Map All The Things

Geolocation for Mobile Forensics Practitioners

Kim Thomson
H-11 Digital Forensics
kim@h11dfs.com

who am I?

- Retired SIGINT Soldier
- Nerd
- I love all things wireless
- My passion is extraction/recovery and decoding of phone/device data
- I teach courses in mobile forensics, chipoff for mobile forensics, JTAG-ISP for mobile forensics, Python for mobile forensics, smartphone analysis, and wireless tracking, mapping, and analysis
- I settled into mobile forensics because of the variety of areas within the field
- kim@h11dfs.com

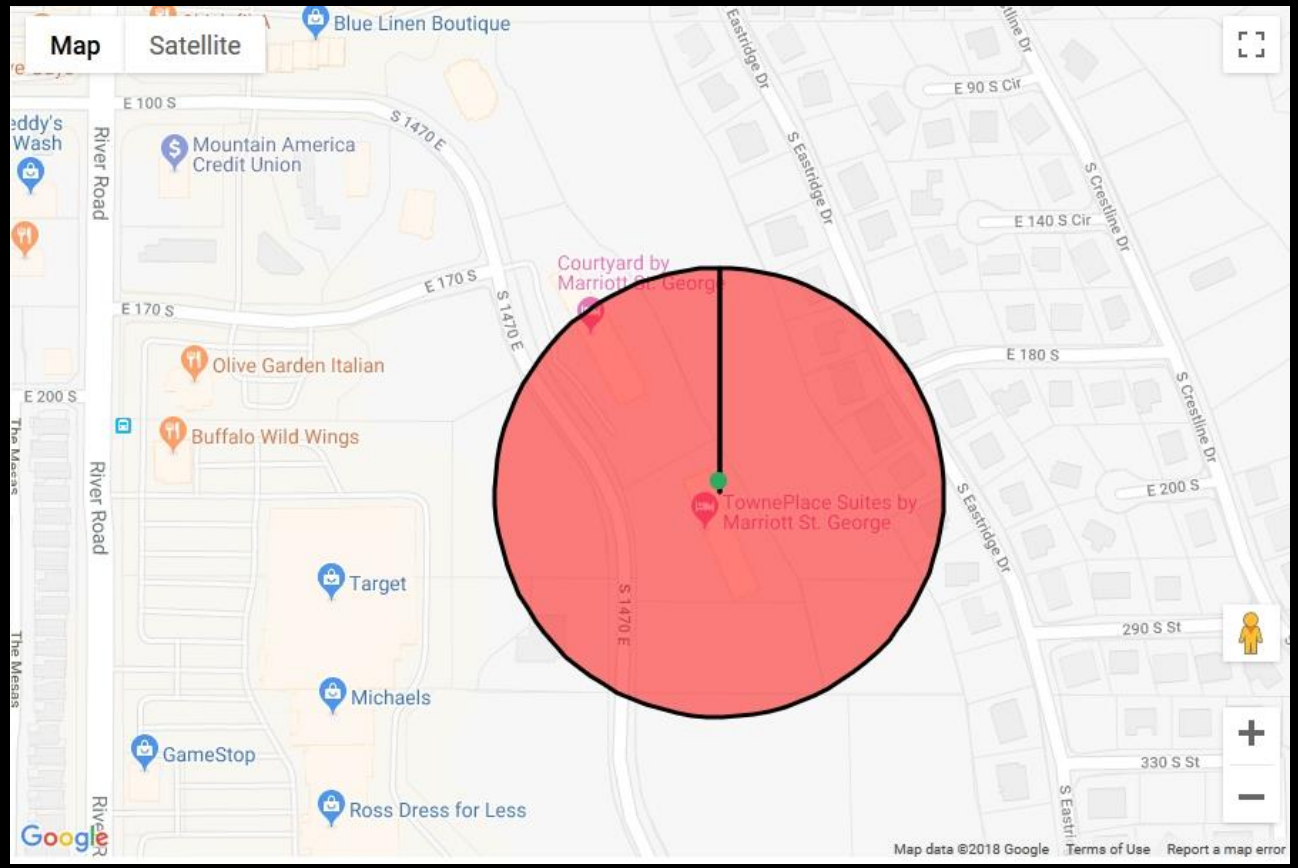
geo-location = what + when + where

Any method or set of methods that returns a location on the ground, which may correlate an action with the location where that action took place. May include a timestamp as well.

WHAT + WHEN + WHERE

what can we do with it?

- Find out what happened where and when
- Trace the location history and general trends of a device
- Get general locations of activity
- Get more specific locations of activity
- Exclude locations



geo-location limitations

- Can be weird, inconsistent , or just inaccurate
- Often must be taken in groups and trends, not specific points
- You must understand where the data is coming from
- Phones sometimes just have the cell tower, which is a couple miles away
- Haters gonna hate…

where does it come from?

- WiFi connections or observations
- BT connections of observations
- Cell connections or observations
- weather apps
- navigation apps
- mapping apps
- social-media apps
- DHCP records
- Geotags on media
- Google location timeline
- CDRs (probably the most accurate)

android dumpsys

- need ADB (USB Debugging)
- root not required
- adb shell “dumpsys” > somefile.txt
- adb shell “dumpsys netstats detail” > otherfile.txt
- netstats
- wifiscanner
- wifi
- netstats!
- usagestats
- batterystats
- netstats…

dump_all_the_sys.py

```
Latest scan results:
    BSSID              Frequency  RSSI  Age(sec)    SSID
  58:b6:33:28:f2:3c       5745    -51    28.979    Marriott_GUEST
  58:b6:33:a8:f2:3c       5745    -51    28.979
  58:b6:33:a8:d5:cc
  58:b6:33:28:d5:cc
  58:b6:33:29:0d:b8
  88:ad:43:84:c2:28
  58:b6:33:a9:0d:bc
  58:b6:33:a9:10:cc
  58:b6:33:29:2e:1c
  58:b6:33:29:10:cc
  58:b6:33:29:0d:bc
  22:86:8c:6b:7a:49
  86:f2:9e:99:e1:20
  58:b6:33:29:07:ac
  42:86:8c:6b:7a:49
  52:86:8c:6b:7a:49
  58:b6:33:a9:07:ac
  9a:ad:43:84:c2:28
  58:b6:33:28:d3:bc
  96:ad:43:84:c2:28
  78:f2:9e:99:e1:20
```

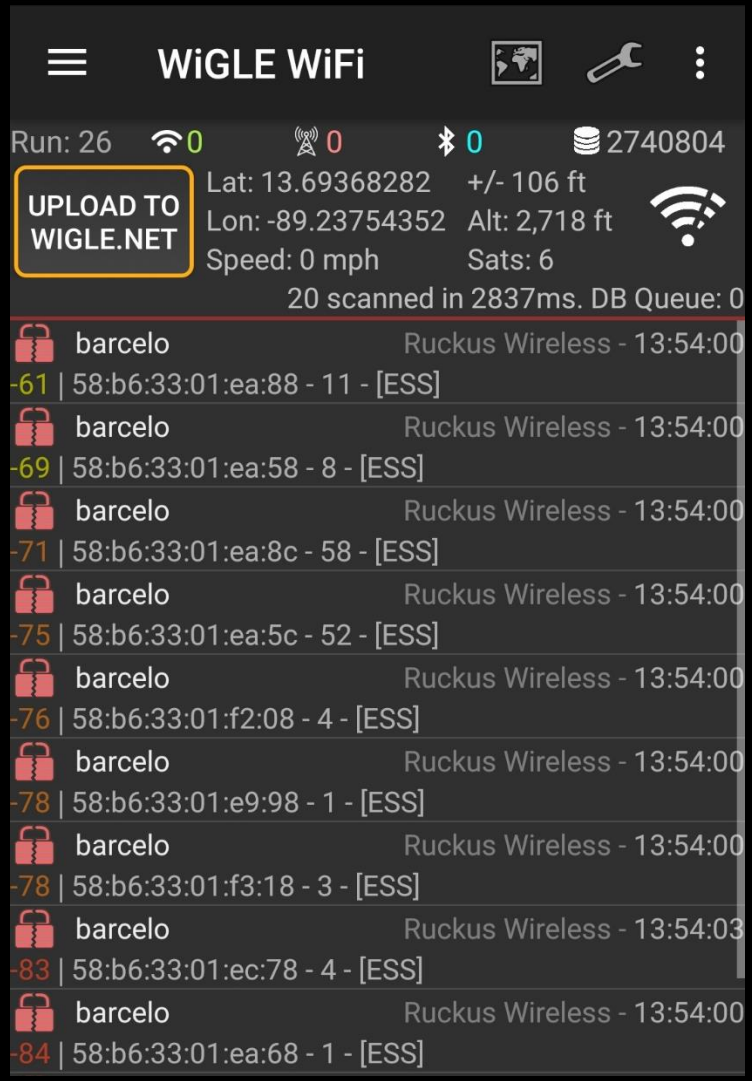
wifiscanner

dumpsys_files

Name	Date modified	Type	Size
dumpsys.txt	2018-09-05 09:08	TXT File	20,284 KB
package.txt	2018-09-05 09:05	TXT File	4,477 KB
wifi.txt	2018-09-05 09:05	TXT File	2,314 KB
netstats.txt	2018-09-05 09:05	TXT File	2,105 KB
dbinfo.txt	2018-09-05 09:05	TXT File	1,199 KB
bluetooth_manager.txt	2018-09-05 09:05	TXT File	773 KB
notification.txt	2018-09-05 09:05	TXT File	503 KB
usagestats.txt	2018-09-05 09:05	TXT File	462 KB
storaged.txt	2018-09-05 09:05	TXT File	347 KB
batterystats.txt	2018-09-05 09:05	TXT File	336 KB
activity.txt	2018-09-05 09:04	TXT File	286 KB
jobscheduler.txt	2018-09-05 09:05	TXT File	262 KB
nfc.txt	2018-09-05 09:05	TXT File	246 KB
content.txt	2018-09-05 09:05	TXT File	214 KB
graphicsstats.txt	2018-09-05 09:05	TXT File	203 KB
CustomFrequencyManagerService.txt	2018-09-05 09:04	TXT File	189 KB
shortcut.txt	2018-09-05 09:05	TXT File	174 KB
procstats.txt	2018-09-05 09:05	TXT File	152 KB
appops.txt	2018-09-05 09:04	TXT File	135 KB
media.camera.txt	2018-09-05 09:05	TXT File	127 KB
settings.txt	2018-09-05 09:05	TXT File	117 KB
wifiscanner.txt	2018-09-05 09:05	TXT File	117 KB

WiFi

- Range < 150m generally
- Fairly precise
- Terms:
  - SSID (Service-set identifier) – network name
    - up to 32 characters
  - BSSID (Basic SSID) – real physical name, unique
    - six bytes
- BSSID Breakdown:
  - Organizationally-Unique ID (OUI) – 1st three bytes, identifies vendor/manufacturer
  - Last three bytes – specific device



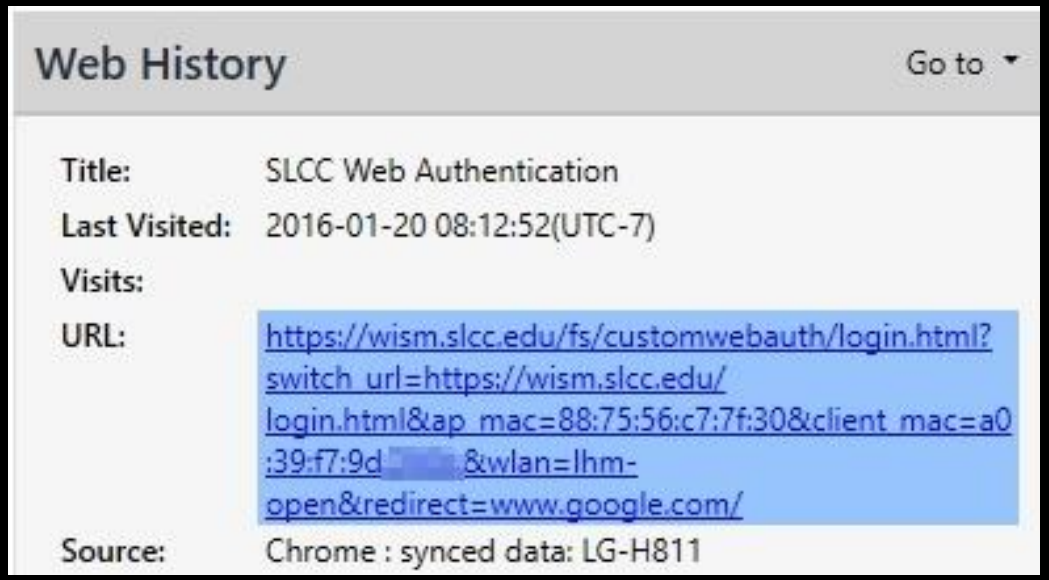
WiFi – connected vs. seen

Seen

- herrevad DB local – Android
- google maps
- iOS geolocation
- google geolocation in general

Connected

- wpa_supplicant.conf – Android
- dhcp records – Android
- URLs from captive portals



get_bssids_google.py

cell towers

- Range > 1000m

- Identified by (GSM):
  - MCC – Mobile Country Code
  - MNC – Mobile Network Code
  - LAC – Location Area Code
  - CID – Cell ID

- Identified by (CDMA):
  - SID – System ID
  - NID – Network ID
  - BID – Basestation ID

http://mcc-mnc.com/ http://wiki.opencellid.org/wiki/Public:CDMA http://www.roamingzone.com/sid/

