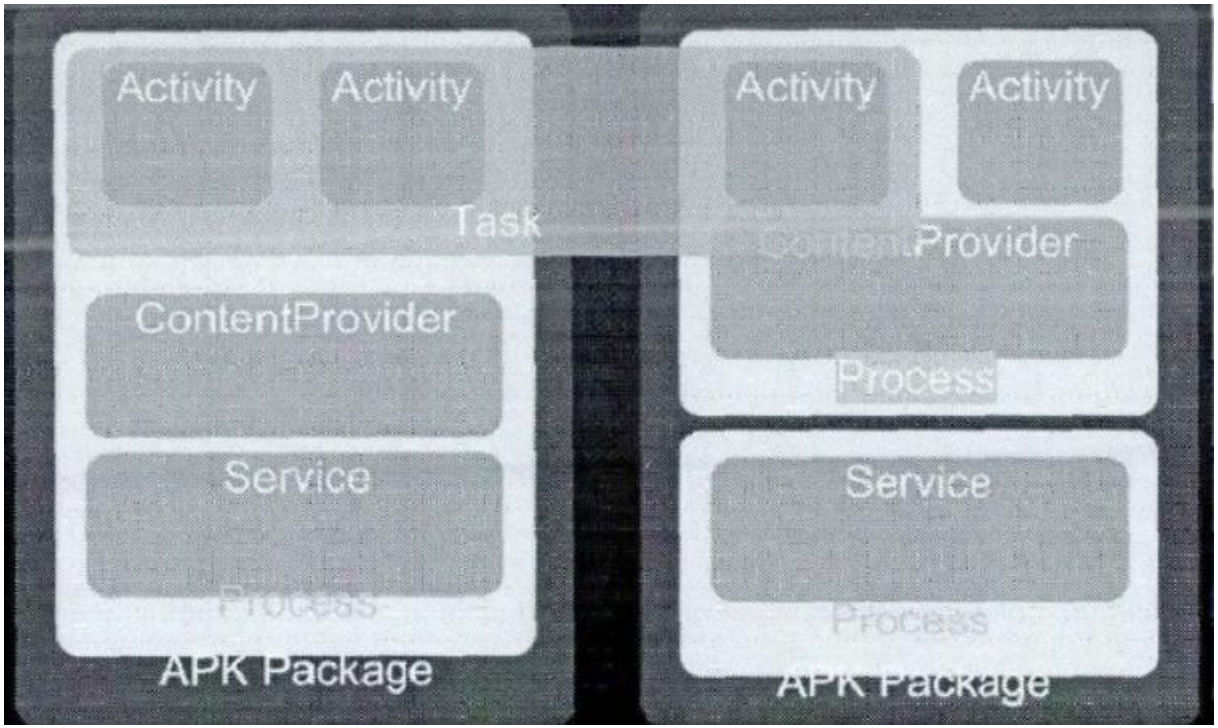
# 基于日志的Android平台恶意软件检测反感的研究与实现

- 1.基于源码的静态检测方法
- 2.基于沙箱或者虚拟机的动态检测方法
- => Android系统日志的特性，提出基于日志的Android平台手机恶意软件检测方案

# Android四大组件



# Android 系统中的UID、GID、GIDS与PID

- UID：应用程序在安装时被分配用户 UID
- GID：对于普通的应用程序，GID即等于UID
- GIDS：Application 申请的具体权限
- PID：是host应用程序的沙箱，里面一般有一个UID和多个GIDS

