Android geolocation using GSM network

« Where was Waldroid? »

Renaud Lifchitz
renaud.lifchitz+27c3@gmail.com

#27c3
27-30 December 2010, Berlin
Speaker's bio

- French computer security engineer
- Main activities:
  - Penetration testing & security audits
  - Security trainings
  - Security research
- Main interests:
  - Security of protocols (authentication, cryptography, information leakage, zero-knowledge proofs...)
  - Number theory (integer factorization, primality tests, elliptic curves)
Why Android?
Why Android?

- Why not?
- In just 2 years, 300,000 Android phones activated each day (Andy Rubin, Google, 2010/12/09)
- Android sales overtake iPhone in the U.S. since summer
- Because hacking on Android is sooooo cool (Linux kernel 🙄)
Why Android?

Operating System Share: 6 Month Recent Acquirers

Smartphone Subscribers, National, US

Source: The Nielsen Company
Geolocation: different approaches
GPS

- **Pros:**
  - Very accurate

- **Cons:**
  - Phone needs a built-in GPS
  - User must switch it on
  - Doesn't work inside buildings nor underground
Wi-Fi

• Pros:
  – Works inside buildings

• Cons:
  – Phone needs built-in Wi-Fi
  – User must switch it on
  – Less accurate than GPS
  – Needs access points
GSM location

- **Pros:**
  - No need for built-in GPS or Wi-Fi
  - Can be done from the network side

- **Cons:**
  - Medium accuracy
  - Needs GSM coverage
Cell location resolution

Every GSM cell (BTS) is identified by 4 numbers:

- MCC: Mobile Country Code
- MNC: Mobile Network Code
- LAC: Location Area Code
- CID: Cell ID

(MCC: 262, MNC: 01) = T-Mobile® Deutschland
Cell location resolution

• There have been several attempts to build databases of GSM cells:

<table>
<thead>
<tr>
<th>Name</th>
<th>Cells</th>
<th>Countries (MCC)</th>
<th>Operators (MNC)</th>
<th>Measures</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.location-api.com/">http://www.location-api.com/</a></td>
<td>11 182 473</td>
<td>215</td>
<td>1050</td>
<td>424 000 000</td>
</tr>
<tr>
<td><a href="http://labs.ericsson.com/apis/mobile-location/">http://labs.ericsson.com/apis/mobile-location/</a></td>
<td>3 900 000</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><a href="http://opencellid.org">http://opencellid.org</a></td>
<td>610 168</td>
<td>168</td>
<td>208</td>
<td>49 101 675</td>
</tr>
<tr>
<td><a href="http://cellid.telin.nl">http://cellid.telin.nl</a></td>
<td>133 637</td>
<td>61</td>
<td>165</td>
<td>832 474</td>
</tr>
<tr>
<td><a href="http://cellspotting.com">http://cellspotting.com</a></td>
<td>111 287</td>
<td>591</td>
<td></td>
<td></td>
</tr>
<tr>
<td><a href="http://celldb.org">http://celldb.org</a></td>
<td>138 582</td>
<td>221</td>
<td>640</td>
<td>2 649 453</td>
</tr>
<tr>
<td><a href="http://developer.yahoo.com/yrb/zonetag/">http://developer.yahoo.com/yrb/zonetag/</a></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><a href="http://www.cellumap.com">http://www.cellumap.com</a></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><a href="http://openbmap.org">http://openbmap.org</a></td>
<td>204 226  (582 964)</td>
<td>169</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Cell location resolution

- Why not use Google fantastic indexing power?
- Huge and continuously updated database thanks to:
  - Google cars
  - Android phones
Cell location resolution

- Google API? Quite confidential...

- Reverse-engineer:
  - What is used when you run Android Google Maps without GPS nor Wi-Fi
  - What is used by Google Gears plugin when you do a Google local search in your browser
Cell location resolution

- Android Google Maps internals:
  - tcpdump ARM compilation
  - Proprietary binary protocol
  - HTTP POSTed to http://www.google.com/glm/mmap
  - See “Poor Man's GPS” by Dhaval Motghare for reference: http://www.orangeapple.org/?p=82
  - Buggy...
Cell location resolution

- Google Gears internals:
  - Sniff Firefox plugin network traffic
  - See it's simple JSON!
  - Some (confidential!) reference here: http://code.google.com/p/gears/wiki/GeolocationAPI
  - “Officially deprecated” but updated and works a lot better than previous binary protocol
Cell location resolution

POST /loc/json HTTP/1.1
Accept-Charset: utf-8
Accept-Encoding: plain
Cache-Control: no-cache
Connection: close
Content-Length: 242
Content-Type: application/json
Host: www.google.com

{"radio_type": "gsm", "address_language": "fr_FR", "host": "maps.google.com", "version": "1.1.0", "cell_towers": ["mobile_network_code": 1, "cell_id": 32755, "mobile_country_code": 208, "location_area_code": 24832]}, "request_address": true}

