ACComplice: Location Inference using Accelerometers on Smartphones

Jun Han, Emmanuel Owusu, Thanh-Le Nguyen
Adrian Perrig, Joy Zhang
Carnegie Mellon University
Introduction

• Smartphones feature numerous **sensors**
• Many users concerned about GPS sensors leaking **location privacy**
  • Mobile OS companies were reported to collect GPS location information of their customers
• Other sensors (e.g., accelerometers) considered benign
• **Should** we consider **acceleration data (accelerometer)** to be privacy sensitive?
Yes, we should!

- Using only accelerometer data, we can derive a car’s **traveled route** and **starting location** within a **200 meter radius** (without knowing the starting location!)

- Main Challenges
  1. Initial position is unknown.
     - ACComplice does not rely only on using dead reckoning
     - Dead reckoning: Calculates one’s position using previous position and advances that position with **estimated speed and time**
  2. Trajectories are very noisy.
Accelerometer

- Measure the acceleration experienced by a device
  - Used in many smartphone applications
    (e.g., Gaming, Activity Recognition)

- Can expose sensitive private information to a malicious application!
Adversary Model

• Malicious application logs accelerometer data in the background

• Malicious application execute on the mobile device without special privileges
  – Needs permission to access accelerometer

• The application can communicate with the external server

• OS is not compromised
Background

• Probability Inertial Navigation (ProbIN) uses statistical model to provide estimated trajectory from noisy sensor readings
  – Nguyen and Zhang, “Probabilistic Infrastructureless Positioning in the Pocket”, MobiCase 2011

• Outperforms traditional approaches
  – Traditional physics approach uses double integral of acceleration data to obtain the displacement – which is less accurate due to aggregated error
  – Dead reckoning is noisy because error accumulates over time and traveled distance

• Statistical model consists of two parts:
  – Translation model
  – Trajectory model
ProbIN

• Translation Model:
  – **Quantized** using K-means clustering into *motion labels* sequence, M
  – From M, extracts the optimal *displacement labels* sequence, D
    • Green cluster = Left turn
    • Red cluster = Forward
    • Blue cluster = Right turn

• Trajectory Model:
  – Using D, establishes a “**grammar**” for vehicle motion based on past information (equivalent to n-gram language model)
  – Assigns higher probability to “**normal**” trajectory patterns
    • [Fwd-Fwd-Right-Fwd] vs. [Fwd-Back-Fwd-Back]
Trajectory Reconstruction

- Car driven for a total of 22 km

- ProbIN vs Physics Based Approach

- Although ProbIN seems to be similar to GPS data…
  - When overlayed onto a map, differs significantly
  - ProbIN alone cannot provide useful information about user’s traveled route
Contribution

- ACComlice identifies the **trajectory** and **starting point** of an individual driving in a vehicle based **solely on** accelerometer measurements

- **Overview**
  1. “Snap” trajectory obtained using ProbIN to map
  2. Predict starting location
Challenge 1: Map Matching

• Final task is to “snap” trajectory to a real-world roadways

• Map each motion trajectory (ProbIN) point to the best corresponding candidate segment
  – Similarity measures (distance and angle) used to map motion trajectory onto road segments
  – Realign the motion trajectory to mitigate inherent noise effects
Map Matching Algorithm

\[ Ci = \text{Candidate Segment } i \quad \text{Pi} = \text{Point } i \text{ on a motion trajectory} \]
Map Matching Algorithm

$C_i = \text{Candidate Segment } i \quad P_i = \text{Point } i \text{ on a motion trajectory}$
Map Matching Algorithm

Ci = Candidate Segment i
Pi = Point i on a motion trajectory
Map Matching Algorithm

\[ C_i = \text{Candidate Segment } i \quad P_i = \text{Point } i \text{ on a motion trajectory} \]
Map Matching Algorithm

(a) Pittsburgh, PA
(b) Mountain View, CA

Indicates the starting point
Challenge 2: Starting Point Prediction

- Align trajectory to all starting points on map and perform map matching algorithm → Compare different starting points to find the *most likely starting point*

- Compute difference scores, $DS$
  - $DS$ indicates how similar *mapped points* are to the corresponding points on the *motion trajectory*
    - Sum up distances between the two corresponding point pairs
  - Mapped points that are similar to the motion trajectory will have smaller sum of the distances → Leading to a *lower $DS$*

- Starting point with a lower $DS$ is more likely to be the actual starting point

- Rank all valid difference scores
Challenge 2: Starting Point Prediction
Pittsburgh, PA
Pittsburgh, PA
Pittsburgh, PA
Mountain View, CA

GPS Trajectory
- Stretched Factor: 1.50
- Stretched Factor: 1.25
- Stretched Factor: 1.00
Mountain View, CA
Mountain View, CA

GPS Trajectory
- Stretched Factor: 1.50
- Stretched Factor: 1.25
- Stretched Factor: 1.00
- Forms a cluster because all points in a cluster get mapped to a similar route
- Locate device owner to within a 200 meters of the true location
Starting Point Prediction

• Longer trajectories provide more information
  – Counted the number of predicted starting points that form in the 200 meter cluster region, as the length of the trajectory was varied
  – More globally unique constraints
  – More points in a cluster region → More likely to be a true starting point
Discussion / Conclusion

• Derive location using only displacements
  – Starting point localized to 200 meter radius.
  – Trajectory aligned in close correspondence to the ground truth.

• Limitations?
  – Road network with perfect grid-like structure?

• Number of portable devices equipped with accelerometers increased → may lead to prevalent attacks

• Accelerometers cannot be shielded

• Opportunities for more sophisticated attacks
  – Include more constraints
    (e.g., traffic light timings, speed limits, pot holes, incline changes, road angles)
  – Tracking device?
Jun Han (junhan@cmu.edu)
CyLab / Carnegie Mellon University