Should I Protect You? Understanding Developers’ Behavior to Privacy-Preserving APIs

Shubham Jain and Janne Lindqvist
Department of Electrical and Computer Engineering
Rutgers University

Workshop on Usable Security (USEC’14)
February 23, 2014
Developers? Developers! Developers!
Privacy-Preserving API design?

- **Observation**: today developers have options
  - take all,
  - or nothing
- **Evidence**: some developers are trying to follow least privilege
- **1. Question**: Can we design a privacy-preserving clean-slate API?
Can We Nudge Developers?

• 1. *Question*: Can we design a privacy-preserving API?
  – Yes
  – Other have done it, too!

• *What we should be asking*: Can we nudge developers to make better user privacy decisions with API designs?
Localization Options (Permissions)

- **ACCESS_FINE_LOCATION (GPS)**

  Your location
  Precise location (GPS and network-based)

- **ACCESS_COARSE_LOCATION (WiFi or cell network)**

  Your location
  Approximate location (network-based)

- “To meet the privacy expectations of users when your app requests permission for coarse location (and not fine location), the system will not provide a user location estimate that’s more accurate than a city block.” – Android 4.2.
// Acquire a reference to the system Location Manager
LocationManager locationManager = (LocationManager) this.getSystemService(Context.LOCATION_SERVICE);

// Define a listener that responds to postal code updates
LocationListener locationListener = new LocationListener() {
    public void onLocationChanged(Location location) {
        String msg = "Updated Location: " +
                      Double.toString(location.getLatitude()) + "," +
                      Double.toString(location.getLongitude());
    }

    // And then reverse geocoding
Example Modified API

// Acquire a reference to the system Location Manager
LocationManager locationManager = (LocationManager)
this.getSystemService(Context.LOCATION_SERVICE);

// Define a listener that responds to postal code updates
LocationListener locationListener = new LocationListener() {
    public void onPostalCodeChanged(Location location) {
        String zipCode = location.getPostalCode();
        getMyWeather(zipCode);
    }
};
Method

• Participants screened and randomly divided to five groups
• Non-Android Group (Some Java experience)
  – Control Group (using just the baseline API)
  – Treatment group A (TA)
  – Treatment group B (TB)
• Android Group (Some Experience with Java/Android)
  – Treatment group C (TC)
  – Treatment group D (TD)

• No mention about privacy to avoid biasing participants.
  – Questions about privacy after completing the tasks
Method

- Android Location baseline API documentation
- Treatment Android Location API
  - Everything in the baseline API
  - And our modified APIs
  - Order of the presentation varied between treatment groups (TA, TC) vs (TB, TD)

- Programming Tasks:
  - Weather app
  - Running app
  - Address app
Method: Lo-fi programming

Prototyping for Tiny Fingers

<DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Challenge #2</title>
  </head>
  <body>
    <h1>The interesting challenge #2</h1>
    <p>All my attention on Challenge #2. This code is the funniest code I ever wrote.</p>
  </body>
</html>
## Results

<table>
<thead>
<tr>
<th>Group</th>
<th>Participants</th>
<th>Used Our API</th>
<th>Reverse Geo</th>
<th>Copied example</th>
<th>???????????</th>
</tr>
</thead>
<tbody>
<tr>
<td>CG</td>
<td>6</td>
<td>N/A</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>TA</td>
<td>5</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>TB</td>
<td>6</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>TC</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TD</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Why?

• “I tried to make it the postal code or city because that is usually what people want. They don’t usually want latitude and longitude” - TA2, on using the getPostalCode(), requestPostalCodeUpdates() and onPostalCodeChanged() for the weather task.

• “Geocoder was the most confusing part” - TB5.
Why not?

• “You get them [geocoordinates] from location manager. Then you have to use this part - geocoding. I tried to do that for this one but I didn’t really know how to” - TB2.

• “I may have chosen this [Geocoder class] because it was first. I was reading through and I saw this and I was like, oh that will work” - TD4.
When Asked About Privacy

• “I know about them [location privacy issues]. It flashed my mind for a second, like do you want to track every single detail? But then I just continued doing what I was doing” - TA3 (used our API).

• “That’s why I tried to avoid GPS when possible because lots of people are sensitive to giving fine location data away. And I tried to use the network when possible because even if they’re sure they know you’re connected to this tower, still towers cover such a vast area and depending on where you are there is such a huge number of people attached to that network they can’t identify who you are without more information on that” - TC1 (used our API).
When Asked About Privacy

- “Your phone is capable of sending your coordinates at all times to a server. I chose to use postal code as opposed to street address or coordinates because I didn’t want to send out too much information” - TC4, discussing his code on weather application.

- “I didn’t think about it [location privacy] because I just assume that once they [users] install the application they’ve already given permission for it.” - TC3
Limitations

• Participants Rutgers CS/ECE undergrads/grads

• Small group sizes, no statistical analysis

• Monetary incentives: 3rd party ad-network libraries

• StackOverflow?
Conclusions

• When approaching API documentation from a “blank slate” participants tend to follow the sample code closely.

• First step to indicate that if developers have privacy-preserving examples in official documentation, developers could be using them instead of less privacy-preserving alternatives.
Shameless Plug

• Afternoon session: Huiqing Fu et al. “A Field Study of Run-Time Access Disclosures on Android Smartphones”

• Over 200 articles around the world.
  – New Age Online (?), US liberal and conservative media
Thank you
janne@winlab.rutgers.edu
BACKUP SLIDES
Caché Architecture (Amini et al., MobiSys’11)