



一种对 Android 应用资源索引表混淆方案的实现

刘洋,文伟平

(北京大学软件与微电子学院,北京 102600)

摘要:由于 Android 应用具有被逆向、篡改、二次打包等风险,通过对 Android 应用中资源索引表文件进行混淆操作可以加大攻击者对 Android 应用逆向分析的难度。Android 应用中的资源索引表记录着该应用所有的字符串信息。通过对资源文件索引表格式的研究,提出了一种基于字符串置换的混淆方法,实现了对资源文件索引表的混淆,加大了攻击者对 Android 应用逆向分析的难度。

关键词:Android;资源文件索引表;混淆

中图分类号:TP309

文献标识码:A

doi: 10.11959/j.issn.1000-0801.2016252

An obfuscate implementation for Android resources index table

LIU Yang, WEN Weiping

School of Software and Microelectronics, Peking University, Beijing 102600, China

Abstract: Android application has risks of reverse, rewrite and repackage. Obfuscate Android resources index table will increase the difficulty of reverse analysis attack. Resources index table records the strings information in Android App. Through analyzing the structure of resources index table, a new obfuscate method was proposed to achieve obfuscate resources index table and increase the difficulty of reverse Android application.

Key words: Android, resources index table, obfuscate

1 引言

随着移动互联网时代的到来,传统 PC 设备逐渐退出大众视线,智能移动设备作为 PC 设备的替代品被大多数用户所接受。Android 系统凭借其开源的优势,占据了移动设备操作系统的主要份额。截至 2014 年 11 月,Android 官方应用市场 Google Play 上的应用数量已接近 140 万个^[1]。

由于 Android 应用的迅猛发展和安全问题的日益突出与反制措施极不协调^[2],大量攻击者将目光转移到逆向 Android 应用。攻击者在对应用实现逆向分析后,一方面可

以通过抄袭他人源码窃取劳动成果;另一方面可以通过阅读他人代码逻辑,找寻 Android 应用中的漏洞、修改应用源码、对应用实施攻击。通过对应用中的核心文件进行混淆操作,可有效保护 Android 应用程序开发者的知识产权,在一定程度上避免了 Android 应用程序被逆向分析、盗版以及恶意篡改^[3]。

当前对 Android 应用内 dex 文件的混淆技术较为成熟,有多种实现方案,且具有较高的兼容性。但是对于 Android 应用中资源文件索引表文件的混淆研究较少。当前资源文件索引表混淆方法使用范围最广的是由微信团

收稿日期:2016-07-31;修回日期:2016-09-25

基金项目:国家自然科学基金资助项目(No.61170282);信息网络安全公安部重点实验室基金资助项目(No.C14604)

Foundation Items: The National Natural Science Foundation of China (No.61170282), The Laboratory of the Ministry of Public Security Information Network(No.C14604)

队提出的,通过将 Android 应用包作为输入,修改资源索引表及 res 目录下其他资源文件的方法对资源文件进行混淆。微信团队对资源索引文件的混淆操作主要是将/res/drawable/hello.jpg 修改为/r/d/a.jpg 文件。经此类方法处理后的资源文件索引表长度会被压缩。本文通过对 Android 应用中资源文件索引表结构进行研究,实现一种在不修改资源索引表长度的前提下,对资源文件索引表进行混淆操作的方法。目前在 Android 应用 res 文件夹内资源文件的研究主要有:

- 参考文献[4]实现一种对 res 目录下资源文件名称及资源索引字符串的混淆操作;
- 参考文献[5]阐述了 Android 系统打包资源文件原理及安装 Android 应用时资源文件的载入过程;
- 参考文献[6]阐述了 resources.arsc 文件格式分析。

2 arsc 文件分析

Android 应用中的资源文件,分别存储在 apk 包下的 assets 文件夹和 res 文件夹下。存放在 res 目录下的资源文件会在 R.java 中生成对应的资源 ID,通过调用资源 ID 的方式可以对此类资源文件进行访问。存放在 assets 目录下的资源文件通过 AssetManager 类提供的 API,以字节流的形式被读取。在使用 AAPT 对 apk 应用进行打包时,res 文件夹中非 Bitmap 类型文本文件将被编译为二进制。

