静态分析工具SuperAndroid简介

Last edited by **caoquanli** 1 month ago

Table of Contents

- 1. SuperAndroid简介
- 2. 安装
  - 2.1. 从github上克隆项目
  - 2.2. 下载工具
- 3. 用法
  - 3.1. 命令行指令使用说明
  - 3.2. 脚本示例(demo)
  - 3.3. 报告格式
- 4. SuperAndroid漏洞检测一览表（以QQ音乐APK为例）
- 5. SuperAndroid验证测试
  - 5.1. 定位权限测试：
  - 5.2. 相机权限测试：
  - 5.3. Exported activity测试
  - 5.4. 互联网权限测试：
  - 5.5. 拨打电话权限测试：
  - 5.6. IP信息泄漏漏洞
  - 5.7. 阅读手机状态权限
  - 5.8. 录制音频权限测试
  - 5.9. 发送短信权限测试
  - 5.10. 阅读联系人权限
  - 5.11. 写入外部存储权限
  - 5.12. 执行系统命令
  - 5.13. 弱算法
  - 5.14. 抛出的异常
  - 5.15. Rooted设备检测
  - 5.16. 未检查的日志输出
  - 5.17. sleep方法漏洞
  - 5.18. 编码安全漏洞

v1.0, 2019-1-8
本文对SuperAndroid的用法以及适用场景展开说明。

1. SuperAndroid简介

SuperAndroid是一个可以在 Windows，MacOS X和Linux中使用的命令行应用程序，它可以分析 .apk文件来搜索漏洞。它通过解压缩APK并应用一系列规则来检测这些漏洞来实现此目的。

2. 安装

SuperAndroid提供了针对于 Windows8.1+ (64位)， MacOS X(64位)和 Linux(64位)三种不同系统的二进制文件。如果需要支持32位版本的文件需要 从源代码编译，还需要另外使用 rustup.rs来安装 Rust。

注:SuperAndroid 运行需要基于jdk1.7+

以Linux版本安装为例：

2.1. 从github上克隆项目

```
https://github.com/SUPERAndroidAnalyzer/super.git
```

2.2. 下载工具

选择适合自己主机系统的二进制文件进行下载，完成之后进行安装。

Windows8.1+ (64位)， MacOS X(64位)， Linux(64位)

3. 用法

在命令行界面输入指令

```
$ super-analyzer -h
```

若之后显示以下内容则安装成功

```
SUPER Android Analyzer 0.5.0
SUPER Team <contact@superanalyzer.rocks>
Audits Android apps (.apk files) for vulnerabilities
```

```
USAGE:
           super-analyzer [FLAGS] [OPTIONS] <package>
```

```
FLAGS:
        --bench       Show benchmarks for the analysis
        --force       If you'd like to force the auditor to do everything from the beginn
ing
           -h, --help        Prints help information
        --html        Generates the reults in HTML format
        --json        Generates the reults in JSON format
        --open        Open the report in a browser once it is complete
            -q, --quiet       If you'd like a zen auditor that won't output anything in s
tdout
           -a, --test-all    Test all .apk files in the downloads directory
            -V, --version     Prints version information
            -v, --verbose     If you'd like the auditor to talk more than necessary
```

```
OPTIONS:
        --dex2jar <dex2jar>                    Where to store the jar files
        --dist <dist>                          Folder where distribution files will be ex
tracted
        --downloads <downloads>                Folder where the downloads are stored
        --jd-cmd <jd-cmd>                      Path to the jd-cmd file
        --min-criticality <min_criticality>    Set a minimum criticality to analyze (Crit
ical, High, Medium, Low)
        --results <results>                    Folder where to store the results
        --rules <rules>                        Path to a JSON rules file
        --template <template>                  Path to a results template file
            -t, --threads <threads>                Number of threads to use, by defau
lt it will use one thread per logical
                                                   CPU core
```

```
ARGS:
        <package>    The package string of the application to test
```

快捷使用指令如下：

```
$ super-analyzer ~/home/test.apk --open
中间参数为需要分析的APK文件路径，最后一个参数表示分析完毕后在浏览器中自动打开HTML报告，（生成的报告在当前目录下的results文件夹中）
```

