Android reverse engineering: understanding third-party applications

Vicente Aguilera Díaz
OWASP Spain Chapter Leader
Co-founder of Internet Security Auditors
vicente.aguilera@owasp.org
Twitter: @vaguileradiaz
www.vicenteaguileradiaz.com

OWASP EU Tour 2013

June 5, 2013. Bucharest (Romania)

The OWASP Foundation
http://www.owasp.org

Who I am?

VICENTE

AGUILERA

DÍAZ

- Co-founder of Internet Security Auditors
- OWASP Spain Chapter Leader
- More info: www.vicenteaguileradiaz.com

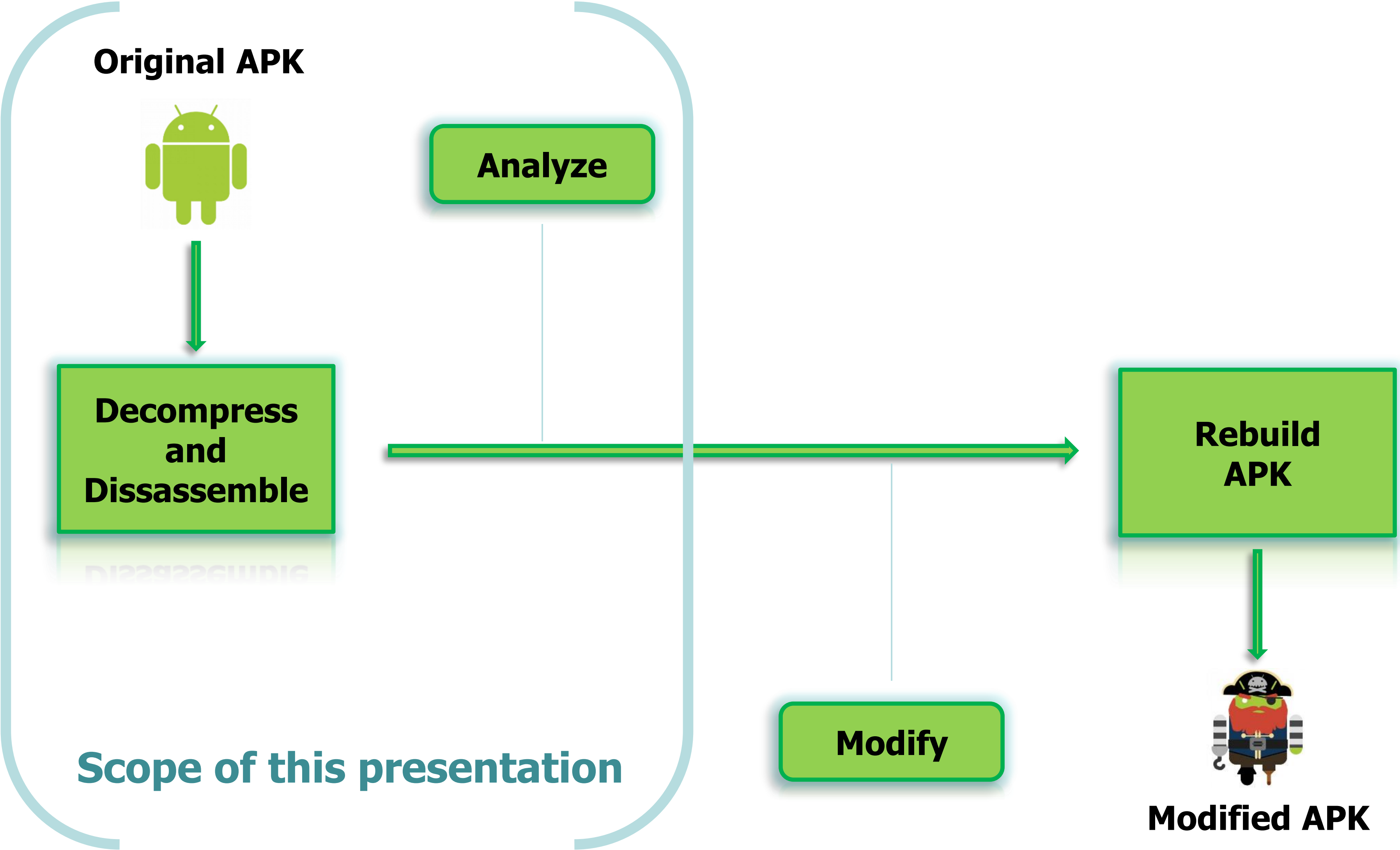
Agenda

- Reverse engineering: definition and objectives
- Application analysis workflow
- Malware identification in Android apps

Reverse engineering: definition and objectives

- Definition
  - Refers to the process of analyzing a system to identify its components and their interrelationships, and create representations of the system in another form or a higher level of abstraction. [1]
- Objetives
  - The purpose of reverse engineering is not to make changes or to replicate the system under analysis, but to **understand** how it was built.

