(In)Security of Mobile Banking
...and of Other Mobile Apps

Eric Filiol & Paul Irolla
ESIEA
Operational Cryptology and Virology Lab
I am so sorry not to be here with you but a family member of mine is extremely ill in hospital and his days are numbered. I had to be with my family in these circumstances.

Please feel to contact me for further technical details about this talk.
Agenda

• Introduction & Background: The OpenDAVFI Project
• Our Analysis Tools
• The Paradigm of Insecurity: the Facebook App
• Analysis of Banking Apps
  — The Protocol & a Few Statistics
  — Case Studies: JP Morgan Access, BNP Paribas, Sberbank, Bradesco, Bank of China...
  — Banking Apps Security Comparison: Western World versus Asian World
• Conclusion & Future Work
Introduction & Background

The Open DAVFI Project
Tools and Analysis
The OpenDAVFI Project

- Open and free fork of two-year project DAVFI to develop a sovereign and trusted AV (Android, Linux, Windows) by ESIEA/(C + V)° Lab
  — Release scheduled in 2015 (administrative stuff pending)
- Funded partly by the French Government (6 millions euros with 0.35 % of funding)
• One of the key features is that all apps available on a secure market is fully analyzed (static & dynamic analysis including a reversing step).

• Whenever safe AND compliant to our security policy (see further), the app is certified & signed before made available on the secure market.
Our Trust Policy

• Legit apps can be malevolent when it comes to targeted marketing and user tracking capabilities.
• A few apps contains severe vulnerabilities.
• The “malware” definition needs to be extended.
• An app is trustworthy according to our Trust Policy if and only if:
  — It does not contain hidden functionalities.
  — User information collection must be motivated by explicit functionalities.
  — Web communications involving personal user informations must be encrypted.
  — The app does not contain known vulnerabilities.
Why Bank Apps?

• Progressively, banks are forcing users to move towards mobile banking.
• Because our money is a serious business.
• Our privacy and data confidentiality is an even more critical issue!
• So, we expect them to be at the edge of security and confidentiality and to take care of our core interests.
• Most banks have been contacted to provide (for free) all technical details. Up to now, only a very few have answered.
• A few (BNP Paribas, CA) are currently correcting part of the problems reported.
Our Analysis Tools
The Approach

• Based on advanced and innovative data-mining techniques
• The tools we have developed:
  — Egide: advanced static analysis and malware detection tool
  — Panoptes: advanced dynamic analysis tool (network communications analysis at runtime)
  — Tarentula: web crawling tool to collect apps
• These tools are non-public at the present time
A program that reverses apps and generates a report which is a map and a guide in the source code

Tasks: reverses to smali/java, detects risky behaviors/methods/sources/sinks, computes the control flow graph through entry point methods, computes statistics on group of apps, computes similarities between an app and a group of apps

Generates a neural network and trains it on an app database, generates reports and graphs...

Demos and examples of reports
Dynamic Analysis - Panoptes

• Task: reveals communications between an app and the Internet.
• Opens a fake access point and listens to HTTP/HTTPS/POP/IMAP communications
• Generates a tree of communication information
• Required material list:
  — Wifi card with Master mode available
  — Ethernet connection available
  — Rooted Android phone
Dynamic Analysis - Panoptes
Bypassing SSL Encryption with Panoptes

- A fake Certification Authority is installed in the phone.
- SSL/TLS requests are intercepted, terminated and a new one is initiated to the original destination address.
- The server response is copied, embedded in a SSL layer and signed with our fake Certification Authority.
The Issue: Extending the APK DB

- Database of classified applications is the sinews of antiviral war
- A subject rarely explained or detailed in security papers
- A sophisticated data mining algorithm is useless with a poor database.
- So how to populate a database for training machine learning algorithms?
- How do the others make it?
The Issue: Extending the APK DB (2)

• Several universities gather malware and propose to share them
  — http://www.malgenomoproject.org/
  — http://user.informatik.uni-goettingen.de/~darp/drebin/

• A few websites share Android malware
  — http://virusshare.com/
  — http://contagiodump.blogspot.fr/

• It is a good starting point but not enough.
Appetizer - The Paradigm of Insecurity: the Facebook App
Facebook App

