doi 10.3969/j.issn.1671-1122.2016.02.003

一种安卓平台下提权攻击检测系统的设计与实现

张涛[1]，裴蓓[2]，文伟平[3]，陈钟[1]

（1. 北京大学信息科学技术学院，北京 100871 ；2. 信息网络安全公安部重点实验室，上海 201204 ；3. 北京大学软件与微电子学院，北京 102600）

摘 要 ：随着安卓系统的盛行，其安全性问题也逐渐成为人们关注的焦点。在安卓系统中进行敏感操作必须向系统申请相应的权限。虽然安卓系统中已经设计了与权限控制相关的系统模块，但是攻击者仍然可以借助系统漏洞或第三方程序漏洞进行提权攻击，进而非法使用一些超越其申请权限的功能。此种攻击不但对系统安全威胁较大，还具有一定的隐蔽性。文章通过对以往研究进行分析和创新，提出了一种新型的基于控制流检测和安卓敏感权限词典匹配的轻量级提权攻击检测方法，并在此基础上完成了自动化程度高、检测效率高的检测软件的设计与实现。

关键词 ：提权攻击 ；检测 ；安卓 ；控制流 ；权限字典

中图分类号 ：TP309 文献标识码 ：A 文章编号 ：1671-1122（2016）02-0015-07

中文引用格式 ：张涛，裴蓓，文伟平，等 . 一种安卓平台下提权攻击检测系统的设计与实现 [J]. 信息网络安全，2016 (2) ：15-21.

英文引用格式 ：ZHANG Tao，PEI Bei，WEN Weiping，et al. Design and Implementation of Privilege Escalation Attack Detecting System Based on Android Platform[J].Netinfo Security, 2016 (2) : 15-21.

Design and Implementation of Privilege Escalation Attack Detecting System Based on Android Platform

ZHANG Tao[1], PEI Bei[2], WEN Weiping[3], CHEN Zhong[1]

(1. *School of Electronics Engineering & Computer Sciences, Peking University, Beijing* 100871,*China*;2. *Key Lab of Information Network Security of Ministry of Public Security, Shanghai* 201204,*China*; 3. *School of Software and Microelectronics, Peking University, Beijing* 102600, *China*)

Abstract: Along with the rapid development of Android mobile operation system, its security issue has taken attentions. In the Android, it is necessary to apply the authorities to the system for sensitive operations. Although some system modules related to authority control have been designed in the Android, the attackers still can use the system vulnerabilities or third party program vulnerabilities to carry out the privilege escalation attack, and then illegally use some functions beyond their application permissions. This kind of attack is not only a great threat to the security of the system, but also has the feature of concealment. Based on the analysis and innovation on the past research, this paper proposes a new light weight method for detecting the privilege escalation attack, which uses the control flow detection and Android sensitive authority dictionary matching. In addition, detection software with high degree of automation and high detection efficiency is designed and implemented on the basis of privilege escalation attack detecting method.

Key words: privilege escalation attack; detection; Android; control flow; authority dictionary

收稿日期 ：2015-12-27

基金项目 ：国家自然科学基金 [61170282] ；信息网络安全公安部重点实验室资金 [C14604]

作者简介 ：张涛（1987—），男，江西，博士研究生，主要研究方向为系统与网络安全 ；裴蓓（1983—），女，安徽，研究实习员，硕士研究生，主要研究方向为信息网络安全认证和项目管理 ；文伟平 (1976—)，男，湖南，副教授，博士，主要研究方向为网络攻击与防范、恶意代码研究、信息系统逆向工程等 ；陈钟（1963—），男，江苏，教授，博士，主要研究方向为系统与网络安全、密码学。

通信作者 ：裴蓓 peibei@stars.org.cn

0 引言

安卓系统是一款基于 Linux 内核并开放源代码的移动平台操作系统。基于 Linux 内核使得安卓系统能够长期不断电运行，十分稳定 ；而开放源代码使得任何公司和个人都可以通过修改其源代码进行定制，从而使得安卓系统能够适应不同的硬件平台。自从安卓初创团队于 2005 年被 Google 收购以来，安卓系统借助 Google 强大的科研、资本和运营能力，逐步侵占其他移动平台操作系统市场占有率，成为近几年市场占有率第一的移动平台操作系统。大量开发者选择在该平台上进行移动软件开发，使得安卓系统成为全球应用最多的移动平台。

