



Android 恶意应用 HTTP 行为特征生成与提取方法

罗亚玲¹, 黎文伟², 苏欣^{2,3}

(1. 广东松山职业技术学院计算机系, 广东 韶关 512126;

2. 湖南大学信息科学与工程学院, 湖南 长沙 410082;

3. 湖南警察学院网络侦查技术湖南省重点实验室, 湖南 长沙 410138)

摘要: Android 恶意应用数量的不断增加不仅严重危害 Android 市场安全, 同时也为 Android 恶意应用检测工作带来挑战。设计了一种基于 HTTP 流量的 Android 恶意应用行为生成与特征自动提取方法。该方法首先使用自动方式执行恶意应用, 采集所生成的网络流量。然后从所生成的网络流量中提取基于 HTTP 的行为特征。最后将得到的网络行为特征用于恶意应用检测。实验结果表明, 所设计的方法可以有效地提取 Android 恶意应用行为特征, 并可以准确地识别 Android 恶意应用。

关键词: Android 恶意应用; HTTP 流量; 网络行为特征; 安全

中图分类号: TP393.08

文献标识码: A

doi: 10.11959/j.issn.1000-0801.2016222

HTTP behavior characteristics generation and extraction approach for Android malware

LUO Yaling¹, LI Wenwei², SU Xin^{2,3}

1. Department of Computer, Guangdong Songshan Polytechnic College, Shaoguan 512126, China

2. College of Computer Science and Electronics Engineering, Hunan University, Changsha 410082, China

3. Hunan Provincial Key Laboratory of Network Investigational Technology,
Hunan Police Academy, Changsha 410138, China

Abstract: Growing of Android malware, not only seriously endangered the security of the Android market, but also brings challenges for detection. A generation and extraction approach of automatic Android malware behavioral signatures was proposed based on HTTP traffic. Firstly, the behavioral signatures were extracted from the traffic traces generated by Android malware. Then, network behavioral characteristics were extracted from the generated network traffic. Finally, these behavioral signatures were used to detect Android malware. The experimental results show that the approach is able to extract Android malware network traffic behavioral signature with accuracy and efficiency.

Key words: Android malware, HTTP traffic, network behavioral characteristic, security

收稿日期: 2016-04-25; 修回日期: 2016-08-04

通信作者: 罗亚玲, versace0922@163.com

基金项目: 国家自然科学基金资助项目 (No.61173168, No.61471169); 广东省教育厅资助项目 (粤教高函[2012]54 号-A12); 网络犯罪侦查湖南省普通高等学校重点实验室开放研究基金资助项目 (No.2016WLFZZC008)

Foundation Items: The National Natural Science Foundation of China (No.61173168, No.61471169), Foundation of the Education Department of Guangdong Province of China (No.[2012]54-A12), The Open Research Fund of Key Laboratory of Network Crime Investigation of Hunan Provincial Colleges (No. 2016WLFZZC008)

1 引言

近年来,智能手机数量及其业务量发展迅速。国内移动数据服务商 QuestMobile 发布的《2015 年中国移动互联网研究报告》显示,截至 2015 年 12 月,国内在网活跃移动智能设备数量达到 8.99 亿台,其中,Android 操作系统占 70% 左右^[1]。Android 系统代码的开放性以及 Android 智能手机上存储的大量用户隐私信息,使得 Android 平台成为了众多恶意代码开发者的活跃地盘。

在已有的 Android 恶意应用检测研究中,很多工作都关注于提取系统层的行为特征(system-level behavioral signature)来对 Android 恶意应用进行检测,此类方法主要分为静态分析和动态分析。其中,静态分析的典型方法有 RiskRanker^[2]和 Drebin^[3]等。RiskRanker 提出了一种基于代码分析的主动检测策略,检测当前 Android 市场中的 Android 应用,并通过分析应用中是否存在危险行为来检测恶意应用。该方法把危险行为分为 3 类:高风险、中风险和低风险。例如,一个应用如果被检测到存在 root 手机这一行为,就会被定义为具有高风险行为的应用;如果被检测到使用短信服务但是没有调用 onClick 这个方法,就会被定义为具有危险行为的应用。Drebin 设计了一个轻量级、低开销、能实时在 Android 手机上检测 Android 恶意应用的系统。该系统从应用的 Manifest 文件、dex 代码等系统层属性中选取了 8 个属性,并使用线性 SVM 算法^[4]对实时运行在 Android 手机中的应用进行分类,从而实现检测恶意应用的目的。然而,由于 Android 恶意应用制造者广泛使用代码混淆和加密等技术,静态分析不能很好地处理这些情况。因此,有些研究人员提出使用动态分析,该类型的方法有 PUMA^[5]、TaintDroid^[6]等。其中,PUMA 设计了一种 PUMA 脚本语言来自动执行、动态分析 Android 应用。该脚本语言通过使用 Android Monkey^[7]并针对不同的应用控件设定多个执行事件,比如点击按钮、拖拉屏幕等。该工具可以在用于测试 Android 应用 bug、广告欺诈检测、触发 Android 恶意应用恶意行为等多个工作中。TaintDroid 提出了使用动态污染分析变量层、方法层、消息层和文件层的信息,用于检测隐私泄露方面的漏洞。

由于基于系统层行为特征的检测方法存在不易部署、开销大等不足,基于网络流量行为特征的检测方法受到研究人员的关注。Wei 等人^[8]对 27 个 Android 应用所产生的网络流量进行分析,然而该方法通过手动的方式来执行应