Application analysis workflow



Original APK

Analyze

Decompress and Dissassemble

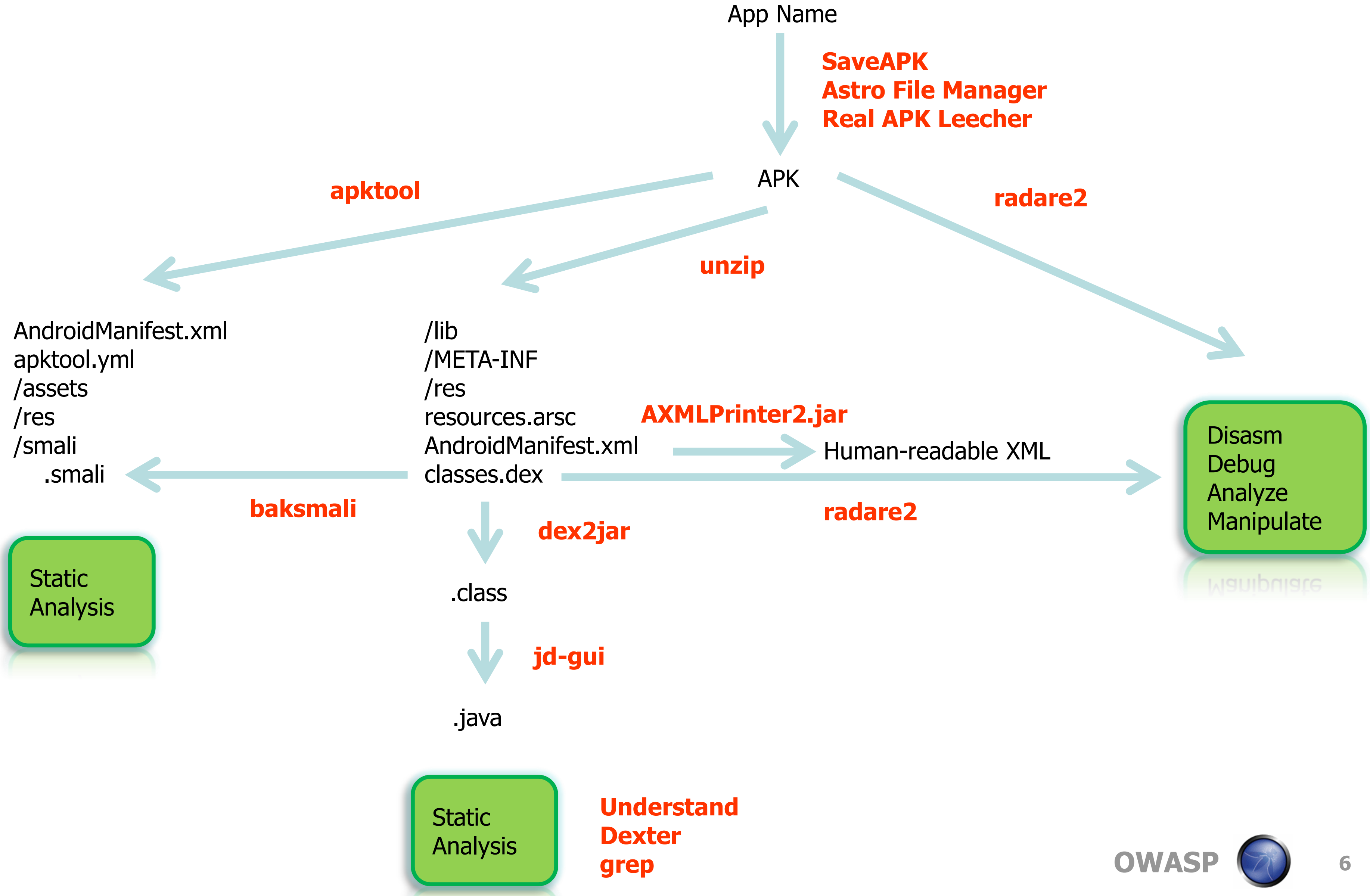
Rebuild APK

Modify

Modified APK

Scope of this presentation

Application analysis workflow



App Name

SaveAPK
Astro File Manager
Real APK Leecher

APK

apktool

unzip

radare2

AndroidManifest.xml
apktool.yml
/assets
/res
/smali
.smali

/lib
/META-INF
/res
resources.arsc
AndroidManifest.xml
classes.dex

AXMLPrinter2.jar

Human-readable XML

baksmali

radare2

Disasm
Debug
Analyze
Manipulate

Static
Analysis

dex2jar

.class

jd-gui

.java

Static
Analysis

Understand
Dexter
grep

Application analysis workflow

- Static Analysis Tools for Android Apps

TOOL	DESCRIPTION	URL
Dexter	Static android application analysis tool	https://dexter.bluebox.com/
Androguard	Analysis tool (.dex, .apk, .xml, .arsc)	https://code.google.com/p/androguard/
smali/baksmali	Assembler/disassembler (dex format)	https://code.google.com/p/smali/
apktool	Decode/rebuild resources	https://code.google.com/p/android-apktool/
JD-GUI	Java decompiler	http://java.decompiler.free.fr/?q=jdgui
Dedexer	Disassembler tool for DEX files	http://dedexer.sourceforge.net/
AXMLPrinter2.jar	Prints XML document from binary XML	http://code.google.com/p/android4me/
dex2jar	Analysis tool (.dex and .class files)	https://code.google.com/p/dex2jar/
apkinspector	Analysis functions	https://code.google.com/p/apkinspector/
Understand	Source code analysis and metrics	http://www.scitools.com/
Agnitio	Security code review	http://sourceforge.net/projects/agnitiotool/