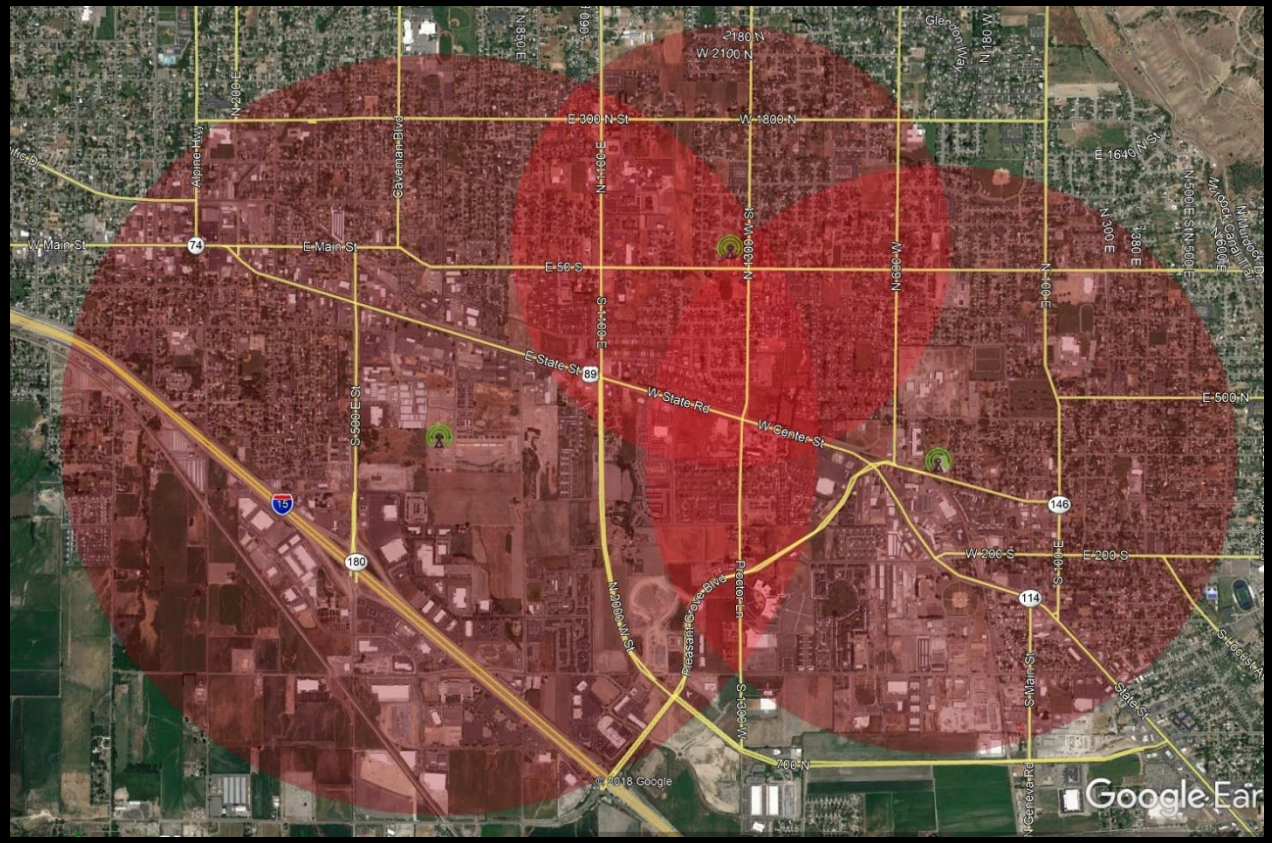
cell towers – connected vs. seen

Seen

- herrevad DB local – Android
- google maps
- iOS geolocation
- google geolocation in general

Connected

- herrevad local reports table
  - look for network quality
- URLs from captive portals



resources

- MapMaker
  - https://www.darrinward.com/lat-long/
- Earth Point
  - http://www.earthpoint.us/Convert.aspx
- Wigle
  - https://wigle.net/
- opencellid.org
  - https://opencellid.org/

- Skyhook Wireless
  - https://www.skyhook.com/
- Mozilla Location Services
  - https://location.services.mozilla.com/
- County Parcel Maps
  - Each county has their own
- Voter-registration records online
  - https://voterrecords.com/

geolocation APIs

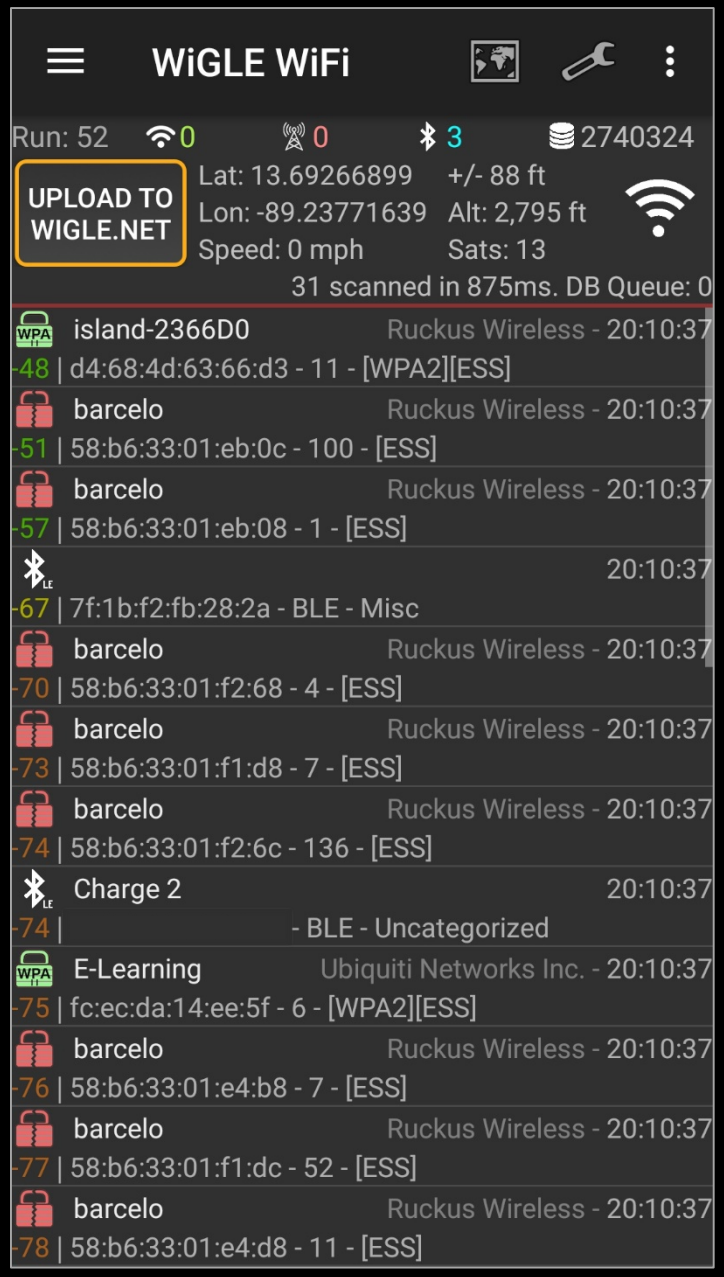
- Wigle.net (WiFi, Cell, Bluetooth)
- OpenCellid.org (Cell)
- Mozilla Location Services (Cell, WiFi)
- Google Geolocation (lat/long from WiFi/Cell)
- Google Geocoding (address <-> lat/long)
- OpenStreetMaps
- Skyhook

```
import googlemaps
from googlemaps.geolocation import geolocate

gmaps = googlemaps.Client(key='{your key}')

nets = [{'macAddress':'04:27:58:1E:F3:2B'},\
        {'macAddress':'AC:EC:80:80:5B:E0'},\
        {'macAddress':'0C:EA:C9:00:6A:3B'},\
        {'macAddress':'E2:88:5D:46:E6:00'},\
        {'macAddress':'0C:EA:C9:11:8B:BA'},\
        {'macAddress':'0C:EA:C9:11:8B:BC'}]

location_result = geolocate(gmaps, wifi_access_points=nets)

lat = location_result[u'location'][u'lat']
lng = location_result[u'location'][u'lng']
accuracy = location_result[u'accuracy']
address = gmaps.reverse_geocode((lat, lng))[0][u'formatted_address']

print 'Position is: {},{}. Precision is {} meters.'.\
      format(lat, lng, accuracy)
print 'Address is {}'.format(address)
```

Google Geolocation/Geocoding API – Python

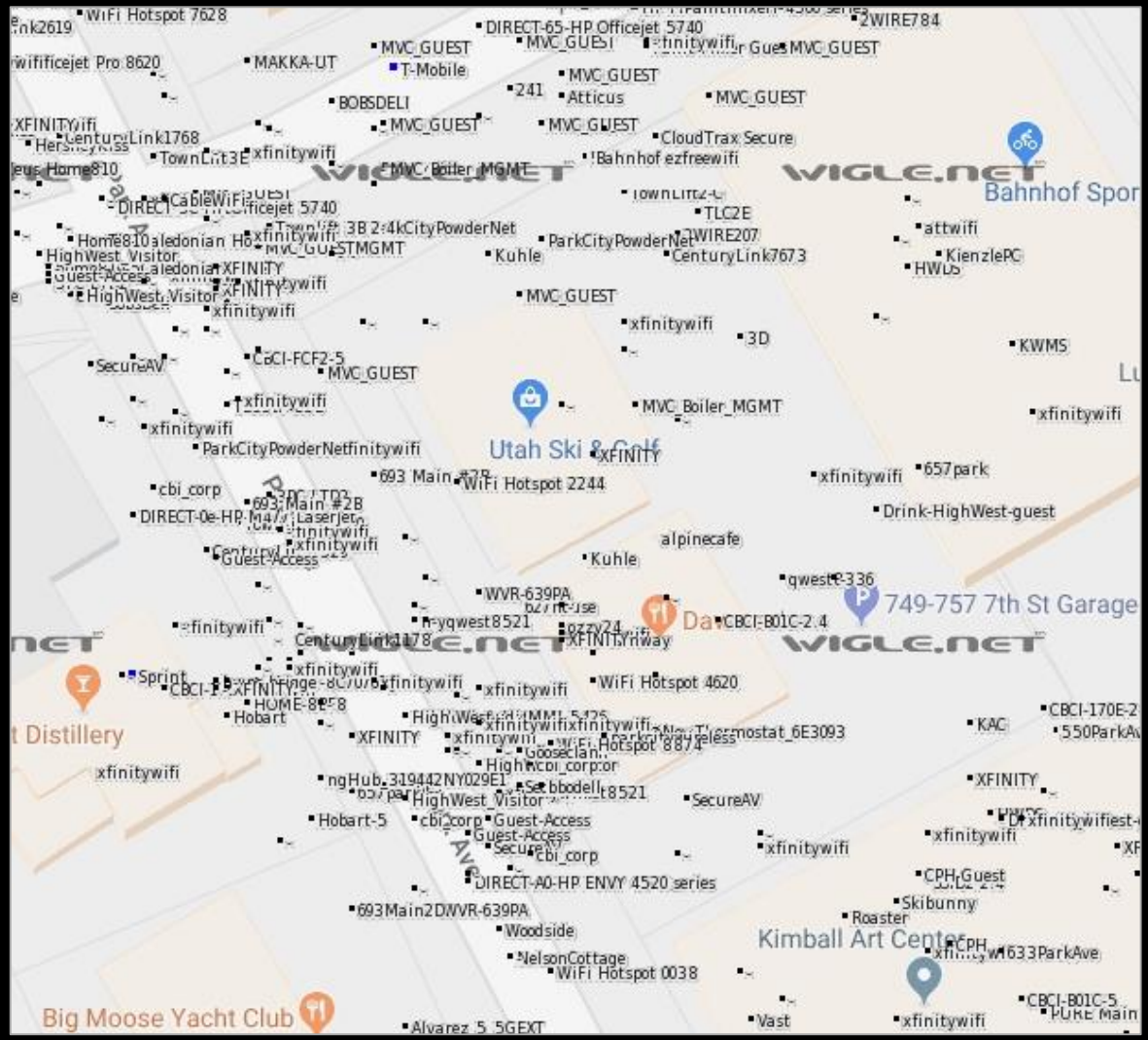
wigle.net

- Nearly 500 million wireless devices mapped
- Crowd-sourced (83%+ through their Android app)
- Free to use and contribute
- Mobile app is easy to use (Wigle Wardriving)
- User-friendly website
- Competitive component
- Registration is free (username/email/password)
- Special data request: wigle-admin@wigle.net



wigle.net

- Map is browsable on the Internet
- Searches are easy
- For more queries, request them from wigle-admin@wigle.net, stating your use case
- Go to https://api.wigle.net/ for more info and live searching to practice with the API



wigle.net – API Testing

Curl

```
curl -X GET "https://api.wigle.net/api/v2/network/search?onlymine=false&first=0&freenet=false&paynet=false&ssid=Marriott_GUEST" -H "accept: application/json"
```

Request URL

```
https://api.wigle.net/api/v2/network/search?onlymine=false&first=0&freenet=
```

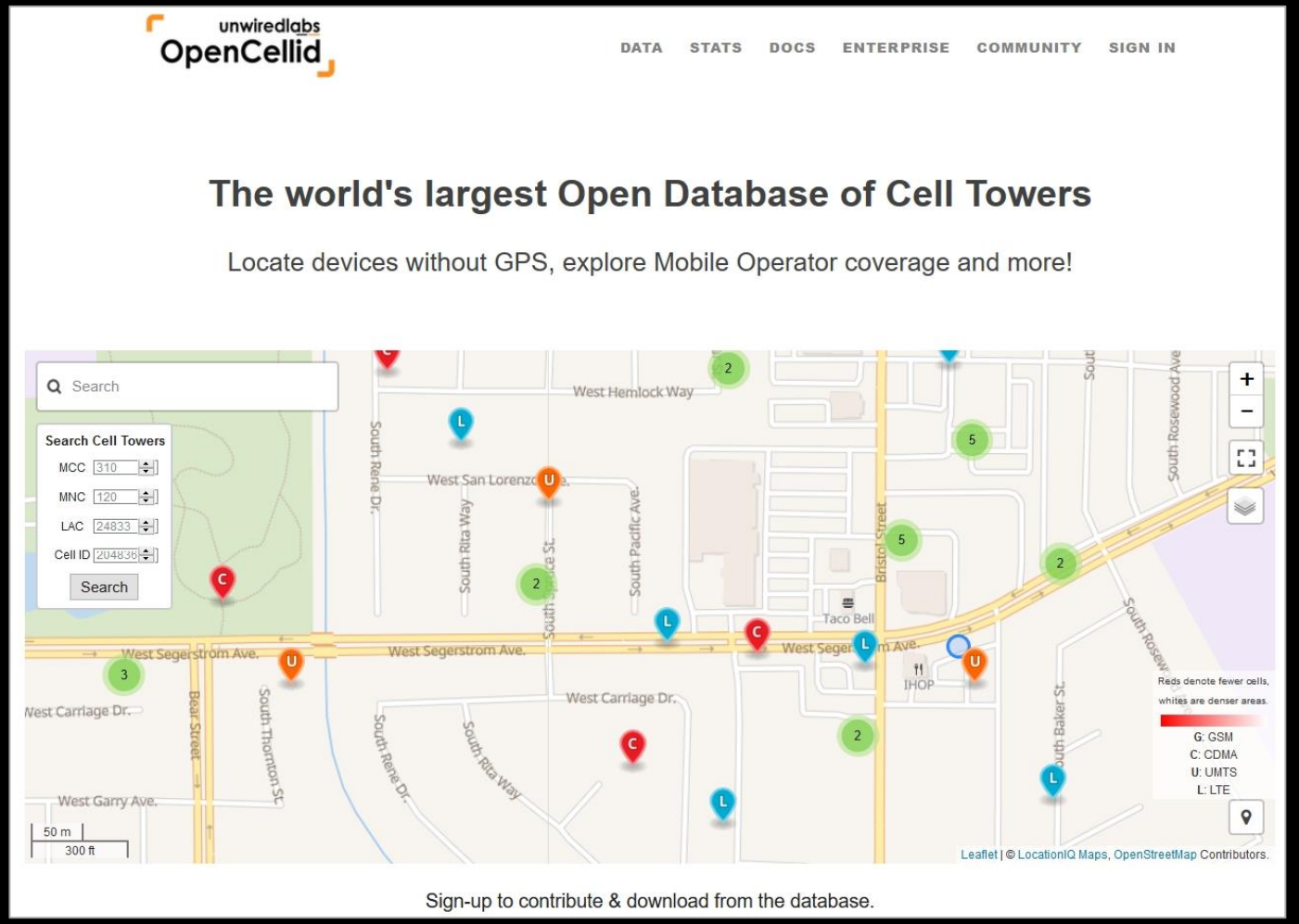
Response body


```
{
  "success": true,
  "totalResults": 65964,
  "search_after": 3701374,
  "first": 1,
  "last": 100,
  "resultCount": 100,
  "results": [
    {
      "trilat": 33.45885086,
      "trilong": -111.98564911,
      "ssid": "Marriott_Guest",
      "qos": 7,
      "transid": "20080417-00000",
      "firsttime": "2008-04-17T18:00:00.000Z",
      "lasttime": "2014-04-25T21:00:00.000Z",
      "lastupdt": "2014-11-11T10:00:00.000Z",
      "housenumber": "",
      "road": "Red Mountain Freeway",
      "city": "Phoenix",
      "region": "AZ",
      "country": "US",
      "netid": "00:1C:B0:E9:97:40",
      "name": null,
```