Google Gears GSM Geolocation API full query
Cell location resolution

Google Gears GSM Geolocation API response body

- Interesting details:
  - Latitude&longitude
  - Full human-readable address (including street number, street name, zip code, city, region and country!)
  - Accuracy (in meters) → cell coverage?
Cell location resolution

- Going further: mapping the GSM network using sniffing with a SDR (Software Defined Radio) or an old phone (Nokia 3310)
- USRP 1 from Ettus Research LLC:

![Image of USRP 1 and Nokia 3310 phone]
Cell location resolution

- Use excellent AirProbe project:
  https://svn.berlin.ccc.de/projects/airprobe/

1. Scan with GnuRadio
2. Demodulate with AirProbe
3. Decode with Wireshark
Cell location resolution

$ tshark -V gsm_a.cell_ci -r out1.xml | grep -A2 'Cell CI'

Cell CI: 0x3198 (12696)
Location Area Identification - LAC (0x1005)

Cell CI: 0x31fe (12798)
Location Area Identification - LAC (0x1005)

Cell CI: 0x3806 (14342)
Location Area Identification - LAC (0x044c)

Cell CI: 0xe0ba (57530)
Location Area Identification - LAC (0x044c)
Cell location resolution

• Result!

GSM mapping 1 square kilometre of Paris from my bed 😊

#27c3 – 27-30 December 2010 – Berlin

“Android geolocation using GSM network”

Renaud Lifchitz
Attack vectors
Attack basics

- Android uses a specific logging facility
- Enabled by default
- 3 or 4 different logs
- Circular memory buffers
- Handled by character device files
- Built-in `logcat` tool to manipulate the logs
# Attack basics

```bash
# ls -l /dev/log

crw-rw--w-  1 root  log  10, 36 Dec 25 15:15 system

crw-rw--w-  1 root  log  10, 37 Dec 25 15:15 radio

crw-rw--w-  1 root  log  10, 39 Dec 25 15:15 main

crw-rw--w-  1 root  log  10, 38 Dec 25 15:15 events

cd /dev/log ; for f in *; do logcat -b $f -g; done

/dev/log/events: ring buffer is 256Kb (255Kb consumed), max entry is 4096b, max payload is 4076b

/dev/log/main: ring buffer is 64Kb (63Kb consumed), max entry is 4096b, max payload is 4076b

/dev/log/radio: ring buffer is 64Kb (14Kb consumed), max entry is 4096b, max payload is 4076b

/dev/log/system: ring buffer is 64Kb (6Kb consumed), max entry is 4096b, max payload is 4076b
```

Playing with logging facility
Attack basics

# hexdump -C radio | head

Playing with logging facility

#27c3 – 27-30 December 2010 – Berlin
“Android geolocation using GSM network”
Renaud Lifchitz
$ logcat -v time -b radio -d -s RILJ:D
12-26 14:53:25.147 D/RILJ ( 371): [3114]> QUERY_NETWORK_SELECTION_MODE
12-26 14:53:25.177 D/RILJ ( 371): [3112]< GPRS_REGISTRATION_STATE {1, null, null, 
9}
12-26 14:53:25.197 D/RILJ ( 371): [3113]< REGISTRATION_STATE {1, 0403, 00061E10,
9, null, null, null, null, null, null, null, null, null, null, null, null}
12-26 14:53:25.207 D/RILJ ( 371): [3114]< QUERY_NETWORK_SELECTION_MODE {0}
12-26 14:53:27.457 D/RILJ ( 371): [3117]< GPRS_REGISTRATION_STATE {1, null, null, 
9}
12-26 14:53:27.477 D/RILJ ( 371): [3118]< REGISTRATION_STATE {1, 0403, 00061E00,
9, null, null, null, null, null, null, null, null, null, null, null, null}

History of user's visited MCCs+MNCs, LACs, CIDs in radio logs

#27c3 – 27-30 December 2010 – Berlin
“Android geolocation using GSM network”
Renaud Lifchitz
Attack basics

• Attack scenario:
  - Collect history of visited GSM cells on the victim's side (no prior access needed)
  - Send them to the attacker
  - Resolve them into latitude&longitude

• Attack range:
  - Local (i.e. physical attack)
  - Remote (here remote means using a local vulnerability!)
Physical attack

- Connect the victim's phone to the attacker computer via USB
- Requires:
  - Physical access to the victim's phone for a few seconds
- Works even if the victim's phone is locked! (using USB debugging function)
Remote attack

- Remotely spy the victim
- Malware application who abuse either:
  - User trust
  - Android security model
- Requires:
  - A bit of social engineering (or not 😊)
Remote attack

- Android permissions model: Dalvik (java) sandbox
- Permissions: android.permission.*
- What can a user fear?
  - Dangerous combination of 2 permissions:
    - ACCESS_COARSE_LOCATION
    - or ACCESS_FINE_LOCATION
    - + INTERNET
Remote attack

- **1st attack - Use both permissions:**
  - Internet permission is needed for free ad-sponsored applications
  - Official geolocation permission is needed for location-aware applications

💡 most users won't care!
Remote attack

- **2nd attack** – Use the radio logs:
  - Instead of using Android geolocation API, read radio logs (READ_LOGS permission) to collect Cell IDs
  - Write results into the system log (no permission needed!)
  - Voluntarily crash the application when needed (no permission needed!)
  - If the user reports the crash, system log is sent to the developer using the integrated Google Feedback client 😊
Remote attack

Android geolocation using GSM network

Renaud Lifchitz
Remote attack

![Remote attack message]

Google Feedback client

The application Launcher (process com.android.launcher2) has stopped unexpectedly. Please try again.