3.1. 命令行指令使用说明

```
$ super-analyzer ~/home/test.apk
分析apk文件，报告默认生成html格式，所处路径默认为当前路径下的results文件夹中
```

```
$ super-analyzer ~/home/test.apk --open
增加参数 --open  分析apk文件，报告默认生成HTML格式,且分析完毕后自动打开报告
```

```
$ super-analyzer -v  ~/home/test.apk
增加参数 -v 分析过程中会显示一些不必要的细节，
```

```
$ super-analyzer --json  ~/home/test.apk
设置报告以json格式进行显示，所处路径默认为当前路径下的results文件夹中
```

3.2. 脚本示例(demo)

```
#!/bin/bash
super-analyzer ~/home/test.apk --open
```

3.3. 报告格式

默认生成HTML格式，
也可加参数 --json 生成json形式的报告，生成的报告在当前路径下的results文件夹中

4. SuperAndroid漏洞检测一览表（以QQ音乐APK为例）

级别	漏洞分类	漏洞描述
严重	接受所有SSL证书漏洞	可检测不安全的应用程序SSL实现。此应用程序接受所有证书，包括默认情况下自签名。这是一个严重问题，可能会受到中间人攻击。
	SQL注入漏洞	可检测应用程序是否受SQL注入攻击。若存在SQL注入攻击，存储在数据库中的任何数据都可以暴露，因为任何攻击者都能够检索，修改和删除存储的信息。
	WebView XSS	可检测Webview不安全的实现。此问题可能允许远程攻击者在WebView中执行代码并执行跨站点脚本攻击。
	WebView忽略SSL错误	可检测WebView是否存在忽略SSL的错误，如若存在可导致它接受任何SSL证书，此应用程序可能受到中间人攻击。
高危	执行系统命令	应用程序可以直接执行系统命令。
	弱算法漏洞	可检测到是否执行了弱算法。使用弱算法易使攻击者破坏加密通信，从而获得对纯文本内容的访问。
	可读权限	可检测到权限是否未加限制被赋予了所有人。
	Rooted 设备检测	可检测到此应用程序正在对root设备执行检查。如果应用程序已经控制设备，则可以使用它来执行特定代码。
	外部存储中的写入读取	应用程序可以在外部存储器中读/写。任何应用都可以读取外部存储中写入的数据。
低危	抛出的异常	方法抛出的异常应该是特定类型的。通用异常类型不安全并可能导致错误。
	未检查的日志输出	敏感信息不应该在Log中输出，会导致信息泄漏。
	未知权限漏洞	应用程序可以自己创建权限，可能导致开发人员之间的误解。
	sleep方法参数漏洞	Sleep方法使用变量作参数。如果这些变量被修改，它可能会迫使应用程序无限期地停止。
警告	访问粗略位置权限	允许该应用获取您的大致位置。此位置由位置服务使用网络位置源（如手机信号塔和Wi-Fi）获得。必须打开这些位置服务并将其提供 给您的设备，以便应用使用它们。应用可以使用它来确定您的位置。检查是否确实需要权限。
	访问精确位置权限	允许该应用使用全球定位系统（GPS）或网络位置信息源（如手机信号塔和Wi-Fi）获取您的精确位置。必须打开这些位置服务并将其 提供给您的设备，以便应用使用它们。应用程序可能会使用它来确定您的位置，并可能消耗额外的电池电量。检查是否确实需要权限。

级别	漏洞分类	漏洞描述
	编码安全漏洞	此应用程序是否使用Base64编码。这不是一种安全的数据编码方法。
	拨打电话许可	该应用在没有您干预的情况下拨打电话号码。这可能会导致意外收费或通话，恶意应用可能会在未经您确认的情况下拨打电话而花费您的 费用，检查是否确实需要权限。
	证书或密钥库泄漏	源代码的反编译可能导致硬编码证书或密钥库的泄露。
	Exported activity	Exported activity可以被其他程序使用。
	Exported receiver	Exported receiver可以被其他应用使用。
	Exported service	Exported service可以被其他程序使用。
	获取账户列表权限	获取手机已知的帐户列表。这可能包括您安装的应用程序创建的任何帐户。检查是否确实需要权限。
	IP信息泄漏	源代码的反编译可能导致私有IP的泄露。
	联网权限	允许该应用创建网络套接字并使用自定义网络协议。浏览器和其他应用程序提供了将数据发送到互联网的方法，因此不需要此权限即可 将数据发送到互联网。检查是否确实需要此权限。
	拨出电话权限	应用程序拨出电话并更改要拨打的号码。此权限允许应用程序监控，重定向或阻止拨出电话。检查是否确实需要权限。
	读取手机状态权限	该应用拥有访问设备功能。此权限允许应用程序确定电话号码和设备ID，并呼叫远程号码。检查是否确实需要权限。
	录制音频权限	此权限允许应用程序随时录制音频而无需您的确认。检查是否确实需要权限。
	发送短信权限	这可能会导致意外收费。恶意应用可能会在未经您确认的情况下发送消息而花费您的金钱 检查是否确实需要权限。
	URL泄漏	源代码的编译可能导致私有URL的泄漏。
	写入外部权限	允许该应用写入SD卡。检查是否确实需要权限。