用生成流量,不适用于大量 Android 应用的情况。参考文献[9]提出了一种 DNS 域名分析方法,来检测网络中存在的 Android 恶意应用所产生的流量。然而很多 Android 恶意应用会经常变换 DNS 域名,从而导致该方法检测率较低。Su 等人^[10]提出了使用 HTTP 特征签名和时间关联度的方法来识别 Android 应用。该方法以特征签名为基础,通过自定义的时间窗口来发现不具备特征签名的 HTTP 流,从而达到 Android 应用识别的目的。Dai 等人^[11]提出了一种自动生成移动应用 HTTP 流量并从中提取特征字符串对 Android 应用进行识别的方法,然而并未对 Android 应用识别的准确率进行实验证明。Xu 等人^[12]设计一种名为 FLOWER 的自动学习 Android 应用特征签名的系统。该系统通过观察同时出现的应用属性,从 HTTP 头部文件中学习特征签名。但是该系统并不能准确地识别网络流量由哪个应用产生,只能把网络流量产生的应用数量范围限定在 5 个以下。因此,不适用于 Android 恶意应用的检测。

本文设计了一种针对 Android 恶意应用的 HTTP 行为特征自动生成与提取方法,解决上述工作存在的不足。在 Zhou 等人的研究工作^[13]中发现,大部分的 Android 恶意应用都是通过 HTTP 与外部服务器进行通信,并执行恶意活动,比如隐私窃取、下载恶意代码等。因此,本文重点关注 Android 恶意应用产生的 HTTP 流量,并从中提取基于网络流量的行为特征。不同于已有研究中所提取的端口或者主机域名等网络特征,本文所提取的网络特征包括流量的五元组、HTTP 头部文件中的主机域名、HTTP 请求方法和 URI 等,可以准确地捕获到 Android 恶意应用的网络行为。

所提出的方法首先自动地执行 Android 恶意应用并捕获其所产生的网络流量;其次,解析所捕获的网络流量数据,提取出相应的 HTTP 流的行为特征;随后对所提取的 HTTP 流的行为特征进行聚类,并合并具有重叠元素的特征;最后得到一个由 HTTP 流行为特征所组成的 Android 恶意应用网络流量行为特征的特征库,并将该特征库用于 Android 恶意应用检测。实验结果表明,本文提出的方法可以有效地从 HTTP 流量中提取 Android 恶意应用的行为特征,并准确地检测 Android 恶意应用。

2 行为特征提取方法设计

设计并实现了一种自动执行 Android 恶意应用,并从



所生成的网络流量中提取行为特征的方法。该方法通过提取网络流量中的 HTTP 流的行为特征来检测 Android 恶意应用。首先调用 Android 的测试工具(比如 Monkey 等)的程序接口(application programming interface, API)来自动执行 Android 恶意应用,并使用 TCPDUMP^[14]捕获应用在运行过程中生成的网络流量。随后,对捕获到的网络流量数据中的 HTTP 流的行为特征进行提取,并通过聚类算法对提取到的行为特征进行聚类 and 合并。最后建立一个基于 HTTP 流的 Android 恶意应用网络流量的行为特征库,用于 Android 恶意应用检测工作。图 1 是本文方法的框架。

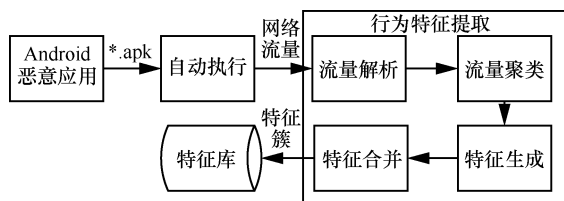


图 1 Android 恶意应用网络行为特征自动提取框架

2.1 自动执行

由于手动执行 Android 恶意应用会造成时间和精力的大量浪费^[8],因此使用 Android 自动测试工具 Monkey 所提供的自动执行功能来解决这个不足。Monkey 是 Android 中的一个命令行工具,可以运行在模拟器或实际设备中。它向系统发送伪随机的用户事件流(如按键输入、触摸屏输入、手势输入等),以实现正在开发的应用程序进行压力测试。由于 Monkey 所提供的自动操作 Android 应用的事情都是随机产生的,无法根据应用中不同的控件类型而生成相对应的操作事件,导致一些 Android 恶意应用的网络行为无法被触发。针对这个不足,本文所设计的方法在 Monkey 的基础上增加了如何根据不同的 Android 应用控件类型生成相对应的自动操作事件。自动执行模块的工作流程如图 2 所示。图 2 由 4 个组件组成,分别是事件模拟器、路径生成器、控件识别器和流量捕获。

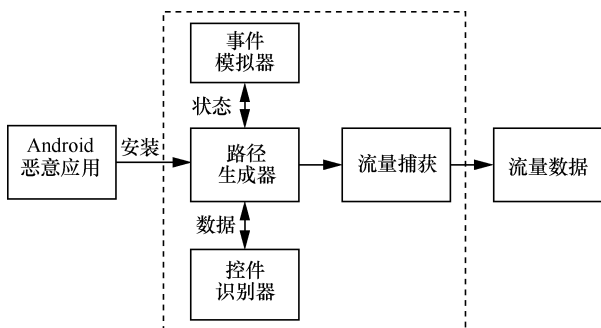


图 2 自动执行模块流程

2.1.1 事件模拟器

该组件基于 Android 的测试工具 MonkeyRunner^[15]的 API 设计实现,主要实现两个功能:一是连接 Android 设备(如 Android 智能手机或模拟器),自动安装和卸载所需要运行的应用,并启动需要采集流量的应用;二是向 Android 设备发送应用操作事件,比如点击按钮、拖拉屏幕等,并返回执行应用的状态。根据控件识别器所提供的不同控件类型生成不同的操作事件,保证 Android 恶意应用可以被正常地执行。

