向与 4 个尺度的小波滤波来提取图像的纹理特征信息。

4) 每一幅图像最终获得一个  $16 \times 8 \times 4 = 512$  维特征向量。

采用上述步骤对 6 个 Android 恶意代码家族的部分样本进行了特征提取, 并将得到的高维数

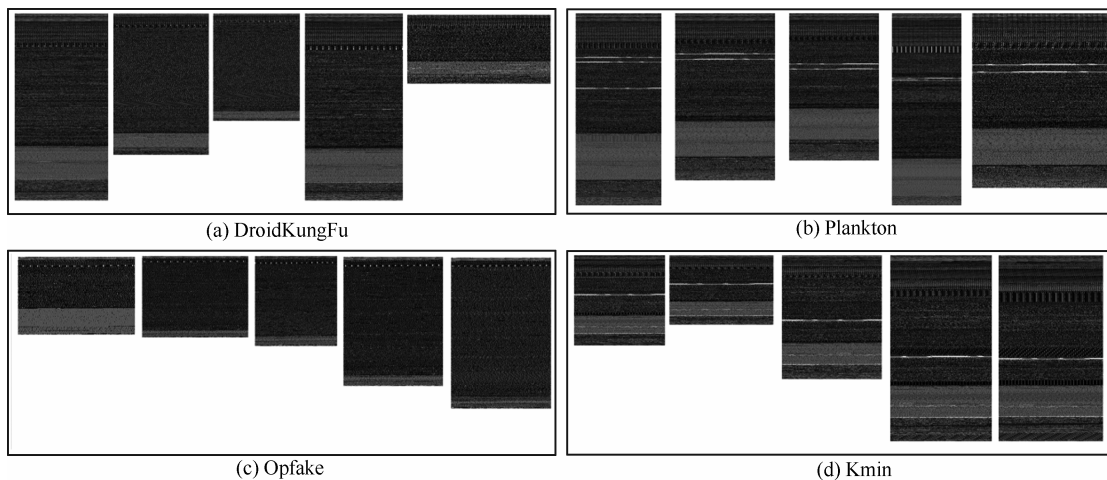


图 3 Android 恶意家族 DroidKungFu、Plankton、Opfake、Kmin 的字节码图像

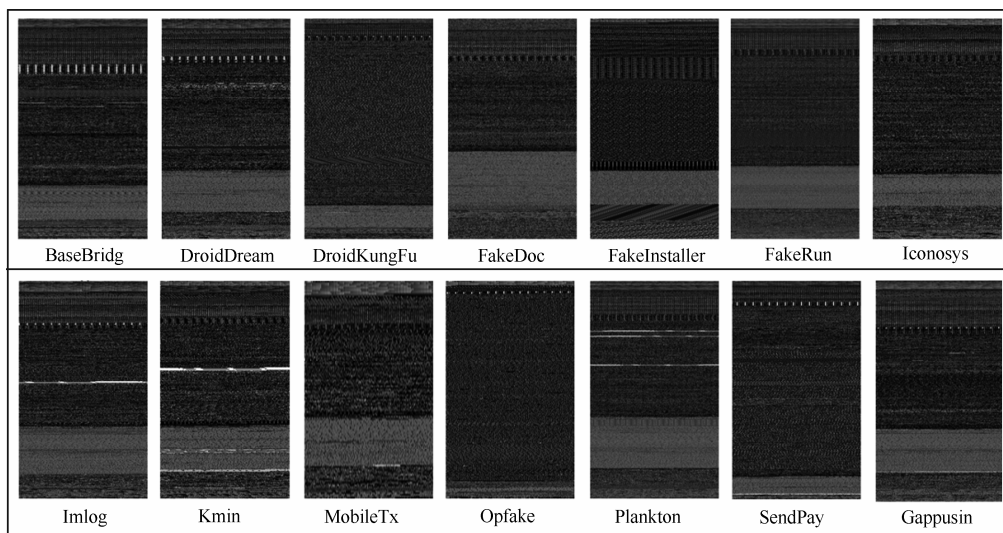


图 4 14 个常见 Android 恶意家族的字节码图像

据进行降维与数据可视化<sup>[15]</sup>。结果如图 5 所示，GIST 特征有较好的分类效果。

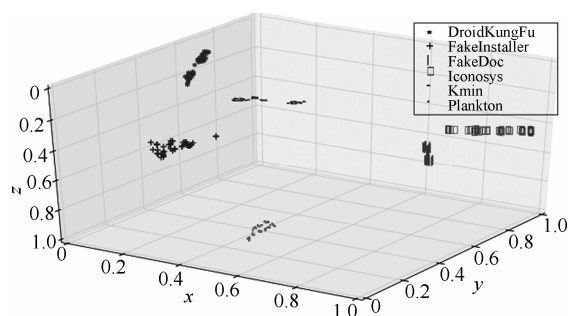


图 5 GIST 特征的数据可视化

### 3.4 随机森林算法

随机森林 (random forest) 是一种常见的机器学习分类算法，利用多棵决策树对样本进行训练并预测。在样本训练阶段，通过有放回抽样的方式抽取子样本集，对每组子样本集进行决策树建模，由多棵决策树构成随机森林。在样本预测阶段，先统计每棵决策树的预测结果，再通过投票形式给出最终的分类结果<sup>[16]</sup>。