但是，由于安卓系统的开放性以及开发者水平良莠不齐，使得安卓系统存在许多安全问题和漏洞 [1-4]。安卓系统的安全问题和漏洞主要涉及通信系统、应用软件、隐私信息与设备安全四个方面。通信系统方面的安全风险包括攻击者在用户不知情的情况下拨打电话或挂断电话、发送垃圾短信等 [5]。应用软件方面则主要涉及恶意软件发起的攻击 [6]。隐私信息方面则多指用户隐私信息及敏感信息的泄露。设备安全方面则主要多指手机被盗而遭遇的信息泄露以及因此带来的经济损失 [7]。

1 提权攻击概念及模型

1.1 提权攻击概念

在所有针对安卓系统的攻击方式中，提权攻击是一种危害性较大的攻击方式。传统意义上，提权攻击指的是低授权、受限制的用户或者程序通过非法手段提升自身权限以实施控制计算机或者其他超出自身权限的行为。攻击者一旦攻击成功，系统原有的安全设置将无法保护用户的隐私和安全，给用户带来严重威胁 [8]。

安卓系统中的提权攻击有着普通提权攻击没有的特点。安卓系统中的提权攻击是指本来低权限或者不具备权限的恶意程序通过其他具有权限的第三方应用程序的漏洞而行使了第三方应用程序的功能。这种基于安卓系统的攻击行为由 DAVI[9] 等人第一次提出，他们在文献 [9] 中对安卓系统中的提权攻击做了如下定义 ：拥有较低（较少）权限的应用程序访问拥有较高（较多）权限的应用程序组件（活动、服务、内容提供器和广播接收器等）而不受到任何限制的攻击行为称为安卓系统中的提权攻击。这种攻击具有很强的隐蔽性，且攻击者常常并不破坏用户设备，仅仅盗取用户信息，因此很难被用户察觉。此外，这种攻击行为的前提是用户系统中存在着拥有较高权限，但同时也有漏洞的第三方软件，具有一定的限制性，因此也一直未受到人们的关注。随着智能手机用户的增加以及人们对移动设备依赖性的增强，越来越多的手机中保存着用户越来越多的个人信息，如银行账户信息、个人身份信息等，这大大增加了安卓系统提权攻击的危害。

2009 年 ,ENCK[4] 等人首次系统介绍了安卓安全机制 , 公布了一款名为 Kirin 的应用安装工具 , 并第一次采用静态检测组合策略 , 即在安装应用的时候就对其申请的所有权限进行审核，如果违背了配置策略，就拒绝对其进行安装。虽然该工具有效阻止了申请敏感权限的程序的安装，但是其误报率较高，原因在于仅依据已知危险软件的权限申请组合并不能够判断出申请同样权限的其他应用也为危险软件。

在此之后，FELT[10] 等人开发出 Stowaway。它的原理是通过反编译安卓应用程序代码，检测其调用的系统 API，并与 manifest.xml 文件中声明使用的权限进行比较，查看应用程序是否为危险程序。该工具误报率同样较高，因为开发人员对系统 API 所需要的权限了解不足，常常申请超出必要的权限，因此在配置文件中申请过高权限也不能完全证明其为恶意程序。

随后，对提权攻击的研究扩展到多个应用之间的串谋攻击，而不仅仅局限于对单个应用的研究 [11]。

FELT 等人提出 IPC Inspection，其基本方法是当应用程序接收其他应用程序的消息时，它必须减小自己使用的权限集合，删去其他应用程序没有的权限，以防提权攻击。

DIETZ[12] 等人在 IPC Inspection 的基础上提出 Qurie。Qurie 是安卓安全机制的一个扩展，它提供了一个轻量级的来源系统以防止所谓的混淆代理人攻击。其基本思路与 IPC Inspection 类似，不同点在于，当收到低权限的应用程序的消息后，Qurie 会让开发者自己来决定如何处理。由于开发者可能并不具有软件安全的相关知识，因而无法对复杂情况做出正确的判断。