2.1.2 控件识别器

该组件基于 Android 另一个测试工具 Hierarchy Viewer^[15]的 API 设计实现,主要功能是识别当前 Android 设备上显示的界面中所定义的控件类型,比如按钮、文本框等,并发送给事件生成器,以便事件生成器根据不同类型的控件生成不同的应用操作事件。

2.1.3 路径识别器

该模块调用 Hierarchy Viewer 所提供的 API 进行设计,主要功能是识别当前手机或者模拟器屏幕显示的界面(activity),并获取该 activity 上所有控件的信息(如控件名称、ID、是否可点击等),从而为事件生成模块提供控件信息生成相应的执行事件。典型的 Android 应用包含一个或多个 activities。一个 activity 可以被看作一个容纳了很多控件元素的容器,如按钮、列表、文本框等。不同的 activity 之间都是执行这些控件进行转换。

一个 Android 应用包含一条或多条可执行路径,执行路径不同可能触发不同网络行为,从而导致产生的流量具有不一样的属性。因此,为了识别 Android 应用的可执行路径,提出一个基于深度优先搜索的可执行路径识别算法,即深度优先识别算法(depth-first identification algorithm)。该算法是基于传统的深度优先遍历树算法,根据 Android 应用的特点进行改进得到的。该算法最初把一个 Android 应用启动时首先出现在屏幕的 activity 作为根节点;随后对根节点的 activity 中控件进行点击,当前 activity 就会跳转到下一级节点,一直重复这个过程直到跳转到某个 activity 上不存在可点击的控件或者点击后该 activity 不再转换。该算法则返回上一级节点的 activity 去寻找没有点击过的控件进行点击。那么从根节点到叶子节点的这条路径就可以记录为 Android 应用的一条可执行路径。深度优先识别算法将详细介绍可执行路径识别的过程,如算法 1 所示。

算法 1 基于深度优先的路径识别算法

输入 Android 应用的操作界面;

输出 Android 应用执行路径;

```

(1) function DFI(|A|)
(2)   for all activities  $A_i, A_j$  in |A| do
(3)     change to activity  $A_i$ ;
(4)     DEPFIRSTIDENT ( $A_i$ );
(5)   end for
(6) end function
(7) function DEPFIRSTIDENTI ( $A_i$ )
(8)   widgetset ← GET_WIDGET( $A_i$ );
(9)   for each widget in widgetset do
(10)    if (widgetset.isclickable == TURE) then
(11)      if ( $A_i \rightarrow A_j$ )
(12)        record this activity transition;
(13)        DEPFIRSTIDENTI ( $A_j$ );
(14)        back to activity  $A_i$ ;
(15)      end if
(16)    end if
(17)  end for
(18) end function

```

在深度优先路径识别算法中,首先从 Android 应用的 manifest 文件中提取应用启动时显示的 activity 名称 (比如 main activity),这些 activity 将作为可执行路径的起点。然后从这个起点开始执行算法 (第 1~5 行)。对于应用中的每个 activity(A_i),算法都会提取其所有的控件的属性 (第 9 行)。该算法将逐个判断每个控件是否可点击,如果该控件可点击,则进行点击。当发现当前 activity A_i 跳转到一个新的 activity A_j 时,算法将在 A_j 上递归地调用 DEPFIRSTIDENT 函数,并记录这个转换。该算法将以深度优先的形式执行下去,直到当前显示的 activity 中没有可点击的控件或者点击可点击的控件后 activity 不再转换,算法将跳转到上一级的 activity,并重复前述的过程。

假设 Android 应用具有 N 个 activity。每个 activity 调用 DEPFIRSTIDENT 函数的次数是根据该 activity 中所定义的可点击控件的个数来决定的,即 widgetset 所包含控件个数 W 。在执行一个 Android 应用时,DEPFIRSTIDENT 函数被调用的次数应该是该应用所具有的可点击控件的个数,即

$$M = \sum_{k=1}^N W_k. \text{ 所以该算法的时间复杂度为 } O(M).$$

2.1.4 流量捕获

该组件是基于 TCPDUMP^[14]设计实现的,主要功能就是捕获 Android 恶意应用在运行时产生的网络流量,并保存为 pcap 格式文件,便于后续的分析工作。

2.2 网络行为特征提取

该模块通过对 Android 恶意应用运行时产生的网络流量进行捕获、解析,并从中提取 HTTP 的行为特征。将提取的行为特征保存到 Android 恶意应用网络行为特征库中。网络行为可以映射为多个不同的网络特征,比如一个 Android 恶意应用可能访问一个或者多个命令与控制 (command and control, C&C) 服务器,访问不同的 C&C 服务器可能导致不同的网络行为,从而产生不同的网络流量行为特征。而根据 Zhou 等人的研究发现^[13],绝大部分的 Android 恶意应用都是通过 HTTP 和 C&C 服务器进行通信。因此,Android 恶意应用的网络流量特征主要集中在 HTTP 流中。所以本文将从 Android 恶意应用产生的 HTTP 流中提取 Android 恶意应用的网络流量行为特征。如图 1 所示,该模块由 4 个不同的组件组成,接下来将详细介绍每个组件。