在 res 文件夹下,包括 animator、anim、color、drawable、layout、menu、raw、values 文件夹,其中 animator 和 anim 文件夹下存放应用中使用的动画文件,color 目录下存放配置应用的配色方案,drawable 目录内存放布局图形文件,layout 目录内存放布局文件,menu 目录存放菜单资源文

件,raw 目录存放不被编译为二进制的通用文件,values 目录内存放资源字符串与资源 ID 对应关系。在 AAPT 对 res 目录下文件进行打包时,会将 values 文件夹下的对应关系重新描述到名为 resources.arsc 的文件下,即资源索引表下,这个后缀为 arsc 文件就是 Android 应用的资源索引表。本文中提出的混淆方案基于对 resources.arsc 文件的混淆,为方便描述,使用 arsc 文件表示 resources.arsc 文件。

arsc 文件中存放着 Android 应用中所有的字符串资源,在安装一个 APK 应用包时,arsc 文件先于 AndroidManifest.xml 载入。由于 arsc 文件载入时机过早,通过对 arsc 文件进行全文加密,在载入前通过 hook 解密函数解密的过程不易被实现。对于 arsc 文件的操作大多使用对资源 key 混淆的方式进行加固。在对 arsc 文件结构的分析过程中,实现了一种对于 arsc 文件的新混淆方案。

3 arsc 文件混淆原理

对于 arsc 文件内字符串的混淆可以增加逆向难度,对其阅读源码的过程造成干扰,延长攻击者解读源码信息的时间。通过对 arsc 文件进行修改,可找到一种对资源 key 置换操作的混淆实现。

完成 arsc 文件混淆操作,需实现在对资源字符串进行置换混淆后不影响程序对于资源文件的查找。在对 Android 应用打包时,res 目录下的 XML 文件和 AndroidManifest.xml 文件会由文本文件格式编译为二进制格式。如图 1 所示,以 AndroidManifest.xml 为例,打包前的 AndroidManifest.xml 根据资源 key 和资源内容间建立关联关系,打包过程中,根据资源索引表内资源 key、资源 ID、资源内容的关系重新建立资源 ID 和资源内容的索引关

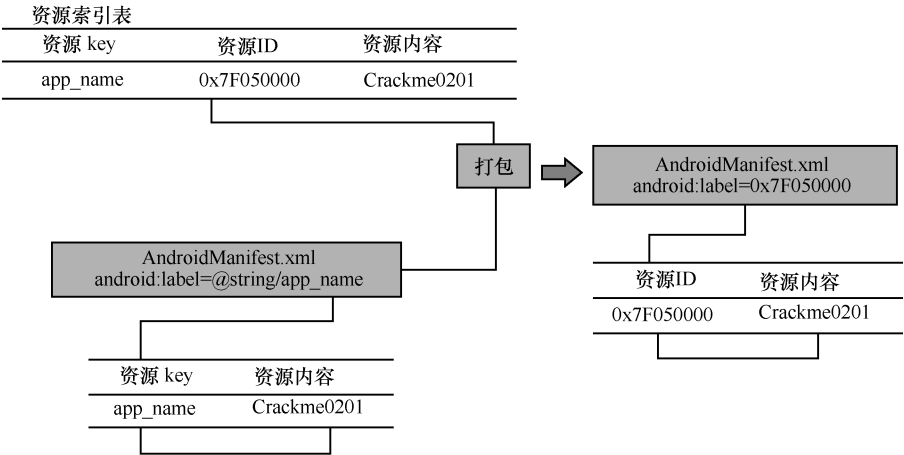


图 1 XML 文件编译二进制过程中资源内容索引更换示意



系。即在 Android 应用完成一次打包后,XML 中使用到的资源 key 不在程序运行时对资源的查找起作用,Android 应用运行时通过对二进制格式 XML 中资源 ID 的查找即可实现对资源内容的查找。

使用反编译软件将 Android 应用资源文件的恢复过程中,会根据 arsc 文件的索引表,通过资源 ID 和资源 key 的关联关系,找寻二进制的 XML 中资源 ID 对应的资源 key,然后将资源 ID 恢复为对应的资源 key。利用这一特性,只要不破坏资源 ID 和资源内容的关联关系,就不会影响运行时 XML 内对资源内容的调用。通过修改资源 ID 和资源 key 的对应关系,进而可以在反编译时置换资源 ID 与资源 key 的对应关系。资源 ID 和资源 key 的对应关系存储在 res 目录下的 arsc 文件下。