https://api.wigle.net/swagger#/Network search and information tools/

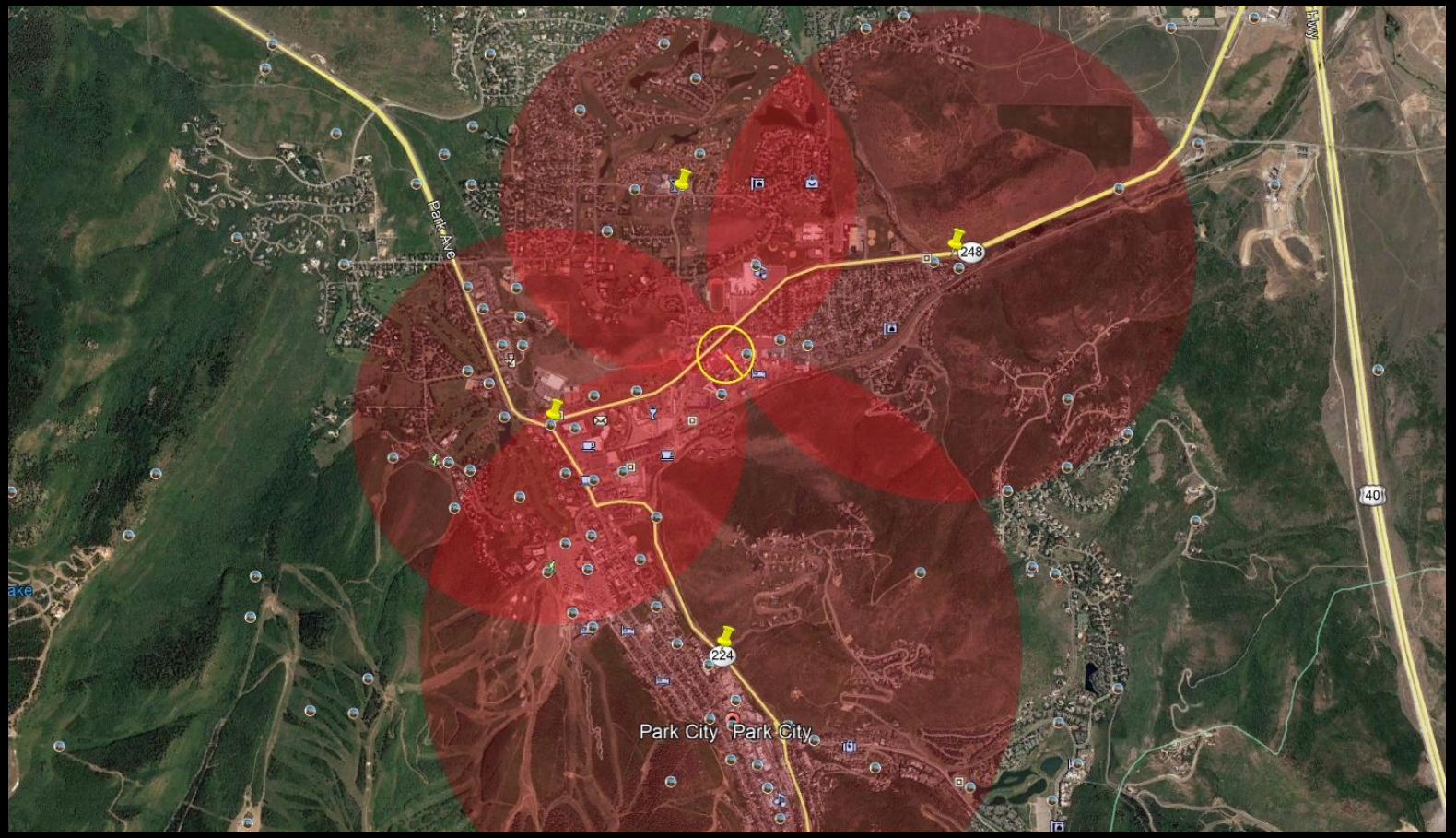
opencellid.org

- Owned by Unwired Labs
- Free to sign up and download entire database (csv.gz)
- Crowdsourced through at least two different apps
- Data in CSV format
- API has generous free limits
- For cell, it's easier to just download



http://opencellid.org/downloads

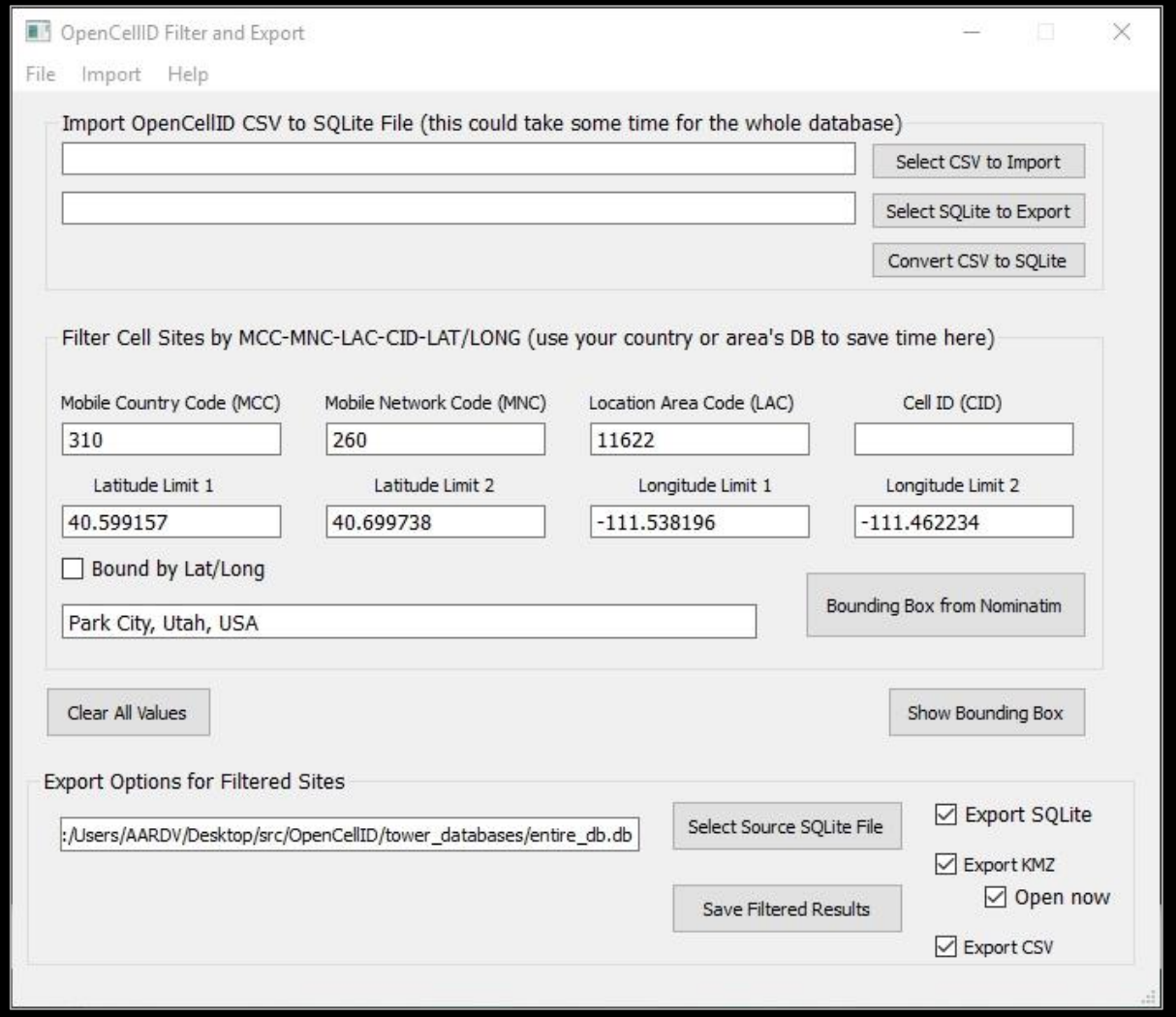
weighted-centroid trilateration



Multiple Measurements with differing signal strengths

opencellid.org

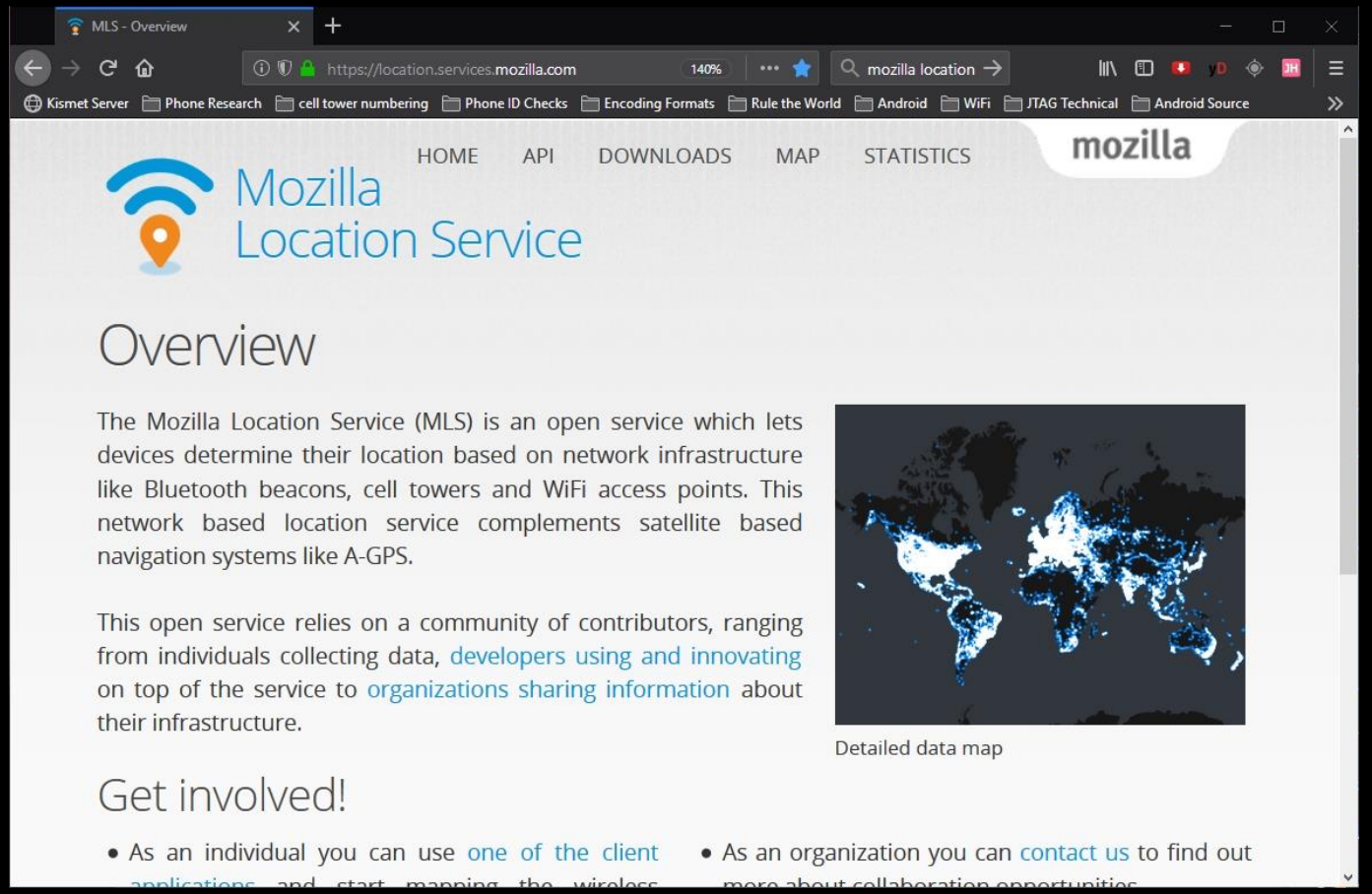
- OpenCellID Filter and Export
- Python2.7 and QT4 (yeah, yeah, I know)
- Currently rewriting in Python3 and QT5