2.2.1 流量解析

每一个被捕获的流量数据文件都包含一条或多条 HTTP 请求的信息。由于受限于最大分段长度,很多 HTTP 请求被分割成多段进行传输。因此,本组件需要把这些被分割的段整合为一个整体。本组件将根据网络流量属性中的具有相同的源/目的 IP 地址和源/目的端口的流判断为同一条流。HTTP 流整合完成后,提取如下的网络流量的属性: {fid, srcIP, dstIP, srcPort, dstPort, host, method, URI, header, isad, isalexa}。其中, fid 表示 Android 恶意应用的识别符, < srcIP, dstIP, srcPort, dstPort > 分别表示源/目的 IP 地址和源/目的端口, host 表示所访问的服务器域名, method 表示 HTTP 请求的方法,如 get 或者 post, URI 表示 HTTP 请求的统一资源定位符, header 表示 HTTP 头部信息, isad 表示是否为广告库生成的流量, isalexa 表示是否是正常流量。

2.2.2 流量聚类

为了更好地区分 Android 恶意应用的不同 HTTP 行为,本组件将相似度较高的 HTTP 行为聚类到同一个簇中。聚类的原则是,基于 URI 相似度将具有相同的 HTTP 请求方法并具有相同或者相似的 URI 的 HTTP 流聚类到同一个簇。相似度的计算方法是基于 Jaccard index^[16]进行



计算的,该模块对每一对 URI 进行 Jaccard 指标计算,如果计算得到的 Jaccard 指标的值大于 0.6 (第 3.3 节详述),则认为这两条流具有相似性,并可以聚类到同一个簇。

2.2.3 特征生成

HTTP 流的行为特征代表了 Android 恶意应用的网络行为,该特征是由多个网络属性所组成的。由于大多数的 Android 恶意应用都通过 HTTP 与外部服务器进行通信,因此本组件重点关注于 HTTP 流的行为特征提取。HTTP 流的行为特征由 HTTP 请求流中的主机域名、请求方法和 URI 3 个部分组成。其中,URI 又包括 URL 路径或者查询路径以及查询的参数,这些元素可以映射为应用不同的网络行为。图 3 通过一个名为 ADRD^[7]的 Android 恶意应用所产生的 HTTP 请求中所包含的 URI 为例子来更好地解释 Android 恶意应用的行为特征。

其中,page 表示该应用所访问的 URL 或者请求的资源路径,*f* 表示应用所访问的具体页面或者请求具体的资源文件类型,query 表示查询部分,*k* 表示查询的参数,*v* 表示查询参数的值。

该组件首先根据分隔符“?”把整个字段划分成两个子字段,分别是请求页面字符串和查询字符串。随后去掉查询字段里面的查询参数值,因为有些参数表示 Android 设备的参数,比如 IMEI、屏幕分辨率等,对于不同的 Android 设备这些值是不一样的,但是并不能说明所产生的网络行为不同。最后,该模块从 URI 中提取出的 page 和 *k* 两类字段,与流量解析部分描述的其他网络属性共同组成了 Android 恶意应用的网络流量行为特征。

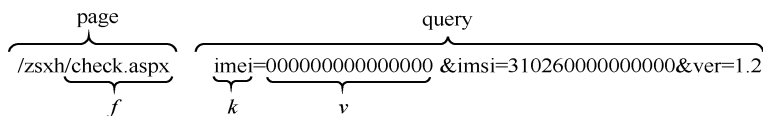


图 3 ADRD 恶意应用的 URI

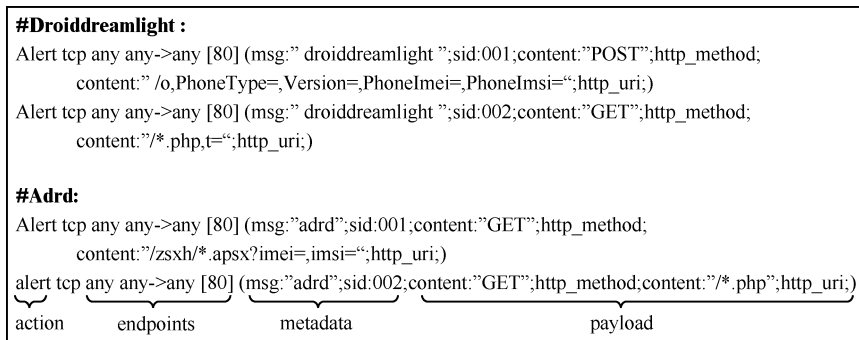


图 4 基于 HTTP 流量的 Android 恶意应用的行为特征格式

2.2.4 特征合并

在 HTTP 流量聚类后,不同的簇中可能存在 HTTP 流包含同样的或者相似的 URI,导致所提取的行为特征具有一定的冗余性,增加后续检测工作的开销。因此,本文对所提取的行为特征进行合并。假设有两个 URI,提取后得到的字段集合分别用 *S1* 和 *S2* 表示。

S1: /zsxh/*.aspx,type=, imei=,imsi=,area=

S2: /zsxh/*.aspx, imei=,imsi=,ver=

从 *S1* 和 *S2* 可以看出,两个字段集合的 page 部分是完全一样的,而查询参数部分也具有一定的重合度。因此具有这样特征的行为特征字段是需要进行合并的。合并的方法就是对查询参数取交集操作。最后得到 *S*: /zsxh/*.aspx, imei=,imsi=。