# 静态、动态检测流程图

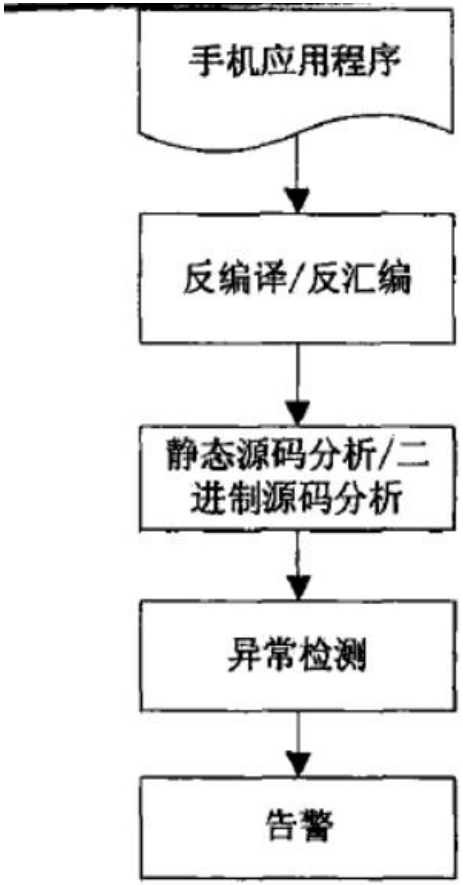



图 2-3　静态检测流程图

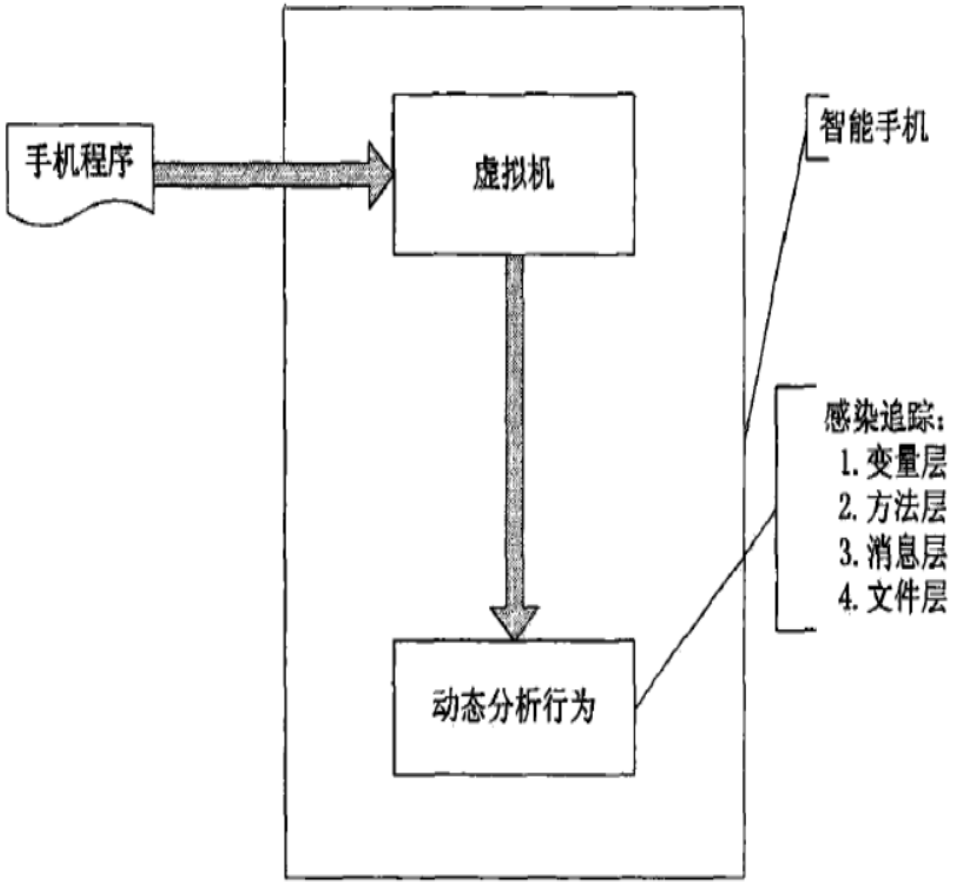



图 2-4　动态检测流程图

# 文章的检测方法

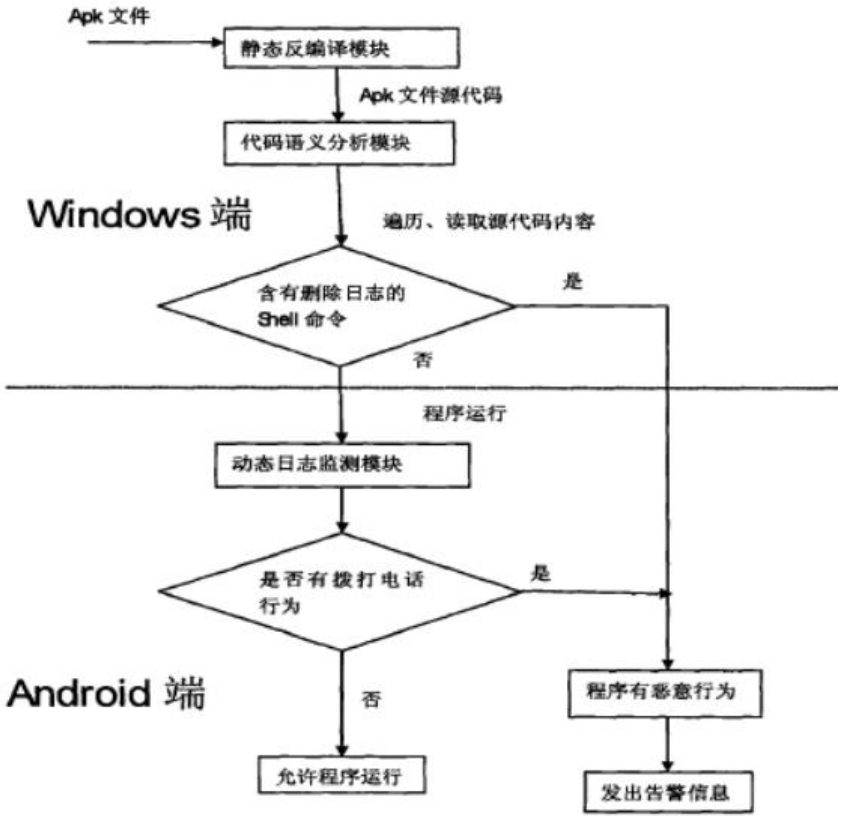
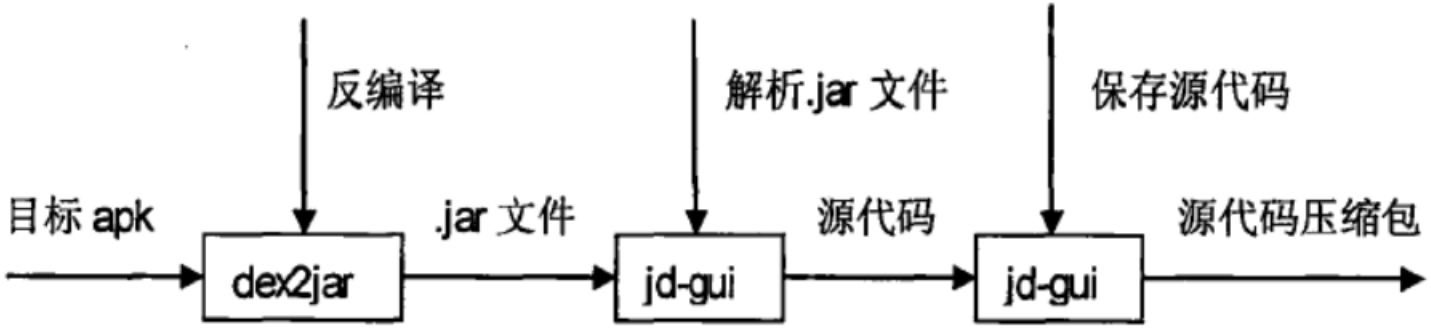



图 3-1　基于日志的 Android 平台恶意软件检测方案功能模块图

# 反编译apk


反编译
解析.jar 文件
保存源代码
目标 apk
.jar 文件
源代码
源代码压缩包
dex2jar
jd-gui
jd-gui

# 语义分析

- BM算法采用从右向左比较 的方法，同时应用到了两种启发式规则，即坏字符规则 和好后缀规则，来决定向右跳跃的距离
- 坏字符：$$skip(x) = \begin{cases} m;\ x \neq P[j](1 \leq j \leq m)\ ，即\ x\ 在\ P\ 中未出现 \\ m - max(x);\ \{k|P[k] = x, 1 \leq k < m\};\ \ x\ 在\ P\ 中出现 \end{cases}$$
- 好后缀：$$Shift(j) = \min\{s|(P[j+1..m] = P[j-s+1..m-s])\&\&(P[j] \neq P[j-s])(j > s), P[s+1..m] = P[1..m](j \leq s)\}$$

关注：“logcat -c”

# 日志分析

- V、D、I、W、E
- Tag、Pid、Message