- Facebook collects information submitted by users (some sort of voluntary STASI)
- But what about information the app sends without the user knowledge?
- And what personal information are stored on the phone without users’ awareness?
- Reminder: you can access any data on the phone by physical access in less a minute (video)
Facebook Insecurity

- After a connection to the Facebook account and some basic navigation, get all data created locally:
  
  ```
  adb shell su -c 'cat /data/data/com.facebook.katana/**/*' > facebook-data.dump
  ```

- Time to parse this thing!

  ```
  /"displayName":"([\^\"]*?)"ng.*?"friendshipStatus":"([\^\"]*?)
  ".*?"contactType":"([\^\"]*?)".*?"cityName":"([\^\"]*)"/
  ```

```
[...]
Name : ########  ######
Status :  ARE FRIENDS
Type : USER
City : Nimes

Name :  Paul Irolla
Status :  CANNOT_REQUEST
Type : USER
City : Laval (Mayenne)

Name :  ########  ######
Status :  ARE FRIENDS
Type : USER
City : Paris
[...]
```
Facebook Insecurity (2)

Other data that are unconsciously collected and stored on your smartphone:

- Private messages
- Private photos
- Private wall content
- Many other private and non-private data...
Facebook: Network Analysis

• FB typical one-kilometer POST request (demo: Panoptes graph).

• Reverse procedure:
  — Unescape url codes recursively
  — Parse the output string as a JSON object
  — Until the data super-structure is entirely reversed
    • Try to parse each string in the JSON object as a JSON object
    • Try to decode each strings which seems to be in a base64 format, then
      — Try to unzip the result with gzip if the magic number is ’1F8B’, and finally
      — Read the result string with a WINDOWS minidump reader like WinDBG (no joke)
FB Needs to Know You Better!

- Bootloader used
- Device model/manufacturer/serial/hardware/ROM
- CPU model/architecture/version + Kernel version
- Screen settings
- List of system applications
- All environment variables
- Open file descriptors count
- Software and hardware file descriptors limit
- Locations settings + Developer settings + Lockpattern settings

```plaintext
LOCK PATTERN ENABLED=1
LOCK PATTERN SIZE=3
LOCK PATTERN VISIBLE=1
```
FB Needs to Know You Better! (2)

- Application settings
- **Security settings**
- Sound used for alarm alert
- Spell checker settings + Screensaver settings
- Notification settings - including used sound
- Battery settings - including current energy level
- Sounds/music settings + Camera settings + Wifi connection settings
- Sdcard and memory size/free space/used space
- ++ Usual user tracking info (timestamp for each user action)

```
connection = WIFI
connection class = POOR
network extra info = Panoptes-AP
```
The Analysis of Banking Apps
The Bank Apps Analyzed (up to now)

- BNP Paribas (France)
- LCL (France)
- Crédit Agricole (France)
- Sofinco (France)
- Société Générale (France)
- BforBank (France)
- Finaref (France)
- Bradesco (Brazil)
- BMCE (Morocco)
- Barclay (UK)
- UBS (Switzerland) - JP Morgan (USA)
- Wells Fargo (USA) - Bank of America (USA)
- Burke and Herbert (USA)
- PNC Financial Service (USA)
- Commerzbank (Germany)
- Deutsche Bank AG (Germany)
- HSBC (UK) - Santander Group (Spain)
- Sberbank (Russia) - Hapoalim Bank (Israel)
- Shahr Bank (Iran)
- VTB (Russia)
- LandKredit (Norway)
- Nordea Mobilbank (Norway)

- Oversea-Chinese Banking Corporation (Singapore)
- DBS Bank (Singapore)
- United Overseas Bank (Singapore)
- Bank of China (Hong Kong)
- Bank Negara (Indonesia)
- Commonwealth Bank of Australia
- National Australia Bank Limited
- Bank of Communications (China)
- Mitsubishi UFJ Financial Group (Japan)
- Advanced Bank Of Asia (Cambodia)
- Public Bank Berhad (Cambodia)
- Bangkok Bank (Thailand)
- State Bank of Mongolia
- HanaNBank (Korea)
- Agricultural Bank of China (China)
- Industrial Bank of Korea (Korea)
- Mizohobank (Japan)
- State Bank of India (India)
## A Few Statistics - Permissions