filter_OCID.py

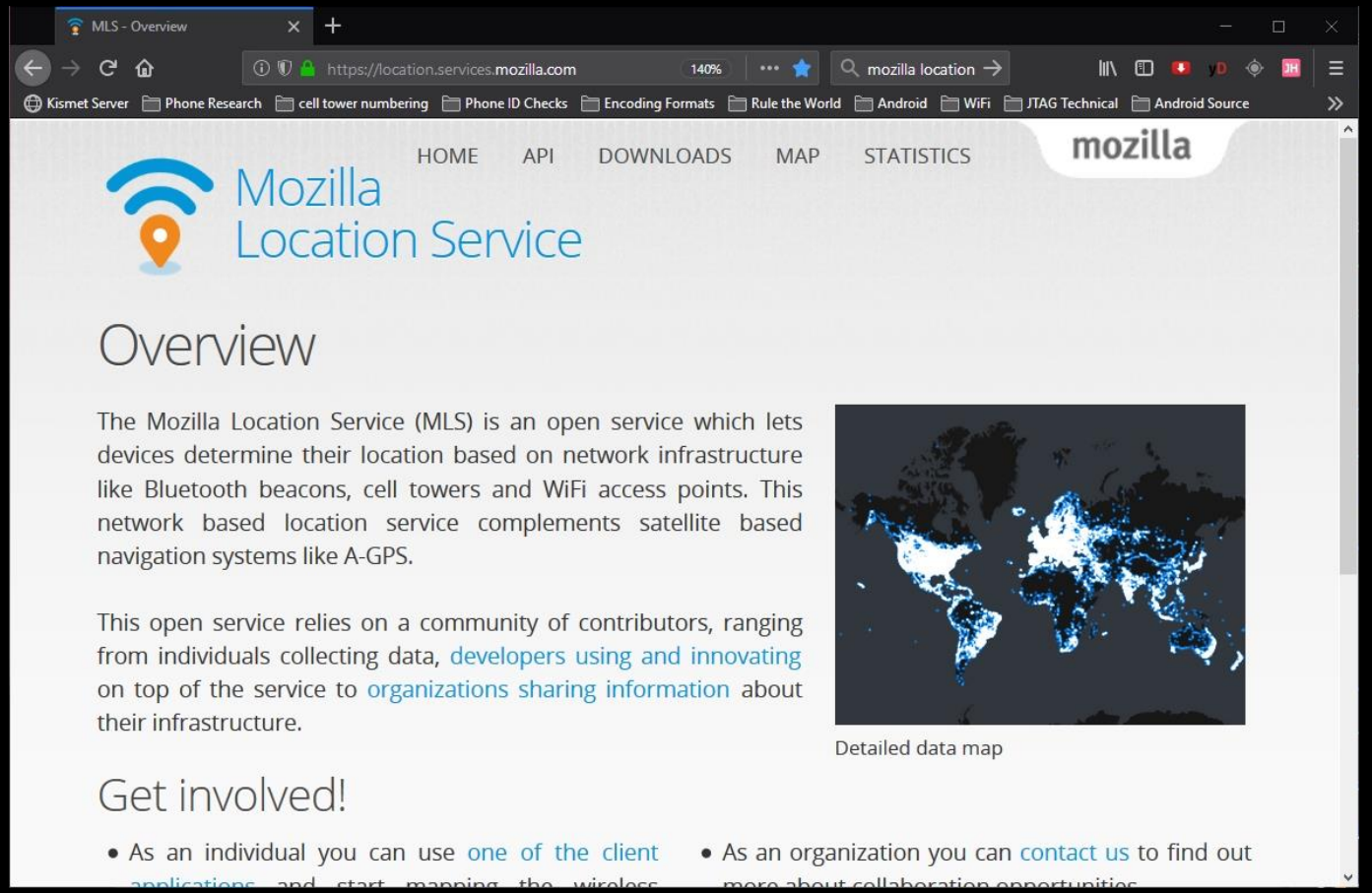
mozilla location services

- The Mozilla you know and love
- Formerly partnered with opencellid, so the format is the same
- Cell and WiFi
- Daily Cell DB downloadable (csv), along with hourly differentials
- Live API with generous free limits



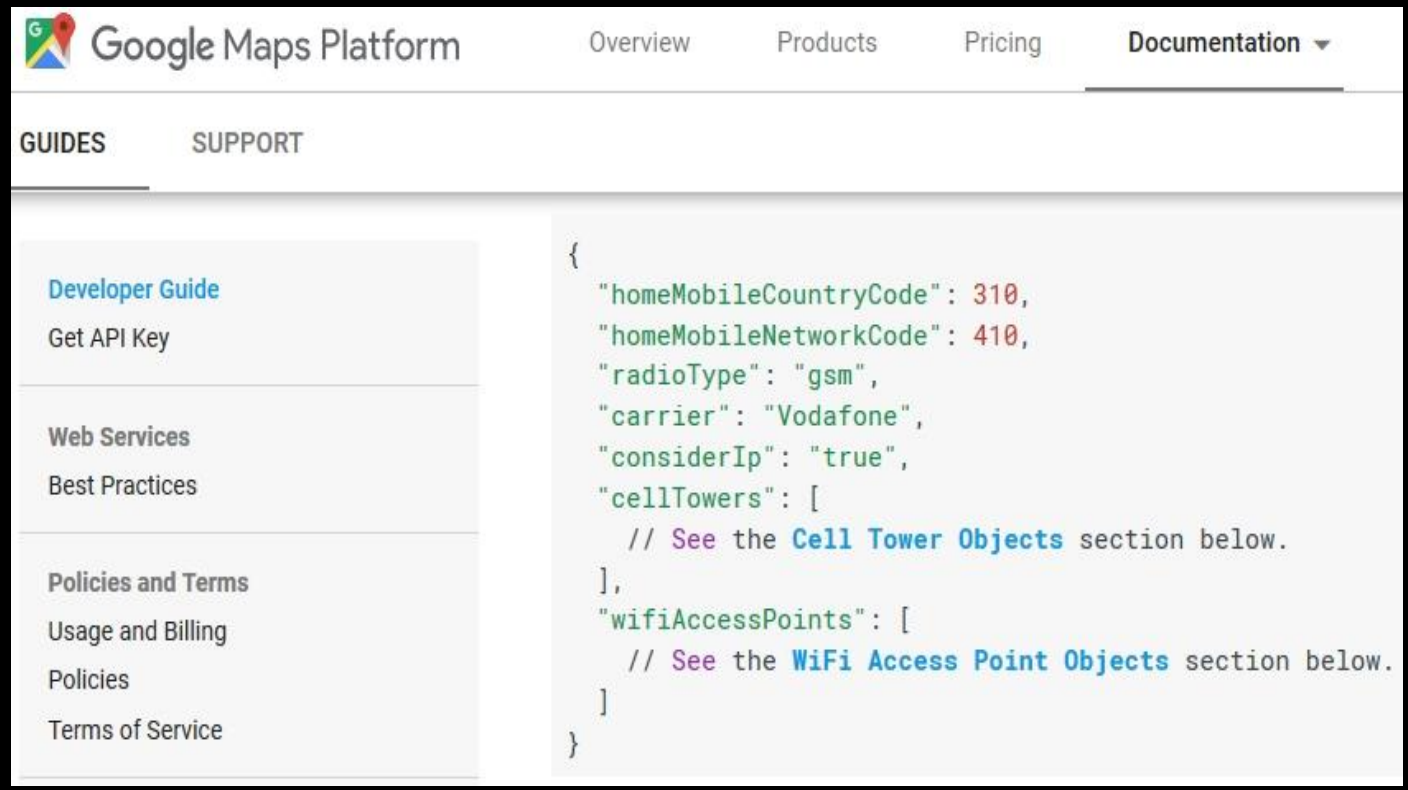
mozilla location services

- The Mozilla you know and love
- Formerly partnered with opencellid, so the format is the same
- Cell and WiFi
- Daily Cell DB downloadable (csv), along with hourly differentials
- Live API with generous free limits



google geolocation api

- Yes, Google is watching us…
- Or rather, we're watching us for Google
- The data is crowdsourced as well as collected by Google vehicles
- Used by Google devices to geolocate themselves
- Simple and free within reasonable limits (formerly 2500 requests/day)



get_bssids_google.py

google geolocation api

- Find position on the ground (lat/long) based on cell towers or wifi positions seen around your position
- Useful for analyzing data from phone dumps
- Just need BSSID/MAC of AP and (optionally) RSSI

```
C:\Users\AARDV\Desktop\src\google_geo>python google_geolocate.py
MAC: 24:A4:3C:7D:BD:D0  RSSI: -74
MAC: 00:15:6D:BB:6C:BF  RSSI: -89
MAC: 24:A4:3C:7D:BD:CE  RSSI: -90
MAC: 9C:D3:6D:A0:58:1E  RSSI: -90
MAC: 24:A4:3C:DA:DA:CE  RSSI: -91
MAC: 5C:DC:96:DA:67:4A  RSSI: -92
MAC: D0:05:2A:98:CB:F2  RSSI: -94

Latitude: 31.7438496
Longitude: -106.4519682
Accuracy: 94.0 meters
Address: Chetumal, Zona Pronaf Condominio La Plata, 32315 Cd Juárez, Chih., Mexico
```

google geocoding api

- Get addresses from lat/long or vice versa
- Need Google Developers account and API key

geocode_simple.py

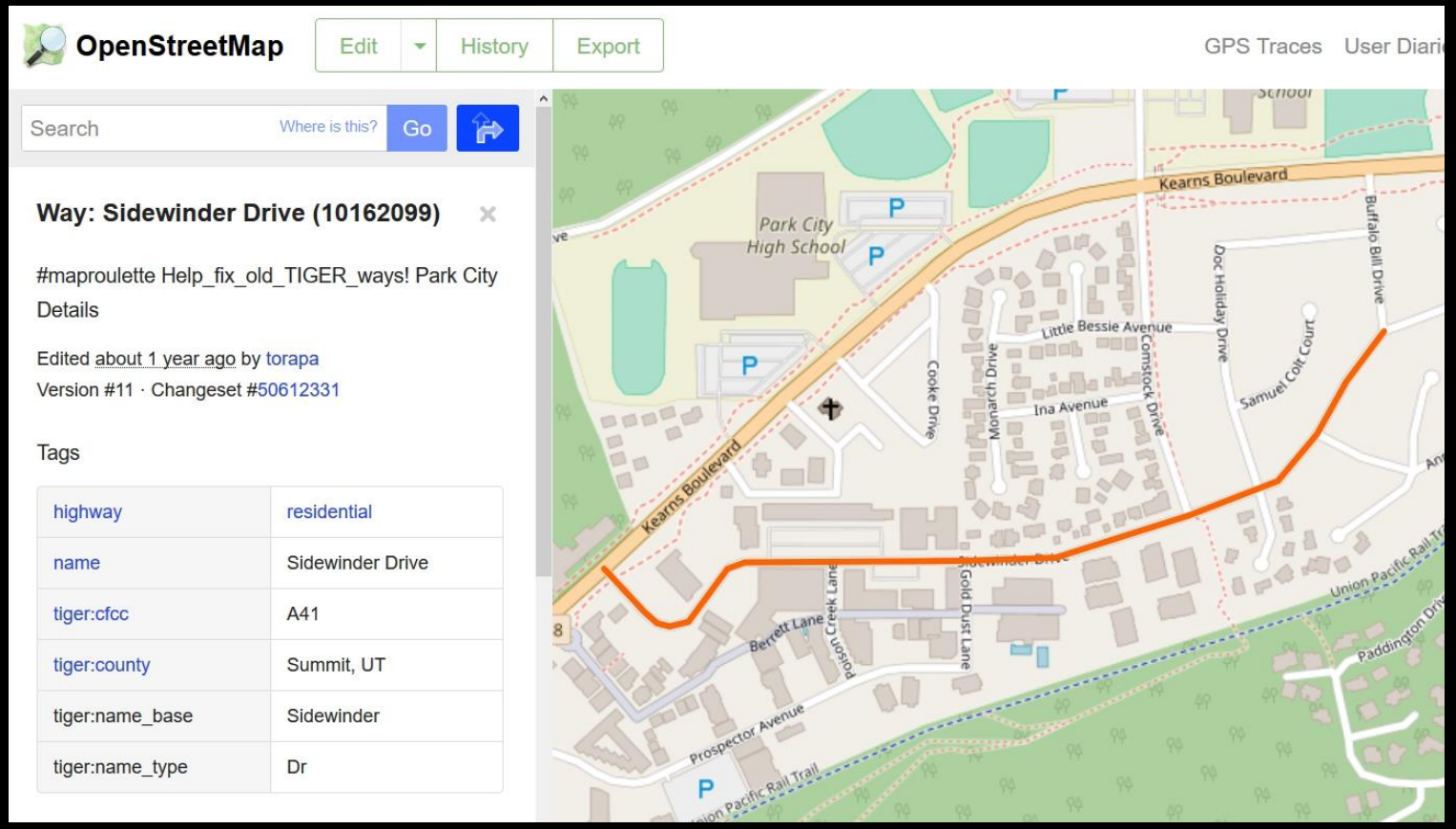
```
import googlemaps

lat, lng = 40.663749, -111.497065

gmaps = googlemaps.Client(key='YOUR_API_KEY_HERE')

addresses = gmaps.reverse_geocode((lat, lng))

#result returns a list of hits

print 'Geocoding {}, {} into an address...'.format(lat,lng)
print addresses[0][u'formatted_address']
```

```
C:\Users\AARDV\Desktop\src\geo_location_tools\google_geolocation>python geocode_simple.py
Geocoding 40.663749, -111.497065 into an address...
1895 Sidewinder Dr, Park City, UT 84060, USA
```

