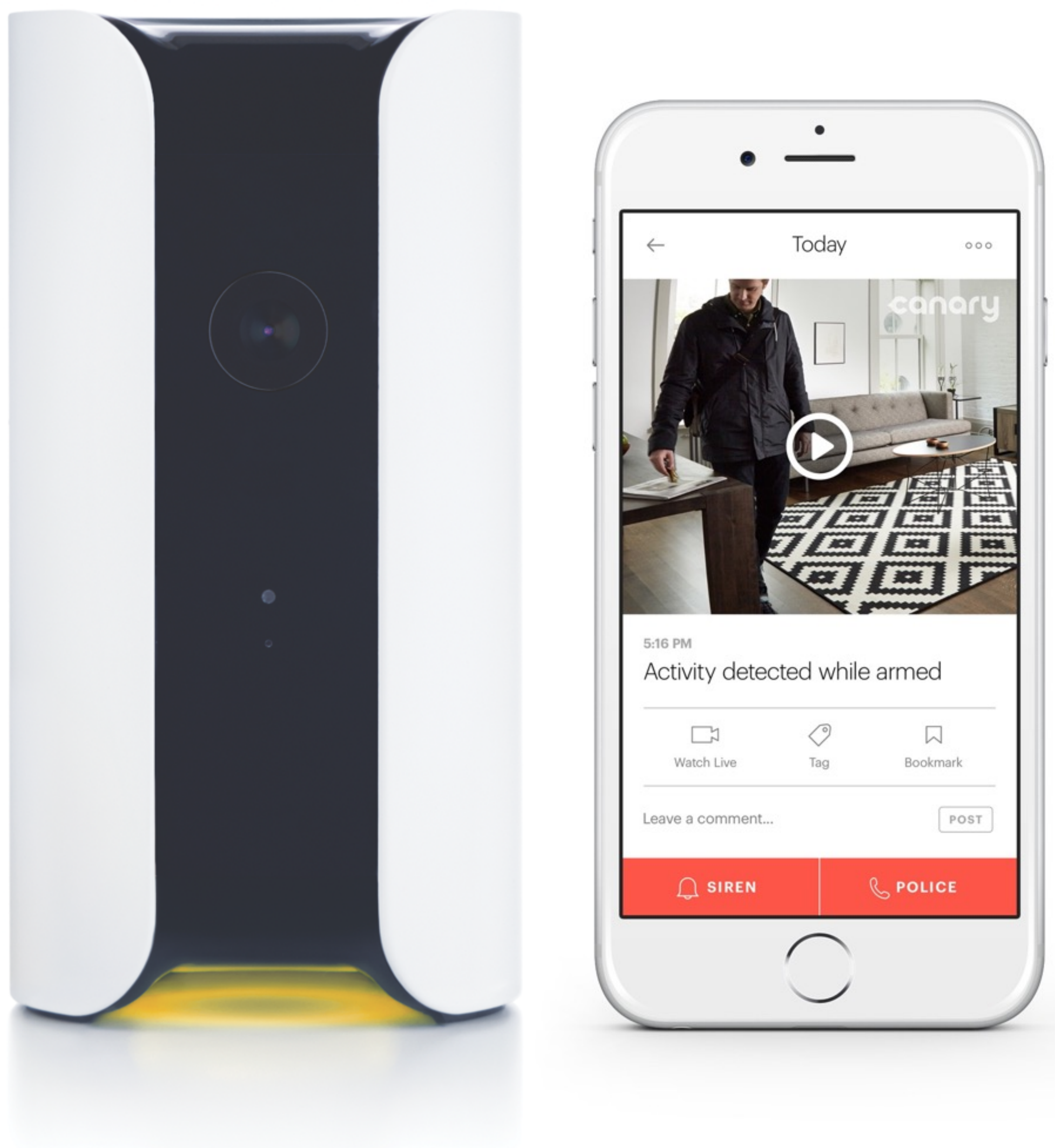
Canary Geofencing

May 17, 2016

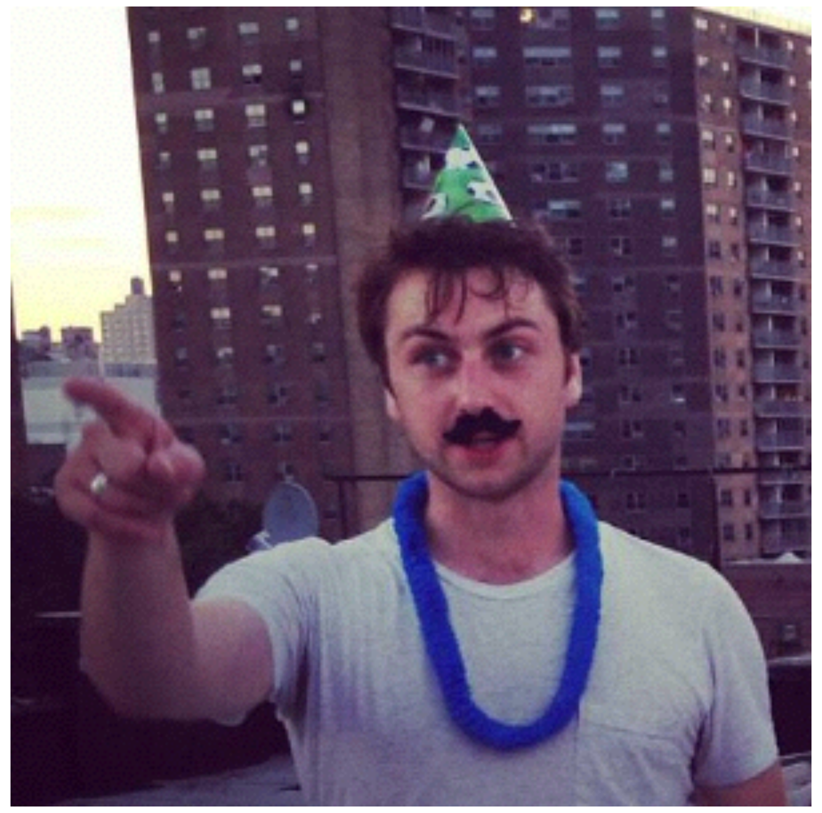
canary

Welcome to Canary!