<table>
<thead>
<tr>
<th>PERMISSIONS</th>
<th>Western Banks</th>
<th>ASIAN BANKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTERNET</td>
<td>100%</td>
<td>93%</td>
</tr>
<tr>
<td>ACCESS_NETWORK_STATE</td>
<td>96%</td>
<td>87%</td>
</tr>
<tr>
<td>ACCESS_FINE_LOCATION</td>
<td>71%</td>
<td>60%</td>
</tr>
<tr>
<td>WRITE_EXTERNAL_STORAGE</td>
<td>68%</td>
<td>73%</td>
</tr>
<tr>
<td>READ_PHONE_STATE</td>
<td>61%</td>
<td>60%</td>
</tr>
<tr>
<td>CAMERA</td>
<td>54%</td>
<td>53%</td>
</tr>
<tr>
<td>ACCESS_COARSE_LOCATION</td>
<td>54%</td>
<td>73%</td>
</tr>
<tr>
<td>c2dm.permission.RECEIVE</td>
<td>46%</td>
<td>40%</td>
</tr>
<tr>
<td>CALL_PHONE</td>
<td>39%</td>
<td>47%</td>
</tr>
<tr>
<td>ACCESS_WIFI_STATE</td>
<td>39%</td>
<td>47%</td>
</tr>
<tr>
<td>READ_CONTACTS</td>
<td>32%</td>
<td>33%</td>
</tr>
<tr>
<td>gsf.permission.READ_GSERVICES</td>
<td>29%</td>
<td>33%</td>
</tr>
<tr>
<td>GET_ACCOUNTS</td>
<td>29%</td>
<td>27%</td>
</tr>
<tr>
<td>ACCESS_MOCK_LOCATION</td>
<td>14%</td>
<td>7%</td>
</tr>
<tr>
<td>READ_EXTERNAL_STORAGE</td>
<td>14%</td>
<td>13%</td>
</tr>
<tr>
<td>RECEIVE_BOOT_COMPLETED</td>
<td>14%</td>
<td>13%</td>
</tr>
<tr>
<td>WRITE_CONTACTS</td>
<td>11%</td>
<td>13%</td>
</tr>
<tr>
<td>NFC</td>
<td>11%</td>
<td>20%</td>
</tr>
<tr>
<td>RECEIVE_SMS</td>
<td>11%</td>
<td>20%</td>
</tr>
<tr>
<td>WRITE_SETTINGS</td>
<td>11%</td>
<td>7%</td>
</tr>
<tr>
<td>CHANGE_WIFI_STATE</td>
<td>11%</td>
<td>20%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>PERMISSIONS</th>
<th>Western Banks</th>
<th>ASIAN BANKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEND_SMS</td>
<td>7%</td>
<td>7%</td>
</tr>
<tr>
<td>RESTART_PACKAGES</td>
<td>7%</td>
<td>12%</td>
</tr>
<tr>
<td>CHANGE_NETWORK_STATE</td>
<td>7%</td>
<td>7%</td>
</tr>
<tr>
<td>READ_SMS</td>
<td>7%</td>
<td>7%</td>
</tr>
<tr>
<td>RECORD_AUDIO</td>
<td>7%</td>
<td>20%</td>
</tr>
<tr>
<td>READ_LOGS</td>
<td>7%</td>
<td>13%</td>
</tr>
<tr>
<td>ACCESS_LOCATION_EXTRA_COMMANDS</td>
<td>7%</td>
<td>13%</td>
</tr>
<tr>
<td>KILL_BACKGROUND_PROCESSES</td>
<td>7%</td>
<td>0%</td>
</tr>
<tr>
<td>ACCESS_NETWORK</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>GET_TASKS</td>
<td>4%</td>
<td>47%</td>
</tr>
<tr>
<td>RECEIVE_MMS</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>MOUNT_UNMOUNT_FILESYSTEMS</td>
<td>4%</td>
<td>13%</td>
</tr>
<tr>
<td>DISABLE_KEYGUARD</td>
<td>4%</td>
<td>7%</td>
</tr>
<tr>
<td>READ_OWNER_DATA</td>
<td>4%</td>
<td>13%</td>
</tr>
<tr>
<td>READCALENDAR</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>WRITECALENDAR</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>BROADCAST_STICKY</td>
<td>4%</td>
<td>7%</td>
</tr>
<tr>
<td>SMARTCARD</td>
<td>4%</td>
<td>7%</td>
</tr>
<tr>
<td>NFC_TRANSACTION</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>ACCESS_DOWNLOAD_MANAGER</td>
<td>4%</td>
<td>0%</td>
</tr>
<tr>
<td>READ_CALL_LOG</td>
<td>4%</td>
<td>0%</td>
</tr>
</tbody>
</table>
### A Few Statistics - Behaviors