在国内，于达 [13] 首次提出监视代码函数里的参数列表，以查看其是否涉及敏感信息，并以此将代码函数分成权限

泄露函数和隐私泄露函数两种。除了监视泄露函数中的参数列表，于达还将 Intent 劫持予以考虑，但其并未就如何构建控制流图进行详细的说明。

1.2 安卓系统下的提权攻击模型

DAVI[9] 等人在 2011 年首次提出安卓系统中的提权攻击时给出了如图 1 所示的提权攻击模型。

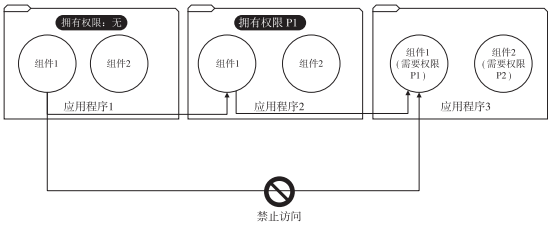

图1 提权攻击模型

图 1 显示，3 个应用程序分别运行在 3 个安卓系统提供的 Dalvik 虚拟机上。正常情况下，它们只能分别访问应用自身内部的信息、图片等资源，不可以访问其他应用程序的资源。但是当有合理的需求需要访问其他应用程序的资源时，就需要应用程序之间的合作。图 1 中，应用程序 1 启用不需要任何权限，自身也不拥有任何权限 ；应用程序 2 启用不需要任何权限，自身拥有权限 P1 ；应用程序 3 的组件 1 启用需要权限 P1，组件 2 启用需要权限 P2。应用程序 1 虽然不能直接访问应用程序 3 的组件 1，但是可以访问应用程序 2 的组件 1。应用程序 2 的组件 1 拥有访问应用程序 3 的组件 1 的权限，那么以应用程序 2 作为跳板，应用程序 1 就可以访问应用程序 3 的组件 1，从而执行自身权限范围之外的功能，也就完成了一次提权攻击。

图 1 中应用程序 2 是具有漏洞的应用程序，起到了跳板的作用，对应用程序 1 的提权攻击负有责任。如果应用程序 2 强制要求调用它的内部组件时都必须拥有权限 P1，就可以完全阻止这种攻击的发生。开发人员可以在 Manifest.xml 中直接静态声明需要的权限 P1 或者在代码里使用 checkPermission() 等 API 函数来动态审查权限。但是大多数开发者并不是信息安全专家，甚至对于信息安全一无所知，他们没有意识到或者不在乎这种攻击的发生，因为即使存在这些漏洞，对于应用程序本身的运行没有任何影响。但是这种攻击可能给用户带来巨大的安全风险，攻击者可以通过这种攻击方式获得用户的隐私信息、银行账户信息，或者进行连接互联网、发送恶意短信、拨打恶意电话等一系列恶意操作。