Canary is the all-in-one home security system you control from your phone.



Meet the Mobile Team

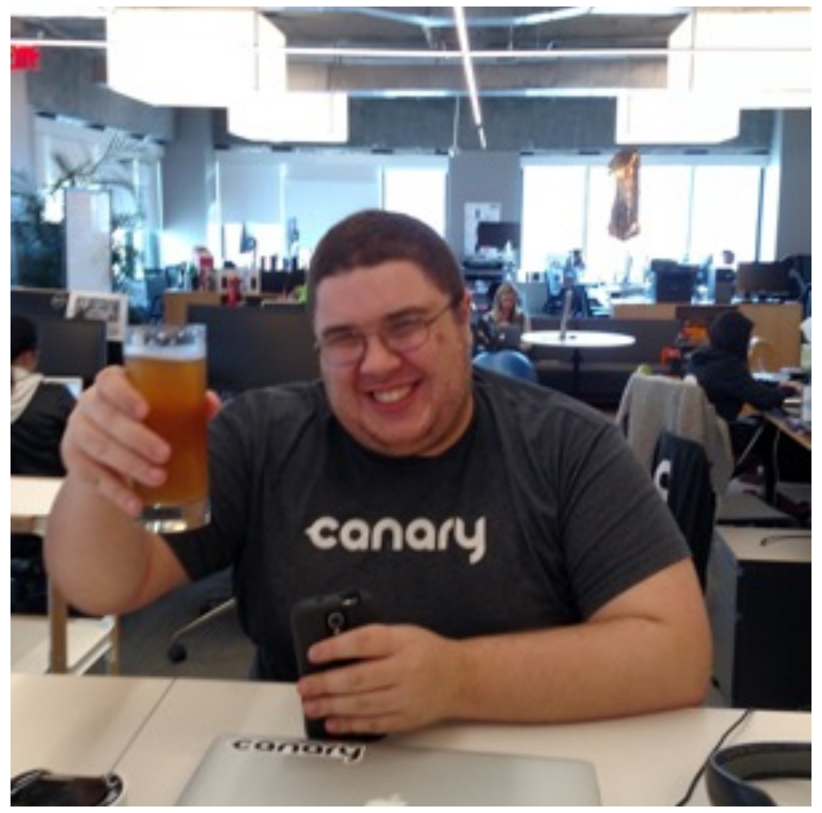


Tiernan Kennedy

Team lead

Michael Klein

iOS



Andrew Whitcomb

iOS

Michael Schroeder

Android

Sergey Morozov

Android

What is a geofence?

- A geofence is a virtual barrier around a location or region.
- Geofences are typically used to alert a user when they have entered or exited this region.
- Example: The iOS Reminders app allows you to remind you when you get home.
- Location services are comprised of:
  1. Cellular
  2. Wi-Fi
  3. GPS

Friends Seminary
E 14th St
E 15th St
Rutherfo
Trader Joe's
E 13th St
Peridance Capezio Center
IHOP
3 Av [L]
M
Baohaus
E 12th St
The Penny Farthing
E 13th St
Webster Hall
momofuku milk bar
Momofuku Ssäm Bar
St
3rd Ave
E 12th St
2nd Ave
The Smith
E 11th St
Angelica Kitchen
St. Mark's

Why does Geofencing matter to Canary?

Let’s first take a look at traditional security system.

- Keypads & pincodes
- A hassle to arm/disarm
- Most people don't bother

Canary is a passive system

- Simple system that lives in the background
- Use your location to arm/disarm your Canary.
- Works with many users in a household

The original approach

Use region monitoring!

- Region monitoring creates a simple geofence around your location.