随机森林算法有许多优点，特别是在处理多种类型数据时可以产生高准确度，且对不平衡的数据集可以平衡误差，非常适用于恶意家族数据集的分类，所以，本文实验选用随机森林算法。

## 4 实验与分析

### 4.1 实验数据

测试所采用的恶意样本来源于德国哥廷根大

学 DREBIN 项目的恶意样本数据库<sup>[17]</sup>，在该样本库中选用了 14 类常见的恶意家族作为最终实验数据，如表 1 所示。

表 1 Android 恶意家族与样本数量

| 家族名称          | 样本数 |
|---------------|-----|
| BaseBridge    | 330 |
| DroidDream    | 81  |
| DroidKungFu   | 667 |
| FakeDoc       | 132 |
| FakeInstaller | 925 |
| FakeRun       | 61  |
| Iconosys      | 152 |
| Imlog         | 43  |
| Kmin          | 147 |
| MobileTx      | 69  |
| Opfake        | 613 |
| Plankton      | 625 |
| SendPay       | 59  |
| Gappusin      | 58  |

### 4.2 恶意家族分类实验

使用随机森林算法对 14 类常见恶意家族的 3 962 个样本，按照 10 折交叉验证模式进行测试，结果如表 2 所示，除了 Gappusin 家族由于样本数量较少导致 TPR 值较低，其他家族的 TPR 值都不低于 0.8；总体上，平均检测率要高于 90%，平均误报率约为 0.6%，表明该方法对恶意家族分

表 2

恶意家族分类效果

| 家族名称          | <i>TPR</i> | <i>FPR</i> | <i>Precision</i> | <i>Recall</i> | <i>F-Measure</i> | <i>AUC</i> |
|---------------|------------|------------|------------------|---------------|------------------|------------|
| BaseBridge    | 0.839      | 0.002      | 0.965            | 0.839         | 0.898            | 0.984      |
| DroidDream    | 0.827      | 0.002      | 0.882            | 0.827         | 0.854            | 0.983      |
| DroidKungFu   | 0.934      | 0.057      | 0.733            | 0.934         | 0.821            | 0.989      |
| FakeDoc       | 0.962      | 0.001      | 0.977            | 0.962         | 0.969            | 0.995      |
| FakeInstaller | 0.969      | 0.011      | 0.956            | 0.969         | 0.962            | 0.999      |
| FakeRun       | 0.951      | 0          | 1                | 0.951         | 0.975            | 1          |
| Iconosys      | 0.961      | 0.004      | 0.885            | 0.961         | 0.921            | 0.999      |
| Imlog         | 0.837      | 0          | 1                | 0.837         | 0.911            | 0.984      |
| Kmin          | 0.98       | 0.001      | 0.973            | 0.98          | 0.976            | 1          |
| MobileTx      | 1          | 0          | 0.986            | 1             | 0.993            | 1          |
| Opfake        | 0.977      | 0.008      | 0.951            | 0.977         | 0.964            | 0.999      |
| Plankton      | 0.974      | 0.005      | 0.965            | 0.974         | 0.97             | 0.998      |
| SendPay       | 0.915      | 0.001      | 0.947            | 0.915         | 0.931            | 0.989      |
| Gappusin      | 0.534      | 0          | 1                | 0.534         | 0.697            | 0.954      |