提权攻击主要包括操作型攻击（如打电话、发短信等）以及数据型攻击（获取敏感数据）两种。

1）操作型攻击

攻击源发出指令最终导致某种越权操作。这类攻击不需要返回数据给攻击源，所以难以追踪。操作型攻击模型如图 2 所示。

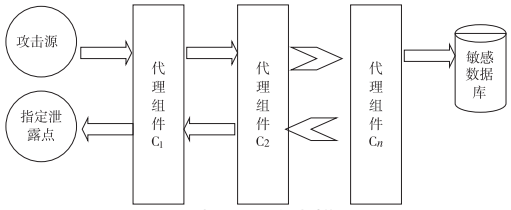

图2 操作型攻击模型

2）数据型攻击

攻击源发出指令后，从敏感数据库中获取敏感信息并返回，或者将敏感信息传回指定泄露点。这类攻击必须返回敏感数据，所以容易跟踪与监控。数据型攻击模型如图 3 所示。

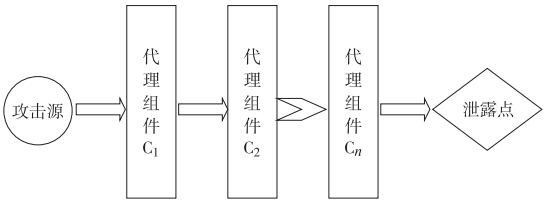

图3 数据型攻击模型

2 检测系统总体设计

图 4 为本文提权攻击检测系统的基本工作原理图。由图 4 可知，该系统主要包括两大核心模块 ：申请权限分析模块和应用程序控制流图构建模块 [14]。

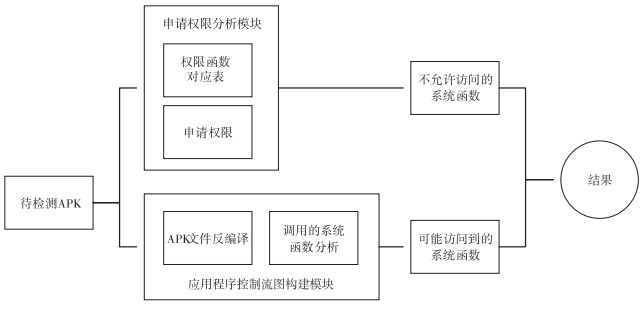

图4 提权攻击检测系统工作原理图

1）申请权限分析模块

Manifest.xml 是整个应用程序的配置文件，应用程序使用的所有权限、用户自定义的权限以及各个组件需要被哪些权限保护等都需要在该文件中进行声明。因此，针对 Manifest.xml 文件进行分析是整个检测系统最基础的部分。完成此分析后，我们将得到一个完整的权限申请列表，并

由此映射得到所能使用的函数列表，从而确定不能访问的系统函数。申请权限分析模块就是针对 Manifest.xml 文件进行分析。

2）应用程序控制流图构建模块

如图 4 所示，从提交待检测 APK 到 APK 文件反编译再到调用的系统函数分析主要是基于 Smali 文件解析来实现的。

针对应用程序的 APK 文件进行反编译将得到 Smali 文件，该文件中保存着应用程序反编译后得到的代码。分析该文件可以得到应用程序使用的函数，调用的系统函数、参数等信息，并以此得到系统函数调用列表与调用顺序，也就是控制流图。

2.1 Manifest.xml文件解析

1）Manifest.xml 文件结构

Manifest.xml 配置文件是每个安卓程序必需的文件。它位于整个安卓应用开发工程项目的根目录，包含了安卓应用程序的每一个组件（活动、服务、内容提供器和广播接收器等），并使用 Intent 过滤器和权限来确定这些组件之间以及这些组件和其他应用程序是如何交互的。

图 5 为 Manifest.xml 文件的粗略结构。

```
<?xml version="1.0" encoding="utf-8"?>
<manifest >
    <uses-permission />
    <permission />
    <application
        android:permission="...">
        <activity
            android:permission="..."
            anprod:exported="...">
            <intent-filter>...</intent-filter>
        </activity>
        <service
            anprod:permission="..."
            anprod:exported="...">
            <intent-filter >...</intent-filter>
        </service>
    </application>
</manifest>
```

图5 Manifest.xml文件结构

其中涉及权限的标签分别有：

（1）<uses-permission>。在安卓系统的安全模型中，应用程序在默认情况下不可以执行任何对其他应用程序、系统或者用户有负面影响的操作。如果应用程序需要执行某个操作，就要声明使用这个操作对应的权限，使用该标签可以完成对权限的声明。

（2）<permission>。应用程序也可以自定义属于自己的 permission 或属于开发者的用同一个证书签名的 permission。定义一个 permission 就是在 Manifest.xml 文件中添加一个 <permission> 标签。

（3）<android:permission>。如果要将一个应用程序或者组件设置为权限保护的，也就是说只有具备要求的权限时才能访问该程序或组件，则需要设置这个标签。

（4）<android:exported>。如果要将一个组件设置为私有的，即无论何种权限都不许访问此组件，则需要设置此标签。当 <android:exported> 为 false 时，组件是私有的。