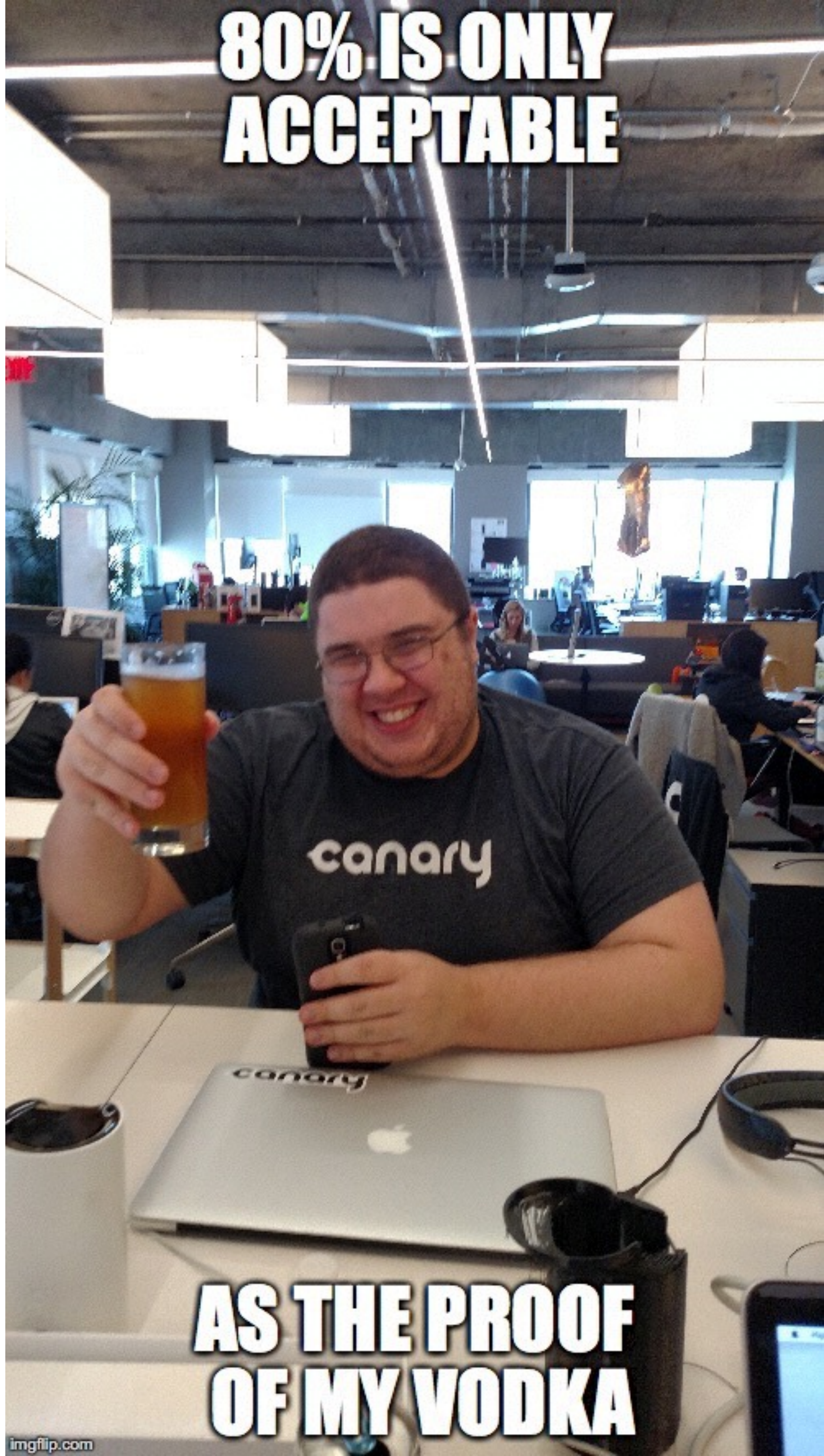
User
Home region

```
CLCircularRegion *region = [[CLCircularRegion alloc] initWithCenter:CLLocationCoordinate2DMake(<Latitude>,
<Longitude>) radius:<radius> identifier:<indentifier>];

[locationManager startMonitoringForRegion:region];

...

-(void)locationManager:(CLLocationManager *)manager didEnterRegion:(CLRegion *)region {
    //User entered the region
}

-(void)locationManager:(CLLocationManager *)manager didExitRegion:(CLRegion *)region {
    //User exited the region
}
```

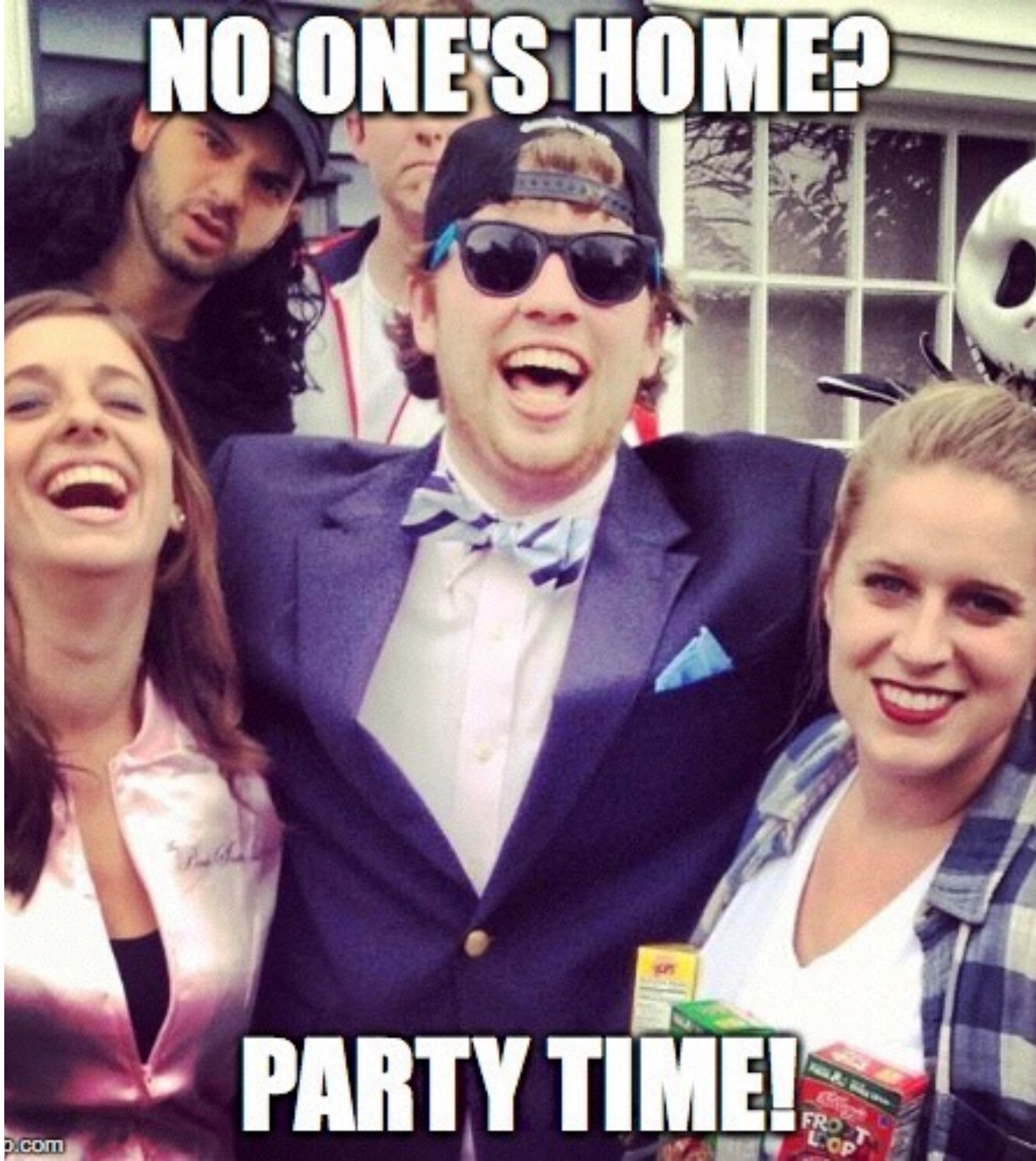
Problems with this approach

Well, really, it does work, but not to our standards; it is only accurate about 80% of the time.