https://developers.google.com/maps/documentation/geolocation/get-api-key

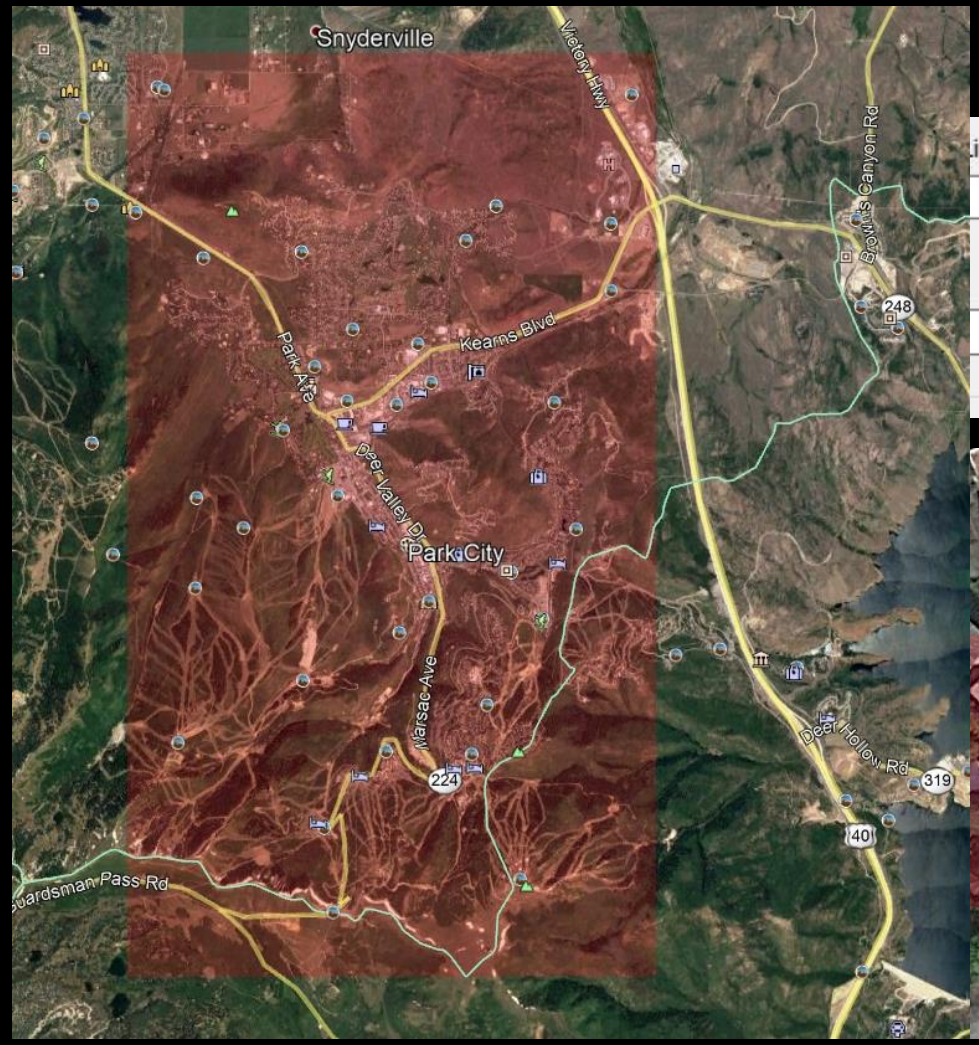
OpenStreetMap

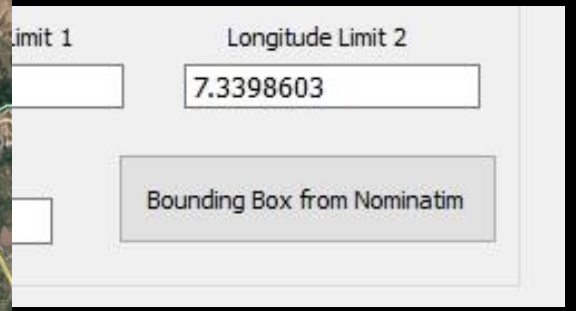
- free to use
- crowdsourced
- can be incomplete
- can be used in Python through the geopy library
- you can contribute to the project as well



OpenStreetMap - Nominatim

- geocoding
- named location searching
- bounding boxes for filtering

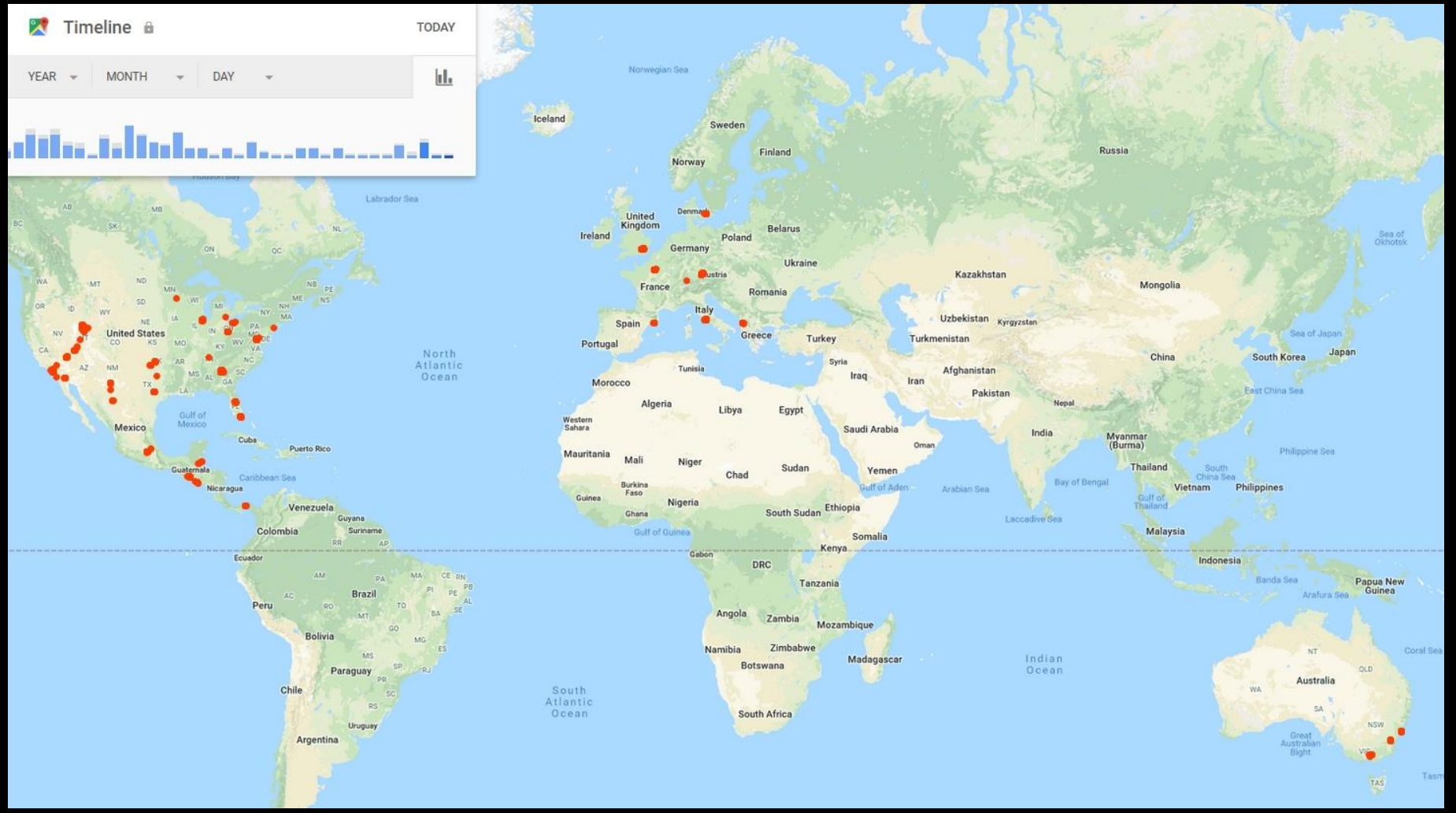




skyhook

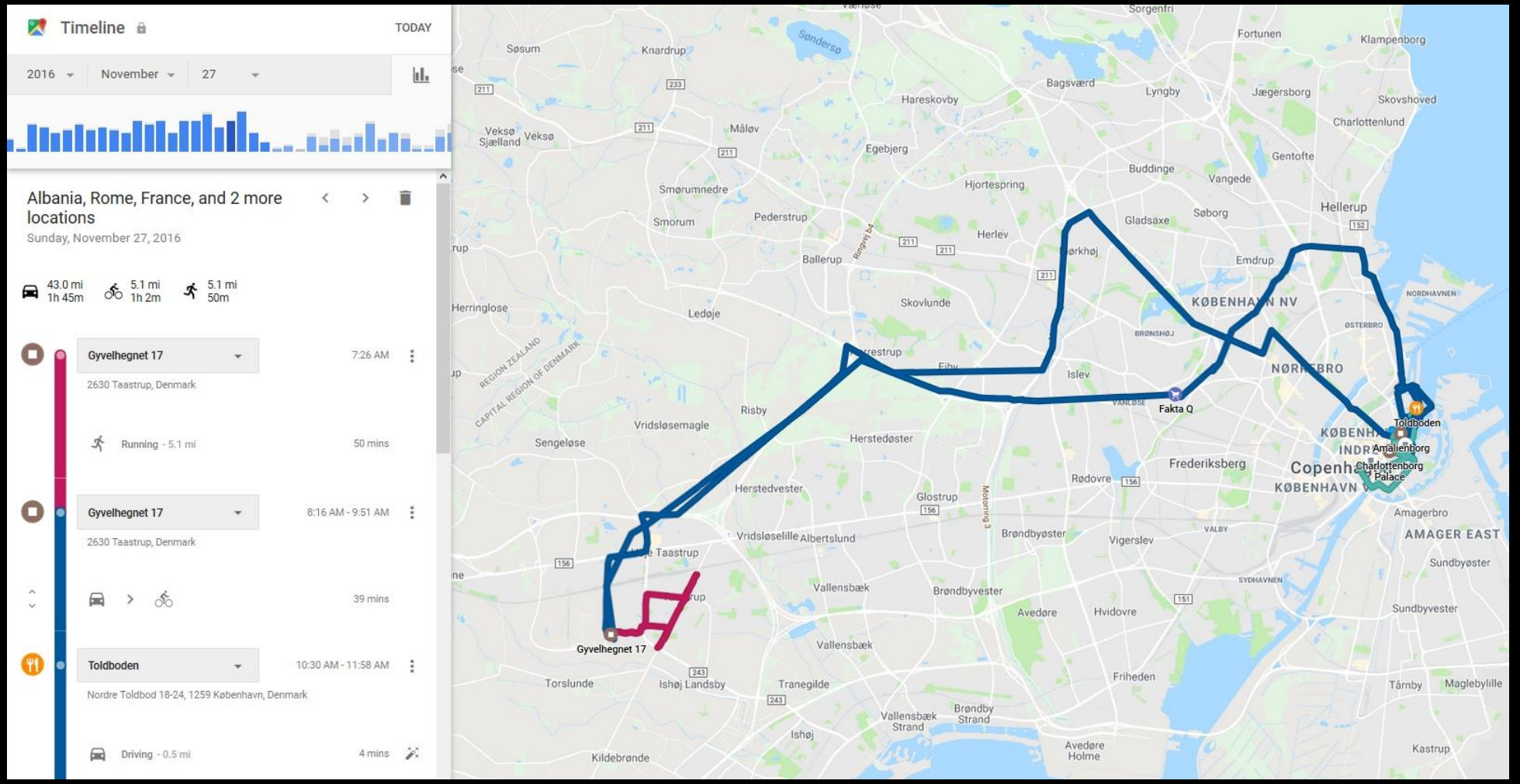
- pioneers of wifi/cell geolocation
- paid
- free-ish access through python-geoclue

google location history



Overall, showing total history

google location history

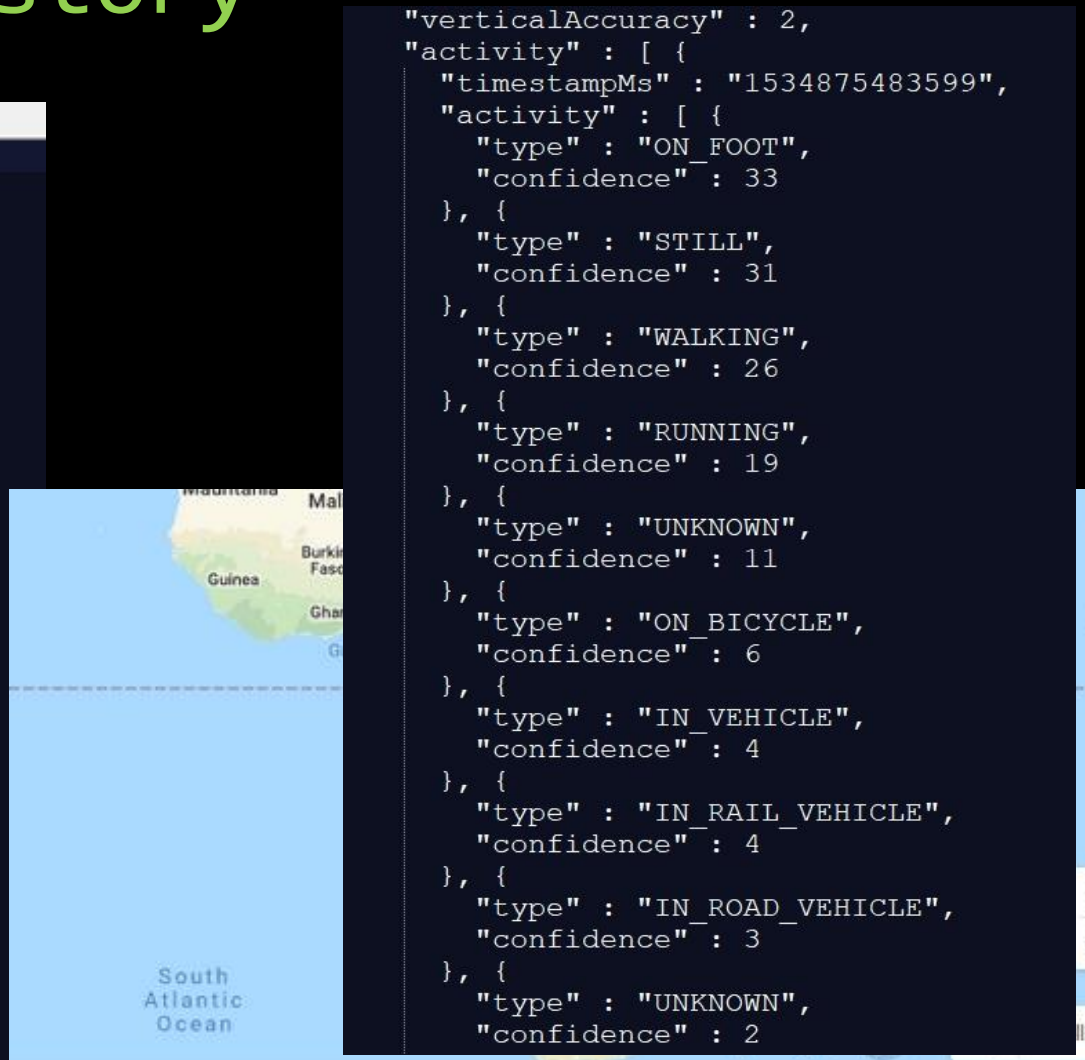


Dailies, including specific activity types

google location history

Location History.json


```
{
  "locations" : [ {
    "timestampMs" : "1534875901466",
    "latitudeE7" : 136928408,
    "longitudeE7" : -892377220,
    "accuracy" : 82,
    "altitude" : 810,
    "verticalAccuracy" : 2,
    "activity" : [ {
      "timestampMs" : "1534875890588",
      "activity" : [ {
        "type" : "STILL",
        "confidence" : 100
      } ]
    } ]
  }, {
    "timestampMs" : "1534875875533",
    "latitudeE7" : 136927999,
    "longitudeE7" : -892377783,
    "accuracy" : 32,
    "altitude" : 810,
    "verticalAccuracy" : 2
  }, {
    "timestampMs" : "1534875853593",
    "latitudeE7" : 136927982,
    "longitudeE7" : -892377634,
    "accuracy" : 38,
    "altitude" : 810,
    "verticalAccuracy" : 2
  }, {
```