（5）<intent-filter>。一个应用程序的 3 个核心组件（活动、服务和广播接收器）都是通过 Intent 消息来激活的。<intent-filter> 标签的主要作用就是区分 Intent 消息中，哪些隐式 Intent 消息可以接受响应，哪些将被拒绝接受响应。需要注意的是，<intent-filter> 对显式 Intent 消息起不到任何区别和过滤作用，所以该标签也不能完全保证组件的安全。

2）Manifest.xml 分析过程

系统主要通过分析 Manifest.xml 文件对 APK 使用的权限做一个初步判断，并根据所声明的权限输出危险控件列表和声明的可使用权限列表两个列表，用来交叉比对出敏感权限和安全权限。分析 Manifest.xml 文件所使用的决策树如图 6 所示。

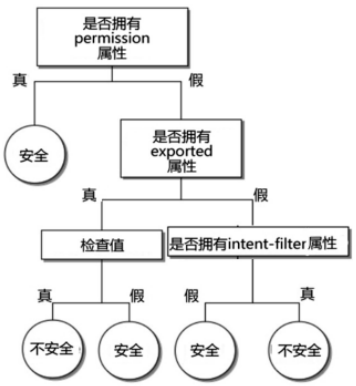

图6 Manifest.xml文件分析决策树[13]

2.2 Smali文件解析

对于一个未加密的 APK，有两种方式将其转换为可读代码：一种是通过 APK 反编译工具 Apktool 将其转换为 Smali 代码；另一种是将 APK 文件解压缩，使用 Java 反编译工具 JD-GUI 打开其中的 classes.dex 文件，将其反编译成 Java 代码。

这里做了一个假设 ：假设原始 APK 总是不透明的，即假设不知道 APK 内部处理和操作流程。但是，无论使用怎样的操作流程，如果 APK 希望采取某些行为（这里特指恶意行为），那么它不可避免地需要调用系统中的函数，否则仅在 APK 内部操作是无法对系统造成实质上的破坏的。将 APK 所调用的所有非该 APK 中的函数（也就是系统中的函数）看作是该 APK 与系统的“ 接口 ”，如果能够掌握该 APK 所有“ 接口 ” 的合集，那么我们就可以预测该 APK 在系统中的最大接触范围，从而对其破坏性活动进行预测。因此，重点应放在如何从 APK 中提取出这些“ 接口 ”，而不是放在 APK 之间函数的相互调用。

本文使用 Samli 文件进行解析，从中获取以下信息 ：

1）该 Smali 代码对应类的名称。

2）该类所包含的所有函数。

3）该类所调用的所有系统函数。

类名位于 Smali 文件第 1 行，并以 .class 开头。对于任何一个单独的 Smali 文件，其所代表的类是唯一确定的。在得到类后，通过分析类中的各个方法（函数），可以得到类中所有调用系统函数的语句。系统函数的调用语句均以 invoke 开头，所调用的函数以虚函数表中的序号标示。