- OS reliability issues: we never get the event
- Connectivity: we get the event when user has no internet connection
- Battery: user’s phone is dead when they enter/exit the region



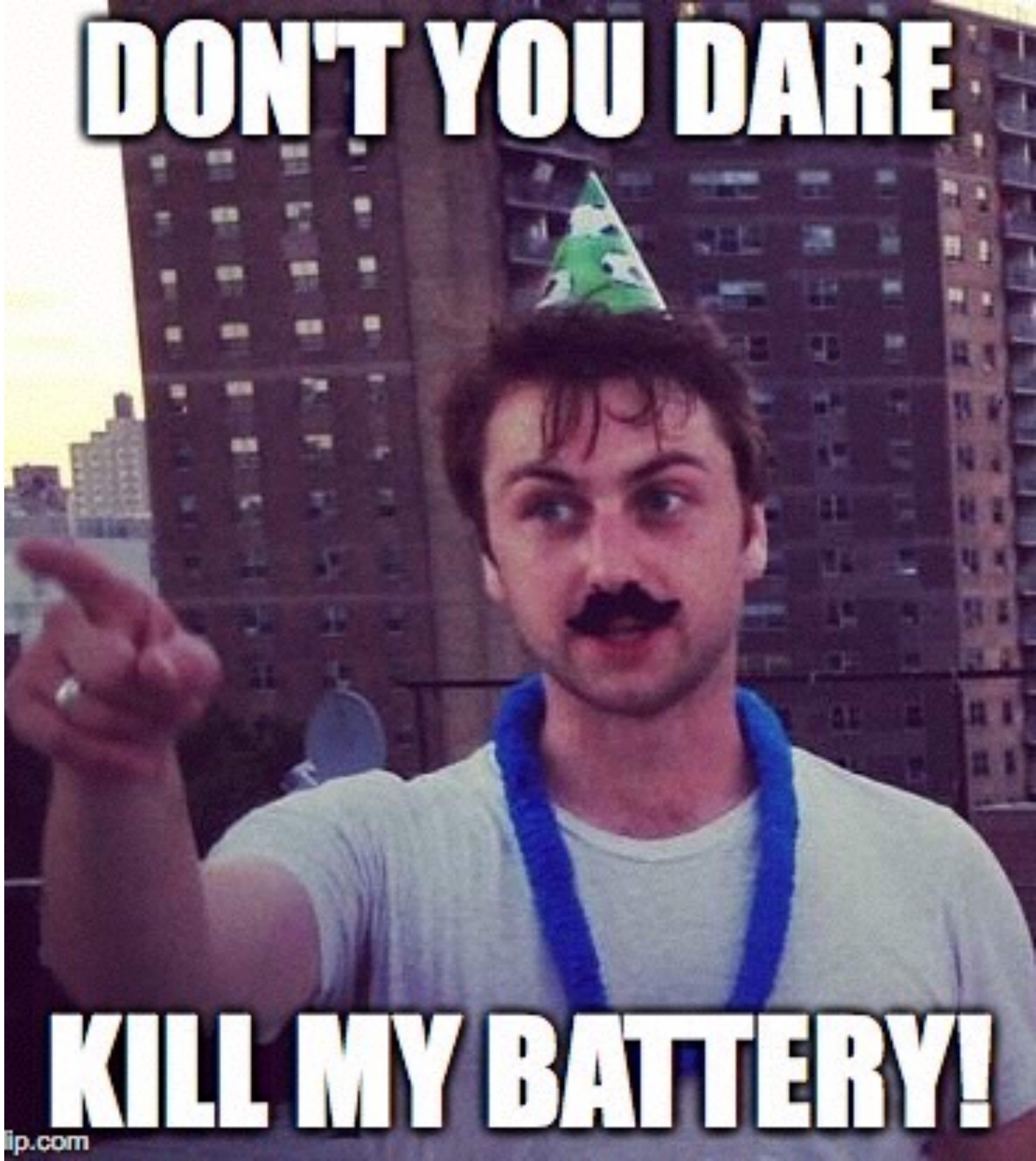
This seriously harms user trust...
Most importantly, this event only fires once, when the user leaves their location. If the Canary never arms: very serious issue for the system.



Research and things to consider

Always need to consider a user’s battery life!

- No battery, no location
- Device never changes modes
- Frustrating user experience



Research and things to consider

CLVisits

User
Home region

- Alerts when a user enters/exits a location they frequent
- Are delayed up to 15 minutes from arrival/departure

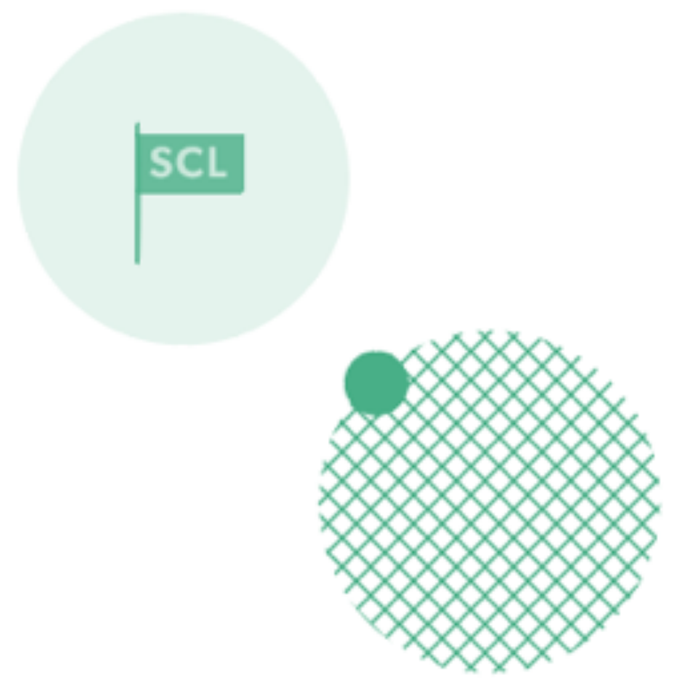
```
if ([locationManager respondsToSelector:@selector(startMonitoringVisits)]) {
    [locationManager startMonitoringVisits];
}

...

-(void)locationManager:(CLLocationManager *)manager didVisit:(CLVisit *)visit {
    CLLocation *visitLocation = [[CLLocation alloc] initWithLatitude:visit.coordinate.latitude
longitude:visit.coordinate.longitude];
    //Check if visit location coordinates are relative to users locations
}
```