Application analysis workflow

- Others (necessary) tools

TOOL	DESCRIPTION	URL
Android SDK	Tools to build, test, and debug apps	http://developer.android.com/sdk/index.html
|--- emulator	Virtual mobile device	developer.android.com/tools/help/emulator.html
|--- adb	Android debug bridge	developer.android.com/tools/help/adb.html
A.R.E.	Android Reverse Engineering VM	https://redmine.honeynet.org/projects/are/wiki

Malware identification in Android apps

- Malware definition
  - Malware is a piece of code which changes the behavior of either the operating system kernel or some security sensitive applications, without a user consent and in such a way that it is then impossible to detect those changes using a documented features of the operating system or the application.[2]
  - A malware is any **malicious code** or piece of software that is designed to perform functions without the consent of the user.

Malware identification in Android apps

- Techniques for introducing malware
  - Exploit any vulnerability in the web server hosting the official store
  - Use the official store to post apps containing malware
  - Install not malicious app that, at some point, install malicious code
  - Use alternatives[3] to official stores to post apps containing malware

Malware identification in Android apps

- A practical example
- Some considerations
  - The analyzed app are in the Play Store
  - The published application does not exploit (supposedly) any vulnerability, but can contains malicious code that exploits the user's trust[4]
  - We will only use static analysis
  - We will analyze Java source code
  - We will use the Android Emulator[5]

Malware identification in Android apps

- What do we need?

… and motivation!

Malware identification in Android apps

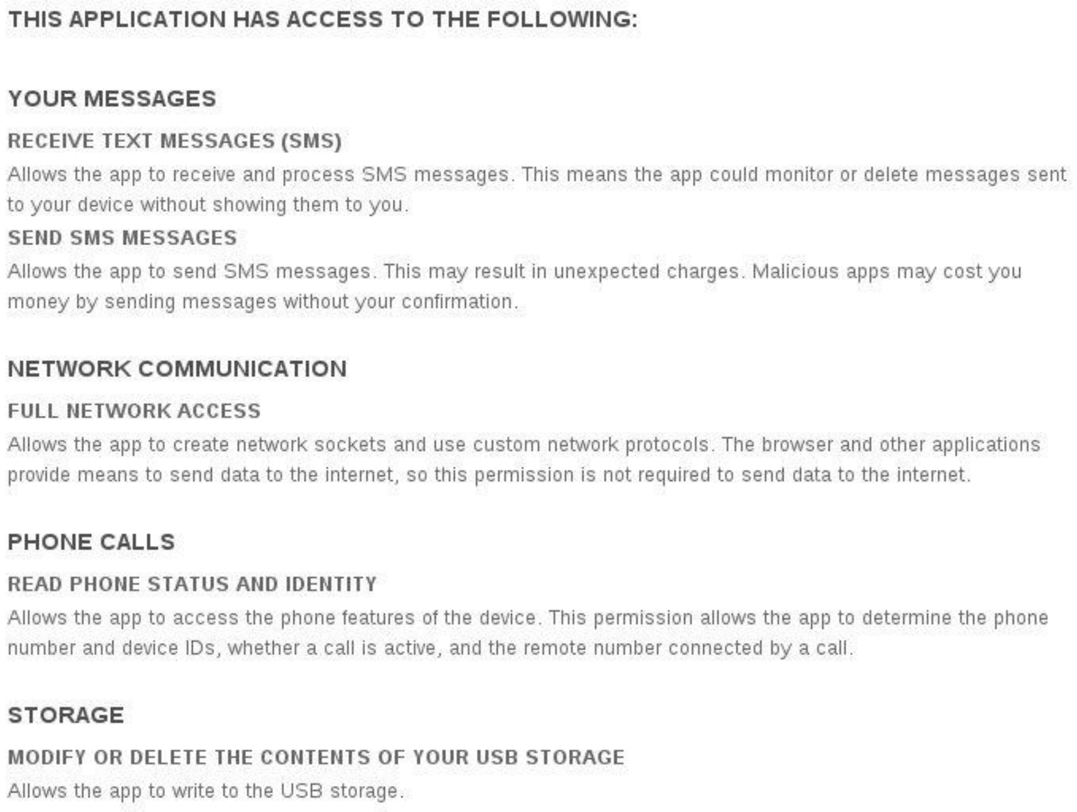
Let's see an example…

Malware identification in Android apps

- Identify a possible malicious application
  - App with unnecessary permissions
    - A wallpaper that requires “SEND SMS MESSAGES”
    - A calculator that requires “DIRECTLY CALL PHONE NUMBERS”
    - …
  - Google:
    - +"send sms messages" +”wallpaper” +site:”play.google.com”

Malware identification in Android apps

- Identify a possible malicious application
  - Example: “Pipe Mania Droid Lite”
    - https://play.google.com/store/apps/details?id=bridge.pipe.lite



THIS APPLICATION HAS ACCESS TO THE FOLLOWING:

YOUR MESSAGES

RECEIVE TEXT MESSAGES (SMS)

Allows the app to receive and process SMS messages. This means the app could monitor or delete messages sent to your device without showing them to you.