经过上述 4 个组件提取了基于网络流量的 Android 恶意应用的行为特征。图 4 以 ADRD 和 DroidDreamLight^[18]两类恶意应用为例,展示了存储于特征库里面的最终行为特征的格式。本文将行为特征存储为类似于 Snort 规则的格式,以便于部署到 Android 恶意应用检测工作中。

3 实验结果

3.1 实验数据集

使用两个不同的数据集,第一个是美国北卡罗莱纳州立大学提供的 Android malware genome project^[13]中所提供的 Android 恶意应用数据集。该数据集包括了来自 49 个不同 Android 恶意应用家族的 1 260 个恶意应用,其中,1 171 个恶意应用都会通过 HTTP 与 C&C 服务器进行通信。该数据

集用于自动执行应用并从中提取 Android 恶意应用的网络流量的行为特征。第二个数据集是由 118 个未知类型的恶意应用组成。该数据集用于验证所提取的行为特征的有效性。表 1 列举了两类数据集的详细信息。

表 1 数据集详细信息

数据集	恶意应用数目/个	具有网络访问/个	正常运行/个	产生网络流量/个
Genome	1 260	1 171	952	952
未知类型	118	107	96	96

由于大多数 Android 恶意应用都是通过正常应用中嵌入恶意代码的方式来生成的。因此,在运行恶意应用时会同时产生正常和恶意两类网络流量。为了更加准确地提取恶意应用产生的网络流量的行为特征,本文根据流量的域名定义了一个白名单,该名单是由 Alexa^[19]网站所提供的世界上最流行的 500 个网站域名组成的。在采集完网络流量数据后先用该白名单过滤掉那些正常的网络流量。此外,Android 应用还具有一个特点,即内嵌的广告库在应用运行时也会产生网络流量。因此这部分流量也需要在提取行为特征之前过滤掉。本文同样建立一个广告名单,由 AdRisk^[20]中所提到的广告域名所组成。同样地在采集 Android 恶意应用生成的网络流量时根据广告名单过滤掉广告流量。表 2 列举了过滤后的网络流量信息。

表 2 流量分类统计信息

数据集	总共产生流个数/个	恶意流个数/个	正常流个数/个	广告流个数/个
Genome	26 889	3 804	10 936	17 189
未知类型	1 748	857	194	828

实验中,每个 Android 恶意应用都是在真实的 Android 智能手机上(系统版本 2.3.3)运行,每个应用运行 5 min。此外,由于在自动执行过程中需要识别控件类型,该阶段平均消耗的时间为 10 s 左右。

3.2 Android 恶意应用自动执行结果

通过使用 activity 覆盖率来验证本文所设计的 Android 恶意应用自动执行方法的有效性。activity 覆盖率是指一个 Android 恶意应用在自动执行过程中执行的 activity 个数占该应用总的 activity 个数的比例。因为在 Android 恶意应用中触发不同的 activity 可能导致不同的网络行为,从而生成不同的 HTTP 流量。所以在自动执行过程中如果可以执行 Android 恶意应用所有的 activity,那么就可以从 HTTP 流量中提取到可以完全表示该恶意应用网络行为的特征。图 5

列举了 5 个流行的 Android 恶意应用的 activity 覆盖率,并和手动执行方法、使用 Monkey 随机执行方法进行比较。这 5 个恶意应用分别来自 5 个不同的恶意家族,分别是 AnserverBot、Basebridge、Beanbot、Geinimi 和 DroidKungfu。

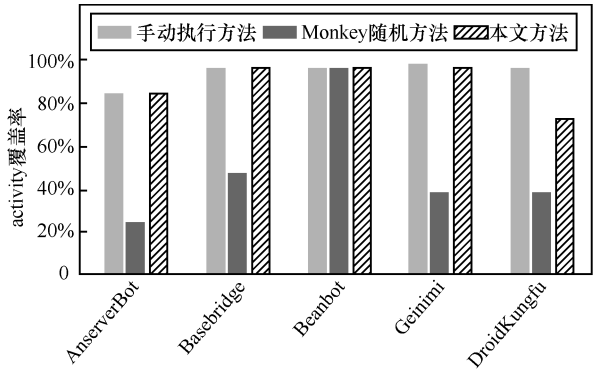


图 5 activity 覆盖率

从图 5 可以看出,本文所设计的 Android 恶意应用自动执行的方法和手动执行的方法在 activity 覆盖率上能达到几乎一样的结果,只是在执行 Droidkungfu 这个 Android 恶意应用时,activity 覆盖率稍微差了一些。但是自动执行方法不需要消耗人力和时间。此外,自动执行方法比 Monkey 随机执行的方法在 activity 覆盖率上要高出很多。这说明自动执行方法通过对所执行控件的类型进行识别后生成相应的执行动作,保证了 Android 恶意应用可以不中断地执行。其中,只有 Beanbot 恶意应用的 activity 覆盖率同另外两种方法一样,这是因为该应用只包含 1 个 activity。activity 覆盖率的结果也验证本文所提出的方法可以有效地执行 Android 恶意应用,尽可能多地触发其网络行为,从而可以提取到更加全面有效的行为特征。

3.3 特征提取结果

首先本实验验证 Jaccard 指标的阈值选取。分别选取 Jaccard 指标为 0.2、0.4、0.6、0.8 进行 10 倍交叉验证的聚类。聚类的结果见表 3。