3 检测系统的实现

3.1系统概述

本系统是一款基于安卓系统的针对提权攻击的静态检测工具，主要分为服务器端和客户端两部分，其总体框架如图 7 所示。

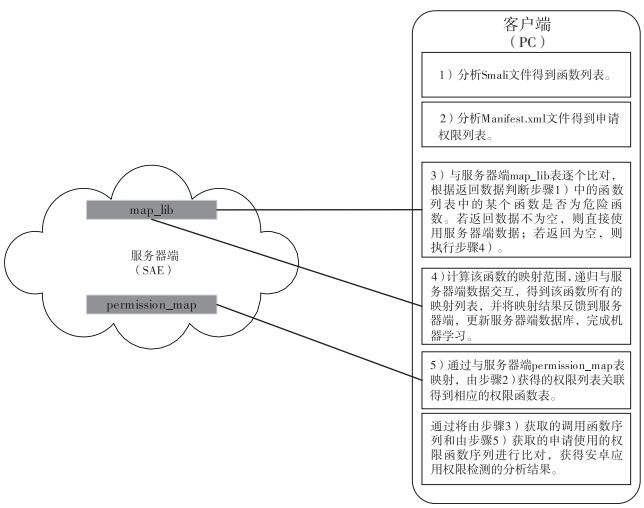

图7 系统总体框架图

1）服务器端

服务器端在系统中起数据存储、更新和查询的作用。服务器端运营于新浪云计算平台 SAE 上，负责为客户端提供所需的数据，并根据客户端返回的结果更新数据。服务器端运行和维护了两张关于安卓系统权限函数的数据库表 map_lib 和 permission_map。map_lib 包含 90848 条数据，主要包括每一个函数的下一步映射函数，如果映射函数为权限敏感函数，则还要包括其涉及的全部敏感感染路径。permission_map 为权限映射表，主要负责将每一个通过查找敏感权限词典得到的敏感权限项映射成为相应的权限操作函数，涉及到 757 条数据。

对于 map_lib 的维护，客户端首先会询问服务器端某个函数是否为安全函数，即查看 map_lib 的相应 father_id 对应的 danger 字段数据是 false(安全)、true(敏感) 还是 null（未曾进行映射运算）。如果返回前两种结果，那么客户端可以立刻得到函数状态 ；如果返回 null，客户端需要计算该函数的映射范围，并将结果返回服务器端，更新 danger 字段和 path 字段 (由于我们只关心敏感函数路径，且一般函数的映射范围都非常巨大，为减轻数据库压力，我们只关心 danger 字段为 true 的 path 字段的值)。

对于 permission_map 的维护，客户端首先将所有 757 个敏感函数 id(即 permission_map 的 id 字段) 存储到内存中，供映射计算使用。在分析 Manifest.xml 时，将所有 permission 权限映射成函数 id 并返回给客户端。在权限映射这一步中，会有一个 permission 对应多个函数 id 的情况，这里我们将所有的 id 都返回给客户端。

2）客户端

客户端可以运行在 Windows vista/7/8 系统上，主要有两个功能模块 ：Manifest.xml 文件分析模块和 Smali 文件解析模块。Smali 文件解析模块负责从输入的 Smali 文件夹中区分出 Smali 文件与 Manifest.xml 文件 ；通过分析 Smali 文件得到函数调用列表，将调用的函数与服务器端进行数据交互，区分出安全函数与敏感函数，如果为敏感函数，则向服务器端查询其敏感感染路径 ；分析 Manifest.xml 文件得到安卓应用申请的权限，如果未在 Manifest.xml 中显式地声明申请此权限，但在 Smali 文件中却调用了涉及该权限的函数，那么系统将认为遭到了提