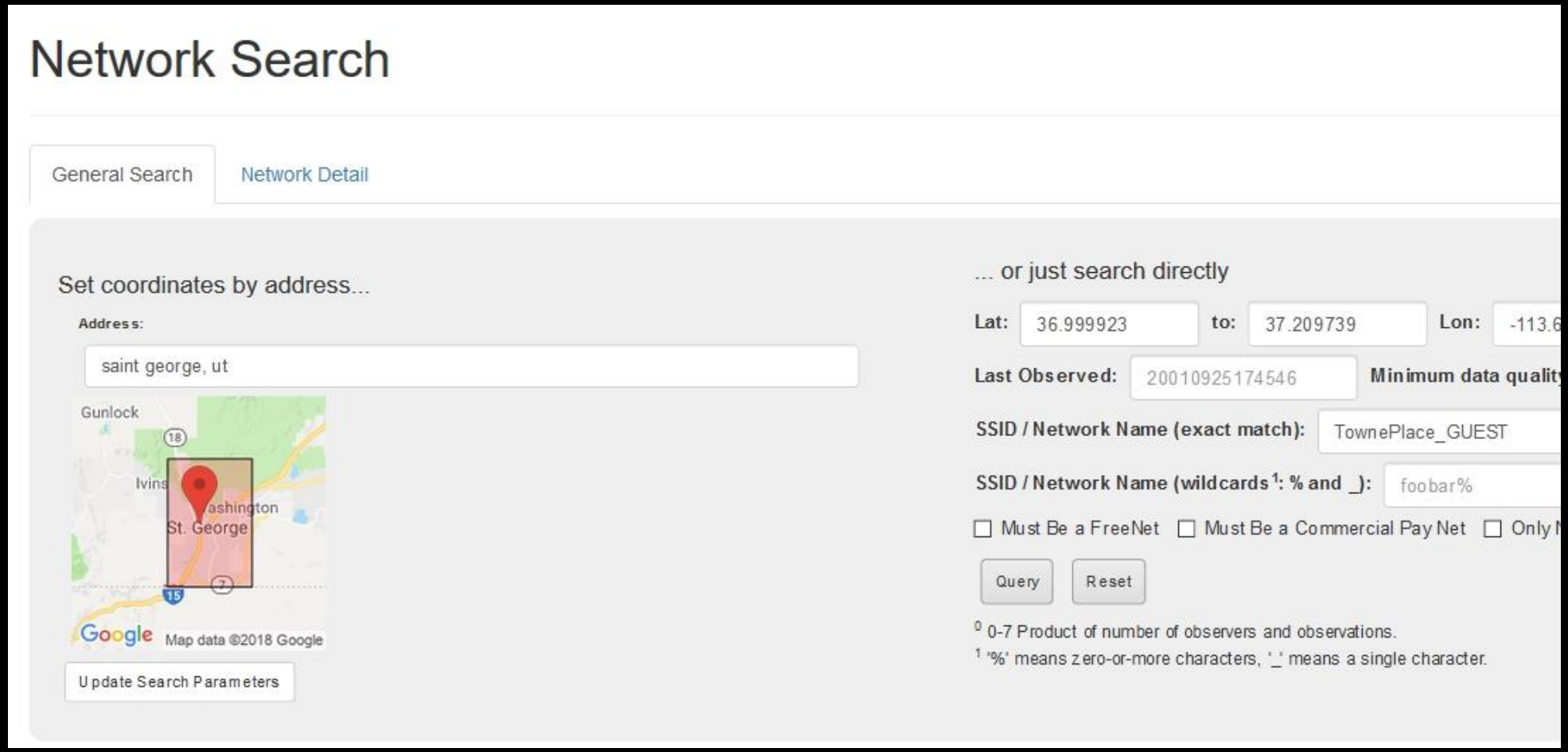



Still working on these

example – dhcp records

Galaxy S7 (SM-G930U)
Only one connected network, no BSSIDs or network records

example – dhcp records



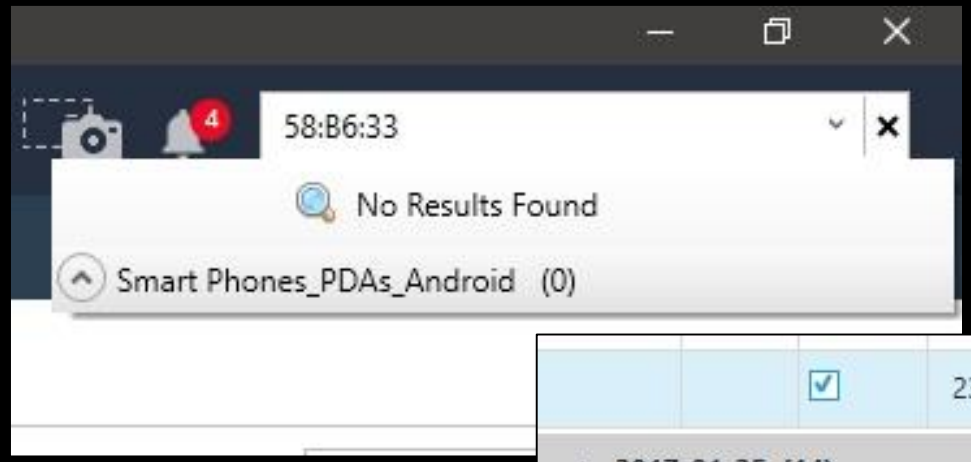
wigle search

example – dhcp records

Map	Net ID	SSID	Name	Type	First Seen	Most Recently	Crypto	Est. Lat	Est. Long
map	58:B6:33:1E:65:E8	TownePlace_GUEST		infra	2016-04-05T20:00:00.000Z	2017-03-18T12:00:00.000Z		37.10405731	-113.55142212
map	58:B6:33:1E:71:FC	TownePlace_GUEST		infra	2001-12-31T17:00:00.000Z	2016-04-06T10:00:00.000Z		37.1036377	-113.55227661
map	58:B6:33:1E:83:88	TownePlace_GUEST		infra	2001-12-31T17:00:00.000Z	2016-04-05T19:00:00.000Z		37.10365295	-113.55252075
map	58:B6:33:1E:D3:D8	TownePlace_GUEST		infra	2016-04-05T16:00:00.000Z	2017-01-09T18:00:00.000Z		37.10384369	-113.55115509
map	58:B6:33:1E:E9:3C	TownePlace_GUEST						37.10135651	-113.55999756
map	58:B6:33:1E:F8:B8	TownePlace_GUEST						37.10277176	-113.55513

wigle.net and macvendors.com

example – dhcp records



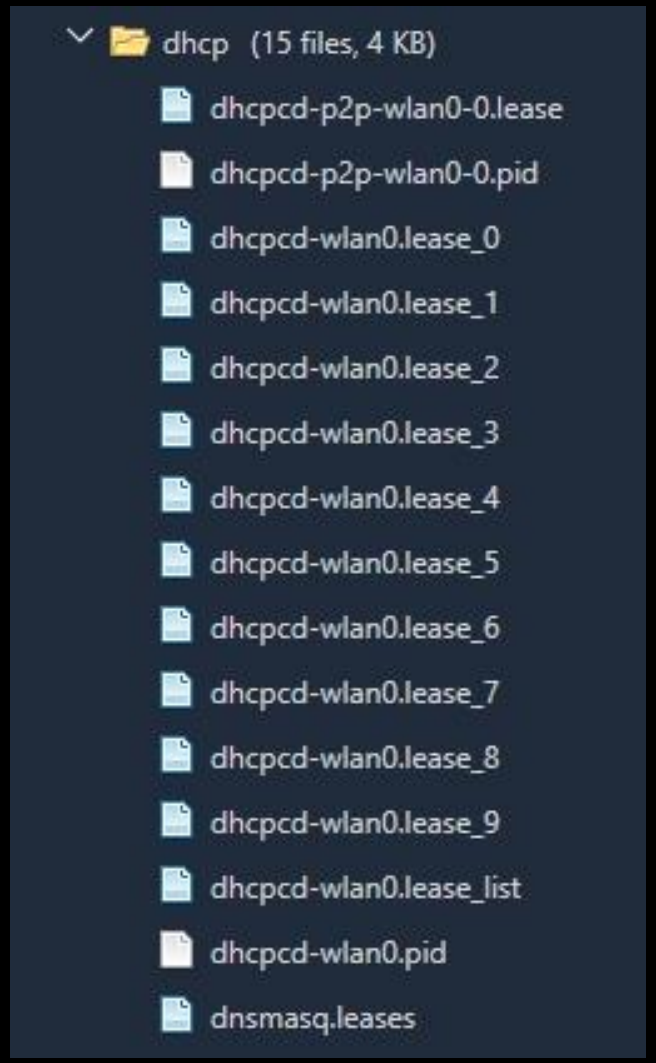
☑	23	✖	Wireless Network...	2016-12-31 18:36:52(UTC-7)
2017-01-25 (14)				
☑	24	✖	Cell Tower: Locati...	2017-01-25 15:49:10(UTC-7)
☑	25	✖	Cell Tower: Locati...	2017-01-25 15:49:50(UTC-7)
☑	26	✖	Cell Tower: Locati...	2017-01-25 15:53:23(UTC-7)

No results parsed for the particular networks

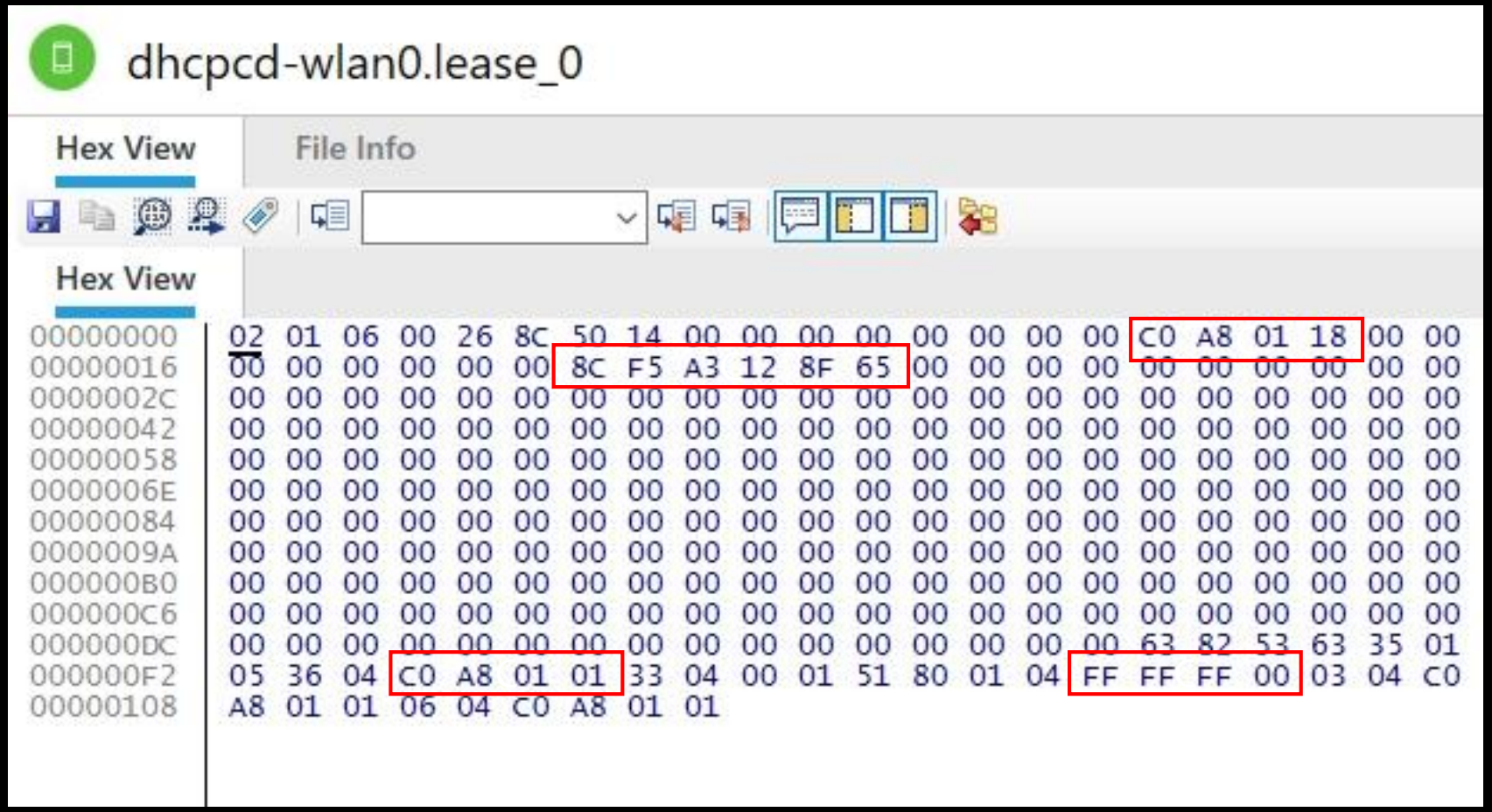
example – dhcp records

- /data/misc/dhcp/
  - dhcpcd-wlano.lease_list (Samsung, ten networks)
  - files named with MAC (BSSID) of access points (LG)
  - dhcp acknowledge packets/files (DHCP ACK)
  - dnsmasq.leases holds records of devices that connected to the phone's hotspot (text file, unix line endings)