4 arsc 文件混淆方法实现

实现上述的资源文件混淆操作核心在于通过修改 arsc 文件中资源 ID 和资源 key 的对应关系实现。因此需对 arsc 文件格式进行分析,在实现过程中通过使用 010 editor,在 hex 模式下对 arsc 文件进行修改。将进行过混淆操作的 arsc 文件添加到修改后缀名为 zip 的 Android 应用包内,使用 resign.jar 对应用包进行重打包操作,得到可以运行的 APK 文件。整体流程如图 2 所示。由于文本形式的 XML 文件需使用资源 key 和资源内容的对应关系,因此只能将二进制编译的 XML 文件与混淆后的 arsc 文件进行重打包操作。

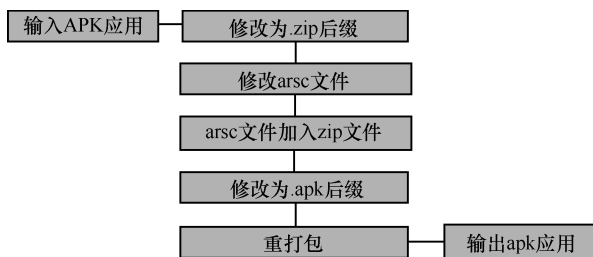


图 2 arsc 文件混淆生成新 Android 应用流程示意

修改资源 ID 和资源 key 的对应关系,需要了解在 arsc 文件中资源 ID 和资源 key 的关系是通过怎样的方式建立的。首先需要了解资源 ID 的构成,资源 ID 由 4 byte 构成,最高字节表示 package_id,向下一位字节表示 type_id,最后两字节表示该资源的 entry_id,通过该 3 类 ID 的组合可根据资源从属的包、类型及自身特性形成一个唯一的资源 ID。基于资源 ID 的唯一性,在同一个 Android 应用中,资源文件

可以被不同 package 共享。

在 arsc 文件结构中,先罗列 Android 应用中所有 package 内容,在 package 结构中声明该 package 下存在 type 的个数。在 type 内容下,声明罗列的 entry 个数。每个 entry 内部均存在一个名称为 ResTable_entry 的数据结构,每个 ResTable_entry 元素可对应唯一的资源 ID。ResTable_entry 元素中的 key 属性和资源 key 之间存在一个对应关系。key 属性内容中的 n 表示的是该资源 ID 对应于资源项名称字符串池中的字符串偏移数组中第 n 个元素,根据字符串偏移数组中的偏移量可找寻到资源 key 的值。其对应关系如图 3 所示。

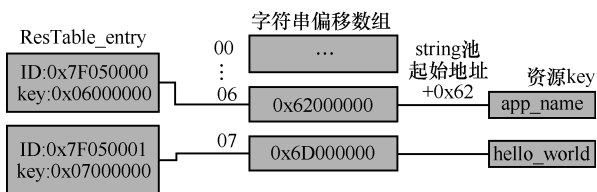


图 3 原始 arsc 文件根据资源 ID 查找资源 key 原理

如图 4 所示,新的置换混淆方法正是基于修改资源 ID 查找资源 key 的过程实现的。通过置换字符串偏移数组的内容,可以成功完成对资源 key 的置换操作。

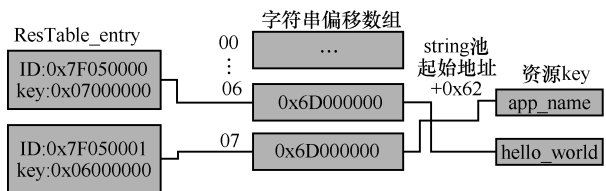


图 4 修改后 arsc 文件根据资源 ID 查找资源 key 原理

在对 arsc 文件格式了解后,打开 arsc 文件,使用 hex 编辑器定位到此资源文件索引表的字符串偏移数组。具体操作如图 5 所示,第一段内容是未混淆前的内容,第二段内容为混淆后的内容。通过置换字符串偏移数组内的偏移值完成混淆操作。每个资源位移可对应不同的资源类型,可以在不同资源类型之间置换资源 key。

5 arsc 文件混淆效果

通过对 arsc 文件的操作可对资源 key 的内容进行混淆置换,当通过反编译工具对资源 key 的内容查看时,得到的是已做混淆操作后的资源 key。图 6 中以 AndroidManifest.xml 为例,对比了在反编译后、混淆前和混