从表 3 可以看出,当 Jaccard 指标从 0.2 增加到 0.6 时,聚类的准确率、精准率(precision)、召回率(recall)和 F 值也随着增加。在 0.6 时达到最高值,误判率随着降低,在 0.6 时达到最低值。当 Jaccard 指标继续增加到 0.8 时,聚类的准确率、精准率、召回率和 F 值有一定程度的降低,而误判率有一定程度的增加。因此,在聚类 Android 恶意应用产生的网络流量时,Jaccard 指标选择为 0.6 可以得到最优的聚类结果。



表 3 不同 Jaccard 指标值的聚类结果

Jaccard 指标	准确率	误判率	精准率	召回率	F 值
0.2	0.851	0.149	0.858	0.851	0.852
0.4	0.937	0.063	0.878	0.937	0.906
0.6	0.989	0.011	0.989	0.989	0.989
0.8	0.974	0.026	0.972	0.971	0.972

表 4 列举了自动提取行为特征中每一阶段的结果。Android 恶意应用的网络流量的行为特征提取是根据第 2 节所描述的方法进行提取的。

表 4 Genome 和未知类型数据集特征提取结果

	数据集	流量解析	流量聚类	特征生成	特征合并
Genome	数目/个	26 889	3 804	660	623
	簇数目/个	–	370	–	99
未知类型	数目/个	1 748	851	84	79
	簇数目/个	–	64	–	11

从表 4 的结果观察到,通过对 Genome 数据集中的 1 171 个 Android 恶意应用所产生的 HTTP 流量进行特征提取,一共提取了 660 条基于网络流量的行为特征。其中经过合并后得到 623 条不同的行为特征,并属于 99 个不同的簇。而未知类型的 Android 恶意应用数据集由于应用个数远少于 Genome 的应用个数。因此,所提取的行为特征也远少于从 Genome 数据集里面提取的特征,生成 84 条行为特征,合并后只有 79 条,分别聚类到 11 个不同的簇。表 5 列举了特征提取过程中每一步所需要的时间开销。

表 5 自动特征提取的时间开销分布

数据集	流量解析	流量聚类	特征生成	特征合并	合计
Genome	5 min 47 s	4.7 s	1.8 s	1.6 s	5 min 55.1 s
未知类型	29 s	1.4 s	1.1 s	1 s	32.5 s

从表 5 的结果可以看出,在自动特征提取的过程中,最为耗时的就是流量解析过程,两类数据集的流量解析时间基本上占据了总体时间开销的 90%以上。但是本文提出的方法主要用于线下分析,比如 Android 市场的应用审核。因此,分钟级别的时间开销应该是可以接受的。

在参考文献[21]中提出了对僵尸网络的 C&C 信道的网络进行聚类分析。该工作使用 X-means 算法对网络流量属性(如数据分组个数等)进行聚类分析。本实验将本文的方法与参考文献[21]的方法在流量解析和流量聚类的时间

开销方面进行对比,以 Genome 数据集为例,结果见表 6。

表 6 聚类分析时间开销对比

方法	流量解析	流量聚类	总共
本文方法	5 min 47 s	4.7 s	5 min 51.7 s
参考文献[21]的方法	5 min 43 s	59 s	6 min 2 s

从表 6 中的结果可以看出,两类方法在流量解析的时间开销上相差不大,因为流量解析工作主要是流量属性的提取,这两种方法都是使用同样的流量解析方法。但是在流量聚类阶段,本文方法的时间开销远低于参考文献[17]的方法。这是因为本文的聚类方法是通过预先设定的 Jaccard 指标值来判断两条流量是否具有相似度,并归类到同一个簇中。而参考文献[21]的方法需要经过多次迭代后才能把具有相似属性的网络流量归类到同一个簇。因此,本文的流量聚类的方法能更加快速地把具有相似属性的网络流量进行聚类。

图 6 以 8 类流行的 Android 恶意应用家族为例,展示了行为特征提取后各个家族所对应的行为特征数目。从图 6 中观察得到,从 Android 恶意应用生成的 HTTP 流中提取到的行为特征数目要多于 Android 恶意应用的数目。这是因为一个 Android 恶意应用一般具有多种不同的网络行为,比如窃取不同的隐私数据、接收 C&C 服务器的命令等。不同的网络行为映射为不同的行为特征。因此,行为特征的数目要多于应用的数目。

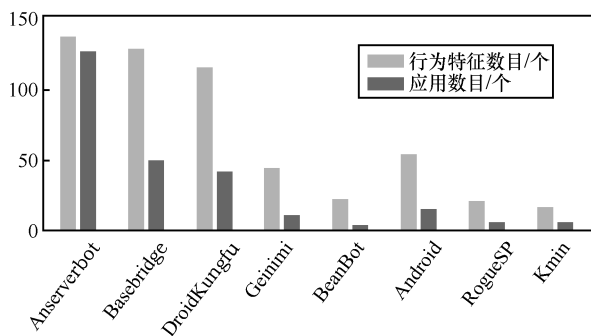


图 6 流行 Android 恶意应用家族行为特征数分布

3.4 Android 恶意应用检测

首先通过使用从 Genome 数据集里面提取到的行为特征,对未知类型的数据集中的 Android 恶意应用进行检测,并判断它们分别是属于哪些 Android 恶意应用家族。检测结果见表 7。

表 7 行为特征验证结果

Android 恶意应用家族	恶意应用数目/个
Anserverbot	13
ADRD	3
Basebrige	6
DroidKungfu	50
Plankton	10
DroidDreamLight	4
其他	10

