Introduction to Repeat Claddroid's Experiment

Mingsong Zhou

May 28, 2019

1. Experiment Environment

1. PC: Ubuntu 18.04 LTS
2. IDE: Idea community 2019, Pycharm community 2019
3. Java 8, python 3.7

2. Run Tool

1. Run idea_ApkIntentAnalysis/configure.sh in idea_ApkIntentAnalysis/ dir with sudo(PC will reboot)
2. Open Android Studio in idea_ApkIntentAnalysis/android-studio/bin/studio.sh. Then set Android SDK with idea_ApkIntentAnalysis/android-sdk and create nexus 5x android 19 x86 emulator.
3. Run idea_ApkIntentAnalysis/run.sh in idea_ApkIntentAnalysis/ dir to open IDEA and then open idea_ApkIntentAnalysis project. Set project sdk with idea_ApkIntentAnalysis/jdk1.8.0_161.
4. Run idea_ApkIntentAnalysis/CladdroidGUI/src/sample/Main.java. View result button will display the result of analysis.
5. How to use exploits: Copy exploits.txt(in idea_ApkIntentAnalysis/logger_file/testLog dir) to dir of detected_app and replace the file("detected_app's name_intentInfoSE.txt"), then run idea_ApkIntentAnalysis/use_exploits/useExploits.sh in idea_ApkIntentAnalysis/use_exploits dir with one arg, the arg is detected_app's absolute path.

3. Project Introduction

The figure 1 is system overview of our work. Our tool consists of two projects. One is *idea_ApkIntentAnalysis* and another is *testApp*. *idea_ApkIntentAnalysis* is is a java project. And *testApp* is a python project.

3.0.1 idea_ApkIntentAnalysis

This project is static analysis part of our tool.

1. *Config.java: idea*$_A$*pkIntentAnalysis/AnalysisAPKIntent/src/com/popoaichuiniu/util/Config.java:* which contains envs. For example: androidJar position, process dir.
2. *GenerateUnitNeedToAnalysis.java:* This java file is used to generate *UnitsNeedAnalysis.txt* of each app. *UnitsNeedAnalysis.txt* include all units that may cause privilege leaks.

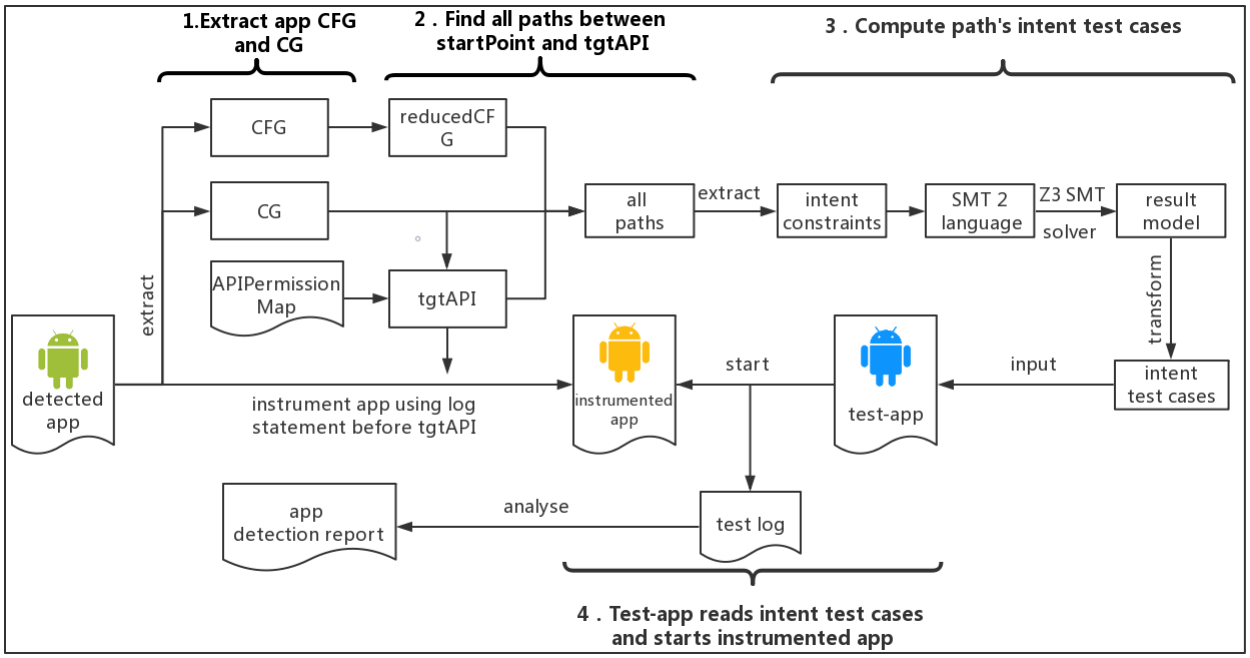

Figure 1: System overview

3. *IntentConditionTransformSymbolicExcutation.java:* At First, This java file find all paths between *startPoint* and *tgtAPI*. Then it generates path constraints into Z3 solver. At last, it gets result model from Z3 solver and change it into intent cases.

4. *InstrumentAPPBeforePermissionInvoke.java:* InstrumentAPPBeforePermissionInvoke.java insert log statements into each app.

5. *ApkSigner.java:* ApkSigner.java signs each app generated by InstrumentAPPBeforePermissionInvoke.java.

3.0.2 testApp

This project is dynamic analysis part of our tool.

1. *testApp.py:* At First, This python file installs *test-app* and *instrumented app* on emulator. Then this python file pushes intent test cases into emulator. At last, it lauches *test-app*. The test-app will reads these intent test cases and send intents to *instrumented app*.