权攻击，提出警告。

客户端分析流程如图 8 所示。

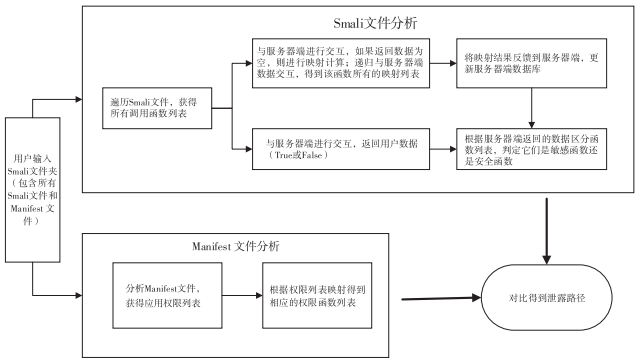

图8 客户端分析流程图

3.2 详细设计

1） Manifest.xml 解析

系统对 Manifest.xml 进行分析并将读取的数据存储到以下数据结构中 ：

```
typedef struct MainfestHead
{
    CString  head_xmlns; // 定义安卓系统命名空间，使得安卓系统中各标准属性能在文件中使用，提供大部分元素中的数据
    CString  head_package; // 本应用使用的 Java 主程序包名
    CString  head_sharedUserId; // 表明数据权限。默认情况下，安卓系统中每个 APK 分配一个唯一的 User Id，因此默认禁止不同的 APK 访问共享数据。如果要共享数据，第一可以采用 shanre preference 方式，第二可以采用 shareUserId 方式，则这些 APK 之间就可以共享数据了
    CString  head_sharedUserLabel; // 一个共享的用户名，只有在设置了 shareUserId 的情况下才有意义
    CString  head_versionCode;// 主要用于版本升级，为 int 类型。第一个版本定义为 1，以后递增，这样只要判断该值就能确定是否需要升级。该值不显示给用户。
    CString  head_versionName ;// 字符串值，代表应用程序的版本信息，需要显示给用户，如 1.1、2.1 等版本信息。
    CString  head_installLocation;// 通过设置该属性可以使得开发者以及用户决定程序的安装位置。选择设置 prefereExternal，会优先考虑安装到 SD 卡上，如果存满，则选择 ROM ；选择设置 auto，系统会根据存储空间自适应 ；选择设置 internalonly，则必须安装到内部才能运行
}MainfestHead;

typedef struct UsesPermission
{
    CString androidName[MAX_USES_PERMISSION];
    int Length;
}UsesPermission;
```

MainfestHead 主要用于存储 Manifest.xml 的基础信息，UsesPermission 用于存储用户申请的权限列表。

2） 函数映射计算

函数映射计算十分困难，函数量大，调用关系复杂，通过建立包含多级缓存的机器学习方法可以大大降低计算时间。一般认为，一个函数只需要遍历一次即可确定其是否安全。通过机器学习，能够避免函数的重复遍历，从而加快函数映射的计算速度。测试表明，一般 APP 分析时间约为 3 分钟。

4 检测系统的测试

运用本文检测系统对 Google Play 平台中的 1179 个应用程序进行了检测，并抽样进行了手工分析，以此测试提权攻击检测系统的实际运行效果，重点测试运行效率和误报率。

4.1 误报率

经过系统检测，有 613 个应用程序在 Manifest.xml 文件分析过程中被检测出含有漏洞，申请了敏感权限。系统对这些应用程序做了更进一步的检测，最终有 123 个应用程序被检测出含有过多的权限，这些权限可以被攻击者利用发动提权攻击。从 123 个报警的应用程序中随机选取 20 个应用程序进行人工分析，主要是对报警的路径进行分析，以验证系统检测的准确性。结果在这 20 个报警的应用程序中，9 个真正存在提权攻击漏洞，具有安全威胁，其他 11 个应用程序都是系统误报。

表 1 给出了 3 种检测工具的实验结果对比。从表 1 可以看出，本文检测方法（CoChecker）的误报率相较于其他两种主流检测软件有了一定的提升，处于中等水平。

表1 3种检测工具实验结果对比

工具名称	程序数目	警告次数	正确警告数		错误警告数	错报率
			单组件	多组件		
ComDroid	50	179	14	0	165	92.2%
DroidChecker	1179	30	15	0	15	50%
CoChecker	1179	131	123	11	22	16.4%

通过分析得出，造成系统误报的主要原因包含以下几点 ：

1） 一些安卓系统的 API 函数的敏感性不能被唯一确定。例如，函数 print() 被认为是一个可能泄露用户隐私数据的函数，但它大多数情况下还是进行正常显示，只有极少数情况下被攻击者利用。

2） 安卓敏感权限词典是查询安卓官方文档得到，在它的基础上又映射得到敏感函数列表。由于工作量的关系，第 1 版的敏感权限只有 403 个，其中也有一定的提升空间。

4.2 效率

本文系统在 36 秒内完成了 4 M 大小的应用程序的分析，读取 536 个类 13249 个函数，包含 3583 个调用函数，

其中有 1096 个函数在敏感权限词典中需要进行分析，所有函数在 58 秒内完成分析。在分析的 1096 个函数中，共有 1078 个安全函数，18 个敏感函数，并得出泄露路径。

测试结果表明，本文系统的检测效率较高，能够在可以容忍的时间范围内完成对较大程序的检测。

5 结束语

本文首先对安卓系统的整体架构以及安全机制进行了详细分析，通过对安卓系统各组件的安全机制及其局限性进行阐述，引出安卓系统存在的诸多安全漏洞及安全风险。

结合大量调研和相关实践工作，文章提出了一种新型的基于控制流检测与安卓敏感权限词典匹配的轻量级提权攻击检测方法，首次给出详尽的控制流图构建方法，并在此基础上实现了一款自动化程度较高、检测效率高的检测软件。