Force close  Report
# Remote attack

## User reports

### Crash errors in `com.android.launcher2`

<table>
<thead>
<tr>
<th>New</th>
<th>Exception Description</th>
<th>Reports</th>
<th>Reports/Week</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td><code>OutOfMemoryError</code> in <code>Bitmap.nativeCreate()</code></td>
<td>232</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td><code>OutOfMemoryError</code> in <code>Bitmap.nativeCreateFromParcel()</code></td>
<td>19</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td><code>OutOfMemoryError</code> in <code>BitmapFactory.nativeDecodeAsset()</code></td>
<td>88</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td><code>NullPointerException</code> in <code>Bitmap.createBitmap()</code></td>
<td>21</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td><code>ArrayIndexOutOfBoundsException</code> in <code>ArrayList.get()</code></td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td><code>NullPointerException</code> in <code>Parcel.readException()</code></td>
<td>23</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td><code>NullPointerException</code> in <code>LauncherModel$Loader.startLoader()</code></td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td><code>NullPointerException</code> in <code>AllAppsView$RollOver.saveAppsList()</code></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td><code>ArrayIndexOutOfBoundsException</code> in <code>System.arraycopy()</code></td>
<td>34</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td><code>SecurityException</code> in <code>Parcel.readException()</code></td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Remote attack

- 3rd attack - Use Android NDK to completely bypass permissions model:
  - Native Development Kit allows developer to call native functions (C/C++ code) from their applications (similar to JNI)
  - Works outside the Dalvik sandbox...
- Arbitrary file access, code execution, network access... 😊
Remote attack

- 4th attack – Man-in-The-Middle attack during application download over Wi-Fi:
  - The new Android Market&Android Download Manager send application name, description, permissions then content in plaintext HTTP
  - It should be possible to change application description, permissions and/or content using active MiTM and install any malware application!
Remote attack

An Android market download

GET /market/download/Download?
assetId=9177147809749553200&userId=XXXXXXXXXXXXXXXXX&deviceId=YYYYYYYYYYYYYYYYYYYY

HTTP/1.1
Cookie: MarketDA=ZZZZZZZZZZZZZZZZZZZZ
Host: android.clients.google.com
Connection: Keep-Alive
User-Agent: AndroidDownloadManager

HTTP/1.0 200 OK
ETag: -1625044586
Content-Type: application/vnd.android.package-archive
Content-Length: 498162
Content-Disposition: inline
Date: Sun, 28 Dec 2010 17:50:13 GMT
Expires: Sun, 28 Dec 2010 17:50:13 GMT
Cache-Control: private, max-age=0
X-Content-Type-Options: nosniff
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
Server: GSE
X-Cache: MISS from proxy
Via: 1.0 proxy (proxy)
Connection: keep-alive

PK........N.<-................res/anim/animation_none.xml....].;n.1.E.q.IG.
Spying users...
Getting more than location

- Much more interesting information in the different logs:
  - Phone calls (numbers & duration)
  - SMS (PDU format)

- Combination of information:
  - Where did phone calls take place?
  - Where were SMS sent/received?
  - Recovery of deleted SMS, call history...
Getting more than location

- History length?
  - It depends on log filling
    - If user has moved quickly: a few hours
    - If not: nearly a whole day

- Logs size can be changed...
Getting more than location

Complete geolocation, calls and SMS history tracking!

(nearly or no permission needed...)

#27c3 – 27-30 December 2010 – Berlin
“Android geolocation using GSM network”
Renaud Lifchitz
How to protect yourself?
How to protect yourself?

- Carefully look at applications using NDK (apk archives embedding .so files)
- Don't install any application requiring READ_LOGS permission
- Don't submit bug reports (or at least choose not to include system logs with submission)
- Reduce logcat buffer size (seems tricky: logcat -r / logcat -n)
- Often clear your logcat (logcat -b radio -c)
- Disable radio logs (seems tricky too!)
Tool demo
Tool demo

Dumping and viewing a user's past location history

#27c3 - 27-30 December 2010 - Berlin

“Android geolocation using GSM network”

Renaud Lifchitz
That's all folks!

Hope you enjoyed the talk!

#27c3 – 27-30 December 2010 – Berlin
“Android geolocation using GSM network”
Renaud Lifchitz
Any questions?

Many thanks for attending!