<table>
<thead>
<tr>
<th>BEHAVIORS</th>
<th>Western Banks</th>
<th>ASIAN BANKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Load app content from web</td>
<td>96%</td>
<td>87%</td>
</tr>
<tr>
<td>Can use clear text communications</td>
<td>89%</td>
<td>87%</td>
</tr>
<tr>
<td>Get OS name</td>
<td>75%</td>
<td>73%</td>
</tr>
<tr>
<td>Get android unique id</td>
<td>71%</td>
<td>20%</td>
</tr>
<tr>
<td>Get IMEI</td>
<td>61%</td>
<td>73%</td>
</tr>
<tr>
<td>Use <code>addJavascriptInterface</code></td>
<td>54%</td>
<td>73%</td>
</tr>
<tr>
<td>Get OS version</td>
<td>50%</td>
<td>100%</td>
</tr>
<tr>
<td>User tracking capabilities</td>
<td>25%</td>
<td>47%</td>
</tr>
<tr>
<td>Get MAC address</td>
<td>13%</td>
<td>40%</td>
</tr>
<tr>
<td>Get MSISDN (Phone number)</td>
<td>11%</td>
<td>27%</td>
</tr>
<tr>
<td>Get IMSI</td>
<td>7%</td>
<td>20%</td>
</tr>
<tr>
<td>Get CID</td>
<td>4%</td>
<td>7%</td>
</tr>
<tr>
<td>Get LAC</td>
<td>4%</td>
<td>7%</td>
</tr>
<tr>
<td>Get SIM serial number</td>
<td>4%</td>
<td>20%</td>
</tr>
<tr>
<td>Get access point MAC address</td>
<td>4%</td>
<td>20%</td>
</tr>
</tbody>
</table>
Appraising of Asian Mobile Banking Security Assessment

- Overall security awareness of Asian banks seems superior to what we have observed for European/American continents
- In particular, the use of custom obfuscation, security routines on the native layer (c libs.), custom trusted SSL root CA... is prevalent and shows a significative care for security
- Therefore the analysis was much harder than what we have performed for Western Banks apps
- But there is always some black sheeps in the flock...
## Technical Summary Asian Bank Apps

<table>
<thead>
<tr>
<th>Banking application</th>
<th>Vulnerability found</th>
<th>Plaintext communications during runtime</th>
<th>Fake root CA countermeasure</th>
<th>User tracking capabilities</th>
<th>Strong use of crypto/obfuscation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oversea-Chinese Banking Corporation</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>DBS Bank</td>
<td>Potential</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>United Overseas Bank</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Bank of China</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Bank Negara Indonesia</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Commonwealth Bank of Australia</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>National Australia Bank Limited</td>
<td>Yes</td>
<td>Yes (But harmless)</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Bank of Communications</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Mitsubishi UFJ Financial Group</td>
<td>No</td>
<td>Yes</td>
<td>Partially</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Advanced Bank Of Asia</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Public Bank Berhad</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>No</td>
<td>N/A</td>
</tr>
<tr>
<td>Bangkok Bank</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>State Bank of Mongolia</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Vnechtorgbank</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Industrial Bank of Korea</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>No</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Case Study – JP Morgan Access

Demos
Case Study – JP Morgan Access (2)

“oxrohccRtI/m1w9NC/7nqwANljaa8fORRXcJ2S1EiThNdeuW6GErL7NQogAnOFtPdYlwP1Gh2+0aNqsnrKeGbw==

MFwwDQYJKoZIhvcNAQEBBQADSwAwSAJBAMx6N9b4yalFC60of8YWU1e08sh4KRoldfJKRmtVazOKg2p3UUwMT5oUwBYYEhWsSl+bTD6DMCIQrwr2iSW09DkCAwEAAQ==

Root Call Blocker, LBE Privacy Guard, Dual Mount SDWidget, Hexamob Recovery Pro, Total Commander, Boot Man”
Case Study – JP Morgan Access (3)

```java
String[] arrayOfString = str2.split("######");
...
RootTools.log("Executing sh Commands : " + arrayOfString5[0] + arrayOfString5[1]);
...
List localList = runcmd(arrayOfString6);
```
Case Study – JP Morgan Access (4)

- Dynamic Analysis:
  — An encrypted string is received
  — APK Instrumentation reveals it contains signatures and lists of strings.