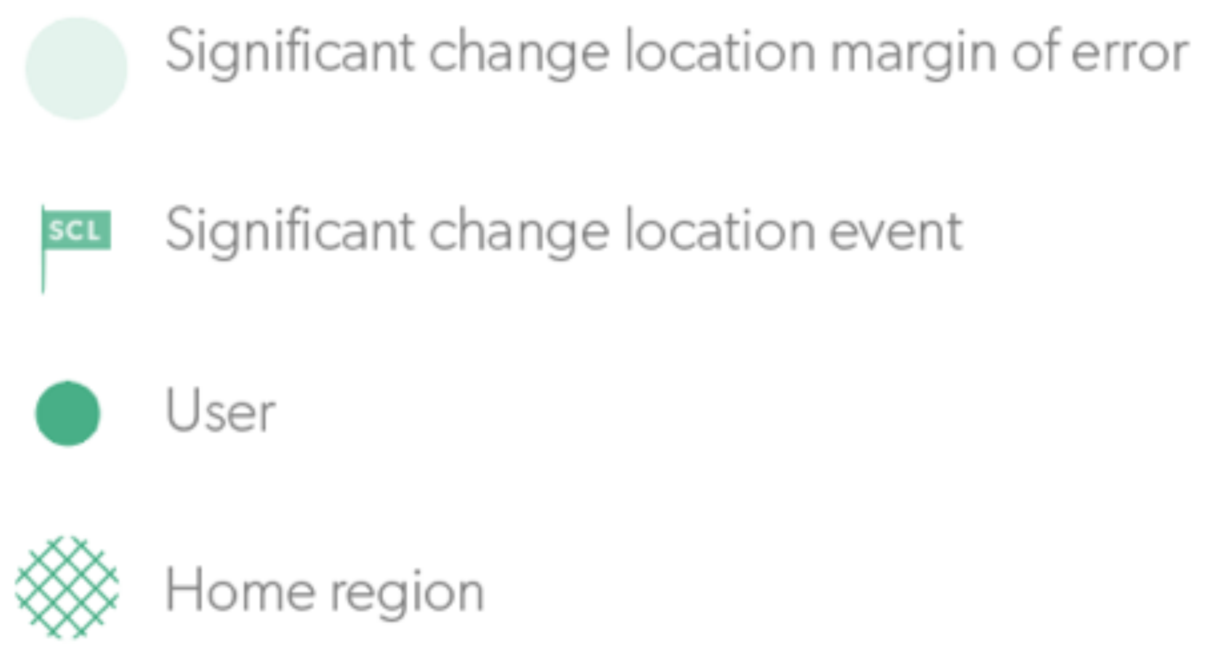
Research and things to consider

Significant Location Changes (SLCs)

- Low accuracy, low battery usage

```
[locationManager startMonitoringSignificantLocationChanges];
```

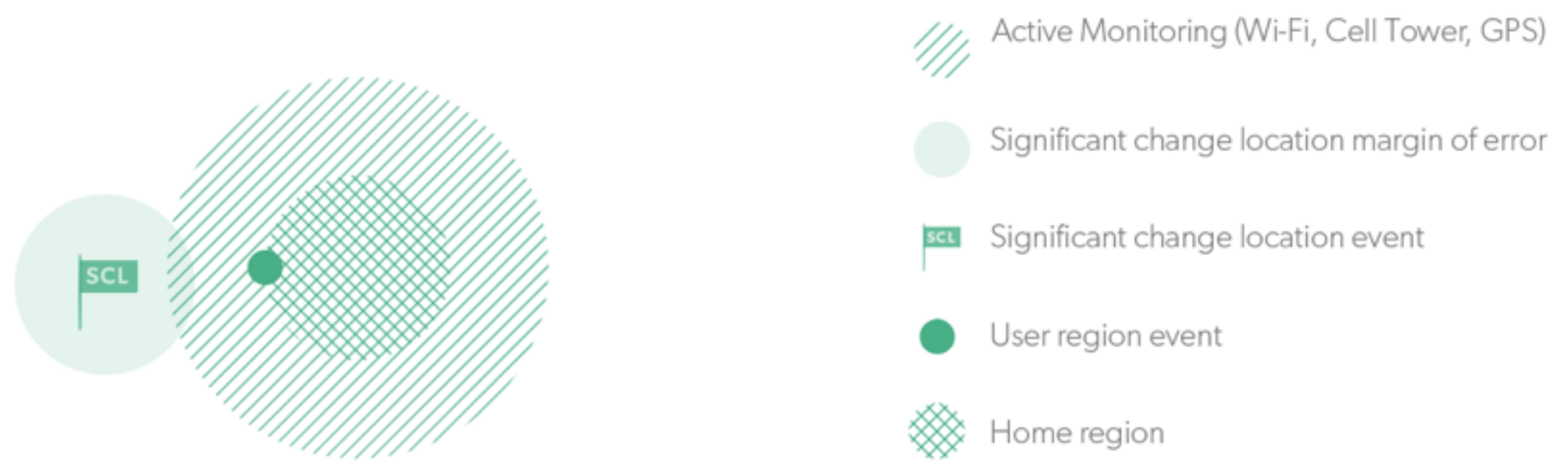



Significant change location margin of error
SCL
Significant change location event
User
Home region