SEND SMS MESSAGES

Allows the app to send SMS messages. This may result in unexpected charges. Malicious apps may cost you money by sending messages without your confirmation.

NETWORK COMMUNICATION

FULL NETWORK ACCESS

Allows the app to create network sockets and use custom network protocols. The browser and other applications provide means to send data to the internet, so this permission is not required to send data to the internet.

PHONE CALLS

READ PHONE STATUS AND IDENTITY

Allows the app to access the phone features of the device. This permission allows the app to determine the phone number and device IDs, whether a call is active, and the remote number connected by a call.

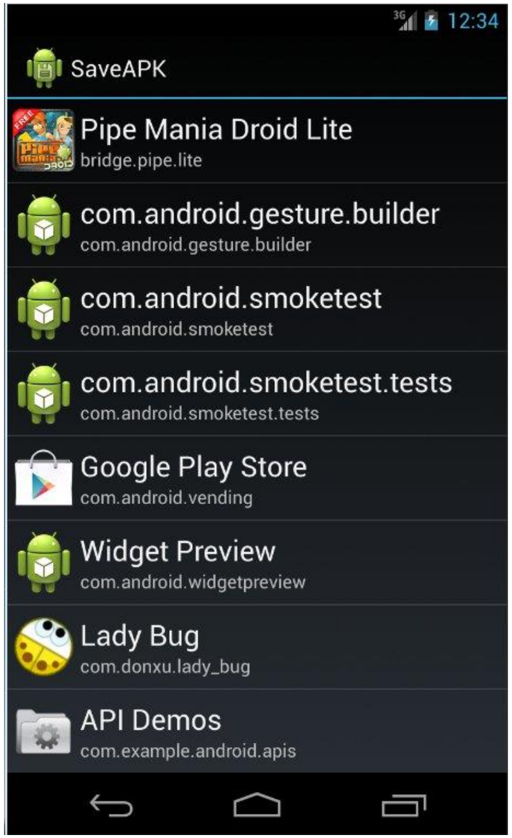
STORAGE

MODIFY OR DELETE THE CONTENTS OF YOUR USB STORAGE

Allows the app to write to the USB storage.

Malware identification in Android apps

- Obtaining the APK file
  - Using the SaveAPK tool (requires IO File Manager)



12:34

SaveAPK

Pipe Mania Droid Lite
bridge.pipe.lite

com.android.gesture.builder
com.android.gesture.builder

com.android.smoketest
com.android.smoketest

com.android.smoketest.tests
com.android.smoketest.tests

Google Play Store
com.android.vending

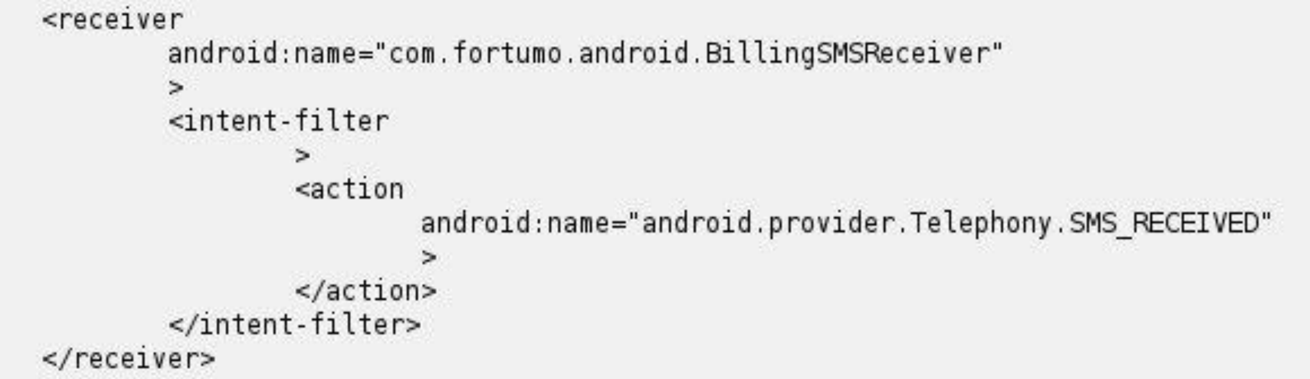
Widget Preview
com.android.widgetpreview