从表 7 中观察到,未知类型的数据集中一共正常运行了 96 个恶意应用,并成功地判断出 86 个恶意应用分别所属的恶意应用家族。还有 10 个恶意应用没有成功判定,是因为行为特征提取的数据集中没有包含这些恶意应用家族。将在未来的工作中继续扩大数据集,争取包含更多的恶意应用家族,从而提高 Android 恶意应用的行为特征提取质量。此外,在未知类型的 Android 恶意应用中,DroidKungfu 恶意应用的数量是最多的。该恶意应用可以入侵 Android 手机存在的漏洞,并躲避目前杀毒软件的检测。

其次,为了验证白名单和广告名单对 Android 恶意应用特征提取的影响。对未过滤的 Android 恶意应用产生的网络流量按照前述方法进行特征提取,并使用这类特征和从过滤后的 Android 恶意应用中提取的特征一起对一个由 100 个从第三方市场下载的 Android 应用组成的数据集进行验证。该数据集通过 VirusTotal^[22]的验证发现一共存在 37 个 Android 恶意应用和 63 个 Android 正常应用。VirusTotal 是一个提供在线检测 Android 恶意应用的网站,其中集成了 40 多个商业反病毒软件。最后,根据两类特征的检测结果见表 8。

如表 8 所示,根据两类不同的提取特征都从 37 个恶意应用中检测到 32 个。但是通过白名单和广告名单过滤所提取出来的特征仅仅误判 5 个正常应用为恶意应用,而未通过白名单和广告名单过滤所提取的特征误判 21 个正常应用为恶意应用。该结果说明了通过白名单和广告名单过滤后所提取的特征更加能反映 Android 恶意应用

表 8 Android 应用检测结果

特征类型	检测到的恶意应用数目/个	误判为恶意应用数目/个
通过名单过滤提取的特征	32	5
未通过名单过滤提取的特征	32	21

的网络行为特征,在实际检测中具有较高的准确率和较低的误判率。

最后,为了验证本文方法的有效性,本实验将对比本文的方法和参考文献[10]所提出的 Android 应用识别的方法进行恶意应用检测的精准率和召回率,实验数据集使用未知类型中的 96 个 Android 恶意应用。其中,精准率和召回率的计算式如下所示:

$$precision = \frac{TP}{TP+FP}, recall = \frac{TP}{TP+FN} \quad (1)$$

其中,TP 表示检测的成功率,FP 表示检测的误判率,FN 表示检测的漏判率。比较结果如图 7 所示。

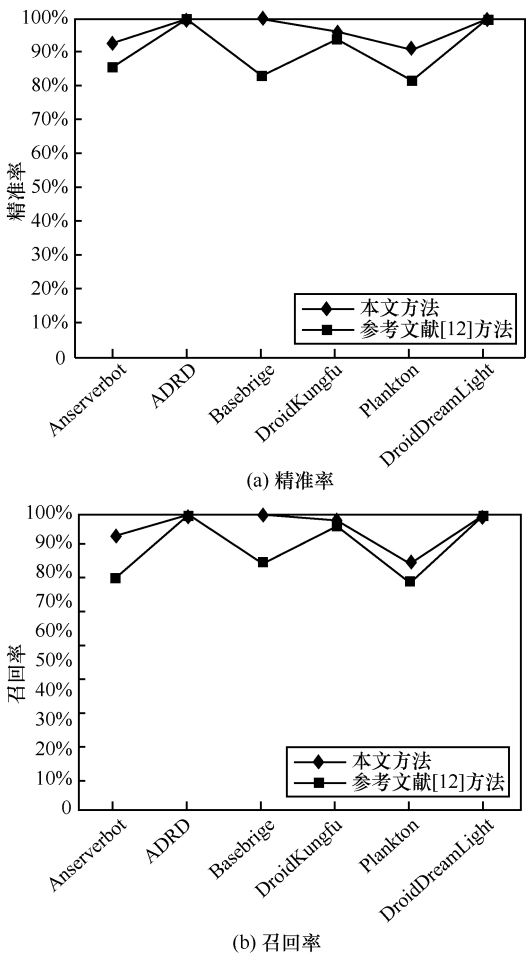


图 7 本文方法和参考文献[10]方法的比较结果



从图7可以看出,在检测6个不同Android恶意应用家族中的恶意应用时,本文方法在精准率和召回率两个指标上均能达到90%以上,而参考文献[10]的方法在这两个指标上只能达到80%左右的程度。这是因为参考文献[10]中的方法主要通过特征签名和时间窗口来生成网络行为特征,这样会导致把一些不属于该应用的流量误判为该应用所生成的网络流量,导致精准率和召回率有所降低。此外,在ADRD和DroidDreamLight两个恶意应用家族上,本文的方法和参考文献[10]的方法具有相同的精准率和召回率。这是因为这两个家族的应用数目较少,变种应用也较少,所以具有较高的精准率和召回率。

4 结束语

针对Android恶意应用提出了一种基于HTTP流的自动网络流量行为生成与特征提取的方法。首先介绍如何设计自动执行Android恶意应用的方法来解决手动执行方法中存在的耗时耗精力的不足;其次,通过对捕获到的网络流量进行流量解析、聚类,特征生成和合并等操作,提取到基于HTTP流的Android恶意应用行为特征。在实验阶段,使用两类不同的数据集对本文所设计的提取方法进行验证。实验结果表明,本文的方法不仅能有效地提取Android恶意应用的HTTP流量的行为特征,并且所提取的特征可以准确地识别未知的Android恶意应用。

参考文献:

- [1] 上年国内在网活跃移动智能设备数量达到8.99亿[EB/OL]. [2016-04-25]. http://www.ce.cn/culture/whcyk/cysj/201601/18/t20160118_8352012.shtml.
The number of online active smart devices reach 0.899 billion in first half of 2015 in China [EB/OL]. [2016-04-25]. http://www.ce.cn/culture/whcyk/cysj/201601/18/t20160118_8352012.shtml.
- [2] GRACE M, ZHOU Y, ZHANG Q, et al. RiskRanker: scalable and accurate zero-day Android malware detection [C]//The 10th International Conference on Mobile Systems, Applications and Services (MobiSys 2012), April 5-11, 2012, San Diego, CA, USA. New York: ACM Press, 2012:281-294.
- [3] 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 2014), June 11-15, 2014, San Diego, CA, USA. Washington: IEEE Computer Society, 2014: 199-210.
- [4] Linear SVM [EB/OL]. [2016-04-25]. <http://www.linearsvm.com>.
- [5] HAO S, LIU B, NATH S, et al. Programmable UI-automation for large-scale dynamic analysis of mobile apps [C]//12th International Conference on Mobile Systems, Applications and Services (MobiSys 2014), September 5-11, 2014, Bretton Woods, New Hampshire, USA. New York: ACM Press, 2014: 204-217.
- [6] ENCK W, GILBERT P, CHUN B, et al. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones[C]//9th USENIX Conference on Operating Systems Design and Implementation, December 9-12, 2010, Vancouver, BC, Canada. New Jersey: IEEE Press, 2010: 1-6.
- [7] Monkey [EB/OL]. [2016-04-25]. <http://developer.android.com/tools/help/monkey.html>.
- [8] WEI X, GOMEZ L, NEAMTIU I, et al. Profile droid: multilayer profiling of android applications [C]//18th Annual International Conference on Mobile Computing and Networking (MobiCom 2012), June 11-15, 2012, Istanbul, Turkey. New York: ACM Press, 2012: 137-148.
- [9] JIANG X, ZHOU Y. Dissecting android malware: Characterization and evolution [C]//2012 IEEE Symposium on Security and Privacy, May 20-23, 2012, San Francisco, USA. New Jersey: IEEE Press, 2012: 95-109.
- [10] SU X, ZHANG D, DAI S, et al. Mobile traffic Identification based on applications network signature [J]. International Journal of Embedded Systems, 2016, 8(2-3): 217-227.
- [11] DAI S, TONGAONKAR A, WANG X, et al. Network profiler: Towards automatic fingerprinting of android apps[C]// 32nd IEEE International Conference on Computer Communications, Infocom 2013, June 5-7, 2013, Turin, Italy. New Jersey: IEEE Press, 2013: 809-817.
- [12] XU Q, LIAO Y, MISKOVIC S, et al. Automatic generation of mobile App signatures from traffic observations [C]// 34th IEEE International Conference on Computer Communications, Infocom 2015, November 3-5, 2015, Hong Kong, China. New Jersey: IEEE Press, 2015: 1481-1489.
- [13] ZHOU Y, JIANG X. Dissecting android malware: characterization and evolution[C]//IEEE Symposium on Security and Privacy, June 5-9, 2012, San Francisco, CA, USA. New Jersey: IEEE Press, 2012: 95-109.
- [14] Tcpdump[EB/OL]. [2016-04-25]. <http://www.tcpdump.org>.
- [15] Tools[EB/OL]. [2016-04-25]. <http://developer.android.com/tools>.

- [16] Jaccard index[EB/OL]. [2016-04-25]. http://en.wikipedia.org/wiki/Jaccard_index.
- [17] Adrd [EB/OL]. [2016-04-25]. <https://www.microsoft.com/security/portal/threat/encyclopedia/entry.aspx?Name=TrojanSpy%3AAndroidOS%2FAdrd.A>.
- [18] Trojan: droid dream light [EB/OL]. (2011-07-23) [2016-04-25]. <http://tools.cisco.com/security/center/viewAlert.x?alertId=23296>.
- [19] Alexa [EB/OL]. [2016-04-25]. <http://www.alexa.com>.
- [20] GRACE M, ZHOU W, JIANG X, et al. Unsafe exposure analysis of mobile in-app advertisements [C]//Fifth ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec 2012), November 3–5, 2012, Tucson, Arizona, USA. New York: ACM Press, 2012: 101-112.
- [21] 苏欣, 张大方, 罗章琪, 等. 基于 Command and Control 通信信道流量属性聚类的僵尸网络检测方法[J]. 电子与信息学报, 2012, 34(8): 1993-1999.
- SU X, ZHANG D F, LUO Z Q, et al. Botnet detecting method based on clustering flow attributes of Command and Control communication channel [J]. Journal of Electronics & Information Technology, 2012, 34(8): 1993-1999.
- [22] Virus total[EB/OL]. [2016-04-25]. <https://www.virustotal.com/>.

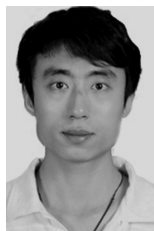
[作者简介]



罗亚玲(1977–),女,广东松山职业技术学院讲师,主要研究方向为 Web 开发与研究、移动应用。



黎文伟(1975–),男,博士,湖南大学信息科学与工程学院副教授,主要研究方向为可信系统与网络、网络测试。



苏欣(1983–),男,湖南大学信息科学与工程学院博士生,主要研究方向为移动应用安全、移动互联网大数据挖掘。