- Static Analysis reveals some strings are sent directly into a shell.

- Well, that’s a remote shell, isn’t it?

- Technical details sent to the bank in December 2014. No News since. The vulnerability is still active. Nothing has been corrected yet 😞غضب
Dynamic analysis failed!
Static analysis reveals that `addJavascriptInterface` loads plaintext javascript.
That means MITM attackers can gain a reverse shell on vulnerable phones (nearly 75 % of the present smartphones)
Demo
Three months later, the bank has corrected nothing. The vulnerability is still exploitable
001122334400

- 001122334400 is the MAC address of our wifi access point used for interception
- So the app sends MAC addresses and signal strength in plaintext to the surrounding wifi networks
Case Study – Sberbank (2)

- Dynamic analysis reveals that all surroundings wifi networks info are sent in plaintext to yandex servers.
- Static analysis reveals that it is used for fine indoor geolocation.
- In fact, Google maps services (installed on every Android phones) does it too.
- That is basically world wifi networks mapping.
- "Hello Google, someone stole my wifi router, can you send me its coordinates please?"
- Demos
Case Study - Bradesco

- Dynamic analysis reveals that a private key for accessing bank services is received in plaintext.
- The embedded *Jquery* JavaScript lib contains vulnerabilities
- Demo
The application can check for available updates
A link on the official market is sent whenever a new update is available
Then the app downloads and installs the file
Moreover other navigation links (loaded by the app) are received
Security issue: this process is done entirely in http
Demo with Panoptes graph
Potential risks with a MitM attack:
— Installation of an arbitrary app by social engineering
— Loading of arbitrary web pages
— Exploiting the confidence of using a bank app, social engineering could be devastating

Demos with Panoptes graph
One more for the Road

The answer to our contact attempt: a link
To the HSBC page below

HSBC Mobile Banking App
This HSBC Mobile Banking app lets you manage your HSBC accounts securely from your mobile device.

Presence of the addJavascriptInterface method in 14 classes. This method is vulnerable for old Android API, see CVE-2012-6836 / CVE-2013-4710 for more informations. If one of these WebView load a http url, a third party can get a remote shell on the phone. The dynamical analysis could not highlight loading of http url but not all functionalities of the application have been tested.

User tracking informations are send to dc.webtrends.com and www1.member-hsbc-group.com.
Conclusion & Future Works
Future Work

- We intend to cover all banking apps throughout the world.
- Other kind of apps will be analyzed (games, email clients, security tools...)
- Develop our tools further with advanced mathematics (Ph D started in January 2015)
- Publish the {Egide, Panoptes} reports once security issues will be corrected by banks
- Verification analysis will be performed to check whether the users’ privacy issues have been solved as well.
Conclusion

- Mobile (Banking) apps are far from being totally clean. Beyond a few cases of vulnerabilities, users’ privacy is not the priority of developpers or outsourcers (here banks)!
  - Difference of awareness and security vision however between Asia and Western world
- There is a strong need for pressure on app developpers to take care of users’ privacy.
- The bank apps market is not mature and has developped too quickly. Functionalities take precedence over security and users’ fundamental rights for privacy and data confidentiality.
- It is very difficult to identify a visible contact point to report security issues
Conclusion (2)

- All the tested apps are on the Google Play!
- This means that Google does not perform apps’ security analysis at all! It does not care about users’ privacy either (but we all already know that)
- Google has the power to force developers to do a better job
- Choose open source apps (when available, for banks, well it is pure Utopia)
- Prefer local/national banks instead of international banks
Many thanks for your attention,
Questions & Answers
Contact: {filiol, irolla}@esiea.fr