5. SuperAndroid验证测试

5.1. 定位权限测试：

源代码：

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
```

工具检测结果：
标签：访问精确位置权限
说明：允许该应用使用全球定位系统（GPS）或网络位置信息源（如手机信号塔和Wi-Fi）获取您的精确位置。必须打

开这些位置服务并将其提供给您的设 备，以便应用使用它们。应用程序可能会使用它来确定您的位置，并可能消耗额外的电池电量。检查是否确实需要权限。
文件： AndroidManifest.xml
行： 23

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

5.2. 相机权限测试：

源代码：

```
<uses-permission android:name="android.permission.CAMERA"/>
```

工具检测结果：
标签：相机权限
说明：允许该应用使用相机拍摄照片和视频。此权限允许应用程序在未经您确认的情况下随时使用相机。检查是否确实需要权限。
文件：AndroidManifest.xml
行：22

```
<uses-permission android:name="android.permission.CAMERA" />
```

5.3. Exported activity测试

源代码：

```
<activity
          android:name=".MainActivity"
          android:exported="true">
</activity>
```

工具检测结果：
标签：activity设置为导出
说明：已找到设置为可导出的activity，可以被其他应用程序使用。
文件：AndroidManifest.xml
行：32

```
<activity android:name="com.example.wangjian.sqlinjection.MainActivity" android:exported
="true" />
```

5.4. 互联网权限测试：

源代码：

```
<uses-permission android:name="android.permission.INTERNET"/>
```

工具检测结果：
标签：互联网许可
说明：允许该应用创建网络套接字并使用自定义网络协议。浏览器和其他应用程序提供了将数据发送到互联网的方法，因此不需要此权限即可将数据发送到互联网。检查是否确实需要权限。
文件：AndroidManifest.xml
行：4

```
<uses-permission android:name="android.permission.INTERNET" />
```

5.5. 拨打电话权限测试：

源代码：

```
<uses-permission android:name="android.permission.PROCESS_OUTGOING_CALLS"/>
```

工具检测结果：
标签：处理拨出电话权限
说明：允许该应用处理拨出电话并更改要拨打的号码。此权限允许应用程序监控，重定向或阻止拨出电话。检查是否确实需要权限。
文件：AndroidManifest.xml
行：17

```
<uses-permission android:name="android.permission.PROCESS_OUTGOING_CALLS"/>
```

5.6. IP信息泄漏漏洞

源码：

```
intent.setData(Uri.parse("http://www.baidu.com"));
```

工具检测结果：
标签：URL Disclosure
说明：源代码的反编译可能导致私有URL的泄露。
文件： classes / com / example / wangjian / sqlinjection / Main2Activity $ 2.java
行：16

```
paramView = new Intent("android.intent.action.VIEW");
    paramView.setData(Uri.parse("http://www.baidu.com"));
    this$0.startActivity(paramView);
```

5.7. 阅读手机状态权限

源码：

```
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
```

工具检测结果：
标签：阅读手机状态权限
说明：允许该应用访问设备的手机功能。此权限允许应用程序确定电话号码和设备ID，呼叫是否处于活动状态以及通过呼叫连接的远程号码。检查是否确实需要权限。
文件：AndroidManifest.xml
行：15

```
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
```

5.8. 录制音频权限测试

源码：

```
<uses-permission android:name="android.permission.RECORD_AUDIO"/>
```

工具检测结果：
标签：录制音频权限
说明：允许该应用使用麦克风录制音频。此权限允许应用程序随时录制音频而无需您的确认。检查是否确实需要权限。
文件：AndroidManifest.xml
行：12

```
<uses-permission android:name="android.permission.RECORD_AUDIO" />
```

5.9. 发送短信权限测试

源码：