00 00 00 00	21 00 00 00	13 00 00 00	41 00 00 00
31 00 00 00	62 00 00 00	52 00 00 00	7B 00 00 00
8B 00 00 00	6D 00 00 00	8B 00 00 00	B3 00 00 00
A1 00 00 00	C3 00 00 00	E8 00 00 00	DD 00 00 00
EA 00 00 00	D3 00 00 00	05 01 00 00	F7 00 00 00
11 01 00 00	1C 01 00 00	28 01 00 00	38 01 00 00
42 01 00 00			
00 00 00 00	13 00 00 00	21 00 00 00	31 00 00 00
41 00 00 00	52 00 00 00	62 00 00 00	6D 00 00 00
7B 00 00 00	8B 00 00 00	A1 00 00 00	A8 00 00 00
B3 00 00 00	E8 00 00 00	C3 00 00 00	D3 00 00 00
DD 00 00 00	EA 00 00 00	F7 00 00 00	05 01 00 00
11 01 00 00	1C 01 00 00	28 01 00 00	38 01 00 00
42 01 00 00			

图 5 字符串偏移数组混淆操作前后对比

淆后的 AndroidManifest.xml 内容。路径 1 中的 AndroidManifest.xml 文件是对应于未进行 arsc 文件混淆的 Android 应用,路径 2 中的 AndroidManifest.xml 文件对应于已对 arsc 文件混淆过 Android 应用。混淆处理后的 Android 应用,资源 key 由 app_name 置换为 padding_large,资源 key 内容发生了明显的变化。此类混淆方法可应用于所有的资源 key 置换,包括对图片名称进行置换操作,res 文件夹下 drawable 目录内图片文件的名称对应于资源 key、图片存放路径对应于资源内容。

在混淆处理后再对打包的 Android 应用进行兼容性测试。测试的应用包括 3 类,分别是未经混淆处理的应用、经过微信团队的混淆工具 AndResGuard 处理后的应用以及使用本文混淆方法处理的应用。测试结果见表 1。

表 1 兼容性测试结构

类型	测试安装 设备数量/个	测试安装通过 设备数量/个
未混淆	50	49
本文混淆方法	50	49
AndResGuard 混淆	50	49

通过测试结果可知使用本类混淆方法处理后的应用文件和使用 AndResGuard 工具后的应用均可以正常被使用。有良好的兼容性。

6 结束语

通过修改 arsc 文件内资源项名称字符串池中的字符串偏移数组内容完成对于资源 key 的混淆置换。此类混淆置换不影响 Android 应用正常运行,依据来自于 Android 应用运行过程中在通过 XML 文件和 R.class 文件对资源内容