Lady Bug
com.donxu.lady_bug

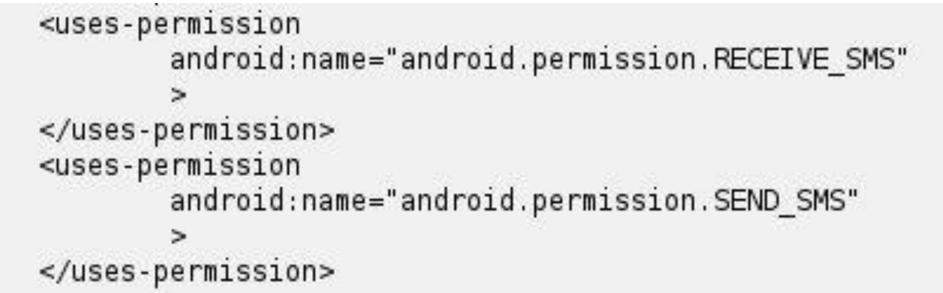
API Demos
com.example.android.apis

Malware identification in Android apps

- Decompress the APK file
  - unzip Pipe\ Mania\ Droid\ Lite.apk
- Verify the permissions and receivers
  - java –jar AXMLPrinter2.jar AndroidManifest.xml > out



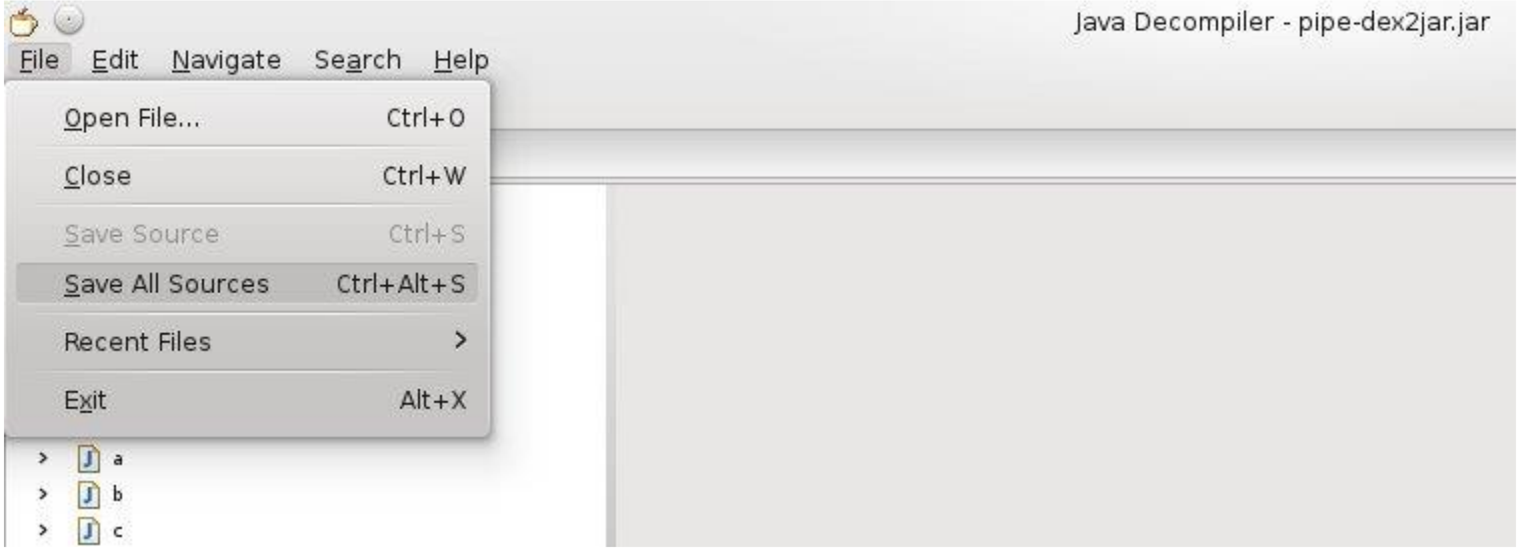
```
<receiver
        android:name="com.fortumo.android.BillingSMSReceiver"
        >
        <intent-filter
                >
                <action
                        android:name="android.provider.Telephony.SMS_RECEIVED"
                        >
                </action>
        </intent-filter>
</receiver>
```



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

Malware identification in Android apps

- Convert from Dalvik EXecutable to Java classes
  - d2j-dex2jar.sh pipe.apk
- Decompile Java classes and download source code
  - jd-gui pipe-dex2jar.jar

