Building a Test Environment for Android Anti-Malware Tests

Hendrik Pilz
Director Technical Lab / Mobile Security
hpilz@av-test.de
Agenda

• Android Malware Landscape
• Real Devices or Emulator?
• Preparation
• Test Scenarios
• Automation
• Problems
Android Malware Landscape

Android Malware Collection Growth

Total Number of Samples

January 2011 to August 2012
Android Malware Landscape

Malware Categories August 2012

- Trojan-SMS
- Trojan-Spy
- Backdoor
- Monitor
- Other
Android Malware Landscape

Malware Families August 2012

FakeInst
Opfake
GinMaster
FakeDoc
KungFu
BaseBrid
Kmin
Plandton
Other

Building a Test Environment for Android Anti-Malware Tests

www.av-test.org
Real Devices or Emulator

Device
• Real user experience
• App activation via SMS
• Real life environment

Emulator
• Cost efficient, scalable
• Root privileges
• Multiple API versions and hardware configurations
• Snapshots
Preparation

System Requirements:

• PC which is capable to run the Android SDK
• Android device, prepaid SIM
• USB cable
• WiFi-Internet for Android device
Preparation

PC with Android SDK -- WWW

USB

Android device
Preparation

• Install Android SDK from developer.android.com/sdk

• Choose Malware Samples according to AMTSO Guidelines

• Install Anti-Malware on test device, update signatures
Preparation

• Connect device to PC

• Create device backup
  $: adb backup -f <file> -apk -shared -all -system
  $: adb restore <file>

• Take Screenshots
  $: android-sdk/tools/ddms
Test Scenarios – On-Demand Scan

• Copy samples to device
  
  `$: adb push <source> /sdcard/samples`

• Perform on-demand scan, delete all malicious files

• Save remaining files
  
  `$: adb pull /sdcard/samples <dest>`

• Save scan reports, if possible
Test Scenarios – On-Demand Scan

Alternative to `adb push/pull`:
Copy files over WiFi from/to network share (e.g. with Astro File Manager)
Test Scenarios – On-Demand Scan

Some Anti-Malware apps scan installed apps only!
An On-Access Test is always required to determine accurate detection rates!
Test Scenarios - On-Access

- Install each sample one-by-one
  
  `$: adb install <apk-file>`

- Check warnings and messages of Mobile Security

- Remove or uninstall sample
  
  `$: adb uninstall <package-name>`
Test Scenarios - On-Access

#!/bin/bash

c=1
# traverse sample directory
for i in `ls $1`
do

    sample="${1}/${i}"
    echo "${c}: Installing ${i}"
    # get android package name
    package=`aapt dump badging ${sample} | \
    grep package: | sed "s/package: name='"" | \
    sed "s/"versionCode.*$/"`
    echo "Package: ${package}"
    # install, enter result, uninstall
    adb install ${sample}
    echo "Was the sample detected? (1 - yes / 0 - no)"
    read result
    echo -e "${sample}\t${result}" >> "on_access_report.txt"
    adb uninstall ${package}
    let c=c+1

done
Test Scenarios – On-Access

android@android-VirtualBox:~$ on-access.sh /media/android/
1: Installing 00a9677cd438dd8b3b3320ad45562d409b929e33587ed7211356d40477725dfc.apk
Package: com.keji.danti34
* daemon not running. starting it now on port 5037 *
* daemon started successfully *
415 KB/s (923684 bytes in 2.168s)
pkg: /data/local/tmp/00a9677cd438dd8b3b3320ad45562d409b929e33587ed7211356d40477725dfc.apk
Success
Was the sample detected? (1 - yes / 0 - no)
1
Success
2: Installing 0153e70bb3573e3fa701e307ae8a5b1b5d2ad72e2339b75acf5770cfda0c9a60.apk
Package: com.keji.Graphisa
422 KB/s (10997244 bytes in 25.399s)
pkg: /data/local/tmp/0153e70bb3573e3fa701e307ae8a5b1b5d2ad72e2339b75acf5770cfda0c9a60.apk
Success
Was the sample detected? (1 - yes / 0 - no)
1
Test Scenarios – On-Access

Building a Test Environment for Android Anti-Malware Tests

www.av-test.org
Test Scenarios – False Positives

- Combination of OA & OD
- Install clean apps via ADB
- Run an OD-scan afterwards
- Note all warnings and detections
Test Scenarios – False Positives

• Be aware of greyware:
  – Ad supported apps
  – Privacy risks
Test Scenarios – Performance

- Install clean apps from Google Play
  - We can’t use ADB here, because we can’t disable USB charging
- Monitor CPU-usage and battery discharge
- Repeat several times
Test Scenarios – Performance

CPU usage

- Product A
- Product B
- Product C
- Product D
- Product E
Test Scenarios – Performance

Discharge rate in % per minute

Estimated battery life in minutes

Product A Product B Product C Product D Product E

Product A Product B Product C Product D Product E
Test Scenarios – Performance

• Measure impact on real-world usage
  – Loading websites
  – Sending/receiving messages
  – Opening apps
  – Playing media files
  – ...

Building a Test Environment for Android Anti-Malware Tests
Test Scenarios – Others?

- Other functions are not common among all AV/mobile security products:
  - Anti-Theft
  - Backup, Encryption
  - Spam, Phishing
  - ...

Building a Test Environment for Android Anti-Malware Tests

www.av-test.org
Test Scenarios – Others?

![Bar Chart]

- Number of Products with this specific Feature

<table>
<thead>
<tr>
<th>Feature</th>
<th>Number of Products</th>
</tr>
</thead>
<tbody>
<tr>
<td>Anti-Malware</td>
<td>45</td>
</tr>
<tr>
<td>Anti-Theft</td>
<td>35</td>
</tr>
<tr>
<td>Message Filter</td>
<td>25</td>
</tr>
<tr>
<td>Backup</td>
<td>15</td>
</tr>
<tr>
<td>Safe Browsing</td>
<td>10</td>
</tr>
<tr>
<td>Parental Control</td>
<td>5</td>
</tr>
<tr>
<td>Data Encryption</td>
<td>1</td>
</tr>
</tbody>
</table>
Automation

• ADB-CLI

• ddmlib.jar (included in SDK)
  – High Level API to control ADB

• Robotium  <http://code.google.com/p/robotium/>
  – GUI automation of Android apps
Problems

• Not all apps support SD card scan
• No proper reporting
• No export of report files
Thank You!

Questions?