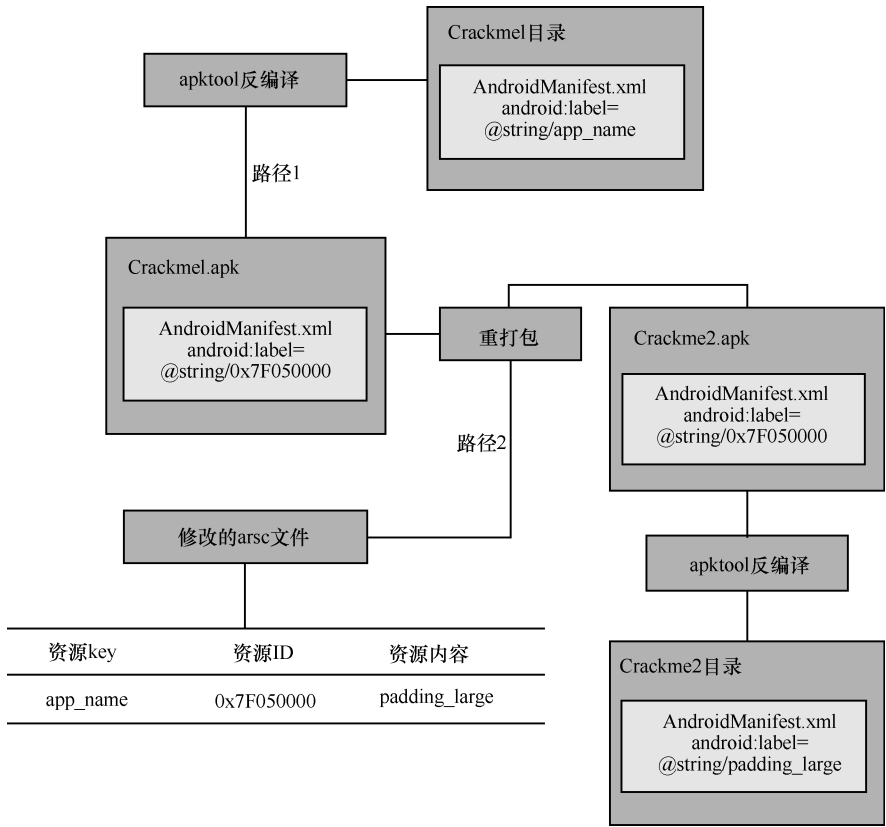


图 6 arsc 文件混淆前后示意



进行调用时,是通过资源 ID。此类混淆操作影响攻击者对 Android 应用进行逆向的依据来源于,在反编译工具运行时会根据 arsc 文件表恢复资源 key 与资源内容的关系。

AndResGuard 混淆工具相比这种通过修改字符串偏移数组内容完成对于资源 key 的混淆置换的方法对 arsc 文件的修改更大。此类置换混淆方法的欺骗性更强,由于资源 key 的内容均来源于原始的 Android 应用,攻击者在阅读资源 key 时更容易误以为是未进行过混淆的元素。但 AndResGuard 工具可自动实现对应用包混淆后进行二次打包操作,使用起来更方便。

对 arsc 文件进行混淆操作只是对 Android 应用 res 目录下资源文件加固的一种实现方法,对于 res 文件夹下 Bitmap 类型文件的加密和 XML 文件的加密将更有利于对抗攻击者对 Android 应用的逆向分析。

参考文献:

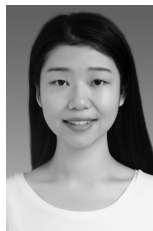
- [1] AppBrain Stats. Number of available Android applications[EB/OL]. (2015-02-16)[2016-06-13]. <http://www.appbrain.com/stats/>.
- [2] 卿斯汉. Android 安全研究进展 [J]. 软件学报, 2016, 27(1): 45-71.
QING S H. Research process on Android security [J]. Journal of Software, 2016, 27(1): 45-71.
- [3] 文伟平, 张汉, 曹向磊. 基于 Android 可执行文件重组的混淆方案的设计与实现[J]. 信息网络安全, 2016(5): 71-77.
WEN W P, ZHANG H, CAO X L. Design and implementation of the scheme of obfuscation based on the recombination of the Android executable file [J]. Netinfo Security, 2016 (5): 71-77.
- [4] Shwen. 微信 Android 资源混淆打包工具[EB/OL]. (2015-10-12)[2016-06-30]. http://mp.weixin.qq.com/s?__biz=MzAwNDY1ODY2OQ==&mid=208135658&idx=1&sn=ac9bd6b4927e9e82f9fa14e396183a8f#rd.

396183a8f#rd.

Shwen. A resource obfuscate tool for wechat Android App[EB/OL]. (2015-10-12)[2016-06-30]. http://mp.weixin.qq.com/s?__biz=MzAwNDY1ODY2OQ==&mid=208135658&idx=1&sn=ac9bd6b4927e9e82f9fa14e396183a8f#rd.

- [5] Beyond702. 手把手教你解析 resources.arsc[EB/OL]. (2016-06-30)[2016-07-01]. <http://blog.csdn.net/beyond702/article/details/51744082>.
Beyond702. Teach you analyze resources.arsc hand by hand[EB/OL]. (2016-06-30)[2016-07-01]. <http://blog.csdn.net/beyond702/article/details/51744082>.
- [6] 罗升阳. Android 应用程序资源的编译和打包过程分析[EB/OL]. (2013-04-15)[2016-07-01]. <http://blog.csdn.net/luoshengyang/article/details/8744683>.
LUO S Y. Compiling and packaging process for Android resource program analysis[EB/OL]. (2013-04-15) [2016-07-01]. <http://blog.csdn.net/luoshengyang/article/details/8744683>.

[作者简介]



刘洋 (1992-), 女, 北京大学软件与微电子学院硕士生, 主要研究方向为网络与系统安全。



文伟平 (1976-), 男, 博士, 北京大学软件与微电子学院副教授, 主要研究方向为网络攻击与防范、恶意代码研究、信息系统逆向工程。