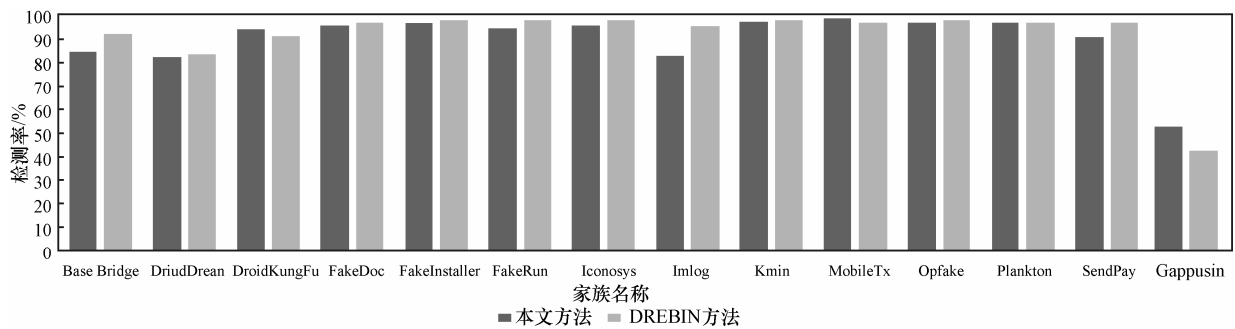


图6 2种检测方法的检测率对比

类有较好的效果。

图6给出了本文方法与DREBIN方法在相同实验数据集上检测率的对比情况,2种检测方法在不同的检测恶意家族的表现上互有优劣,总体效果基本接近,虽然Gappusin家族一项的检测率都不高,但本文方法略好于DREBIN方法。DREBIN方法的平均误报率为1%,而本文方法的平均误报率为0.6%,要略优于DREBIN方法。

## 5 结束语

本文研究了DEX文件结构与字节码图像纹理之间的对应关系,提出了一种基于字节码图像的Android恶意代码家族分类方法,通过将Android恶意应用的Dalvik字节码转化为256阶灰度图形式的字节码,利用GIST算法提取图像

的纹理特征,并结合随机森林算法对特征进行分类。本文对常见的14种Android恶意代码家族的样本数据进行了实验验证,并与DREBIN方法进行了比较,结果表明,该方法可有效进行Android恶意代码家族分类,检测精度高且误报率较低。下一步工作将改进字节码图像纹理特征的提取方法,针对Android平台的特性加入更多恶意代码的可视化方法,以进一步提高检测精度、降低误报率。

## 参考文献:

- [1] 阿里聚安全[EB/OL].<http://jaq.alibaba.com/>.2016.
- [2] Ali poly security[EB/OL].<http://jaq.alibaba.com/>.2016.
- [3] FOSSI M, EGAN G, HALEY K, et al. Symantec internet security threat report trends for 2010[R]. 2010.
- [3] CONTI G, DEAN E, SINDA M, et al. Visual reverse engineering of

- binary and data files[J]. Journal of General Physiology, 2008, 115(5): 637.
- [4] ANDERSON B, STORLIE C, LANE T. Improving malware classification: bridging the static/dynamic gap[C]//The 5th ACM Workshop on Security and Artificial Intelligence. c2012: 3-14.
- [5] NATARAJ L, YEGNESWARAN V, PORRAS P, et al. A comparative assessment of malware classification using binary texture analysis and dynamic analysis[C]//The 4th ACM Workshop on Security and Artificial Intelligence. c2011: 21-30.
- [6] NATARAJ L, KARTHIKEYAN S, JACOB G, et al. Malware images: visualization and automatic classification[C]//The 8th International Symposium on Visualization for Cyber Security, ACM. c2011: 4.
- [7] WU Y, YAP R. Experiments with malware visualization[M]// Detection of Intrusions and Malware, and Vulnerability Assessment. Berlin Heidelberg: Springer, 2012: 123-133.
- [8] HAN K, LIM J, IM E. Malware analysis method using visualization of binary files[C]//The 2013 Research in Adaptive and Convergent Systems, ACM. c2013: 317-321.
- [9] SHAID M, ZAINUDEEN S, MAAROF M. Malware behavior image for malware variant identification[C]// 2014 International Symposium on Biometrics and Security Technologies (ISBAST), IEEE. c2014: 238-243.
- [10] 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 X G, QU W, YAO X X, et al. Malicious code variants detection method based on texture fingerprint study [J]. Journal of Communications, 2014, 35(8): 125-136.
- [11] ZHOU Y, JIANG X. Dissecting android malware: characterization and evolution[C]// 2012 IEEE Symposium on Security and Privacy (SP), IEEE. c2012: 95-109.
- [12] ARP D, SPREITZENBARTH M, HUBNER M, et al. DREBIN: effective and explainable detection of android malware in your pocket[C]// Network and Distributed System Security Symposium (NDSS). c2014.
- [13] ELENKOV N. Android security internals: an in-depth guide to android's security architecture[M]. No Starch Press, 2014.
- [14] OLIVA A, TORRALBA A. Modeling the shape of the scene: a holistic representation of the spatial envelope[J]. International Journal of Computer Vision, 2001, 42(3):145-175.
- [15] T-SNE[EB/OL]. <http://lvdmaaten.github.io/tsne/>. 2016.
- [16] HAN J, KAMBER M, PEI J. Data mining[M]. San Francisco: Morgan Kaufmann, 2011.
- [17] The drebin dataset[EB/OL]. <http://user.informatik.uni-goettingen.de/~darp/drebin/>. 2016.

#### 作者简介:



杨益敏 (1985-), 男, 浙江宁波人, 浙江工业大学硕士生, 主要研究方向为网络与信息安全。



陈铁明 (1978-), 男, 博士, 浙江工业大学教授, 主要研究方向为网络与信息安全。