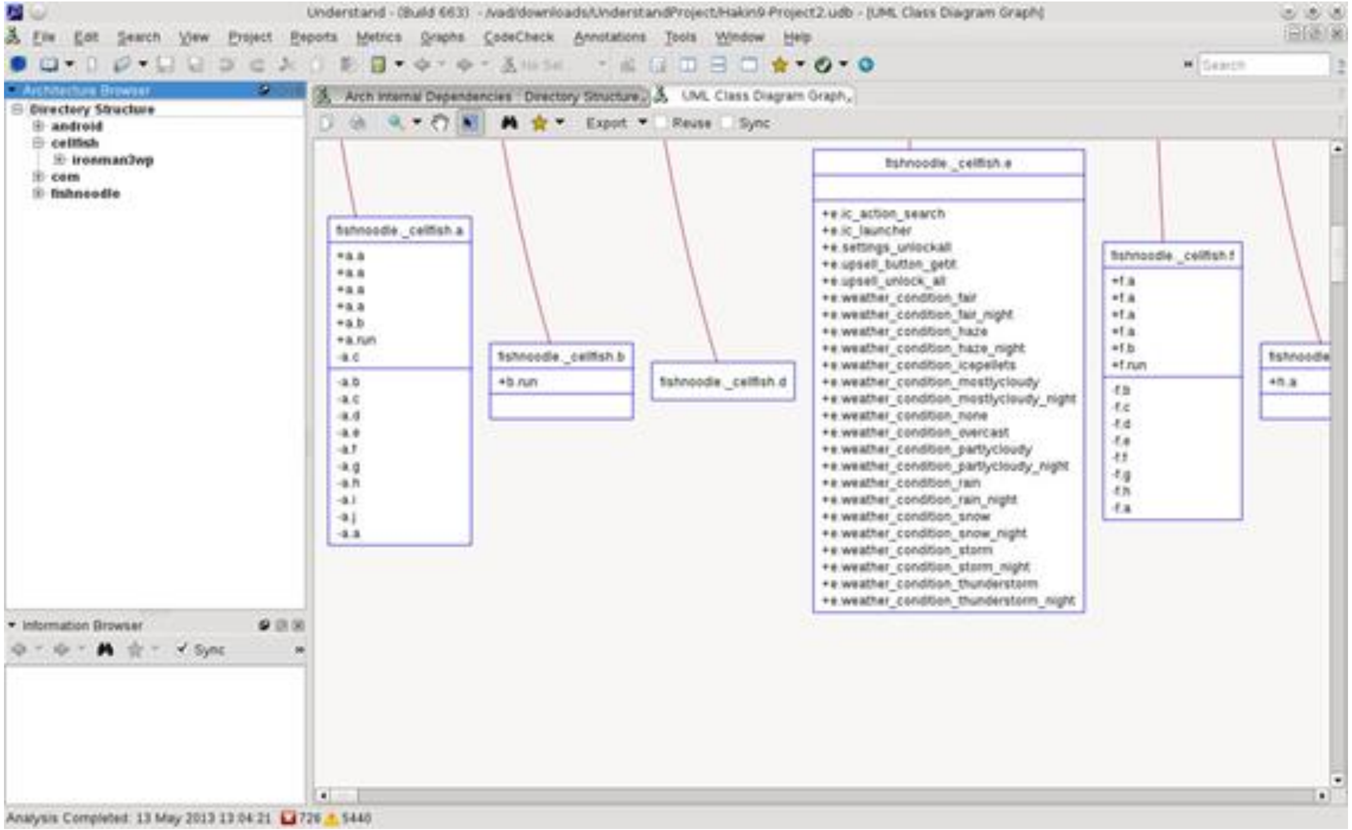


Malware identification in Android apps

- Decompress the source code
  - unzip pipe-dex2jar-src.zip
- Search sensitive strings
  - grep –i telephonymanager –r *
- Analyze the code
  - With tools
  - Manually
- Identifies malicious code