最后，通过对 1179 个安卓应用检测结果进行分析进而验证该软件设计指标的合规性。实验证明，该提权攻击检测系统无论在降低误报率上还是自动化程度上，较传统的安卓应用提权攻击检测工具都有一定的提升。本文仍有进一步的工作要做 ：应用程序控制流图的构造准确性及敏感权限函数的范围还可以继续提升，从而进一步提升该提权攻击检测系统的命中率。（责编 马珂）

参考文献:

[1]360 互联网安全中心 .2015 年第三季度中国手机安全状况报告 [EB/OL]. http:// zt.360.cn/1101061855.php?dtid=1101061451&did=1101460794, 2015 - 10 - 22.

[2] ONGTANG M, MCLAUGHLIN S, ENCK W, et al. Semantically Rich Application - centric Security in Android [J]. Security and Communication Networks，2012，5 (6) : 658 - 673.

[3] 廖明华，郑力明 .Android 安全机制分析与解决方案初探 [J]. 科学技术与工程，2009，26（11）：6351 - 6354.

[4] ENCK W, ONGTANG M, MCDANIEL P. Understanding Android Security[J].IEEE Security & Privacy, 2009 , 7 (1) : 53 - 54.

[5] SHABTAI A，KANONOV U，ELOVICI Y. Intrusion Detection on Mobile Devices Using the Knowledge Based Temporal - abstraction Method[J]. Systems and Software, 2010, 83 (8) :1527 - 1536.

[6] 张帆，钟章队 . 基于权限分析的手机恶意软件检测与防范 [J]. 信息网络安全，2015（10）：66 - 73.

[7]SHABTAI A, FLEDEL Y, ELOVICI Y, et al. Using the KBTA Method for Inferring Computer and Network Security Alerts from Timestamped, Rawsystem Metrics[J]. Computer Virology, 2009, 8 (3): 267 - 298.

[8] 闫梅，彭新光 . 基于 Android 安全机制的权限检测系统 [J] . 计算机工程与设计，2013，34（3）：854 - 858.

[9] DAVI L，DMITRIENKO A，SADEGHI A R, et al. Privilege Escalation Attacks on Android[M]. Berlin : Springer, 2011.

[10]FELT A P, GREENWOOD K, WAGNER D. The Effectiveness of Install - time Permission Systems for Third - party Applications[EB/OL]. http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS - 2010 - 143.pdf, 2010 - 12 - 3.

[11] 刘智伟，孙其博 . 基于权限管理的 Android 应用行为检测 [J]. 信息网络安全，2014（6）：72 - 77.

[12] DIETZ M, SHASHI S, et al. Quire: Lightweight Provenance for Smart Phone Operating Systems[J]. Computing Research Repository, 2011, abs/1102 (2) : 2334.

[13] 于达 . 一种针对 Android 系统提权攻击的检测方法研究与实现 [D]. 北京 ：北京大学，2013.

[14] 张金鑫，杨晓辉 . 基于权限分析的 Android 应用程序检测系统 [J]. 信息网络安全，2014（7）：30 - 34.

《信息网络安全》杂志记者证2015年度核验人员名单公示

根据《新闻记者证管理办法》的有关规定、新闻出版广电总局《关于开展新闻记者证 2015 年度核验工作的通知》（新广出办发〔2015〕140 号）和上海市新闻出版广电局《关于本市开展新闻记者证 2015 年度核验工作的通知》（沪新出报〔2016〕2 号）要求，《信息网络安全》杂志已对持有新闻记者证人员进行了严格审核，现将通过年度核验的人员名单公示如下：

关非（记者证号 ：K31185955000002）；

程斌（记者证号 ：K31185955000001）；

上海市新闻出版局举报电话 ：021 - 64339117。

《信息网络安全》杂志

2016 年 2 月 10 日