Research and things to consider

Active Monitoring (AM)

- Higher accuracy when necessary

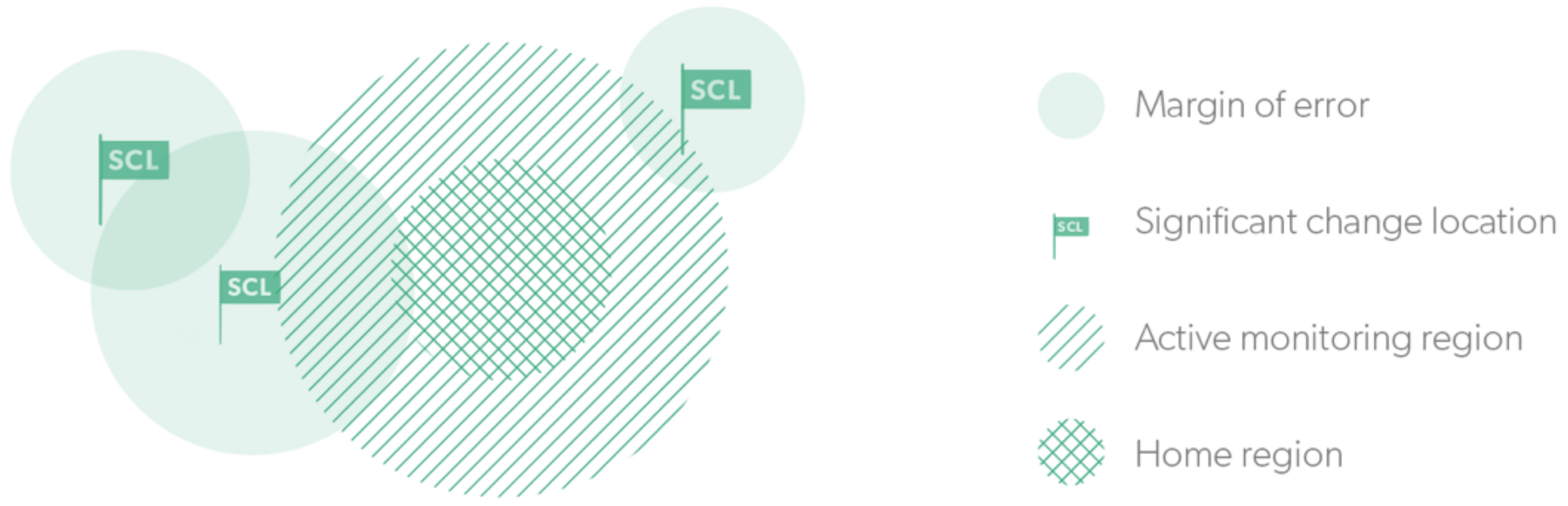

SCL
Active Monitoring (Wi-Fi, Cell Tower, GPS)
Significant change location margin of error
Significant change location event
User region event
Home region

```
locationManager.desiredAccuracy = kCLLocationAccuracyHundredMeters;
[locationManager startUpdatingLocation];

-(void)locationManager:(CLLocationManager *)manager didUpdateLocations:(NSArray *)locations {
    //Collect user location
}
```