- These hold the MACs/BSSIDs of connected networks

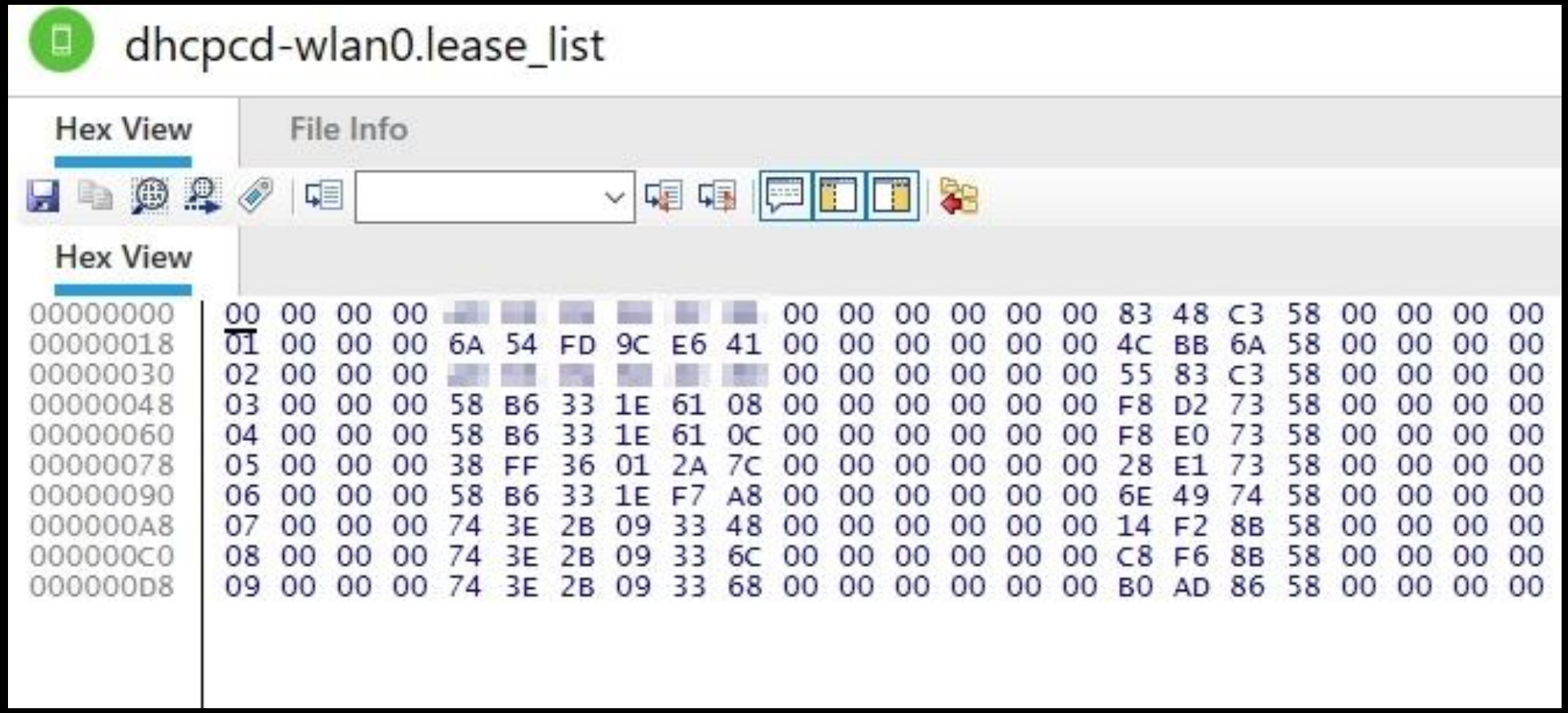


example – dhcp records



https://en.wikipedia.org/wiki/Dynamic_Host_Configuration_Protocol

example – dhcp records



dhcp_lease_list_ext.py mac_mutate.py

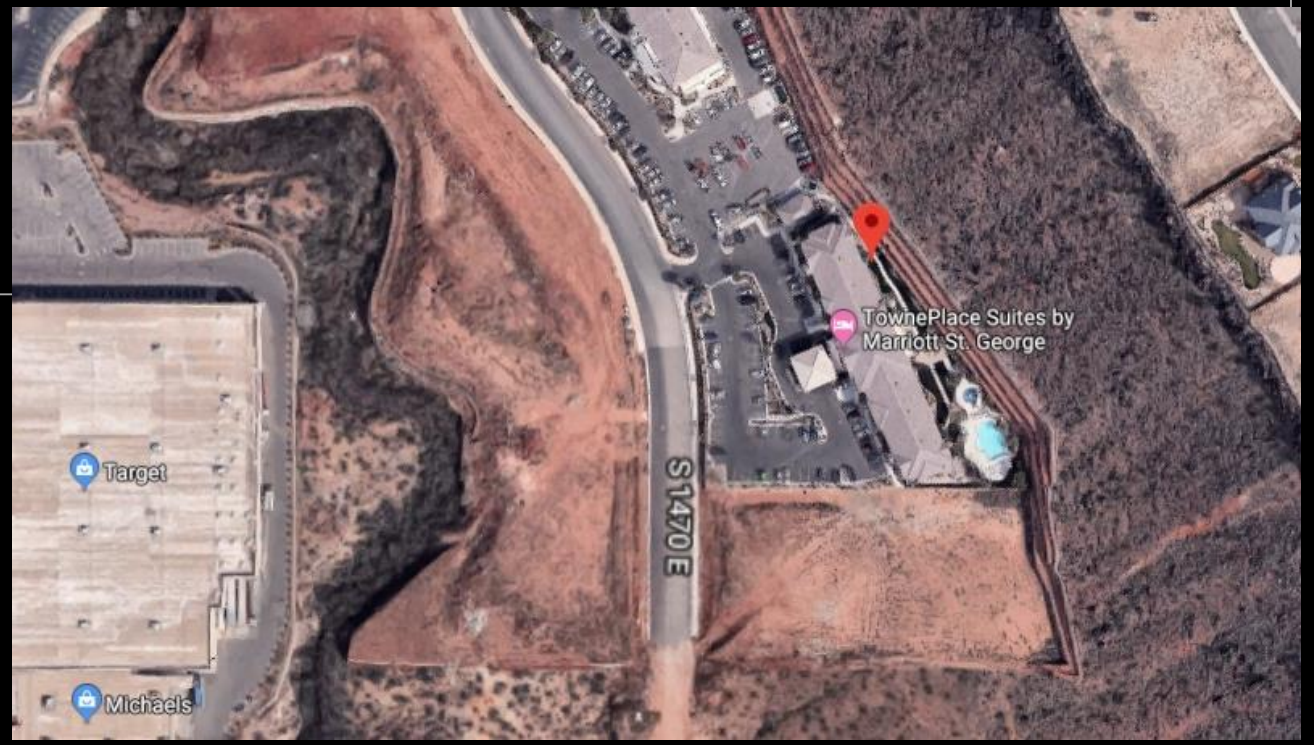
example – dhcp records

	A	B	C
1	DHCP Server MAC Address	TimeStamp	DHCP Server Manufacturer
2	[redacted]	2017-03-10 17:44:51	NETGEAR
3	6A:54:FD:9C:E6:41	2017-01-02 13:42:52	Amazon Technologies Inc.
4	[redacted]	2017-03-10 21:55:49	NETGEAR
5	58:B6:33:1E:61:08	2017-01-09 11:14:16	Ruckus Wireless
6	58:B6:33:1E:61:0C	2017-01-09 12:14:00	Ruckus Wireless
7	38:FF:36:01:2A:7C	2017-01-09 12:14:48	Ruckus Wireless
8	58:B6:33:1E:F7:A8	2017-01-09 19:39:42	Ruckus Wireless
9	74:3E:2B:09:33:48	2017-01-27 18:21:24	Ruckus Wireless
10	74:3E:2B:09:33:6C	2017-01-27 18:41:28	Ruckus Wireless
11	74:3E:2B:09:33:68	2017-01-23 18:28:16	Ruckus Wireless

example – dhcp records

Google Geolocation API search

```
C:\Users\AARDV\Desktop\src\pfic_2018>python google2.py
Enter macs (y/n): y
Enter macs separated by comma only: 38:FF:36:01:2A:7C,58:B6:33:1E:61:08,58:B6:33:1E:61:0C,58:B6:33:1E:F7:A8
MAC: 38:FF:36:01:2A:7C
MAC: 58:B6:33:1E:61:08
MAC: 58:B6:33:1E:61:0C
MAC: 58:B6:33:1E:F7:A8

Latitude: 37.1040487
Longitude: -113.550968
Accuracy: 150.0 meters
Address: 251 S 1470 E, St. George, UT 84790, USA
```



google_macs.py

example – dhcp records

Google Geolocation API search

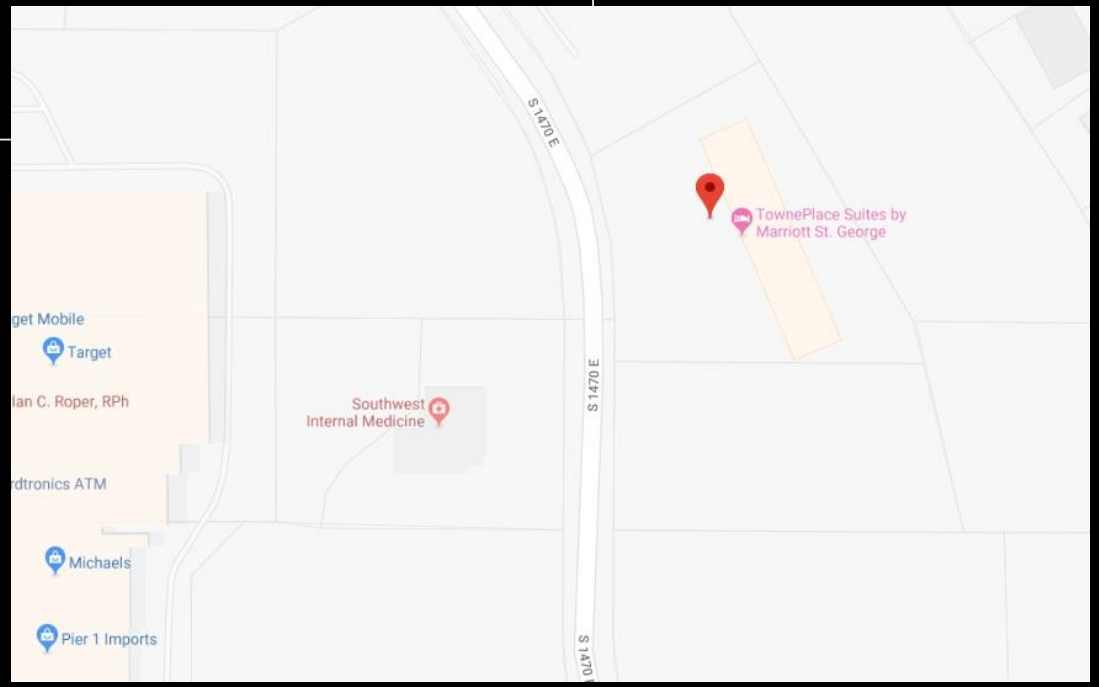
```
C:\Users\AARDV\Desktop\src\pfic_2018>python google2.py
Enter macs (y/n): y
Enter macs separated by comma only: 38:FF:36:01:2A:7C,58:B6:33:1E:61:08,58:B6:33:1E:61:0C,58:B6:33:1E:F7:A8
MAC: 38:FF:36:01:2A:7C
MAC: 58:B6:33:1E:61:08
MAC: 58:B6:33:1E:61:0C
MAC: 58:B6:33:1E:F7:A8

Latitude: 37.1040487
Longitude: -113.550968
Accuracy: 150.0 meters
Address: 251 S 1470 E, St. George, UT 84790, USA
```

google_macs.py

example – dhcp records

Mozilla Location Services API Search

```
C:\Users\AARDV\Desktop\src\pfic_2018>python mozilla_basic.py
Latitude: 37.1038519
Longitude: -113.5512025
Accuracy: 20.1745904
```