Malware identification in Android apps

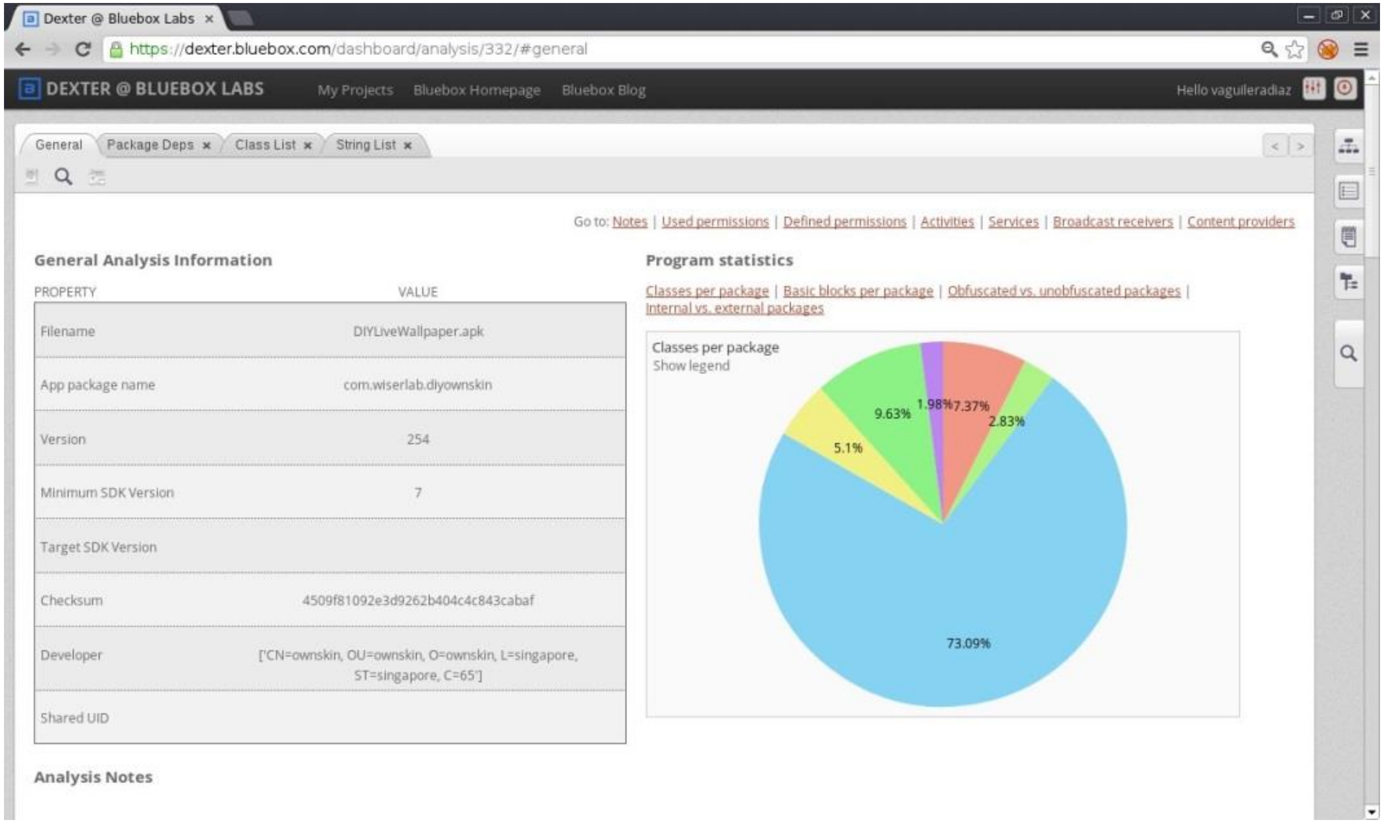
- "Understand" tool


Understand - (Build 663) - /vad/downloads/UnderstandProject/Hakin9-Project2.udb - [UML Class Diagram Graph]
File
Edit
Search
View
Project
Reports
Metrics
Graphs
CodeCheck
Annotations
Tools
Window
Help
No Sel
Search
Architecture Browser
Directory Structure
android
cellfish
ironman3wp
com
fishnoodle
Arch Internal Dependencies : Directory Structure
UML Class Diagram Graph
Export
Reuse
Sync
fishnoodle._cellfish.a
+a.a
+a.a
+a.a
+a.a
+a.b
+a.run
-a.c
-a.b
-a.c
-a.d
-a.e
-a.f
-a.g
-a.h
-a.i
-a.j
-a.a
fishnoodle._cellfish.b
+b.run
fishnoodle._cellfish.d
fishnoodle._cellfish.e
+e.ic_action_search
+e.ic_launcher
+e.settings_unlockall
+e.upsell_button_getit
+e.upsell_unlock_all
+e.weather_condition_fair
+e.weather_condition_fair_night
+e.weather_condition_haze
+e.weather_condition_haze_night
+e.weather_condition_icepellets
+e.weather_condition_mostlycloudy
+e.weather_condition_mostlycloudy_night
+e.weather_condition_none
+e.weather_condition_overcast
+e.weather_condition_partlycloudy
+e.weather_condition_partlycloudy_night
+e.weather_condition_rain
+e.weather_condition_rain_night
+e.weather_condition_snow
+e.weather_condition_snow_night
+e.weather_condition_storm
+e.weather_condition_storm_night
+e.weather_condition_thunderstorm
+e.weather_condition_thunderstorm_night
fishnoodle._cellfish.f
+f.a
+f.a
+f.a
+f.a
+f.b
+f.run
-f.b
-f.c
-f.d
-f.e
-f.f
-f.g
-f.h
-f.a
fishnoodle
+h.a
Information Browser
Sync

Analysis Completed: 13 May 2013 13:04:21 726 5440

Malware identification in Android apps

- "Dexter" online service



Go to: Notes | Used permissions | Defined permissions | Activities | Services | Broadcast receivers | Content providers

General Analysis Information

PROPERTY	VALUE
Filename	DIYLiveWallpaper.apk
App package name	com.wiserlab.diyownskin
Version	254
Minimum SDK Version	7
Target SDK Version	
Checksum	4509f81092e3d9262b404c4c843cabaf
Developer	['CN=ownskin, OU=ownskin, O=ownskin, L=singapore, ST=singapore, C=65']
Shared UID	