Putting it all together

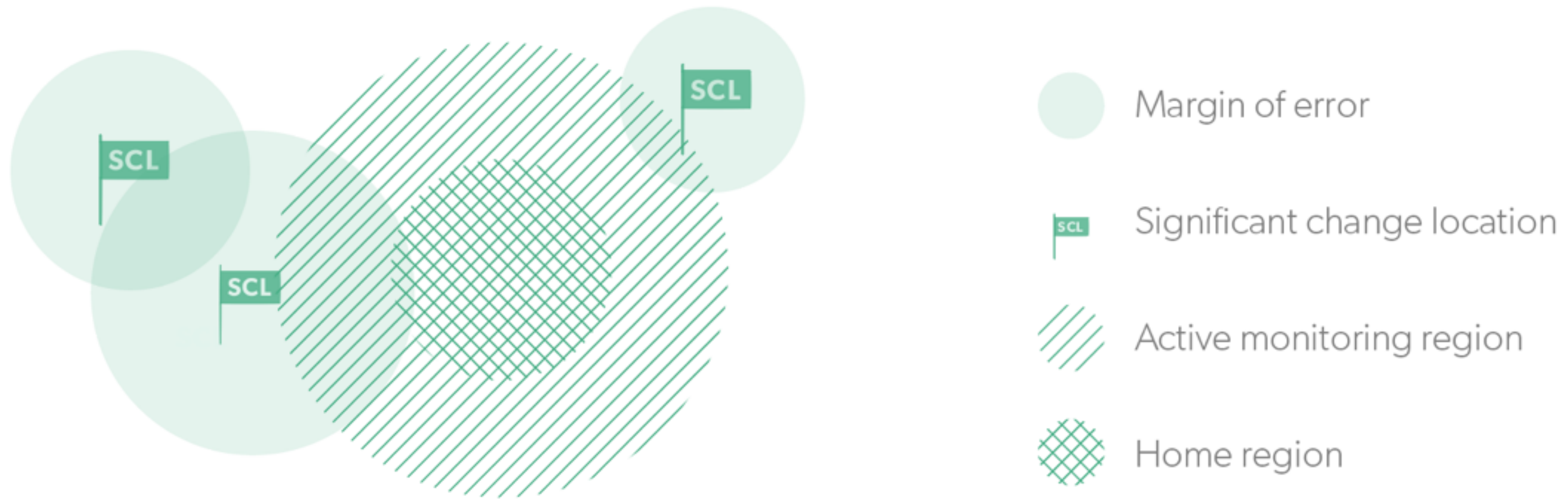
Cut the problem into 3 sections

1. The Home region
2. The Active Monitoring region
3. Everywhere else



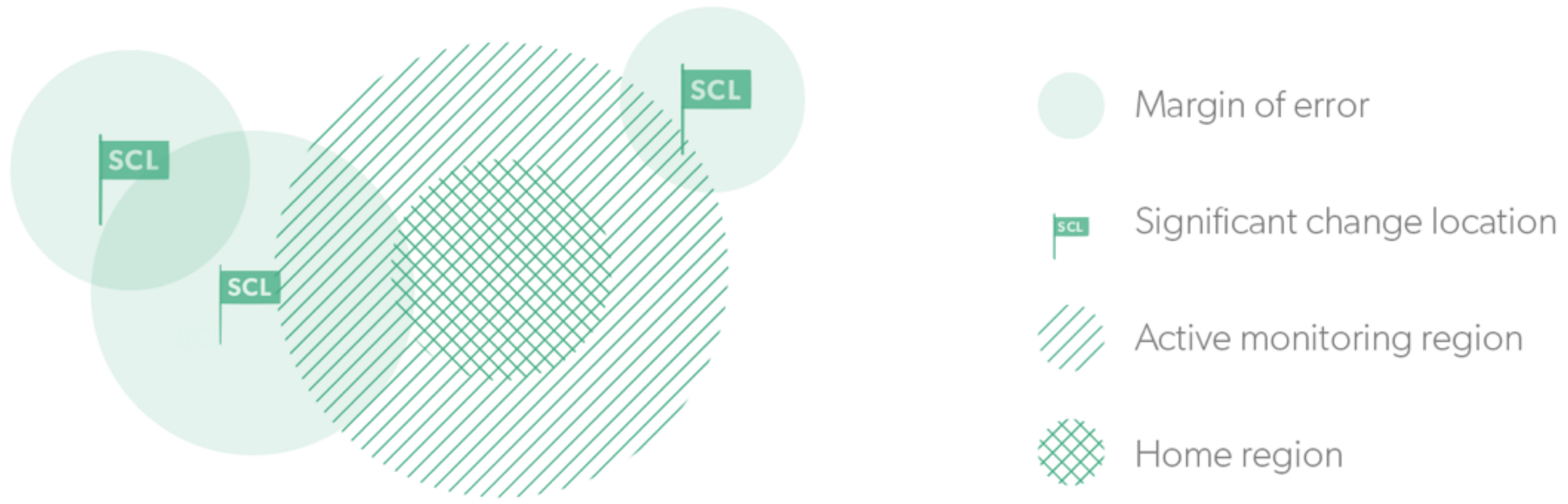
Putting it all together

1. Always monitor for SLCs and Visits
2. Always monitor for Region changes
3. Use Active Monitoring when we are close to a region



Putting it all together

Only enable highest location services when we are unsure of location.

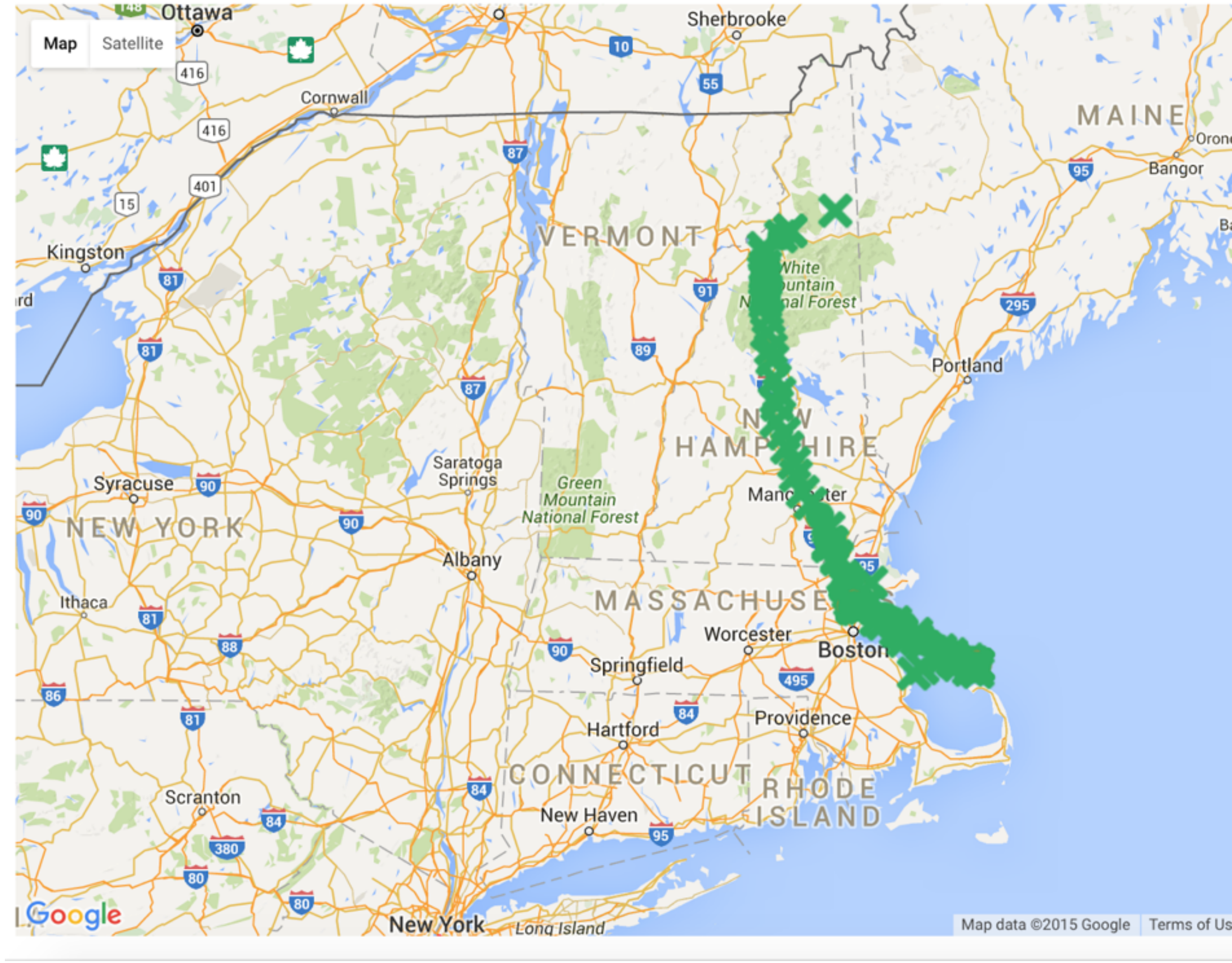


```
self.locationManager.desiredAccuracy = kCLLocationAccuracyNearestTenMeters;
```

Ready to test!