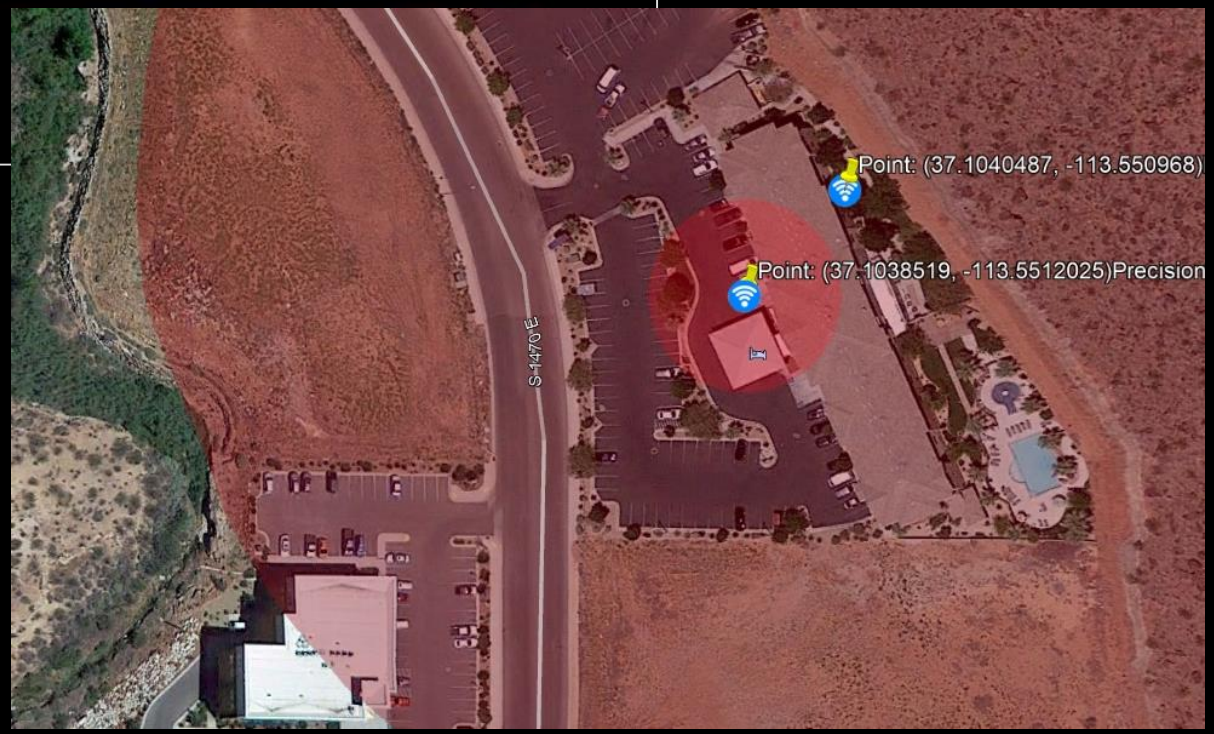


mozilla_basic.py

example – dhcp records

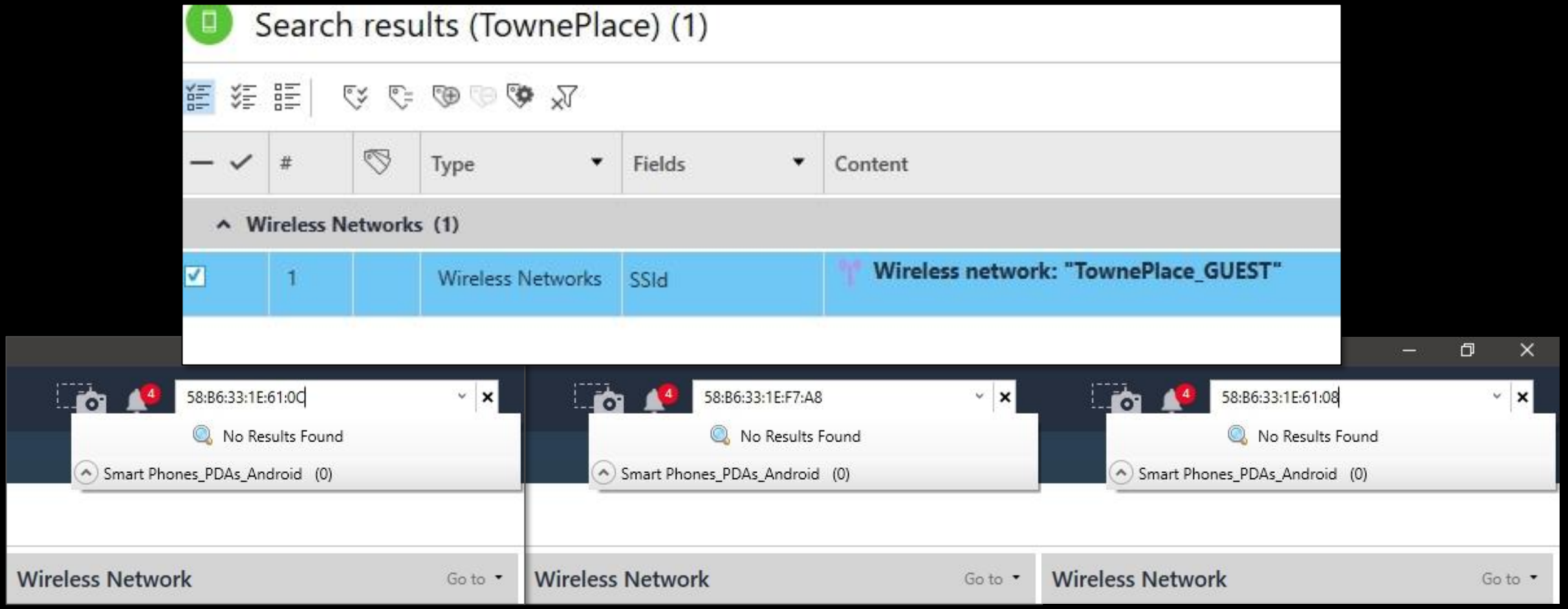
Mozilla Location Services API Search

```
C:\Users\AARDV\Desktop\src\pfic_2018>python mozilla_basic.py
Latitude: 37.1038519
Longitude: -113.5512025
Accuracy: 20.1745904
```

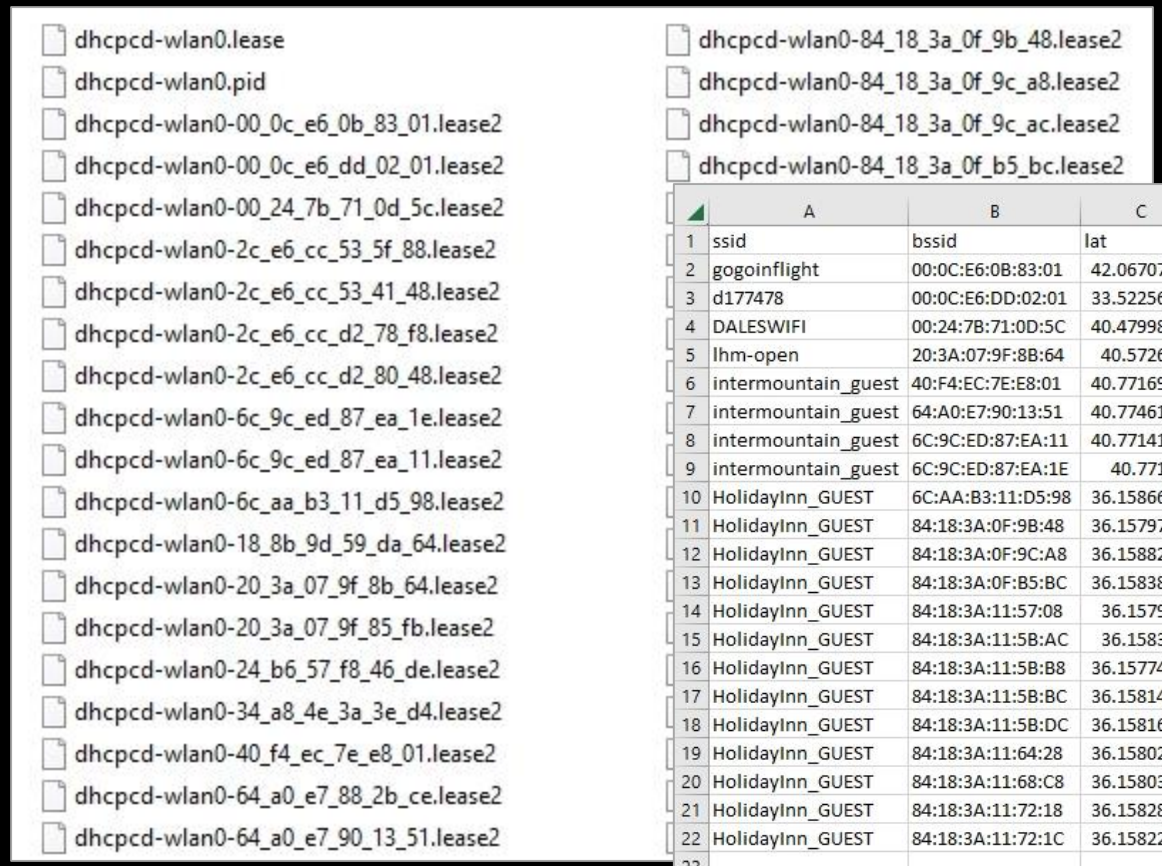


mozilla_basic.py

example – dhcp records



demo – lg dhcp records



	A	B	C	D	E	F	G	H	I	J
1	ssid	bssid	lat	lng	lasttime	country	region	city	house_number	road
2	gogoinflight	00:0C:E6:0B:83:01	42.06707001	-94.48470306	2018-08-18T19:00:00.000Z	US	IA			202nd Street
3	d177478	00:0C:E6:DD:02:01	33.52256393	-101.790123	2016-06-01T05:00:00.000Z	US	TX			Slaton Road
4	DALESWIFI	00:24:7B:71:0D:5C	40.47998047	-111.9581986	2017-11-22T20:00:00.000Z	US	UT	Bluffdale		2700 West
5	lhm-open	20:3A:07:9F:8B:64	40.5726738	-111.9001846	2016-03-27T19:00:00.000Z	US	UT	Sandy		I 15
6	intermountain_guest	40:F4:EC:7E:E8:01	40.77169037	-111.839386	2017-10-23T13:00:00.000Z	US	UT	Salt Lake City	100	Mario Capecchi Drive
7	intermountain_guest	64:A0:E7:90:13:51	40.77461624	-111.8579483	2017-10-30T19:00:00.000Z	US	UT	Salt Lake City		Cypress Avenue
8	intermountain_guest	6C:9C:ED:87:EA:11	40.77141571	-111.8392334	2016-07-03T17:00:00.000Z	US	UT	Salt Lake City	100	Mario Capecchi Drive
9	intermountain_guest	6C:9C:ED:87:EA:1E	40.771492	-111.8392944	2016-07-12T19:00:00.000Z	US	UT	Salt Lake City	100	Mario Capecchi Drive
10	HolidayInn_GUEST	6C:AA:B3:11:D5:98	36.15866089	-86.78357697	2018-05-15T11:00:00.000Z	US	TN	Nashville-Davidson	900	Broadway
11	HolidayInn_GUEST	84:18:3A:0F:9B:48	36.15797424	-86.78433228	2018-05-25T08:00:00.000Z	US	TN	Nashville-Davidson		10th Avenue North
12	HolidayInn_GUEST	84:18:3A:0F:9C:A8	36.15882492	-86.78555298	2018-03-30T22:00:00.000Z	US	TN	Nashville-Davidson		10th Avenue North
13	HolidayInn_GUEST	84:18:3A:0F:B5:BC	36.15838242	-86.78323364	2018-05-15T11:00:00.000Z	US	TN	Nashville-Davidson		Broadway
14	HolidayInn_GUEST	84:18:3A:11:57:08	36.1579628	-86.78437042	2018-05-25T08:00:00.000Z	US	TN	Nashville-Davidson		10th Avenue North
15	HolidayInn_GUEST	84:18:3A:11:5B:AC	36.1583519	-86.78456879	2017-07-11T18:00:00.000Z	US	TN	Nashville-Davidson		10th Avenue North
16	HolidayInn_GUEST	84:18:3A:11:5B:B8	36.15774155	-86.78580475	2018-04-13T21:00:00.000Z	US	TN	Nashville-Davidson		Broadway
17	HolidayInn_GUEST	84:18:3A:11:5B:BC	36.15814972	-86.78451538	2018-05-15T11:00:00.000Z	US	TN	Nashville-Davidson		10th Avenue North
18	HolidayInn_GUEST	84:18:3A:11:5B:DC	36.15816498	-86.78405762	2018-05-15T11:00:00.000Z	US	TN	Nashville-Davidson		10th Avenue North
19	HolidayInn_GUEST	84:18:3A:11:64:28	36.15802383	-86.78492737	2018-03-17T14:00:00.000Z	US	TN	Nashville-Davidson	1010	Broadway
20	HolidayInn_GUEST	84:18:3A:11:68:C8	36.15803909	-86.78444672	2018-05-25T08:00:00.000Z	US	TN	Nashville-Davidson		10th Avenue North
21	HolidayInn_GUEST	84:18:3A:11:72:18	36.15828705	-86.78375244	2018-05-15T11:00:00.000Z	US	TN	Nashville-Davidson		10th Avenue North
22	HolidayInn_GUEST	84:18:3A:11:72:1C	36.15822983	-86.78440094	2018-05-15T11:00:00.000Z	US	TN	Nashville-Davidson		10th Avenue North
23										

LG DHCP records use the BSSID in the filename and the file's modified date is the last connect time

lg_dhcp.py

websites I use

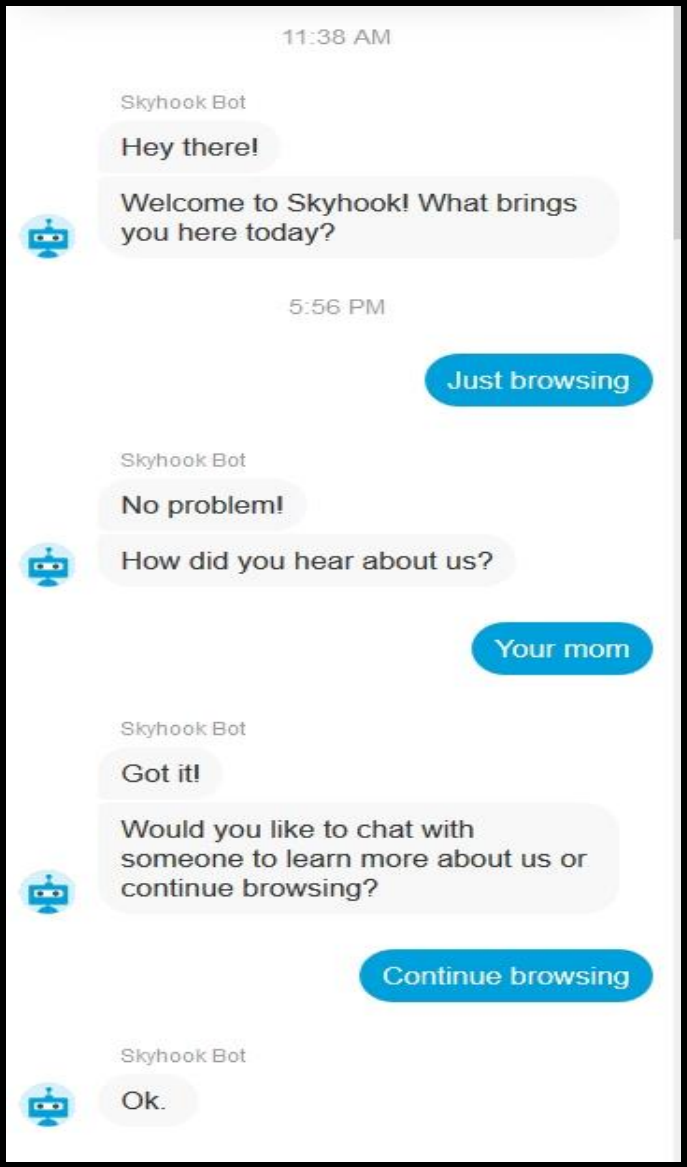
- wigle.net
- opencellid.org
- https://location.services.mozilla.com/
- https://www.openstreetmap.org
- https://www.gpsbabel.org/
- https://www.darrinward.com/lat-long/
- http://www.earthpoint.us/Convert.aspx
- https://github.com/googlemaps/google-maps-services-python

modules/libraries I use

- os, sys, json, struct, binascii, base64, hashlib, argparse, time, datetime, re, etc…
- geopy – geolocation/geocoding, mapping, openstreetmaps, etc.
- simplekml – create KML files
- pygle – work with wigle.net
- requests, urllib2, tweepy – for grabbing thingies off the interwebz
- subprocess – running other programs, opening Google Earth instances, etc.
- sqlite3, csv
- pandas, matplotlib – chewing on data and plotting it

online APIs I use

- wigle – wireless mapping, wardrving
- macvendors – OUI vs vendor/manufacturer
- Google Geolocation and Geocoding
- OpenStreetMaps, Nominatim from geopy
- opencellid.org (db download available)
- Mozilla Location Services (db download available)
- https://github.com/scivision/python-geoclue
- https://github.com/scivision/mozilla-location-wifi-python



Summary

- geolocation is for everyone
- it’s fun
- kim@h11dfs.com
- https://github.com/gkt-aardvark/geo_location_tools
- @ArdJect on Twitter