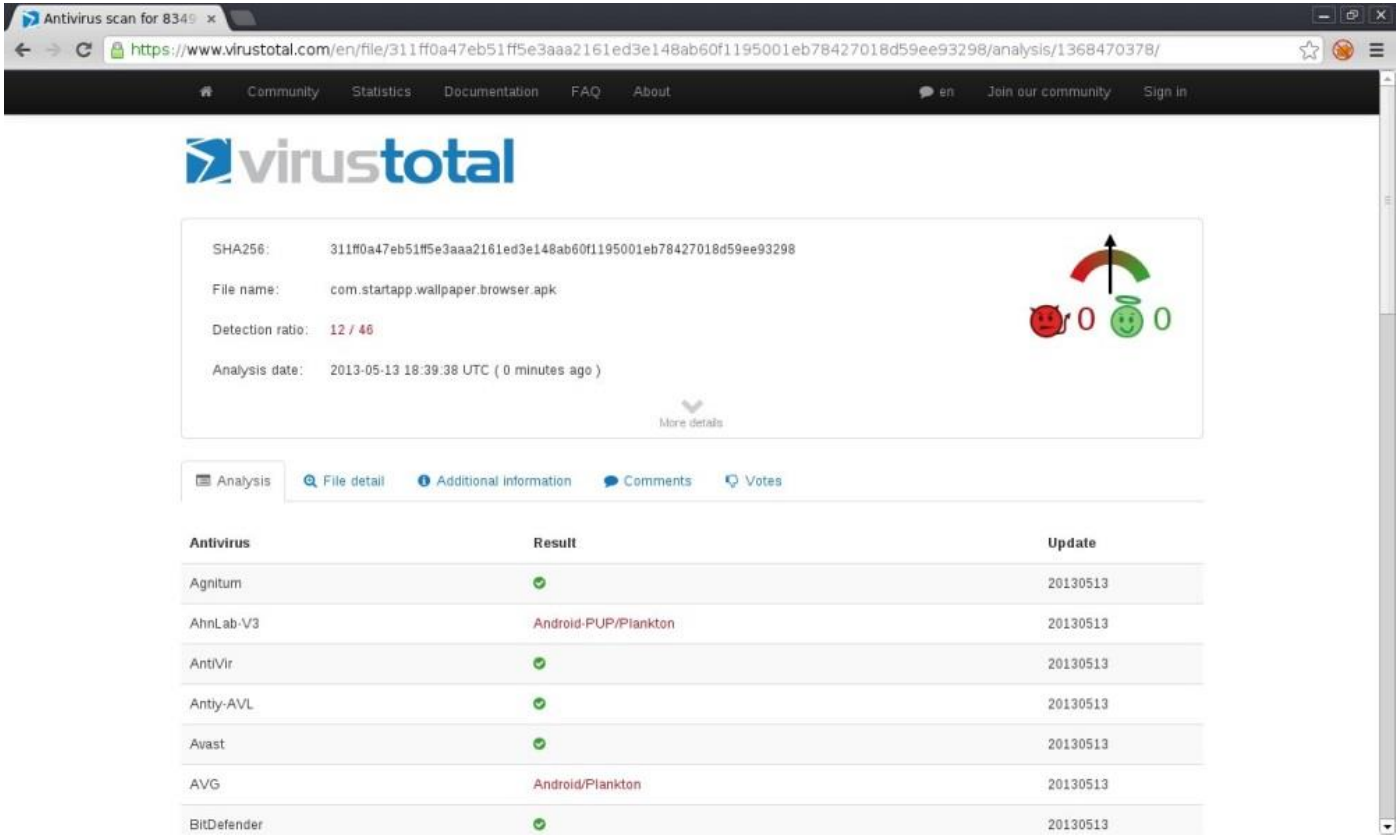
Program statistics

Classes per package | Basic blocks per package | Obfuscated vs. unobfuscated packages | Internal vs. external packages

Analysis Notes

Malware identification in Android apps

- “virustotal.com” online service



SHA256: 311ff0a47eb51ff5e3aaa2161ed3e148ab60f1195001eb78427018d59ee93298

File name: com.startapp.wallpaper.browser.apk

Detection ratio: 12 / 46

Analysis date: 2013-05-13 18:39:38 UTC (0 minutes ago)

Antivirus	Result	Update
Agnitum	✓	20130513
AhnLab-V3	Android-PUP/Plankton	20130513
AntiVir	✓	20130513
Antiy-AVL	✓	20130513
Avast	✓	20130513
AVG	Android/Plankton	20130513
BitDefender	✓	20130513

References

[1] “Reverse Engineering and Design Recovery: A Taxonomy”. Elliot J. Chikofsky, James H. Cross.

[2] “Introducing Stealth Malware Taxanomy”. J. Rutkowska.

[3] “Alternative markets to the Play Store”. http://alternativeto.net/software/android-market/

[4] “Security features provided by Android”. http://developer.android.com/guide/topics/security/permissions.html

[5] “Using the Android Emulator”. http://developer.android.com/tools/devices/emulator.html

References

[6] “Android malware database”

http://code.google.com/p/androguard/wiki/DatabaseAndroidMalwares

Thank’s!

Vicente Aguilera Díaz
@vaguileradiaz

www.vicenteaguileradiaz.com