Create an endpoint to upload system information to:

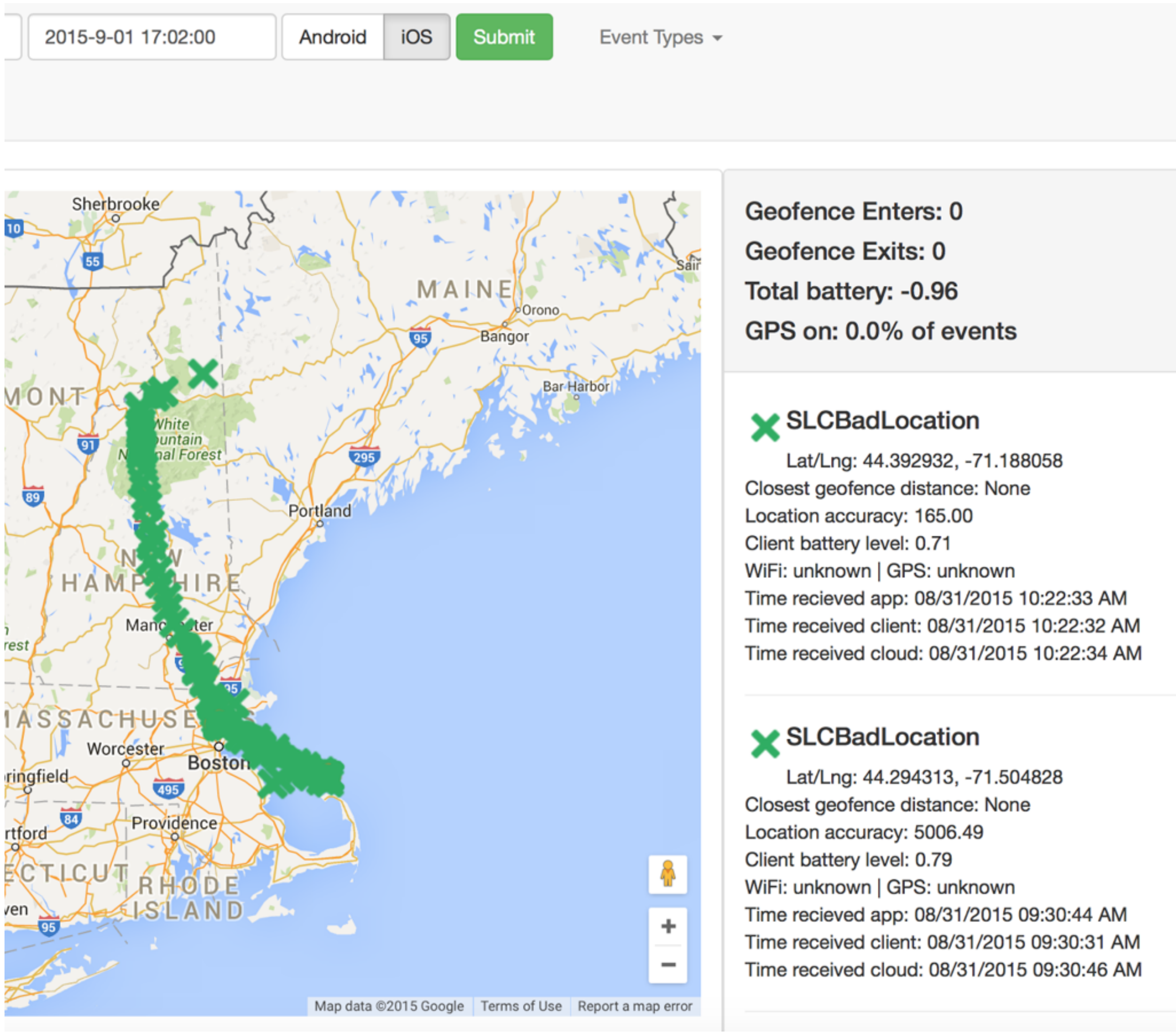
- Battery level
- Latitude/Longitude
- Wi-Fi/GPS on
- Time
- Accuracy
- Closest location



Ready to test!

- We built a custom web app in flask called “Geofencer”
- Placed the pins on a Google Map for analysis
- Run scripts every day on our testers to check for issues
- **Aims:**
  - Increasing accuracy
  - Only turning on GPS chip when absolutely necessary
  - Observing real world behavior to identify cases to where battery use could be increased

Not tracking production users!



Geofence Enters: 0
Geofence Exits: 0
Total battery: -0.96
GPS on: 0.0% of events

SLCBadLocation
Lat/Lng: 44.392932, -71.188058
Closest geofence distance: None
Location accuracy: 165.00
Client battery level: 0.71
WiFi: unknown | GPS: unknown
Time recieved app: 08/31/2015 10:22:33 AM
Time received client: 08/31/2015 10:22:32 AM
Time received cloud: 08/31/2015 10:22:34 AM

SLCBadLocation
Lat/Lng: 44.294313, -71.504828
Closest geofence distance: None
Location accuracy: 5006.49
Client battery level: 0.79
WiFi: unknown | GPS: unknown
Time recieved app: 08/31/2015 09:30:44 AM
Time received client: 08/31/2015 09:30:31 AM
Time received cloud: 08/31/2015 09:30:46 AM

Conclusions

1. Battery life changes are negligible
2. System recovers from missed events
3. Overall improvement in accuracy of arm/ disarm events
4. Many memes created and miles walked

Questions?