```
<uses-permission android:name="android.permission.SEND_SMS"/>
```

工具检测结果：
标签：发送短信权限
说明：允许该应用发送短信。这可能会导致意外收费。恶意应用可能会在未经您确认的情况下发送消息而花费您的金钱 检查是否确实需要权限。
文件：AndroidManifest.xml
行：11

```
<uses-permission android:name="android.permission.SEND_SMS" />
```

5.10. 阅读联系人权限

源码：

```
<uses-permission android:name="android.permission.READ_CONTACTS" />
```

工具检测结果：
标签：阅读联系人权限
说明：允许该应用读取有关您手机上存储的联系人的数据，包括您以特定个人的其他方式致电，通过电子邮件发送或

与之通信的频率。此权限允许应用保存您的联系人数据，恶意应用可能会在您不知情的情况下共享联系人数据。检查是否确实需要权限。
文件：AndroidManifest.xml
行：16

```
<uses-permission android:name="android.permission.READ_CONTACTS" />
```

5.11. 写入外部存储权限

源码：

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
```

工具检测结果：
标签：写入外部存储权限
说明：允许该应用写入SD卡。检查是否确实需要权限。
文件：AndroidManifest.xml
行：6

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

5.12. 执行系统命令

源码：

```
<uses-permission android:name="com.hkmc.permission.WRITE_MODE" />
```

工具检测结果：
标签：执行系统命令
说明：应用程序可以执行系统命令。
文件：classes / com / example / wangjian / test / RunnableDemo.java
行：32

```
Runtime.getRuntime().exec("chmod 777 /data/anr");
```

5.13. 弱算法

源码：

```
<uses-permission android:name="android.permission.READ_PHONE_STATE"/>
```

工具检测结果：
标签：数学随机算法
说明：此方法不像随意使用的那样随机。它不应该用于生成OTP代码。
文件：classes / com / example / wangjian / test / RunnableDemo.java
行：43

```
int j = (int)(Math.random() * 10.0D * 10.0D + 1.0D);
```

5.14. 抛出的异常

源码：

```
catch (Exception e) {
     System.out.println("Thread " +  threadName + " interrupted.");
 }
```

工具检测结果：
标签：catch中的通用异常
说明：异常捕获应该是特定的。通用异常类型可能不安全并导致错误
文件： classes / com / example / wangjian / test / RunnableDemo.java
行：54

```
catch (Exception localException)
```

5.15. Rooted设备检测

源码：

```
if ((Build.TAGS == null) || (!Build.TAGS.contains("test-keys"))) {
        System.out.println("test");
```

工具检测结果：
标签：Rooted设备检测
说明：此应用程序正在对root设备执行检查。如果设备已植根以控制它，则可以使用它来执行特定代码。
文件： classes / com / example / wangjian / test / RunnableDemo.java
行：82

```
 t = new Thread(this, threadName);
t.start();
if ((Build.TAGS == null) || (!Build.TAGS.contains("test-keys"))) {
  System.out.println("test");
```

5.16. 未检查的日志输出

源码：

```
Log.d(TAG, "run: "+time);
```

工具检测结果：
标签：未检查的日志输出
说明：日志输出中不应包含敏感信息，因为他有可能导致信息泄漏
文件：classes / com / example / wangjian / test / RunnableDemo.java
行：48

```
Log.d("ContentValues", ((StringBuilder)localObject1).toString());
```

5.17. sleep方法漏洞

源码：

```
int time = (int)(1+Math.random()*(10-1+1)*10);
         Log.d(TAG, "run: "+time);
         Thread.sleep(time);
```

工具检测结果：
标签：sleep方法漏洞
说明：Sleep方法与vars一起用作参数。如果这些变量被修改，它可能会迫使应用程序无限期地停止。
文件：classes / com / example / wangjian / test / RunnableDemo.java
行：49

```
Thread.sleep(j);
```

5.18. 编码安全漏洞

源码：

```
localStringBuilder.append(Base64.encodeToString(o, 0));
```

工具检测结果：
标签：编码安全漏洞
说明：此应用程序使用Base64编码。这不是一种安全的数据编码格式
文件：classes / com / example / wangjian / test / LoginActivity $ UserLoginTask.java
行：22

```
   paramVarArgs = new StringBuilder();
paramVarArgs.append(Base64.encodeToString(LoginActivity.access$300(this$0), 0));
paramVarArgs.append("#");
```